Hello!

CS 744: TPU

Shivaram Venkataraman
Fall 2021
Midterm 2, Dec 7\textsuperscript{rd}

- Papers from SCOPE to TPU
- Similar format etc.
- Details on Piazza

Poster session: Dec 14th

- More details soon
MOTIVATION

Capacity demands on datacenters
New workloads → voice search → lots of new queries
(Google Assistant)
Siri

Metrics
Power/operation
Performance/operation
Total cost of ownership → acquiring hardware

Custom ASICs for workloads

Goal: Improve cost-performance by 10x over GPUs
For every byte of data (model) → ML inference

<table>
<thead>
<tr>
<th>Name</th>
<th>LOC</th>
<th>Layers</th>
<th>Nonlinear function</th>
<th>Weights</th>
<th>TPU Ops / Weight Byte</th>
<th>TPU Batch Size</th>
<th>% of Deployed TPUs in July 2016</th>
</tr>
</thead>
<tbody>
<tr>
<td>MLP0</td>
<td>100</td>
<td>5 FC, Conv</td>
<td>ReLU</td>
<td>20M</td>
<td>200</td>
<td>200</td>
<td>61%</td>
</tr>
<tr>
<td>MLP1</td>
<td>1000</td>
<td>4 FC</td>
<td>ReLU</td>
<td>5M</td>
<td>168</td>
<td>168</td>
<td>29%</td>
</tr>
<tr>
<td>LSTM0</td>
<td>1000</td>
<td>24 Conv, Vector</td>
<td>sigmoid, tanh</td>
<td>52M</td>
<td>64</td>
<td>64</td>
<td></td>
</tr>
<tr>
<td>LSTM1</td>
<td>1500</td>
<td>37 Conv</td>
<td>sigmoid, tanh</td>
<td>34M</td>
<td>96</td>
<td>96</td>
<td></td>
</tr>
<tr>
<td>CNN0</td>
<td>1000</td>
<td>16 Conv</td>
<td>ReLU</td>
<td>8M</td>
<td>2888</td>
<td>8</td>
<td>5%</td>
</tr>
<tr>
<td>CNN1</td>
<td>1000</td>
<td>4 Conv, 72 Pool</td>
<td>ReLU</td>
<td>100M</td>
<td>1750</td>
<td>32</td>
<td></td>
</tr>
</tbody>
</table>

DNN: RankBrain, LSTM: subset of GNM Translate
CNNs: Inception, DeepMind AlphaGo
Quantization $\rightarrow$ Lower precision, energy use

8-bit integer multiplies (unlike training), 6X less energy and 6X less area

Need for predictable latency and not throughput

e.g., 7ms at 99th percentile

Particularly important for serving
- not just low avg latency
- but we want predictability!
TPU DESIGN
CONTROL

- Mounted on PCIe interface
- PCIe BW is low and this impacts latency/throughput
- Instructions need to be coarse grained to minimize host to device comm.
COMPUTE

- Biggest area occupied by Matrix Multiply Unit
  → Fully connected
  → Convolutions ...
  → Activation ReLU, tanh etc ...
  → Specialized compute Units

- Activation Rew, \text{tanh} etc ...

\[
\rightarrow \text{specialized compute Units}
\]
DATA $\rightarrow$ model (data) $\rightarrow$ output

1. Models are stored in DDR3.
2. Input examples go to unified buffer. Both are inputs to matrix multiply.

$\Rightarrow$ DDR3 BW is lower than unified buffer BW.
INSTRUCTIONS

CISC format (why ?)

1. Read_Host_Memory ← input
2. Read_Weights ← model
3. MatrixMultiply/Convolve
4. Activate
5. Write_Host_Memory

Complex Instruction Set

PCIe, but not that high

Each instruction can take a long time!

maps very closely to the workload
SYSTOLIC EXECUTION

Problem: Reading a large SRAM uses much more power than arithmetic!

Typical implementation:
- Read from cache/register
- FPU to do MAC
- Write back to cache/register

Structured propagation of data between compute units

Predictable performance, while minimizing power
ROOFLINE MODEL

Capture the performance that is available given arithmetic intensity.

- Slope and a flat line
- Points along the line
- Slope = mem access bound
- Flat line = compute bound

Operational Intensity: MAC Ops/weight byte
HASWELL ROOFLINE

State of the art at next time

86 for TPUs

Operational Intensity: MAC Ops/weight byte
# Comparison with CPU, GPU

<table>
<thead>
<tr>
<th>Model</th>
<th>$mm^2$</th>
<th>nm</th>
<th>MHz</th>
<th>TDP</th>
<th>Measured</th>
<th>TOPS/s</th>
<th>GB/s</th>
<th>On-Chip Memory</th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>Idle</td>
<td>Busy</td>
<td>8b</td>
<td>FP</td>
</tr>
<tr>
<td>Haswell E5-2699 v3</td>
<td>662</td>
<td>22</td>
<td>2300</td>
<td>145W</td>
<td>41W</td>
<td>145W</td>
<td>2.6</td>
<td>1.3</td>
</tr>
<tr>
<td>NVIDIA K80</td>
<td>561</td>
<td>28</td>
<td>560</td>
<td>150W</td>
<td>25W</td>
<td>98W</td>
<td>--</td>
<td>2.8</td>
</tr>
<tr>
<td>TPU</td>
<td>$&lt;331^*$</td>
<td>28</td>
<td>700</td>
<td>75W</td>
<td>28W</td>
<td>40W</td>
<td>92</td>
<td>--</td>
</tr>
</tbody>
</table>

- *Idle power is much lower*
- *Power / performance ratio*
SELECTED LESSONS

• Latency more important than throughput for inference

• LSTMs and MLPs are more common than CNNs

• Performance counters are helpful for benchmarking and debugging

• Remember architecture history
New workloads → new hardware requirements

Domain specific design (understand workloads!)
   No features to improve the average case
   No caches, branch prediction, out-of-order execution etc.
   Simple design with MACs, Unified Buffer gives efficiency

Drawbacks
   No sparse support, training support (TPU v2, v3)
   Vendor specific?
DISCUSSION

https://forms.gle/LFeaeME4pFdHZdMV6
For all hardware, increas batch size

Incr util & throughput but at cost of latency

GPU, CPU sacrifice utilization to reach 7ms SLO

<table>
<thead>
<tr>
<th>Type</th>
<th>Batch</th>
<th>99th% Response</th>
<th>Inf/s (IPS)</th>
<th>% Max IPS</th>
</tr>
</thead>
<tbody>
<tr>
<td>CPU</td>
<td>16</td>
<td>7.2 ms</td>
<td>5,482</td>
<td>42%</td>
</tr>
<tr>
<td>CPU</td>
<td>64</td>
<td>21.3 ms</td>
<td>13,194</td>
<td>100%</td>
</tr>
<tr>
<td>GPU</td>
<td>16</td>
<td>6.7 ms</td>
<td>13,461</td>
<td>37%</td>
</tr>
<tr>
<td>GPU</td>
<td>64</td>
<td>8.3 ms</td>
<td>36,465</td>
<td>100%</td>
</tr>
<tr>
<td>TPU</td>
<td>200</td>
<td>7.0 ms</td>
<td>225,000</td>
<td>80%</td>
</tr>
<tr>
<td>TPU</td>
<td>250</td>
<td>10.0 ms</td>
<td>280,000</td>
<td>100%</td>
</tr>
</tbody>
</table>

Can handle much larger batch size

Able to meet 1 ms while 200K+ throughput
How would TPUs impact serving frameworks like Nexus? What specific effects it could have on distributed serving systems architecture.

- Goals
  - Nexus: high Acc util
  - TPU: pred. latency
    - have predictable latency for a range of input sizes
      - could make Nexus profiles more stable
  - Less chance for stragglers
Next week schedule

Tue: Midterm 2

Thu: Last class! (Fairness in ML, Summary)