Understanding the Limitations of Existing Energy-Efficient Design Approaches for Deep Neural Networks

Vivienne Sze
Massachusetts Institute of Technology

In collaboration with Yu-Hsin Chen, Joel Emer, Tien-Ju Yang

Based on SysML 2018 paper with the same title: Link

Contact Info
email: sze@mit.edu
website: www.rle.mit.edu/eems

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

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

- Popular efficient DNN algorithm approaches

**Network Pruning**

- Before pruning
- After pruning

**Compact Network Architectures**

Examples: SqueezeNet, MobileNet

... also reduced precision

- Focus on reducing **number of MACs and weights**
- Does it translate to energy savings?
Data Movement is Expensive

Energy of weight depends on memory hierarchy and dataflow

Normalized Energy Cost*

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

* measured from a commercial 65nm process
Energy-Evaluation Methodology

DNN Shape Configuration
(# of channels, # of filters, etc.)

DNN Weights and Input Data
[0.3, 0, -0.4, 0.7, 0, 0, 0.1, ...]

Hardware Energy Costs of each MAC and Memory Access

Memory Accesses Optimization

# of MACs Calculation

# acc. at mem. level 1
# acc. at mem. level 2
...#
# acc. at mem. level n

# of MACs

E\text{comp}

E_{data}

Energy

L1 L2 L3 ...

DNN Energy Consumption

Energy estimation tool available at [http://eyeriss.mit.edu](http://eyeriss.mit.edu)
Example: AlexNet vs. SqueezeNet

### # of Weights

- **AlexNet** vs. **SqueezeNet**

### Normalized Energy

- **AlexNet** vs. **SqueezeNet**
  - 1.3x increase

### Energy Breakdown (SqueezeNet)

- **Output Feature Map**: 47%
- **Weights**: 21%
- **Input Feature Map**: 23%
- **Computation**: 10%

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

**Notes:** ImageNet, Batch = 10/64, using active cooling

[Movidius, Hot Chips 2016]
Energy Consumption of Existing DNNs

Deeper DNNs with fewer weights do not necessarily consume less energy than shallower DNNs with more weights

[Yang et al., CVPR 2017]
Reduce number of weights by **removing small magnitude weights**
Energy-Aware Pruning

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

[Yang et al., CVPR 2017]
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)

![Diagram showing the process of NetAdapt with pre-trained networks, adapted networks, and empirical measurements]

- **Pretrained Network**
- **Adapted Network**
- **Budget**
  - **Metric**
    - Latency: 3.8
    - Energy: 10.5
- **Empirical Measurements**
  - **Metric**
    - Latency: 15.6...14.3
    - Energy: 41...46
- **Platform**

[Yang et al., arXiv 2018]

In collaboration with Google’s Mobile Vision Team
Improved Latency vs. Accuracy Tradeoff

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

Reference:


*Tested on the ImageNet dataset and a Google Pixel 1 CPU*
Many Efficient DNN Design Approaches

Network Pruning

before pruning

after pruning

pruning synapses

pruning neurons

Compact Network Architectures

Reduce Precision

32-bit float

8-bit fixed

Binary

No guarantee that DNN algorithm designer will use a given approach. Need flexible hardware!
Eyexam: Understanding Sources of Inefficiencies in DNN Accelerators

A systematic way to evaluate how each architectural decision affects performance (throughput) for a given DNN workload

Tightens the roofline model

Step 1: maximum workload parallelism

Step 2: maximum dataflow parallelism

Number of PEs (Theoretical Peak Performance)

Step 3: # of act. PEs under a finite PE array size

Step 4: # of act. PEs under fixed PE array dims.

Step 5: # of act. PEs under fixed storage cap.

Step 6: lower act. PE utilization due to insuff. avg. BW

Step 7: lower act. PE utilization due to insuff. inst. BW

[Chen et al., In Submission]
Existing DNN Architectures

- Specialized DNN hardware often rely on certain properties of DNN in order to achieve high energy-efficiency

- **Example:** Reduce memory access by amortizing across MAC array
Limitation of Existing DNN Architectures

- **Example:** reuse depends on # of channels, feature map/batch size
  - Not efficient across all network architectures (e.g., compact DNNs)
  - Can be challenging to exploit sparsity
Existing Sparse DNN Architectures

• Sparse DNN architectures translate sparsity from pruning into improved energy-efficiency and throughput
  – Perform only non-zero MACs and move data in compressed format

• Existing sparse DNN architectures optimized for either CONV or FC layer due to different BW and data reuse requirements

• Efficient for sparse DNNs, but **overhead for dense DNNs**
  – Compressed format results in **memory overhead** for dense DNNs
  – Additional control to identify location of non-zero values results in **energy overhead** for dense DNNs

Since there is **no guarantee in degree of sparsity**, it is important to **evaluate the overhead on dense DNNs**
To efficiently support:

- Wide range of filter shapes
  - Large and Compact
- Different Layers
  - e.g., CONV and FC
- Wide range of sparsity
  - Dense and Sparse

Eyeriss (v1)
[Chen et al. ISSCC 2016, ISCA 2016]
http://eyeriss.mit.edu
Need More Comprehensive Benchmarks

Processors should support a **diverse set of DNNs** that utilize different techniques

**Example:**

- Sparse **and** Dense
- Large **and** Compact network architectures
- Different Layers (e.g., CONV **and** FC)
- Variable Bit-width

**Reduce Precision**

- 32-bit float: 101001010100000000000000010100000000001000
- 8-bit fixed: 01100110
- Binary: 0

**Network Pruning**

**Compact Network Architecture**
Eyeriss v2: Balancing Flexibility and Efficiency

• Flexible dataflow for high PE array utilization and data reuse for various layer shapes and sizes

• Flexible NoC that can operate in different modes for different requirements
  – Utilizes multicast to exploit spatial data reuse
  – Utilizes unicast for high BW for weights for FC and weights & activations for compact network architectures

• Processes data in both compressed and raw format to minimize data movement for both CONV and FC layers
  – Exploit sparsity in weights and activations
Benchmarking Metrics for DNN Hardware

How can we compare designs?

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
Metrics for DNN Hardware

• **Accuracy**
  – Quality of result for a given task

• **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**
  – Edge and embedded devices have limited battery capacity
  – Data centers have stringent power ceilings due to cooling costs

• **Hardware Cost**
  – $$$
Specifications to Evaluate Metrics

• **Accuracy**
  – Difficulty of dataset and/or task should be considered

• **Throughput**
  – Number of cores (include utilization along with peak performance)
  – Runtime for running specific DNN models

• **Latency**
  – Include batch size used in evaluation

• **Energy and Power**
  – Power consumption for running specific DNN models
  – Include external memory access

• **Hardware Cost**
  – On-chip storage, number of cores, chip area + process technology
## Example: Metrics of Eyeriss Chip

<table>
<thead>
<tr>
<th>ASIC Specs</th>
<th>Input</th>
</tr>
</thead>
<tbody>
<tr>
<td>Process Technology</td>
<td>65nm LP TSMC (1.0V)</td>
</tr>
<tr>
<td>Total Core Area (mm²)</td>
<td>12.25</td>
</tr>
<tr>
<td>Total On-Chip Memory (kB)</td>
<td>192</td>
</tr>
<tr>
<td>Number of Multipliers</td>
<td>168</td>
</tr>
<tr>
<td>Clock Frequency (MHz)</td>
<td>200</td>
</tr>
<tr>
<td>Core area (mm²) / multiplier</td>
<td>0.073</td>
</tr>
<tr>
<td>On-Chip memory (kB) / multiplier</td>
<td>1.14</td>
</tr>
<tr>
<td>Measured or Simulated</td>
<td>Measured</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Metric</th>
<th>Units</th>
<th>Input</th>
</tr>
</thead>
<tbody>
<tr>
<td>Name of CNN Model</td>
<td>Text</td>
<td>AlexNet</td>
</tr>
<tr>
<td>Top-5 error classification on ImageNet</td>
<td>#</td>
<td>19.8</td>
</tr>
<tr>
<td>Supported Layers</td>
<td>All CONV</td>
<td></td>
</tr>
<tr>
<td>Bits per weight</td>
<td>#</td>
<td>16</td>
</tr>
<tr>
<td>Bits per input activation</td>
<td>#</td>
<td>16</td>
</tr>
<tr>
<td>Batch Size</td>
<td>#</td>
<td>4</td>
</tr>
<tr>
<td>Runtime</td>
<td>ms</td>
<td>115.3</td>
</tr>
<tr>
<td>Power</td>
<td>mW</td>
<td>278</td>
</tr>
<tr>
<td>Off-chip Access per Image Inference</td>
<td>MBytes</td>
<td>3.85</td>
</tr>
<tr>
<td>Number of Images Tested</td>
<td>#</td>
<td>100</td>
</tr>
</tbody>
</table>

**Measured or Simulated**

**Mit**

[ML Research Laboratory at MIT](https://rle.mit.edu)

[Massachusetts Institute of Technology](https://mit.edu)
• **All metrics** should be reported for fair evaluation of design tradeoffs

• Examples of what can happen if 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 bandwidth*, one could build a processor with only multipliers 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?
The evaluation process for whether a DNN system 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
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 that into their design

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

• DNN hardware should be evaluated on a comprehensive set of benchmarks and metrics

For updates on Eyerissv2, Eyexam, NetAdapt, etc. or join EEMS news mailing list

Project Website: http://eyeriss.mit.edu
Acknowledgements

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