Accelerators (I)

Joel Emer

Massachusetts Institute of Technology
Electrical Engineering & Computer Science

“Compute has been the oxygen of deep learning”
– Ilya Sutskever (Open AI)
GPU Usage for ImageNet Challenge

Top 5 Error Rate

2010: 28%
2011: 26%
2012: 16%
2013: 12%
2014: 7%

# of entries using GPUs

2010: 60
2011: 110
CPU vs. GPU Performance

Ratio of (partially-optimized) CPU vs. CUDA library (cuDNN)

Source: Stanford CS231n
Opportunities

From EE Times – September 27, 2016

“Today the job of training machine learning models is limited by compute, if we had faster processors we’d run bigger models...in practice we train on a reasonable subset of data that can finish in a matter of months. We could use improvements of several orders of magnitude – 100x or greater.”

– Greg Diamos, Senior Researcher, SVAIL, Baidu

ACM’s Celebration of 50 Years of the ACM Turing Award (June 2017)

“Compute has been the oxygen of deep learning”

– Ilya Sutskever, Research Director of Open AI
Compute Demands Growing Exponentially

AlexNet to AlphaGo Zero: A 300,000x Increase in Compute

Deep and steep

Computing power used in training AI systems
Days spent calculating at one petaflop per second*, log scale

By fundamentals
- Language
- Speech
- Vision
- Games
- Other

Source: OpenAI

The Economist

## Compute Demands for Deep Neural Networks

### Common carbon footprint benchmarks

<table>
<thead>
<tr>
<th>Activity</th>
<th>CO2 Equivalent (lbs)</th>
</tr>
</thead>
<tbody>
<tr>
<td>Roundtrip flight b/w NY and SF (1 passenger)</td>
<td>1,984</td>
</tr>
<tr>
<td>Human life (avg. 1 year)</td>
<td>11,023</td>
</tr>
<tr>
<td>American life (avg. 1 year)</td>
<td>36,156</td>
</tr>
<tr>
<td>US car including fuel (avg. 1 lifetime)</td>
<td>126,000</td>
</tr>
<tr>
<td>Transformer (213M parameters) w/ neural architecture search</td>
<td>626,155</td>
</tr>
</tbody>
</table>

Chart: MIT Technology Review

[Strubell, ACL 2019]
Compute Challenges for Self-Driving Cars

Cameras and radar generate \(~6\) gigabytes of data every 30 seconds.

Prototypes use around 2,500 Watts. Generates wasted heat and some prototypes need water-cooling!
After training a speech-recognition algorithm, for example, Microsoft offers it up as an online service, and it actually starts identifying commands that people speak into their smartphones. G.P.U.s are not quite as efficient during this stage of the process. So, many companies are now building chips specifically to do what the other chips have learned.

Google built its own specialty chip, a Tensor Processing Unit, or T.P.U. Nvidia is building a similar chip. And Microsoft has reprogrammed specialized chips from Altera, which was acquired by Intel, so that it too can run neural networks more easily.
MUSK SAYS TESLA IS BUILDING ITS OWN CHIP FOR AUTOPILOT

Elon Musk disclosed plans for Tesla to design its own chip to power its self-driving function.

[Also Nvidia, Intel, Qualcomm…]
Growing Demand for HW Designers

Facebook Is Forming a Team to Design Its Own Chips

By Mark Gurman, Ian King and Sarah Frier
April 18, 2018, 3:49 PM EDT

Social network could use semiconductors for consumer devices
Move follows Apple’s chip efforts, early work by Google

High-level Synthesis Design Engineer, Consumer Hardware
Google
Mountain View, CA, US
In this role, you will use your software engineering expertise to help solve complex problems, design and optimize algorithms (for example in the domains of machine learning, ... careers.google.com
44 connections work here
5 days ago

ASIC Design Verification Engineer, Consumer Hardware
Google
Mountain View, CA, US
Experience verifying digital logic at the Register Transfer Level (RTL) using SystemVerilog for FPGAs, ASICs, and/or SoCs. Experience with image processing, computer vision, and... careers.google.com
373 company alumni work here
5 days ago

Global ASIC/SoC Sourcing Manager, Consumer Hardware
Google
Mountain View, CA, US
7 years of experience of ASIC and/or SoC sourcing Management or supply chain management experience in commercial sourcing roles with particular experience in silicon and ... careers.google.com
44 connections work here
5 days ago

HW Development Manager, FPGA and ASIC IP design – CSI / Azure – Cloud Server Infrastructure
Microsoft
Bellevue, WA, US
Microsoft is seeking a highly motivated, FPGA and ASIC IP design engineering manager to build innovative FPGA-based computing systems within a large design team. The group will ... careers.microsoft.com
13 connections work here
1 month ago

Physical Design Engineer
Microsoft
Redmond, WA, US
1–2 years of experience in ASIC physical design flows and methodologies. Job responsibilities will entail taking RTL logic through a full ASIC design flow. Worked with toolsets ... careers.microsoft.com
13 connections work here
2 weeks ago
Startups Building Custom Hardware

The New York Times

By CADE METZ JAN. 18, 2018

Big Bets On A.I. Open a New Frontier for Chips Start-Ups, Too. (January 14, 2018)

“Today, at least 45 start-ups are working on chips that can power tasks like speech and self-driving cars, and at least five of them have raised more than $100 million from investors. Venture capitalists invested more than $1.5 billion in chip start-ups last year, nearly doubling the investments made two years ago, according to the research firm CB Insights.”
Tensors

- Rank-0 - Scalar
- Rank-1 - Vector
- Rank-2 - Matrix
- Rank-3 - Cube
Input Activation/Fmap Tensor

input fmap

I[C][H][W]
DNN Model Structure

Input: Image

Low Level Features

Modified Image Source: [Lee, CACM 2011]

High Level Features

Output: “Volvo XC90”
Convolution (CONV) Layer

Filter Weights

Input fmaps (N)

Output fmaps (N)

Activations

December 7, 2022
CONV Computation

filters

input fmap

output fmap

Filter overlay

Incomplete partial sum
CONV Computation

Cycle through input fmap and weights (hold psum of output fmap)

filters

input fmap

output fmap

Filter overlay

Incomplete partial sum
CONV Computation

Cycle through input fmap and weights (hold psum of output fmap)

filters

input fmap

output fmap

filters

Filter overlay

Incomplete partial sum
CONV Computation

Cycle through input fmap and weights (hold psum of output fmap)

filters

input fmap

output fmap

Filter overlay

Incomplete partial sum
CONV Computation

Cycle through input fmap and weights (hold psum of output fmap)

filters

input fmap

output fmap

Filter overlay

Incomplete partial sum
CONV Computation

Start processing next output feature activations

filters

input fmap

output fmap

Filter overlay

Incomplete partial sum
CONV Computation

Cycle through input fmap and weights (hold psum of output fmap)

input fmap

output fmap

filters

Filter overlay

Incomplete partial sum

December 7, 2022

MIT 6.5900 Fall 2022
CONV Computation

Cycle through input fmap and weights (hold psum of output fmap)

filters

input fmap

output fmap

Filter overlay

Incomplete partial sum

December 7, 2022
CONV Computation

Cycle through input fmap and weights (hold psum of output fmap)

filters

input fmap

output fmap

filters

input fmap

output fmap

Filter overlay

Incomplete partial sum
CONV Layer Implementation

Naïve 7-layer for-loop implementation:

```python
for n in [0..N):
    for m in [0..M):
        for p in [0..P):
            for q in [0..):
                O[n][m][p][q] = B[m];
                for r in [0..R):
                    for s in [0..S):
                        for c in [0..C):
                            O[n][m][p][q] += I[n][c][Up+r][Uq+s] × F[m][c][r][s];
                O[n][m][p][q] = Activation(O[n][m][p][q]);
```
CNN Decoder Ring

- N – Number of input fmaps/output fmaps (batch size)
- C – Number of channels in input fmaps (activations) & filters (weights)
- H – Height of input fmap (activations)
- W – Width of input fmap (activations)
- R – Height of filter (weights)
- S – Width of filter (weights)
- M – Number of channels in output fmaps (activations)
- P – Height of output fmap (activations)
- Q – Width of output fmap (activations)
- U – Stride of convolution
CONV Variants

- Depthwise layer - $M == C$ and $\forall c \neq m F_{c,m,r,s} = 0$
- Pointwise layer - $R == S == 1$
- Matrix multiply - $R == S == H == 1$
- Compress (pointwise) - $M < C$ and $R == S == 1$
- Expand (pointwise) - $M > C$ and $R == S == 1$

Compress…Expand sequences are called a “bottleneck”
Architecture Metrics

- **Speed** – The rate at which the hardware finishes tasks. Limited by the number of computation units and their utilization.

- **Energy** – The total energy, e.g., in Joules, consumed to perform a task. Often constrained by battery capacity or desire to reduce carbon footprint.

- **Power** – The rate at which energy is consumed. Often limited by delivery or packaging constraints.

- **Accuracy** – The precision of the results produced. Can be dictated by bit width of compute units.

- **Flexibility** – The range of problems that can be solved, which is constrained by the limitations of the architecture.
Deep Learning Platforms

• CPU
  – Intel, ARM, AMD…

• GPU
  – NVIDIA, AMD…

• Fine Grained Reconfigurable (FPGA)
  – Xilinx, Altera (Microsoft BrainWave)

• Coarse Grained Programmable/Reconfigurable
  – Wave Computing, Graphcore, Samba Nova…

• Application Specific
  – Neuflow, *DianNao, Eyeriss, TPU, Cnvlutin, SCNN, …
What is Moore’s Law

• CPU performance will double every two years*
• Chip performance will double every two years*
• The speed of transistors will double every two years*
• Transistors will shrink to half size every two years*
• Gate width will shrink by $\sqrt{2}$ every two years*
• Transistors per die will double every two years*
• The economic sweet spot for the number of devices on a chip will double every two years*

* Or 18 months…
Technology Trends

Original data collected and plotted by M. Horowitz, F. Labonte, O. Shacham, K. Olukotun, L. Hammond and C. Batten
Dotted line extrapolations by C. Moore

Source: C Moore, Data Processing in ExaScale-Class Computer Systems, Salishan, April 2011
Energy and Power Consumption

- **Energy Consumption** $= \alpha \times C \times V^2$
  
  - Switching activity factor (between 0 to 1)
  - Capacitance
  - Voltage

- **Power Consumption** $= \alpha \times C \times V^2 \times f$
  
  - Frequency
# Dennard Scaling (idealized)

<table>
<thead>
<tr>
<th></th>
<th>Gen X</th>
<th>Gen X+1</th>
</tr>
</thead>
<tbody>
<tr>
<td>Gate Width</td>
<td>1.0</td>
<td>0.7</td>
</tr>
<tr>
<td>Device Area/Cap</td>
<td>1.0</td>
<td>0.5 0.5</td>
</tr>
<tr>
<td>Capacitance</td>
<td>1.0</td>
<td>0.7 0.7</td>
</tr>
<tr>
<td>Voltage</td>
<td>1.0</td>
<td>0.7</td>
</tr>
<tr>
<td>Energy</td>
<td>$1.0 \times 1.0^2 = 1.0$</td>
<td>$2 \times 0.7 \times 0.7^2 = 0.65$</td>
</tr>
<tr>
<td>Delay</td>
<td>1.0</td>
<td>0.7</td>
</tr>
<tr>
<td>Frequency</td>
<td>$1/1.0 = 1.0$</td>
<td>$1/0.7 = 1.4$</td>
</tr>
<tr>
<td>Power</td>
<td>$1.0 \times 1.0^2 \times 1.0 = 1.0$</td>
<td>$2 \times 0.7 \times 0.7^2 \times 1.4 = 1.0$</td>
</tr>
</tbody>
</table>

[Dennard et al., "Design of ion-implanted MOSFET's with very small physical dimensions“, JSSC 1974]
During the Moore + Dennard’s Law Era

• Instruction-level parallelism (ILP) was largely mined out by early 2000s

• Voltage (Dennard) scaling ended in 2005

• Hit the power limit wall in 2005

• Performance is coming from parallelism using more transistors since ~2007

• But....
Technology Trends

Stuttering

- Transistors per chip, '000
- Clock speed (max), MHz
- Thermal design power*, w

Chip introduction dates, selected

Transistors bought per $, m

Log scale

$10^{-1}$

$10^3$

$10^5$

$10^7$

Sources: Intel; press reports; Bob Colwell; Linley Group; IBM Consulting; The Economist

*Maximum safe power consumption
The High Cost of Data Movement

Fetching operands more expensive than computing on them

Now the key is how we use our transistors most effectively.
Data Movement is the Challenge

Memory Read | MAC* | Memory Write
--- | --- | ---
filter weight | ALU | updated partial sum
fmap activation | | *
partial sum | | multiply-and-accumulate
Data Movement is the Challenge

Memory Read | MAC* | Memory Write

DRAM | ALU | DRAM

* multiply-and-accumulate

Worst Case: all memory R/W are DRAM accesses

- Example: AlexNet [NeurIPS 2012] has 724M MACs
  \[ \rightarrow 2896M \] DRAM accesses required
Data Movement is the Challenge

Extra levels of local memory hierarchy
Data Movement is the Challenge

Extra levels of local memory hierarchy

Opportunities: 1 data reuse
Types of Data Reuse in DNN

Convolutional Reuse
CONV layers only
(sliding window)

Reuse:
Activations
Filter weights
Types of Data Reuse in DNN

Convolutional Reuse
- CONV layers only (sliding window)
- Reuse: Activations, Filter weights

Fmap Reuse
- CONV and FC layers
- Reuse: Activations
Types of Data Reuse in DNN

**Convolutional Reuse**
CONV layers only (sliding window)

Reuse: Activations, Filter weights

**Fmap Reuse**
CONV and FC layers

Reuse: Activations

**Filter Reuse**
CONV and FC layers (batch size > 1)

Reuse: Filter weights
Data Movement is the Challenge

Memory Read | MAC | Memory Write
---|---|---
DRAM | Mem | Mem | DRAM

Extra levels of local memory hierarchy

Opportunities: 1 data reuse

1 Can reduce DRAM reads of filter/fmap by up to 500×**

** AlexNet CONV layers
Data Movement is the Challenge

Opportunities: ① data reuse  ② local accumulation

① Can reduce DRAM reads of filter/fmap by up to 500×
② Partial sum accumulation does NOT have to access DRAM
Data Movement is the Challenge

Opportunities:

1. **data reuse**
   - Can reduce DRAM reads of filter/fmap by up to 500×

2. **local accumulation**
   - Partial sum accumulation does **NOT** have to access DRAM

- Example: DRAM access in AlexNet can be reduced from **2896M** to **61M** (best case)
Leverage Parallelism for Higher Performance

Memory Read | MAC | Memory Write

DRAM | Mem | DRAM

ALU

Memory Read

DRAM

ALU

Memory Write

DRAM

ALU

...
Leverage Parallelism for *Spatial* Data Reuse
1-D Convolution

Weights \( W \) \( \times \) Inputs \( W \) = Outputs \( E = W - \text{ceil}(R/2) \)

\[
\begin{align*}
\text{int } & i[W]; \quad \# \text{ Input activations} \\
\text{int } & f[S]; \quad \# \text{ Filter weights} \\
\text{int } & o[Q]; \quad \# \text{ Output activations} \\
\text{for } & q \text{ in } [0, Q): \\
& \quad \text{for } s \text{ in } [0, S): \\
& \quad \quad w = q + s \\
& \quad \quad o[q] += i[w] * f[s];
\end{align*}
\]

\( ^{\dagger} \) Assuming: ‘valid’ style convolution
1-D Convolution - Movie

Tensor: F[S]
Rank: S
0 1 2

8 5 2

Tensor: I[W]
Rank: W
0 1 2 3 4 5 6 7
1 1 2 3 3 2 7 6

Tensor: O[Q]
Rank: Q
0 1 2 3 4 5
0 0 0 0 0 0 0 0
1-D Output Stationary

Cgen

q

Calc
q+s

Next

Update
Coord Payload
Partial Sums

Latch

MAC

o'[q]
of[s]
io[w]

Coord Payload
Input Activations
Output Stationary (OS)

- Minimize partial sum R/W energy consumption
  - maximize local accumulation

- Broadcast/Multicast filter weights and reuse activations spatially across the PE array
OS Example: ShiDianNao

Top-Level Architecture

- Inputs streamed through array
- Weights broadcast
- Partial sums accumulated in PE and streamed out

PE Architecture

Weights

Activations

psums

[Du et al., ISCA 2015]
OS Example: KU Leuven

[Moons et al., VLSI 2016, ISSCC 2017]
Many Dataflows

- **Output Stationary (OS)**
  - [Peemen, *ICCD* 2013]  [ShiDianNao, *ISCA* 2015]

- **Weight Stationary (WS)**
  - [Chakradhar, *ISCA* 2010]  [nn-X (NeuFlow), *CVPRW* 2014]
  - [Park, *ISSCC* 2015]  [ISAAC, *ISCA* 2016]  [PRIME, *ISCA* 2016]
  - [TPU, *ISCA* 2017]

- **Input Stationary (IS)**
  - [Parashar (SCNN), *ISCA* 2017]

- **Row Stationary (IS)**
  - [Eyeriss, *ISCA* 2016]  [Tetris *ASPLOS* 2017]  [Eyeriss2, *JETCAT* 2019]
Spatial Architecture for DNN

Local Memory Hierarchy
- Global Buffer
- Direct inter-PE network
- PE-local memory (RF)

Processing Element (PE)
- Reg File: 0.5 – 1.0 kB
- Control
And Other Design Options

Per storage level cross product of:

- Dataflow
- Split/Shared storage
- Tiling in time
- Tiling in space
- Bypassing

Plus

- Precision/Quantization
- Scale up
- ....
Scale Up Approaches - Chiplets

(a) Simba Package  (b) Simba Chiplet  (c) Simba Processing Element
Simba Dataflow

```plaintext
//Package level
for p3 = [0 : P3):
    for q3 = [0 : Q3):
        parallel_for k3 = [0 : K3):
            parallel_for c3 = [0 : C3):
// Chiplet level
for p2 = [0 : P2):
    for q2 = [0 : Q2):
        parallel_for k2 = [0 : K2):
            parallel_for c2 = [0 : C2):
// PE level
for r = [0 : R):
    for s = [0 : S):
        for k1 = [0 : K1):
            for c1 = [0 : C1):
                for p1 = [0 : P1):
                    for q1 = [0 : Q1):
// Vector-MAC level
parallel_for k0 = [0 : K0):
    parallel_for c0 = [0 : C0):
        p = (p3 * P2 + p2) * P1 + p1;
        q = (q3 * Q2 + q2) * Q1 + q1;
        k = ((k3 * K2 + k2) * K1 + k1) * K0 + k0;
        c = ((c3 * C2 + c2) * C1 + c1) * C0 + c0;
        OA[p,q,k] += IA[p-1+r,q-1+s,c] * W[r,s,c,k];
```
Scale Up Approaches – Wafer Scale

1.2 trillion transistors
46,255 mm²

Source: Cerebras
And Other Design Options

Per storage level cross product of:

- Dataflow
- Split/Shared storage
- Tiling in time
- Tiling in space
- Bypassing

Plus

- Precision/Quantization
- Scale up
- ....

And flexibility!
Thank you!

Next Lecture:
Accelerators (II)