Research Overview of Energy-Efficient Multimedia Systems Group

Vivienne Sze
Efficient Computing with Cross-Layer Design

**Algorithms**
- Convolutions
- Pooling
- Convs
- Linear Classifier
- Object Categories / Positions

**Architectures**
- Link Clock: Core Clock
- DCNN Accelerator
- 14x12 PE Array
- Filter
- Input Image
- Output Image
- Buffer SRAM
- 108KB
- Psum
- Off-Chip DRAM
- 64 bits

**Systems**

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

---

Vivienne Sze [http://sze.mit.edu/](http://sze.mit.edu/) @eems_mit
Energy-Efficient Deep Neural Networks

Efficient and Flexible Hardware

Eyeriss Accelerator
minimize data movement

[ISSCC 2016, ISCA 2016]

http://eyeriss.mit.edu

Co-Design of Algorithms and Hardware

Energy Modeling for Design Exploration and Optimization

NetAdapt

Platform

[CVPR 2017, ECCV 2018, CVPR 2021]

Vivienne Sze
http://sze.mit.edu/ @eems_mit

[Asilomar 2017, ICCAD 2019, ISPASS 2021]
Design of Efficient DNN Algorithms

Popular efficient DNN algorithm approaches

Network Pruning

Efficient Network Architectures

Example: 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

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]
Design Hardware-Aware DNN Algorithms

Directly target energy and latency and incorporate it into the optimization of DNNs to provide better performance tradeoffs

**Energy-Aware Pruning**
Remove weights based on energy consumption

Normalised Energy (AlexNet)

<table>
<thead>
<tr>
<th></th>
<th>Ori.</th>
<th>Magnitude Based Pruning</th>
<th>Energy Aware Pruning</th>
</tr>
</thead>
<tbody>
<tr>
<td>Energy</td>
<td>2.1x</td>
<td>3.7x</td>
<td></td>
</tr>
</tbody>
</table>

*[Yang, CVPR 2017]*

**NetAdapt: Platform-Aware DNN Adaptation**
Automatically adapt DNN to reach target latency/energy

NetAdapt: Platform-Aware DNN Adaptation

<table>
<thead>
<tr>
<th>Pretrained Network</th>
<th>Budget</th>
<th>Empirical Measurements</th>
</tr>
</thead>
<tbody>
<tr>
<td>Metric</td>
<td>Budget</td>
<td>Proposal A</td>
</tr>
<tr>
<td>Latency</td>
<td>3.8</td>
<td>15.6</td>
</tr>
<tr>
<td>Energy</td>
<td>10.5</td>
<td>41</td>
</tr>
</tbody>
</table>

[Yang, ECCV 2018]

Code available at [http://netadapt.mit.edu](http://netadapt.mit.edu)
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](http://netadapt.mit.edu)

[Yang, CVPR 2021]
Energy Estimation for Accelerator Designs

• **Accelergy** is an architecture-level energy estimator framework
  – Early-stage energy estimation
  – Rapid design space exploration
    • e.g., evaluate and compare different deep learning accelerator designs → performance modeling with Timeloop

• Provides flexibility to
  – Describe a diverse range of accelerator designs
  – Support different technologies
    • e.g., CMOS, RRAM, optical

• Validated on both digital and PIM based accelerators (95% accuracy)

• Bridge architecture, circuit and devices research

---

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

[Wu, *ICCAD 2019*],
[Wu, *ISPASS 2020*]

---

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

In collaboration with Joel Emer
Applications that use Sparse Tensor

Density:

0 % 10 % 50 % 100 %

Density:

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

Finite Element Methods

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


In collaboration with Joel Emer
Efficient Computing for Autonomous Navigation

Monocular Depth Estimation with FastDepth

RGB Prediction

http://fastdepth.mit.edu

~40fps on an iPhone

Visual Inertial Localization with Navion

http://navion.mit.edu

Robot Exploration with Mutual Information

Memory Access Pattern

Diagonal Banking Pattern

In collaboration with 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 actuators, 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 miniature robots (e.g., nanorobot aerial vehicles) is currently a grand challenge for robotics research, due to the need for processing a large amount of sensor data (e.g., cameras 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
References

• Efficient Hardware for Deep Neural Networks and Sparse Tensor Accelerators
  – Project website: http://eyeriss.mit.edu
• Co-Design of Algorithms and Hardware for Deep Neural Networks
References

• Efficient Computing for Autonomous Navigation