In-Data Center Performance Analysis of a Tensor Processing Unit

Norman P. Jouppi, Cliff Young, Nishant Patil, David Patterson, Gaurav Agrawal, Raminder Bajwa, Sarah Bates, Suresh Bhatia, Nan Boden, Al Borchers, Rick Boyle, Pierre-luc Cantin, Clifford Chao, Chris Clark, Jeremy Coriell, Mike Daley, Matt Dau, Jeffrey Dean, Ben Gelb, Tara Vazir Ghaemmaghami, Rajendra Gottipati, William Gulland, Robert Hagmann, C. Richard Ho, Doug Hogberg, John Hu, Robert Hundt, Dan Hurt, Julian Ibarz, Aaron Jaffey, Alek Jaworski, Alexander Kaplan, Harshit Khaitan, Daniel Killebrew, Andy Koch, Naveen Kumar, Steve Lacy, James Laudon, James Law, Diemthu Le, Chris Leary, Zhuyuan Liu, Kyle Lucke, Alan Lundin, Gordon MacKean, Adriana Maggiore, Maire Mahony, Kieran Miller, Rahul Nagarajan, Ravi Narayanaswami, Ray Ni, Kathy Nix, Thomas Norrie, Mark Omernick, Narayana Penukonda, Andy Phelps, Jonathan Ross, Matt Ross, Amir Salek, Emad Samadiani, Chris Severn, Gregory Sizikov, Matthew Snelham, Jed Souter, Dan Steinberg, Andy Swing, Mercedes Tan, Gregory Thorson, Bo Tian, Horia Toma, Erick Tuttle, Vijay Vasudevan, Richard Walter, Walter Wang, Eric Wilcox, and Doe Hyun Yoon

June 26, 2017

TPU Origin Timeline

- 2013: Prepare for success-disaster of new DNN apps
  - If only CPUs, need 2X whole datacenter fleet for DNNs
- Custom hardware to reduce the TCO (total cost of ownership) of DNN inference by 10X vs. GPUs or CPUs
- Running in datacenter in 15 months
  - Architecture, compiler, hardware design, build, test, deploy
- At Google I/O on May 18, 2016 Google CEO Sundar Pichai reveals Tensor Processing Unit as “10X performance/Watt”

TPU Context: Moore's Law

- Moore's Law: The number of transistors per chip increases by $O(n^2)$ with a process scaling by a factor of n

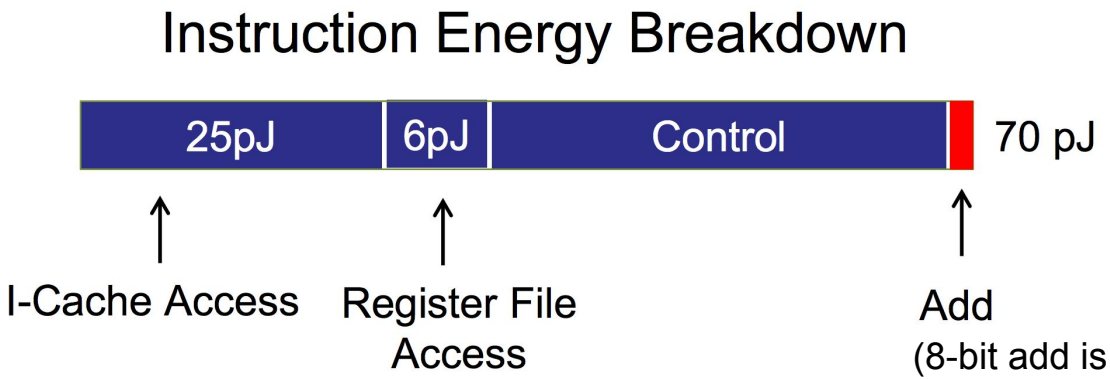
- Historical means of exploiting $O(n^2)$ transistors:
  - Use all the transistors you can to build a faster core and bigger cache memories until you get diminishing returns
  - Then use remaining die area to replicate cores and memories to increase throughput (both in CPUs and GPUs)
  - Number of cores ends up growing as $O(n^2)$

Key Insight

- We want to accelerate tensor math
  - Vectors are tensors of order 1: O(n)
  - 2D matrices are tensors of order 2: $O(n^2)$
- Let's use the $O(n^2)$ transistors from Moore's Law to support multiplication of order 2 tensors natively!
- "Schoolbook" matrix multiply requires $O(n^3)$ operations, so compute in O(n) time
- Use all the die area for just 1 "super brawny" tensor core

Key Insight

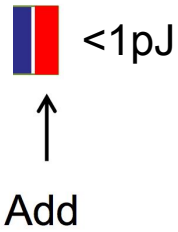
- Energy for control logic, SRAM, and register accesses needed by matrix multiply dominates in conventional processors
- Example from Mark Horowitz's ISSCC 2014 Keynote, slide 33: "Computing's Energy Problem: (and what we can do about it)":



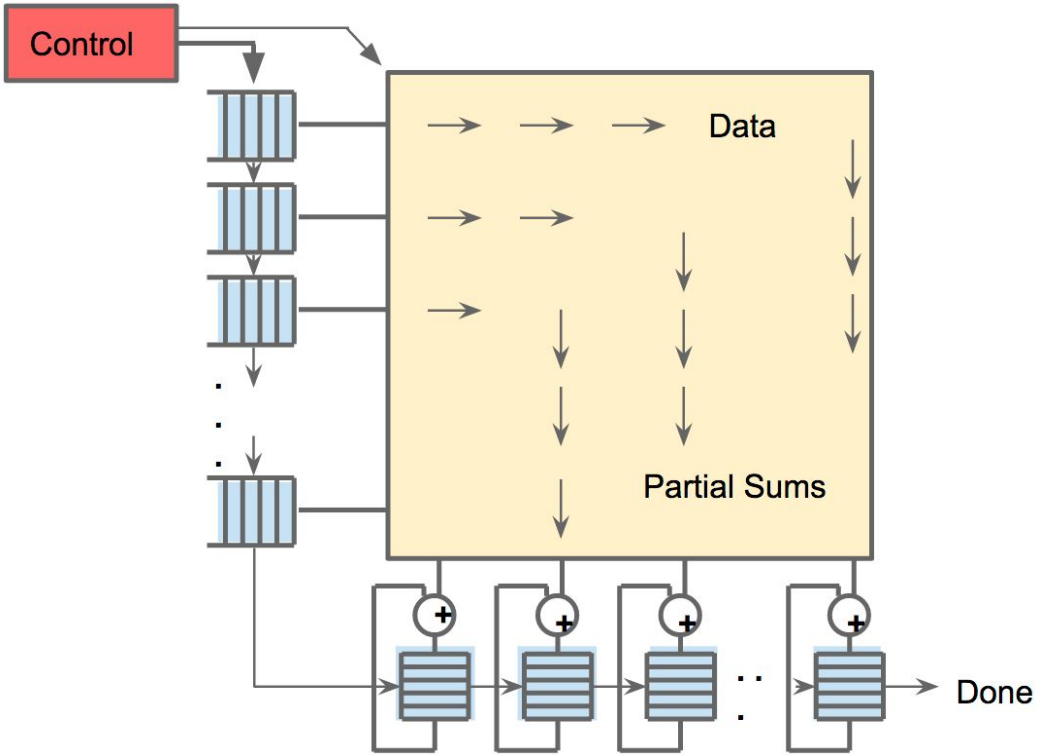
Key Insight

- Solution: matrix operations on a 256x256 systolic array
  - Eliminate complex control logic (use pipelined enable bit)
  - Reuse fetched memory and register data >100X
  - Reduce energy overhead per compute by >10X

Instruction Energy Breakdown

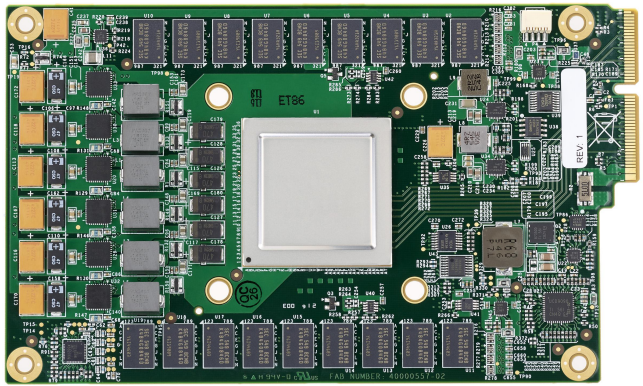


Systolic Execution: Data is Pipelined



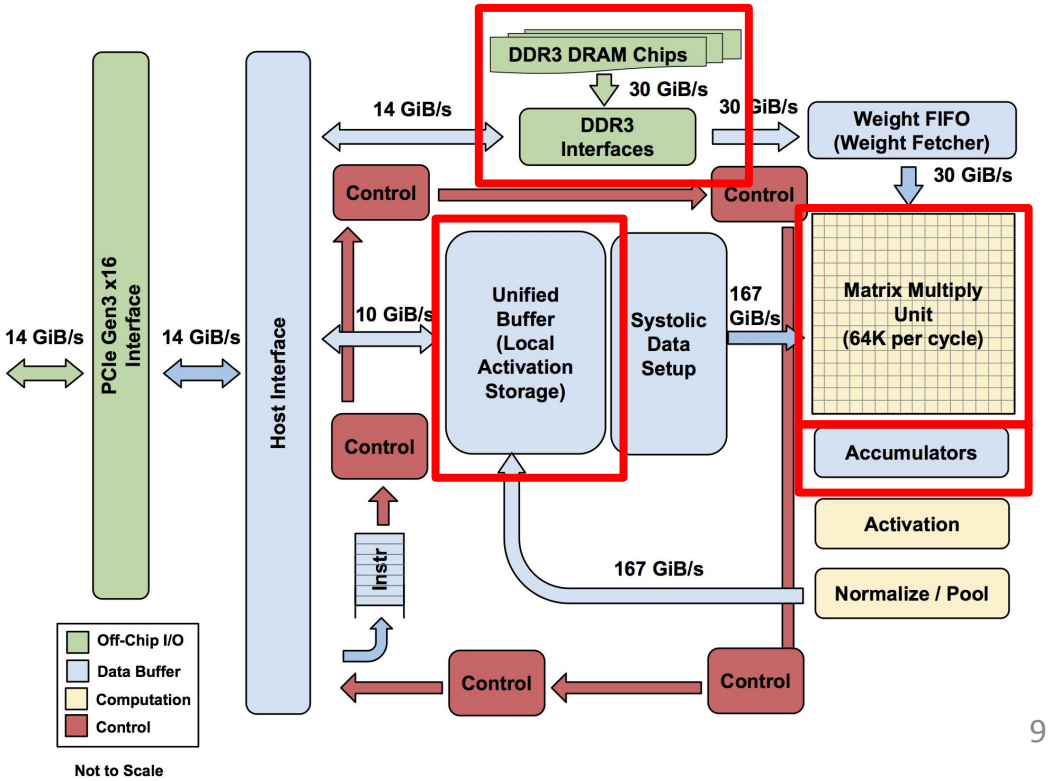
TPU Architecture and Implementation

- Add TPUs to existing servers
  - Up to 4 cards per server
  - Connect over I/O bus (“PCIe”)
- Host server sends it CISC instructions
  - Complexity in SW vs. HW: No branches, only in-order issue, SW controlled buffers, SW controlled pipeline sync

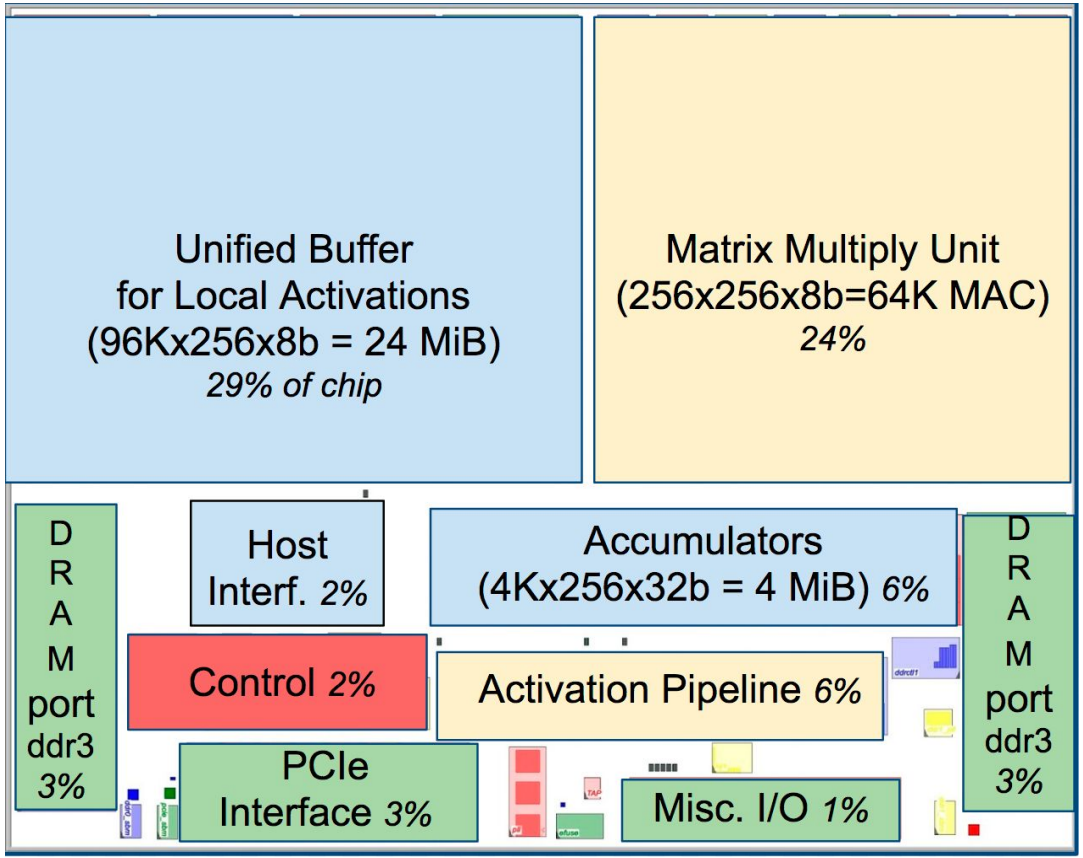


TPU: High-level Chip Architecture

- 700MHz clock rate
- The Matrix Unit: 65,536 (256x256) 8-bit multiply-accumulate ops
- Peak: 92T operations/second
  - 65,536 * 2 * 700M
- >25X as many MACs vs GPU
- >100X as many MACs vs CPU
- 4 MiB of on-chip Accumulator memory
- 24 MiB of on-chip Unified Buffer (activation memory)
- Two 2133MHz DDR3 DRAM channels
- 8 GiB of off-chip weight DRAM memory



TPU: A Neural Network Accelerator Chip



Inference Datacenter Workload (95%)

As of July 2016:

Name	*LOC*	*Layers*					*Nonlinear function*	*Weights*	*TPU Ops / Weight Byte*	*TPU Batch Size*	*% Deployed*
		FC	*Conv*	*Vector*	*Pool*	*Total*					
MLP0	0.1k	5				5	ReLU	20M	200	200	61%
MLP1	1k	4				4	ReLU	5M	168	168	
LSTM0	1k	24		34		58	sigmoid, tanh	52M	64	64	29%
LSTM1	1.5k	37		19		56	sigmoid, tanh	34M	96	96	
CNN0	1k		16			16	ReLU	8M	2888	8	5%
CNN1	1k	4	72		13	89	ReLU	100M	1750	32	

Relative Performance: 3 Contemporary Chips

Processor	*mm²*	*Clock MHz*	*TDP Watts*	*Idle Watts*	*Memory GB/sec*	*Peak TOPS/chip*	
						8b int.	*32b FP*
CPU: Haswell (18 core)	662	2300	145	41	51	2.6	1.3
GPU: Nvidia K80 (13 core, 2 / card)	561	560	150	25	160	--	2.8
TPU	<331*	700	75	28	34	91.8	--

*TPU is less than half die size of the Intel Haswell processor

K80 and TPU in 28 nm process; Haswell fabbed in Intel 22 nm process

These chips and platforms chosen for comparison because widely deployed in Google data centers

Roofline Visual Performance Model

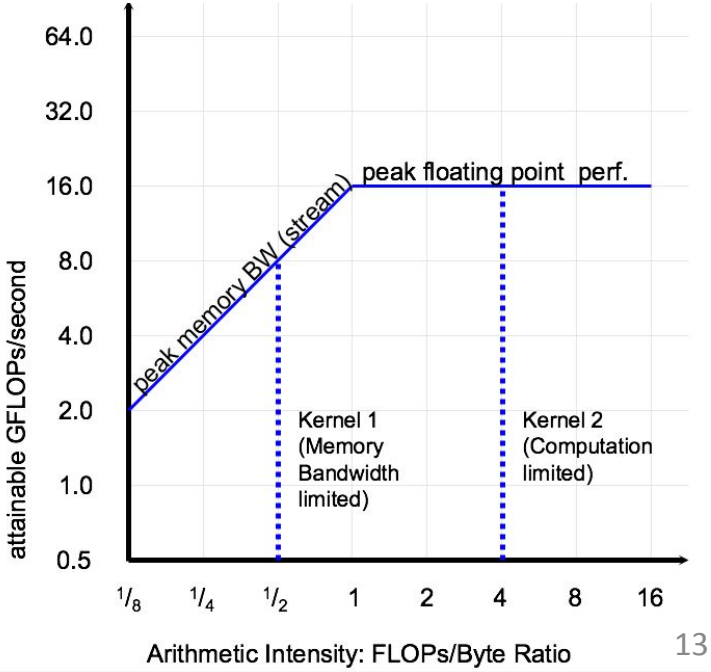
Two limits to performance:

1. Peak Computation
2. Peak Memory Bandwidth
   (For apps with large data that don't fit in cache)

Arithmetic Intensity (FLOP/byte or reuse) determines which limit

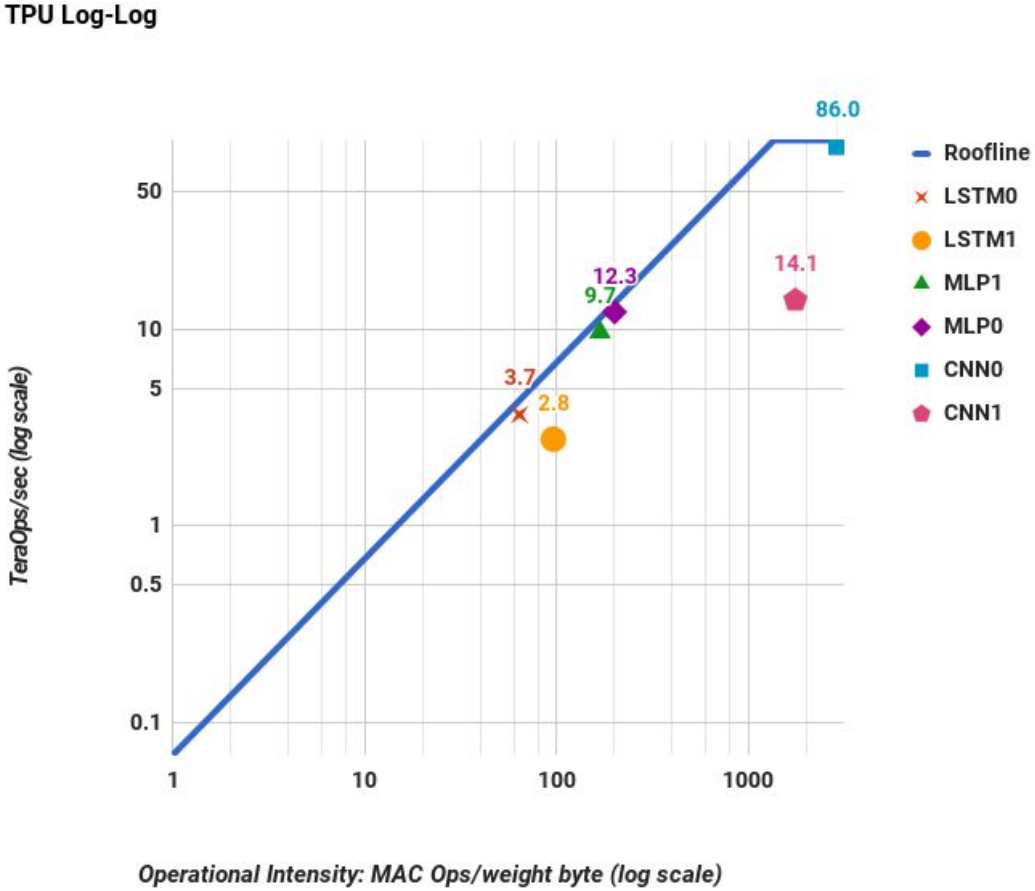
Weight-reuse = Arithmetic Intensity for DNN roofline

GFLOP/s = Min(Peak GFLOP/s, Peak GB/s x AI)

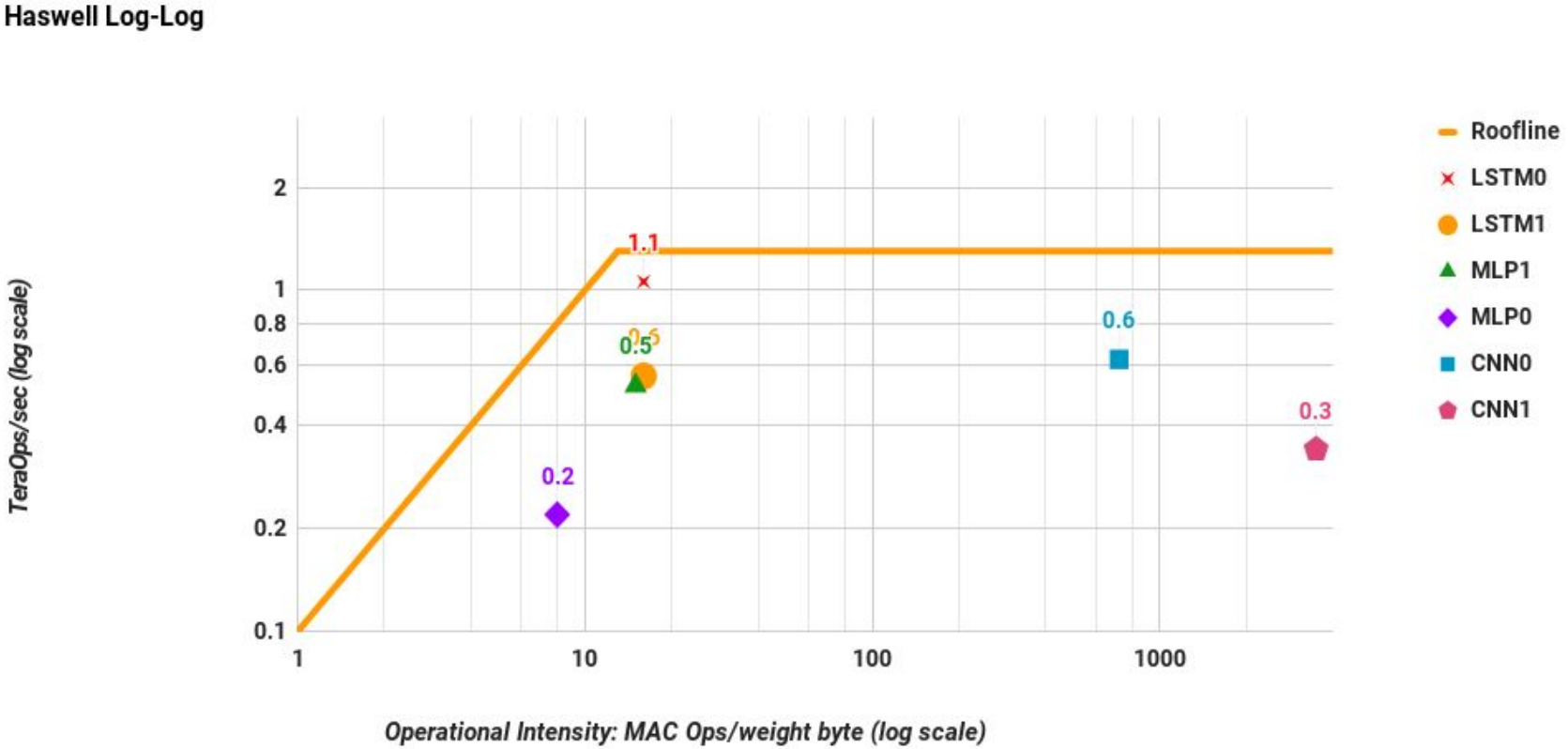


Samuel Williams, Andrew Waterman, and David Patterson. "Roofline: an insightful visual performance model for multicore architectures."*Communications of the ACM* 52.4 (2009): 65-76.

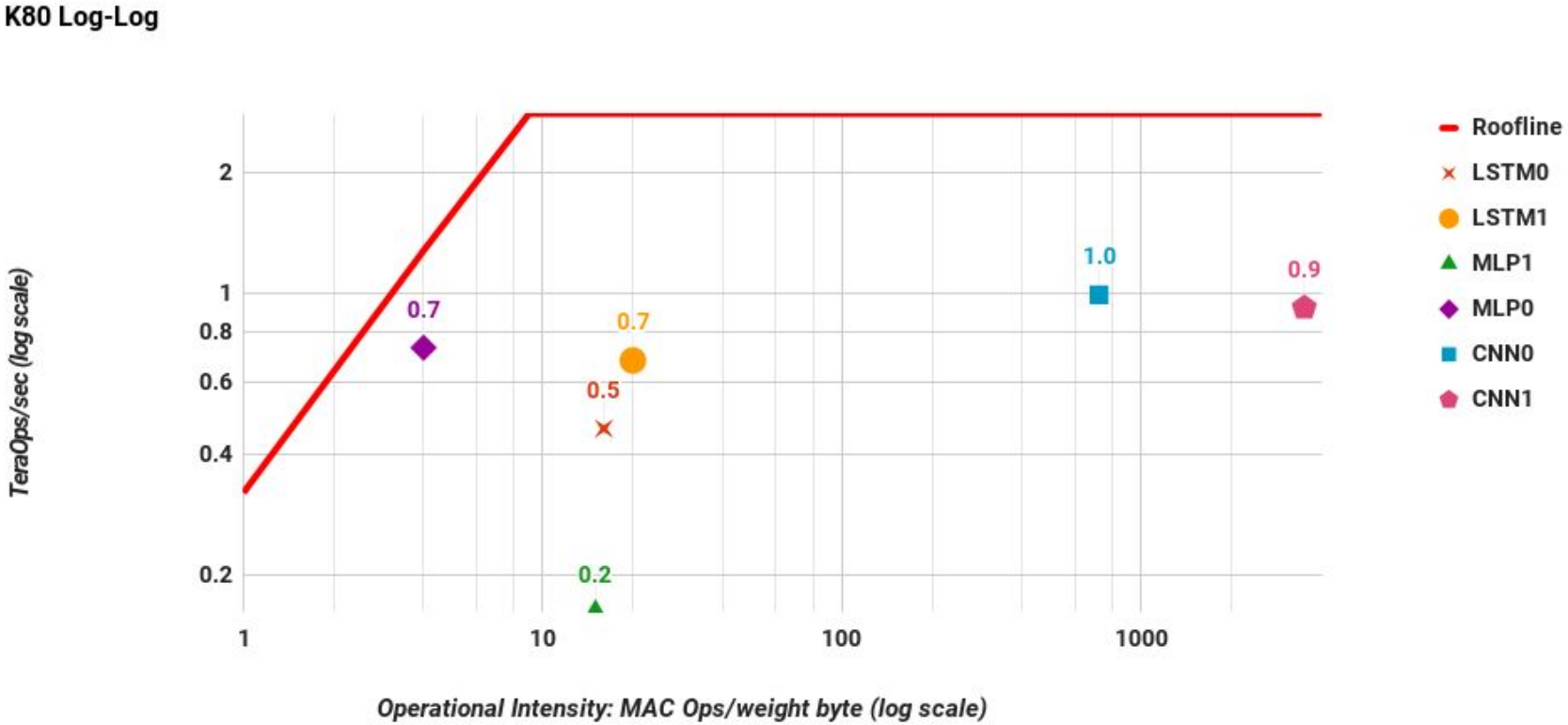
TPU Die Roofline



Haswell (CPU) Die Roofline



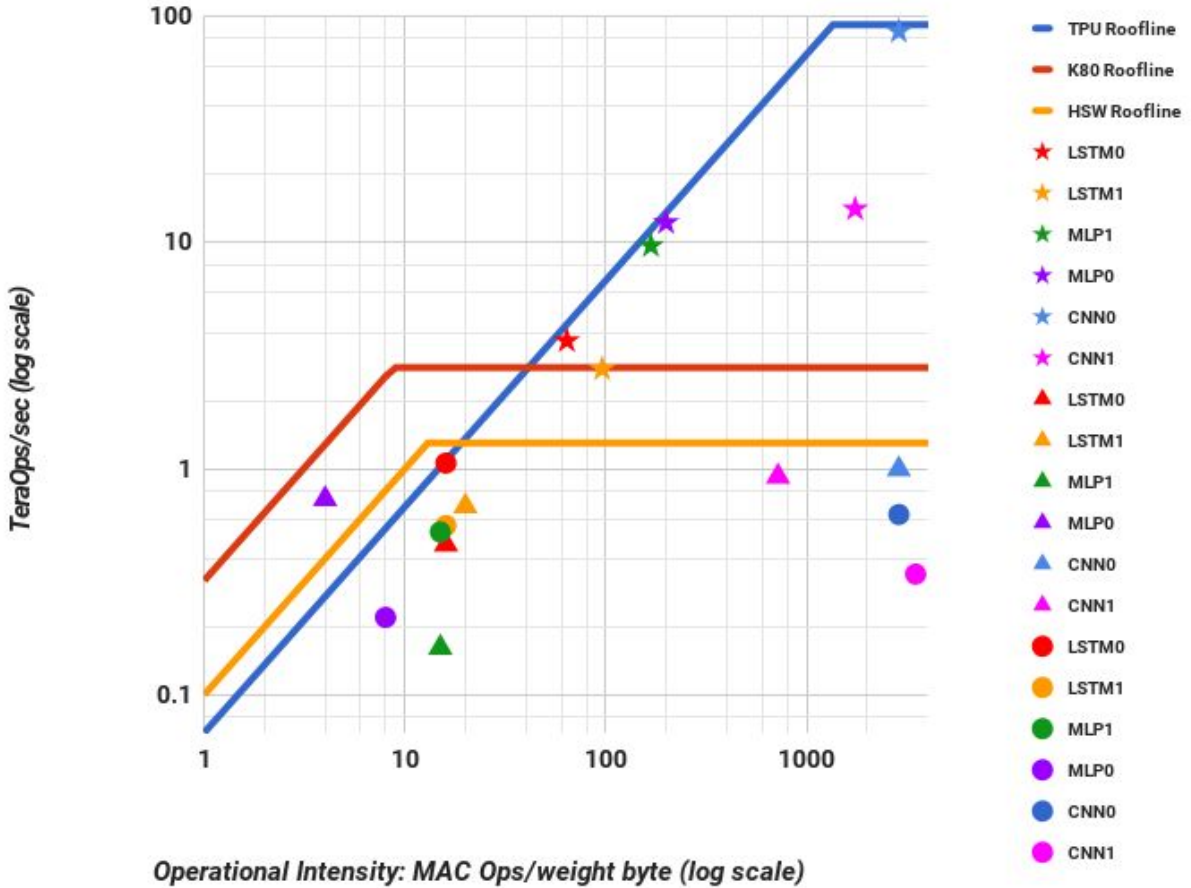
K80 (GPU) Die Roofline



Why so far below Rooflines? (MLP0)

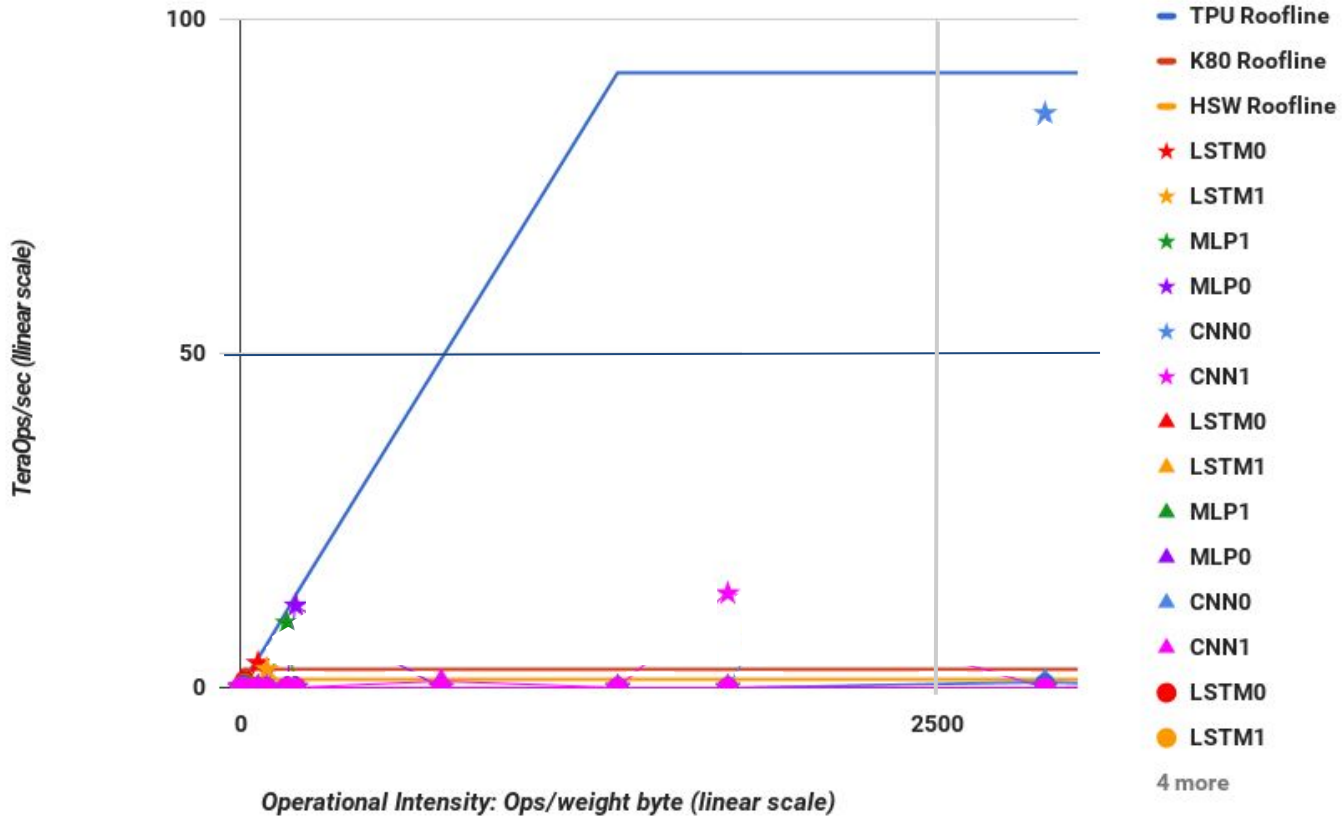
Type	*Batch*	*99th% Response*	*Inf/s (IPS)*		*% Max IPS*
CPU	16	7.2 ms	5,482	↕2.4X	42%
CPU	64	21.3 ms	13,194		100%
GPU	16	6.7 ms	13,461	↕2.7X	37%
GPU	64	8.3 ms	36,465		100%
TPU	200	7.0 ms	225,000	↕1.2X	80%
TPU	250	10.0 ms	280,000		100%

Log Rooflines for CPU, GPU, TPU



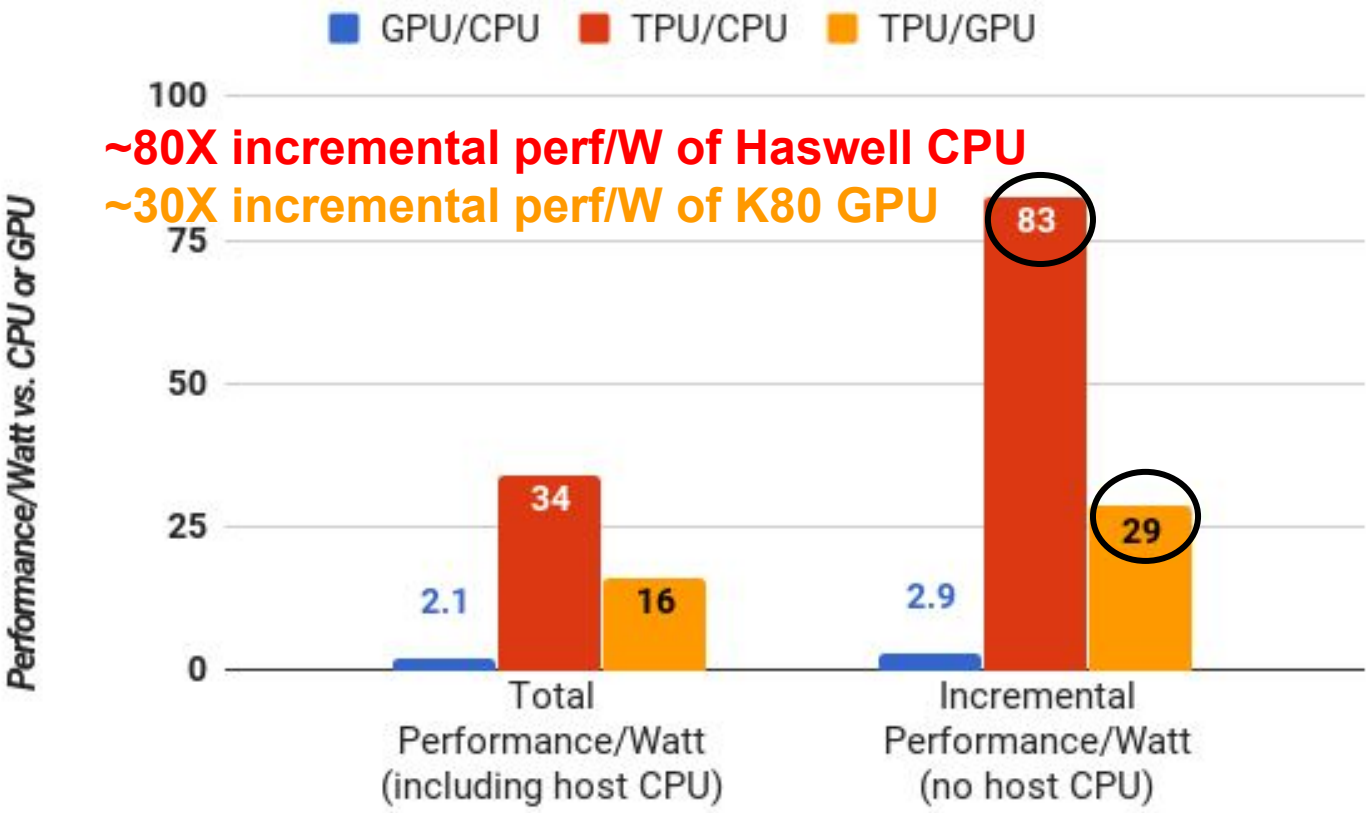
☆ Star = TPU
△ Triangle = GPU
○ Circle = CPU

Linear Rooflines for CPU, GPU, TPU



☆ Star = TPU
△ Triangle = GPU
○ Circle = CPU

Perf/Watt TPU vs CPU & GPU

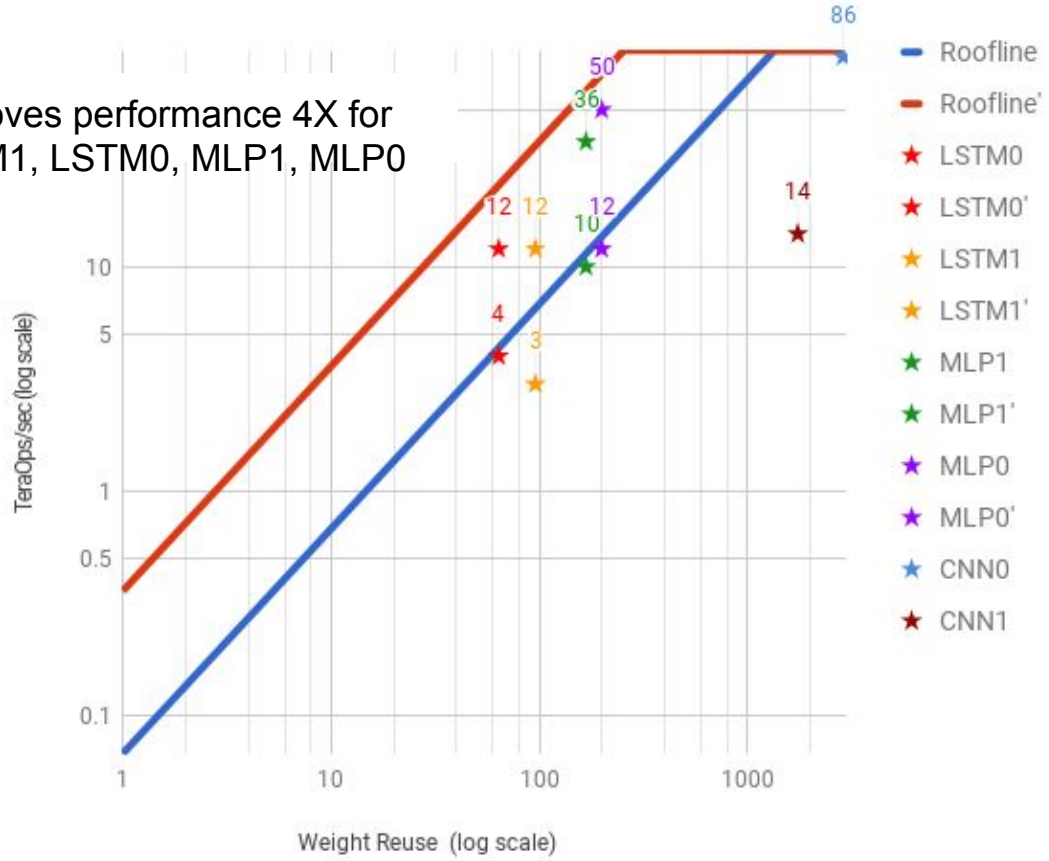


Improving TPU: Move "Ridge Point" to the Left

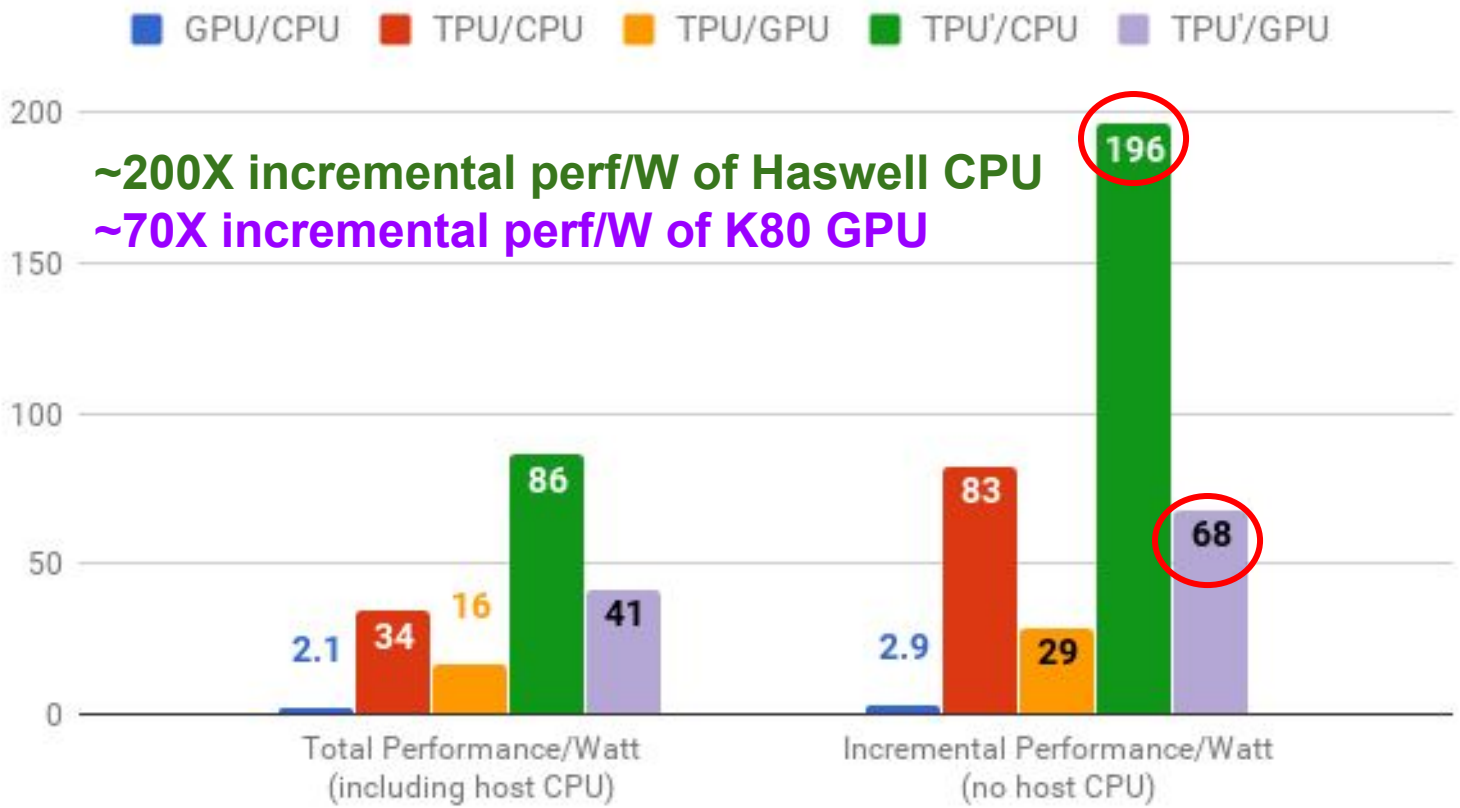
- Current DRAM
  - 2 DDR3 2133 ⇒ 34 GB/s
- Replace with GDDR5 like in K80 ⇒ 180 GB/s
  - Move Ridge Point from 1400 to 256

Revised TPU Raises Roofline

Improves performance 4X for
LSTM1, LSTM0, MLP1, MLP0



Perf/Watt Original & Revised TPU



Conclusions

TPU succeeded because of:

- Large systolic matrix multiply unit, extensive data reuse
- Single “brawny core” provided lower latency

10X difference in computer products are rare:

- 15-month design & live on I/O bus yet TPU 15X-30X faster Haswell CPU, K80 GPU (inference), <½ die size, ½ Watts
- GDDR5 memory could improve TPU >2X at low cost

Questions?