Efficient Computing for AI and Robotics: From Hardware Accelerators to Algorithm Design

Vivienne Sze (@eems_mit)
Massachusetts Institute of Technology

In collaboration with Luca Carlone, Yu-Hsin Chen, Joel Emer, Sertac Karaman, Tushar Krishna, Peter Li, Yi-Lun Liao, Fangchang Ma, Angshu Parashar, Amr Suleiman, Po-An Tsai, Diana Wofk, Nellie Wu, Tien-Ju Yang, Zhengdong Zhang

Slides available at
https://tinyurl.com/ISCAS2021Sze
Processing at “Edge” instead of the “Cloud”

Communication
Privacy
Latency
Existing Processors Consume Too Much Power

< 1 Watt

> 10 Watts
Efficient Computing with Cross-Layer Design

**Algorithms**
- Convolutions
- Pooling
- Convs
- Linear Classifier

**Systems**
- Object Categories / Positions
- F4 maps
- at (x,y)
- C3 feature maps
- at (x,y)
- at (x,y)

**Architectures**
- DCNN Accelerator
- 14x12 PE Array

**Circuits**
- On-Chip Buffer
- Spatial PE Array

Vivienne Sze  http://sze.mit.edu/  @eems_mit
Energy Dominated by Data Movement

<table>
<thead>
<tr>
<th>Operation</th>
<th>Energy (pJ)</th>
</tr>
</thead>
<tbody>
<tr>
<td>8b Add</td>
<td>0.03</td>
</tr>
<tr>
<td>16b Add</td>
<td>0.05</td>
</tr>
<tr>
<td>32b Add</td>
<td>0.1</td>
</tr>
<tr>
<td>16b FP Add</td>
<td>0.4</td>
</tr>
<tr>
<td>32b FP Add</td>
<td>0.9</td>
</tr>
<tr>
<td>8b Multiply</td>
<td>0.2</td>
</tr>
<tr>
<td>32b Multiply</td>
<td>3.1</td>
</tr>
<tr>
<td>16b FP Multiply</td>
<td>1.1</td>
</tr>
<tr>
<td>32b FP Multiply</td>
<td>3.7</td>
</tr>
<tr>
<td>32b SRAM Read (8KB)</td>
<td>5</td>
</tr>
<tr>
<td>32b DRAM Read</td>
<td>640</td>
</tr>
</tbody>
</table>

Relative Energy Cost

Memory access is orders of magnitude higher energy than compute.

[Horowitz, ISSCC 2014]
Autonomous Navigation Uses a Lot of Data

**Semantic Understanding**

- High frame rate
- Large resolutions
- Data expansion

**Geometric Understanding**

- Growing map size

---

2 million pixels

10x-100x more pixels

[Pire, RAS 2017]
Visual-Inertial Localization

Determines location/orientation of robot from images and IMU (also used by headset in Augmented Reality and Virtual Reality)

*Subset of SLAM algorithm (Simultaneous Localization And Mapping)
Localization at Under 25 mW

*First chip* that performs **complete** Visual-Inertial Odometry

**Front-End for camera**
*(Feature detection, tracking, and outlier elimination)*

**Front-End for IMU**
*(pre-integration of accelerometer and gyroscope data)*

**Back-End Optimization of Pose Graph**

Consumes **684×** and **1582×** less energy than mobile and desktop CPUs, respectively

[Joint work with Sertac Karaman (AeroAstro)]

[Zhang, RSS 2017], [Suleiman, VLSI-C 2018]
Key Methods to Reduce Data Size

**Navion**: Fully integrated system – no off-chip processing or storage

Use *compression* and *exploit sparsity* to reduce memory down to 854kB

Navion Project Website: [http://navion.mit.edu](http://navion.mit.edu)

[Suleiman, VLSI-C 2018] Best Student Paper Award
Understanding the Environment

Depth Estimation

State-of-the-art approaches use Deep Neural Networks, which require up to several hundred millions of operations and weights to compute! >100x more complex than video compression
Deep Neural Networks (DNNs) have become a cornerstone of AI.
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

Free download for institutional subscribers
Properties We Can Leverage

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

• Memory Access is the Bottleneck

![Diagram](image)

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

• Example: AlexNet 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**)

![Diagram](Convolutional Reuse (Activations, Weights)
CONV layers only (sliding window)

![Diagram](Fmap Reuse (Activations)
CONV and FC layers

![Diagram](Filter Reuse (Weights)
CONV and FC layers (batch size > 1)
Exploit Data Reuse at Low-Cost Memories

- DRAM
- Global Buffer
- PE
- PE
- PE
- ALU
- Reg File
- Control

Normalized Energy Cost*

<table>
<thead>
<tr>
<th></th>
<th>1x (Reference)</th>
<th>1x</th>
<th>2x</th>
<th>6x</th>
<th>200x</th>
</tr>
</thead>
<tbody>
<tr>
<td>DRAM</td>
<td>ALU</td>
<td>ALU</td>
<td>ALU</td>
<td>ALU</td>
<td>ALU</td>
</tr>
<tr>
<td>0.5 – 1.0 kB RF</td>
<td>ALU</td>
<td>ALU</td>
<td>ALU</td>
<td>ALU</td>
<td>ALU</td>
</tr>
<tr>
<td>100 – 500 kB Buffer</td>
<td>ALU</td>
<td>ALU</td>
<td>ALU</td>
<td>ALU</td>
<td>ALU</td>
</tr>
<tr>
<td>NoC: 200 – 1000 PEs PE</td>
<td>ALU</td>
<td>ALU</td>
<td>ALU</td>
<td>ALU</td>
<td>ALU</td>
</tr>
</tbody>
</table>

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

Farther and larger memories consume more power.

* measured from a commercial 65nm process
Energy-Efficient Dataflow for Deep Neural Networks

Eyeriss: Row-Stationary Dataflow

Exploits data reuse for \textbf{100x} reduction in memory accesses from global buffer and \textbf{1400x} reduction in memory accesses from off-chip DRAM

Overall \textbf{>10x energy reduction} compared to a mobile GPU (Nvidia TK1)

Eyeriss Project Website: \url{http://eyeriss.mit.edu}

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

Exponential

Energy/ Pixel (nJ)

Accuracy (Average Precision)

Measured in 65nm*

1. DPM v5 [Girshick, 2012]

Measured in on VOC 2007 Dataset

* Only feature extraction. Does not include data, classification energy, augmentation and ensemble, etc.
Energy-Efficient Processing of DNNs

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

Hardware Architectures for Deep Neural Networks

ISCA Tutorial
June 24, 2017

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

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

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

We identified various limitations to existing approaches
Design of Efficient DNN Algorithms

Popular efficient DNN algorithm approaches

Network Pruning

Efficient 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?

[Chen*, Yang*, SysML 2018]
Number of MACs and Weights are Not Good Proxies

# of operations (MACs) does not approximate latency well

Similar latency, 3x range in # MACs

Similar # MACs, 2x range in latency

Source: Google

# of weights alone is not a good metric for energy
(All data types should be considered)

Energy breakdown of GoogLeNet

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 w/ similar accuracy
- Outperforms magnitude-based pruning by 1.7x

[Yang, CVPR 2017]

Pruned models available at http://eyeriss.mit.edu/energy.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)
- **Few hyperparameters** to reduce tuning effort
- **>1.7x speed up** on MobileNet w/ similar accuracy

*In collaboration with Google’s Mobile Vision Team*

[Vivienne Sze](http://sze.mit.edu)  
[@eems_mit](https://twitter.com/eems_mit)

[Yang, ECCV 2018]

Code available at  
[http://netadapt.mit.edu](http://netadapt.mit.edu)
FastDepth: Fast Monocular Depth Estimation

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

![Graph showing accuracy versus frames per second](image)

- **This Work**
- Eigen'14
- Eigen'15 (AlexNet)
- Eigen'15 (VGG)
- Laina'16 (UpConv)
- Laina'16 (UpProj)
- Xian'18

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

~40fps on an iPhone

Models available at [http://fastdepth.mit.edu](http://fastdepth.mit.edu)

[Joint work with Sertac Karaman]
NetAdapt v2: Reduce Adaption Time

Reduce time to find efficient DNN that adapts to hardware by up to 5.8x

Typical Steps in Neural Architecture Search (NAS):
1) Train super-network (search space of DNNs)
2) Sample and evaluate different DNNs
3) Fine tune the final DNN

Contributions
• Ordered dropout: train multiple DNNs in single forward pass (reduce step 1)
• Channel-level bypass: merge layer depth and channel width into a single search dimension (reduce step 2)
• Multi-layer coordinate descent optimizer: consider joint effect of multiple layers (reduce step 2 & support non-differentiable metrics, e.g., latency)

More info at http://netadapt.mit.edu

[Vivienne Sze http://sze.mit.edu/ @eems_mit [Yang, CVPR 2021]]
Many Efficient DNN Design Approaches

Network Pruning

before pruning

pruning synapses

pruning neurons

after pruning

Compact Network Architectures

Convolutional Layer

Depth-Wise Layer

Point-Wise Layer

Reduce Precision

32-bit float

8-bit fixed

Binary

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

[Chen*, Yang*, SysML 2018]

Vivienne Sze http://sze.mit.edu/ @eems.mit
Need Flexible Dataflow & Mapping

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

Example: Depth-wise layer

[Chen, JETCAS 2019]
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

[Chen, JETCAS 2019]
Hierarchical Mesh

GLB Cluster
Mesh Network
Router Cluster
PE Cluster
All-to-All

High Bandwidth
High Reuse
Grouped Multicast
Interleaved Multicast

[Chen, JETCAS 2019]
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

<table>
<thead>
<tr>
<th># of PEs</th>
<th>256</th>
<th>1024</th>
<th>16384</th>
</tr>
</thead>
<tbody>
<tr>
<td>AlexNet</td>
<td>17.9x</td>
<td>71.5x</td>
<td>1086.7x</td>
</tr>
<tr>
<td>GoogLeNet</td>
<td>10.4x</td>
<td>37.8x</td>
<td>448.8x</td>
</tr>
<tr>
<td>MobileNet</td>
<td>15.7x</td>
<td>57.9x</td>
<td>873.0x</td>
</tr>
</tbody>
</table>

[Joint work with Joel Emer]

[Chen, JETCAS 2019]
DNN Accelerator Evaluation Tools

• Require systematic way to
  – Evaluate and compare DNN accelerators
  – Rapidly explore design space

• Accelergy [Wu, ICCAD 2019]
  – Early-stage estimation tool at the architecture level
    • Estimate energy based on architecture level components (e.g., # of PEs, memory size, on-chip network)
  – Evaluate architecture level impact of emerging devices
    • Plug-ins for different technologies

• Timeloop [Parashar, ISPASS 2019]
  – DNN mapping tool
  – Performance Simulator → Action counts

• Bridge architecture, circuit, and device research

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

[Joint work with Joel Emer]
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

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

*Total energy might not add up to exact 100.0% due to rounding*
Plug-ins for Fine-Grain Action Energy Estimation

- External energy/area models that accurately reflect the properties of a macro
  - e.g., multiplier with zero-gating

Energy characterizations of the zero-gated multiplier (normalized to idle)

With the characterization provided in the plug-in, we can capture the energy savings for sparse workloads

[Wu, ICCAD 2019]
In-Memory Computing (IMC*)

- Reduce data movement by **moving compute into memory**
- Compute MAC with memory storage element
- **Analog Compute**
  - Activations, weights and/or partial sums are encoded with analog voltage, current, or resistance
  - Increased sensitivity to circuit non-idealities
  - A/D and D/A circuits to interface with digital domain
- Leverage **emerging memory device technology**

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

\[ V_1 \rightarrow G_1 \rightarrow I_1 = V_1 \times G_1 \]

\[ V_2 \rightarrow G_2 \rightarrow I_2 = V_2 \times G_2 \]

Psum is output current
\[ I = I_1 + I_2 = V_1 \times G_1 + V_2 \times G_2 \]

* a.k.a. Processing In Memory (PIM)

Image Source: [Shafiee, ISCA 2016]
Accelergy for IMC

Energy breakdown across layers

<table>
<thead>
<tr>
<th>Layer</th>
<th>Energy Consumption (µJ)</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>0.01%</td>
</tr>
<tr>
<td>2</td>
<td>12.6%</td>
</tr>
<tr>
<td>3</td>
<td>11.4%</td>
</tr>
<tr>
<td>4</td>
<td>17.4%</td>
</tr>
<tr>
<td>5</td>
<td>3.1%</td>
</tr>
<tr>
<td>6</td>
<td>N/A</td>
</tr>
<tr>
<td>7</td>
<td>17.7%</td>
</tr>
<tr>
<td>8</td>
<td>66.9%</td>
</tr>
</tbody>
</table>

Achieves ~95% accuracy

This Work

0.037J

66.9%

12.6%

17.4%

3.1%

0.01%

Cascade [MICRO 2019]

0.035J

67.9%

11.4%

17.4%

3.0%

N/A

Open-source code available at: [http://accelergy.mit.edu](http://accelergy.mit.edu)

Vivienne Sze [http://sze.mit.edu](http://sze.mit.edu) @eems_mit

[Wu, ISPASS 2020]
Accelergy + Timeloop Tutorial

Tutorial material available at [http://accelergy.mit.edu/tutorial.html](http://accelergy.mit.edu/tutorial.html)

*Includes videos and hands-on exercises*

---

**Accelergy + Timeloop Tutorial**

**ISCA Tutorial**

Angshuman Parashar  
Yannan Nellie Wu  
Po-An Tsai  
Vivienne Sze  
Joel S. Emer  

NVIDIA  
MIT  
NVIDIA  
MIT  

NVIDIA, MIT  

**May 2020**

---

**Accelergy + Timeloop Tutorial**

**ISCA Tutorial**

Angshuman Parashar  
Yannan Nellie Wu  
Po-An Tsai  
Vivienne Sze  
Joel S. Emer  

NVIDIA  
MIT  
NVIDIA  
MIT  

NVIDIA, MIT  

**May 2020**
Designing DNNs for IMC

• Designing DNNs for IMC may differ from DNNs for digital processors

• Highest accuracy DNN on digital processor may be different on IMC
  – Accuracy drops based on robustness to non-idealities

• Reducing number of weights is less desirable
  – Since IMC is weight stationary, may be better to reduce number of activations
  – IMC tend to have larger arrays → fewer weights may lead to low utilization on IMC

[Yang, IEDM 2019]
Many Design Considerations for In-Memory Computing

- Number of Storage Elements per Weight
- Array Size
- Number of Rows Activated in Parallel
- Number of Columns Activated in Parallel
- Time to Deliver Input
- Time to Compute MAC

Tradeoffs between energy efficiency, throughput, area density, and accuracy, which reduce the achievable gains over conventional architectures

Available on DNN tutorial website
http://eyeriss.mit.edu/tutorial.html
Applications that use Sparse Tensor

Density:

0 % 10 % 50 % 100 %

Density: (log scale)

>10^-6 % 10^-5 % 10^-4 % 10^-3 % .01 % .1 % 1 %

Sparse neural networks

Finite Element Methods

Dense linear algebra

Dense neural networks

Recommendation systems

Computational Chemistry

Fluid Dynamics

Problems in statistics

Internet & Social media

Circuit Simulation

Electromagnetics

Proteins

[Hedge, MICRO 2019]
Sparseloop: Design Space Exploration for Sparse Tensor Accelerators

- An analytical design exploration framework that comprehends a wide range of
  - Sparse optimizations (e.g., zero-gating, zero-skipping, zero-compression)
  - Data representations (e.g., uncompressed, run length coding, bitmask)

Propose modularized three-step evaluation process

Energy impact of sparse optimizations at different levels of
the memory hierarchy in Eyeriss-based topology

CHAPTER 8

Exploiting Sparsity

A salient characteristic of the data used in DNN computations is that it is (or can be made to be) sparse. By saying that the data is sparse, we are referring to the fact that there are many repeated values in the data. Much of the time the repeated value is zero, which is what we will assume unless explicitly noted. Thus, we will talk about the sparsity or density of the data as the percentage of zeros or non-zeros, respectively in the data. The existence of sparse data leads broadly to two potential architectural benefits: (1) sparsity can reduce the footprint of the data, which provides an opportunity to reduce storage requirements and data movement. This is because sparse data is amenable to being compressed, as described in Section 8.2.1 and (2) sparsity presents an opportunity for a reduction in MAC operations. The reduction in MAC operations results from the fact that 0 × anything is 0. This can result in either savings in energy or time or both. In Section 8.3, we will discuss how the dataflows for sparse data can translate sparsity into improvements in energy-efficiency and throughput. However, first in Section 8.1 we discuss the origins and ways that one can increase sparsity in the data used in DNN computations.

8.1 SOURCES OF SPARSITY

Efficient processing of feature map activations becomes increasingly important as the size of the input to the DNN model grows (e.g., increased image resolution), while efficient processing of filter weights becomes increasingly important as the size of the DNN model grows (e.g., increased number of layers).

This section will discuss various approaches that can exploit properties such as redundancy and correlation in the feature maps and filters to increase their activation sparsity (Section 8.1.1) and weight sparsity (Section 8.1.2), respectively. The requirements for these approaches may differ as activation sparsity is often data dependent and not known a priori, while weight sparsity can be known a priori. As a result, methods to increase sparsity for weights can be performed off-line (as opposed to during inference) and can be more computationally complex than methods applied to increase activation sparsity. For instance, increasing weight sparsity can be incorporated into training.

https://tinyurl.com/EfficientDNNBook
Where to Go Next: Planning and Mapping

Robot Exploration
Where to Go Next: Planning and Mapping

**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  
Updated Map

[Joint work with Sertac Karaman]
Experimental Results (4x Real Time)

Occupancy map with planned path using RRT* (compute MI on all possible paths)

Exploration with a mini race car using motion capture for localization

MI surface

[Zhang, ICRA 2019]
Building Hardware Accelerator to Compute MI

Motivation: Compute MI faster for faster exploration!

\[ I(M; Z) = \sum_{j=1}^{n} \sum_{k=j-\Delta}^{j+\Delta} P(e_j)C_k G_{k,j} \]

Fast Shannon Mutual Information (FSMI)
[Zhang, ICRA 2019]

Algorithm is *embarrassingly* parallel!
High throughput *should* be possible with multiple processing elements (PE)

Process sensor beams in parallel with multiple PEs
Challenge is Data Delivery to All PEs

Power consumption of memory scales with number of ports. **Low power SRAM limited to two-ports!**

Data delivery, specifically memory bandwidth, limits the throughput (not compute)
Optimized Memory Banking Pattern

Memory Access Pattern

PEs read the map at the **same row or column every cycle**

Diagonal Banking Pattern

Reduced conflicts across banks

[Li, RSS 2019]
Experimental Results

Specialized banking, efficient memory arbiter and packing multiple values at each address results in throughput within 94% of theoretical limit (unlimited bandwidth).

Compute MI for an entire map of 20m x 20m at 0.1m resolution in under a second on a ZC706 FPGA (100x faster than CPU at 10x lower power).

<table>
<thead>
<tr>
<th>Metric</th>
<th>Throughput (MI/s)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Baseline (1 bank)</td>
<td>2.5 × 10^4</td>
</tr>
<tr>
<td>16 banks, vertical banking, 1x1 packing</td>
<td>3.1 × 10^4</td>
</tr>
<tr>
<td>16 banks, diagonal banking, 1x1 packing</td>
<td>3.5 × 10^4</td>
</tr>
<tr>
<td>16 banks, diagonal banking, 2x2 packing</td>
<td>3.8 × 10^4</td>
</tr>
<tr>
<td>Unlimited bandwidth</td>
<td>4 × 10^4</td>
</tr>
</tbody>
</table>

Computational MI for an entire map of 20m x 20m at 0.1m resolution in under a second on a ZC706 FPGA (100x faster than CPU at 10x lower power).

[Li, RSS 2019]
Generalize to a Class of Banking Patterns

- **Latin-square banking tile**: cells in each column and row is assigned to different banks

We **rigorously proved** that Latin-square tiles usage minimizes read conflicts between PEs.
Low Power 3D Time of Flight Imaging

- Pulsed Time of Flight: Measure distance using round trip time of laser light for each image pixel
  - Illumination + Imager Power: 2.5 – 20 W for range from 1 - 8 m

- Use computer vision techniques and passive images to estimate changes in depth without turning on laser
  - CMOS Imaging Sensor Power: < 350 mW

...
Results of Low Power Depth ToF Imaging

Mean Relative Error: 0.7%
Duty Cycle (on-time of laser): 11%

[Noraky, TCSVT 2020]
Efficient computing is critical for advancing the progress of AI & autonomous robots → **Critical step to making AI & autonomy ubiquitous!**

In order to meet computing demands in terms of power and speed, need to redesign computing hardware from the ground up → **Focus on data movement!**

Specialized hardware creates new opportunities for the co-design of algorithms and hardware → **Innovation opportunities for the future of AI & robotics!**
Acknowledgements

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

Joel Emer

Sertac Karaman
Low-Energy Autonomy and Navigation (LEAN) Group

A broad range of next-generation applications will be enabled by low-energy, miniature mobile robotics including insect-size flapping wing robots that can help with search and rescue, chip-size satellites that can explore nearby stars, and blimps that can stay in the air for years to provide communication services in remote locations. While the low-energy, miniature actuation, and sensing systems have already been developed in many of these cases, the processors currently used to run the algorithms for autonomous navigation are still energy-hungry. Our research addresses this challenge as well as brings together the robotics and hardware design communities.

We enable efficient computing on various key modules of other autonomous navigation systems including perception, localization, exploration and planning. We also consider the overall system by considering the energy cost of computing in conjunction with actuation and sensing.

**Motion Planning**

Many motion planning and control algorithms aim to design trajectories and controllers that minimize actuation energy. However, in low-energy robotics, computing such trajectories and controls themselves may consume a large amount of energy. We develop algorithms that optimize this trade-off.

**Mutual Information for Exploration**

Computing mutual information between the map and future measurements is critical to efficient exploration. Unfortunately, mutual information computation is computationally very challenging. We develop new algorithms and hardware for efficient computation of mutual information, and demonstrate real-time computation for the whole map in a reasonably-sized map.

**Depth Sensing and Perception**

Depth sensing is a critical function for robotic tasks such as localization, mapping and obstacle detection. State-of-the-art single-view depth estimation algorithms are based on fairly complex deep neural networks that are too slow for real-time inference on an embedded platform, for instance, mounted on a micro aerial vehicle. We address the problem of fast depth estimation on embedded systems.

**Localization and Mapping**

 Autonomous navigation of miniaturized robots (e.g., nano/pico aerial vehicles) is currently a grand challenge for robotics research, due to the need for processing a large amount of sensor data (e.g., camera frames) with limited on-board computational resources. We focus on the design of a visual-inertial odometry (VIO) system in which the robot estimates its ego-motion (and a landmark-based map) from on-board camera and IMU data.

Group Website: [http://lean.mit.edu](http://lean.mit.edu)
Resources on Efficient Processing of DNNs

http://eyeriss.mit.edu/tutorial.html
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 evaluating the energy and area efficiency of DNNs. These metrics include the number of operations per second, the energy consumption, the power consumption, and the area efficiency. Reporting a comprehensive set of metrics is important to allow for a complete picture of the trade-offs in DNN designs.

In this chapter, we will:

- Discuss the importance of the metrics:
  - Address the challenges associated with the metrics and their trade-offs.
  - Discuss how these metrics can be incorporated into design considerations.

- Present several approaches for improving the metrics:
  - Processors using data executing a workload.
  - Processors using data executing a workload.

- Provide examples of processors using data executing a workload.

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

3.1 ACCURACY

Accuracy is required to achieve good results on a wide range of tasks. For instance, for image classification, accuracy is reported as the percent of correctly classified images, while for object detection, accuracy is measured as the mean average precision (mAP), which is related to the trade-off between the true positive rate and the 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 core system. This is commonly referred to by computer architects as the "memory wall." To address the challenges associated with data movement, there have been various efforts to bring compute and memory closer together.

Chapters 5 and 6 focus primarily on how to design spatial architectures that distribute the on-chip memory close to the computation (e.g., scratchpad memory in the PE). This chapter will describe several other architectures that use advanced memory, power, and fabrication technologies to bring the compute and memory together.

First, we will discuss efforts to bring the off-chip high-density memory (e.g., DRAM) closer to the computation. These approaches are often referred to as processing-in-memory on near-data processing, and include memory technologies such as embedded DRAM and 3-D stacked DRAM.

Next, we will discuss efforts to integrate the computation into the memory itself. These approaches are often referred to as processing-in-memory on in-memory computing, and include memory technologies such as Static Random Access Memories (SRAMs), Dynamic Random Access Memories (DRAMs), and emerging non-volatile memory (NVMs). 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-ideality (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 are analogous to these 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, as proven in Chapter 4, this approach is superior to the current state-of-the-art accuracy on a wide range of tasks. For instance, for image classification, accuracy is reported as the percent of correctly classified images, while for object detection, accuracy is measured as the mean average precision (mAP), which is related to the trade-off between the true positive rate and the false positive rate.

Available on DNN tutorial website

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

Vivienne Sze: http://sze.mit.edu/  Email: eems@mit.edu
Additional Resources

Talks and Tutorial Available Online

https://tinyurl.com/ISCAS2021Sze
References

• Efficient Processing for Deep Neural Networks
  - Project website: http://eyeriss.mit.edu
  - Hardware Architecture for Deep Neural Networks: http://eyeriss.mit.edu/tutorial.html
References

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

• Low Power Time of Flight Imaging
References

- **Energy-Efficient Visual Inertial Localization**
  - Project website: [http://navion.mit.edu](http://navion.mit.edu)
• **Fast Shannon Mutual Information for Robot Exploration**
  
  
  
  
  

• **Balancing Actuation and Computation**
  
  