Good morning!

CS 744: TPU

Shivaram Venkataraman

Fall 2020

ADMINISTRIVIA

Midterm 2, Dec 3rd

- Papers from SCOPE to TPU
- Similar format etc.

Presentations Dec 8, 10

- Sign up sheet
- Presentation template
- Trial run?

Next Tue: Fairness in ML, Course summary

Thu: Midterm 2

Piazza

Week after: Project Presentations

- 4 min talks
- 3 to 4 slides
  - Problem statement
  - Approach
  - Initial results
  - In-progress / final report → Dec 17th

MOTIVATION

Capacity demands on datacenters

New workloads → Voice search → ML model to convert speech to text

Metrics

- Power/operation
- Performance/operation → Latency (or Tput) → workloads are latency sensitive
- Total cost of ownership
  - ↳ Buy (Build) + Operate

Goal: Improve cost-performance by 10x over GPUs

WORKLOAD

(a) CNNs are only 5%, MLPs are 61%

(b) CNNs have very high Ops / Byte

(c) Number of weights is not correlated with batch size and Ops

(d) MLP & LSTM have same Ops / byte and batch size

Name	LOC	Layers					Nonlinear function	Weights	TPU Ops / Weight Byte	TPU Batch Size	% of Deployed TPUs in July 2016
		FC	Conv	Vector	Pool	Total					
MLP0	100	5				5	ReLU	20M	200	200	61%
MLP1	1000	4				4	ReLU	5M	168	168	
LSTM0	1000	24		34		58	sigmoid, tanh	52M	64	64	29%
LSTM1	1500	37		19		56	sigmoid, tanh	34M	96	96	
CNN0	1000		16			16	ReLU	8M	2888	8	5%
CNN1	1000	4	72		13	89	ReLU	100M	1750	32	

DNN: RankBrain, LSTM: subset of GNM Translate

CNNs: Inception, DeepMind AlphaGo

WORKLOAD: ML INFERNCE

Quantization → Lower precision, energy use

convert model weights from 32 bit float → 8 bit integer

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

caches → improve average case scenario

branch prediction

- Focus on Inference Only !

TPU DESIGN CONTROL

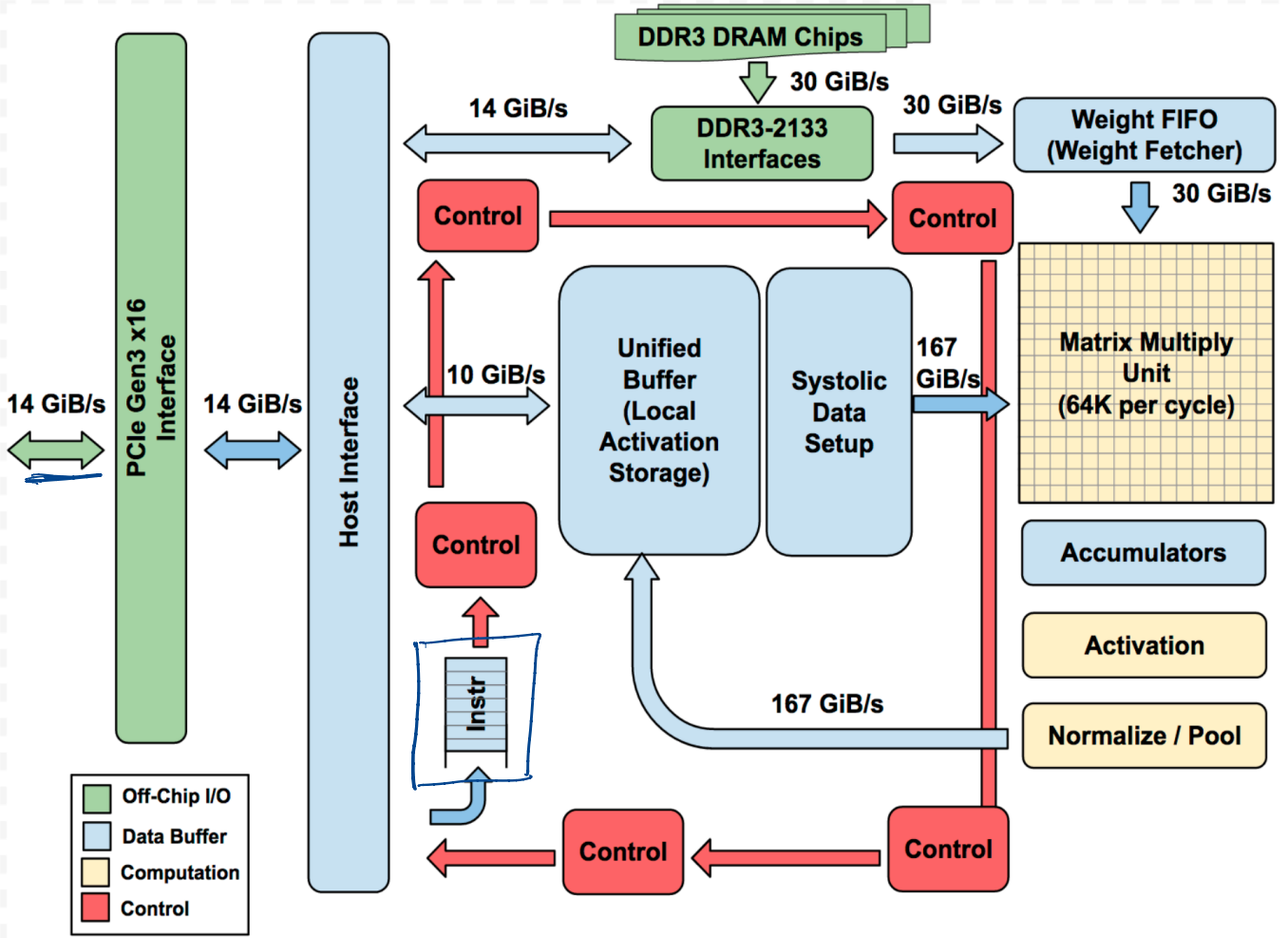
(1) PCIe inter connect for compatibility

→ PCIe has limited bandwidth & latency

(2) Instructions issued from host

↳ Instruction buffer

Simple single thread !



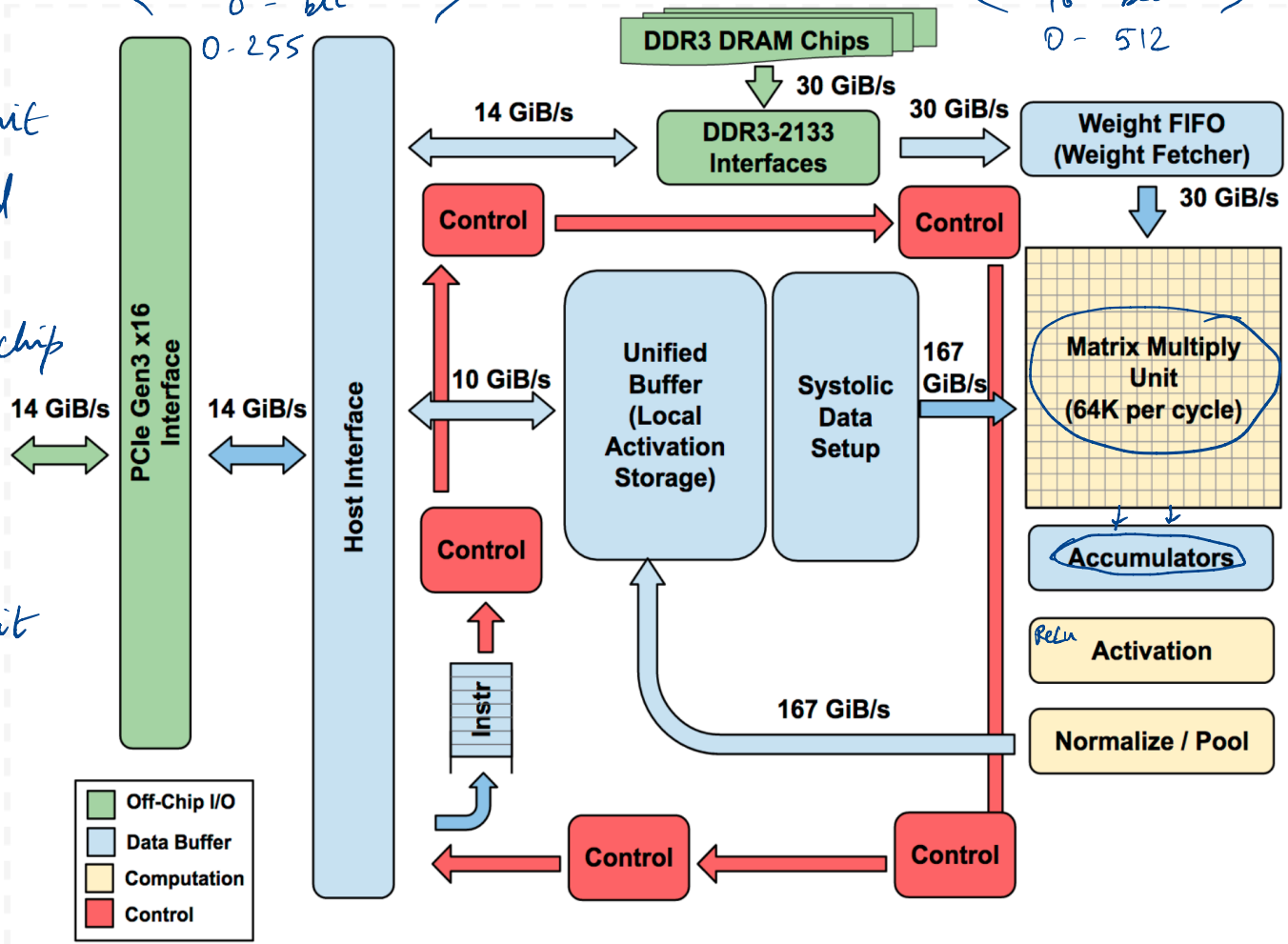
COMPUTE

→ Matrix Multiply Unit
  ↳ Fully Connected
  ↳ Convolutions
→ 24% area of the chip

MAC → Multiply Accumulate

→ 8 bit integer or 16-bit

→ Separate compute units for Activation & Normalize/Pool



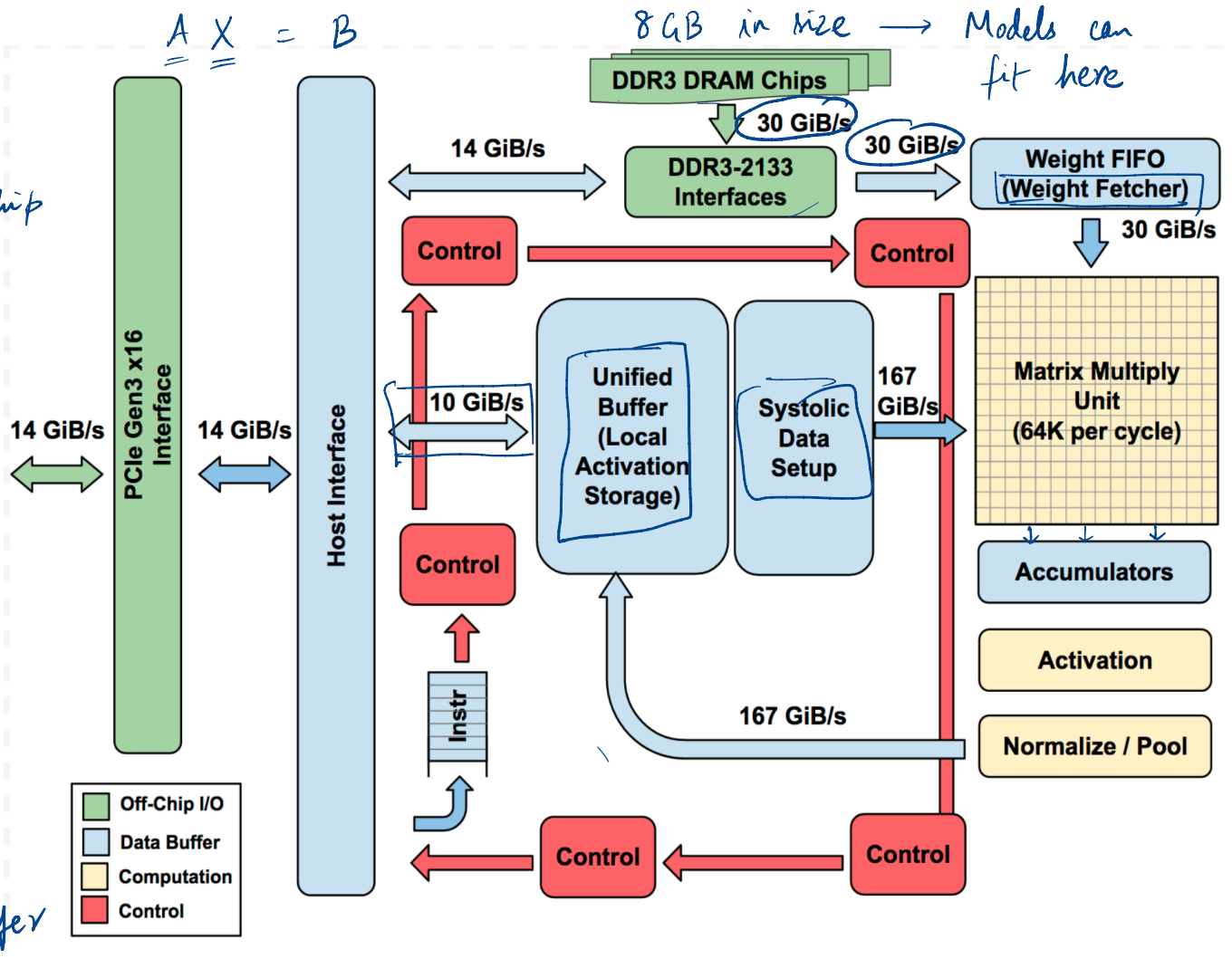
DATA

(1) Models (or weights) are stored in off-chip DRAM

(2) Examples (or inputs) are stored in unified buffer

(3) Pipeline fetching of weights with Matrix Multiply

(4) Intermediate results accumulated and then stored in Unified Buffer



INSTRUCTIONS

CISC format (why ?)

1. Read_Host_Memory
2. Read_Weights →
3. MatrixMultiply/Convolve
4. Activate →
5. Write_Host_Memory

→ Specialized instruction set

→ CISC

↳ Instructions encode operations that take many cycles to run

SYSTOLIC EXECUTION

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

Typical CPU

↳ Caches (L1, L2 etc) or Registers have inputs to compute units

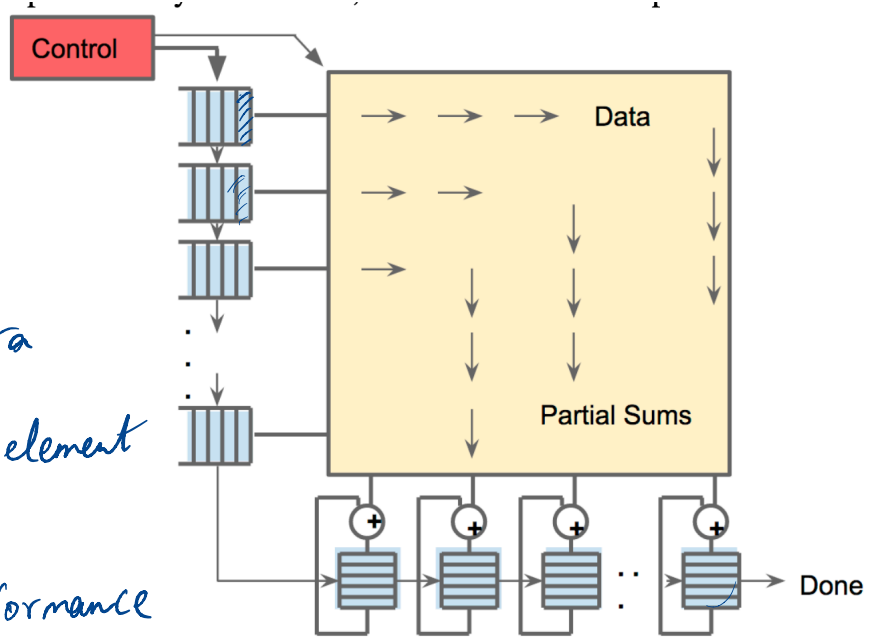
TPU

→ Wave-like propagation of data

→ Data reuse for every element

→ Predictable execution

⇒ predictable performance

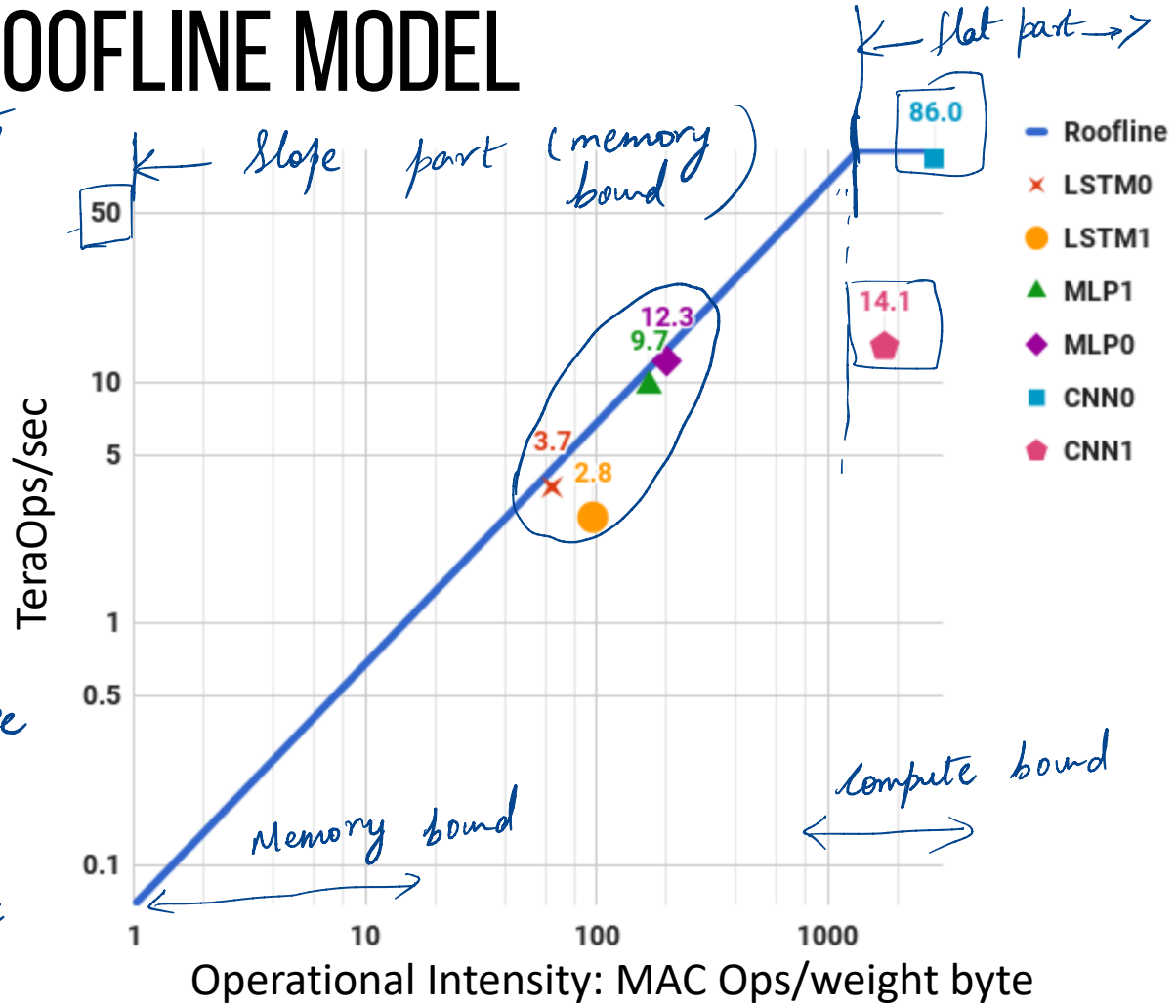


ROOFLINE MODEL

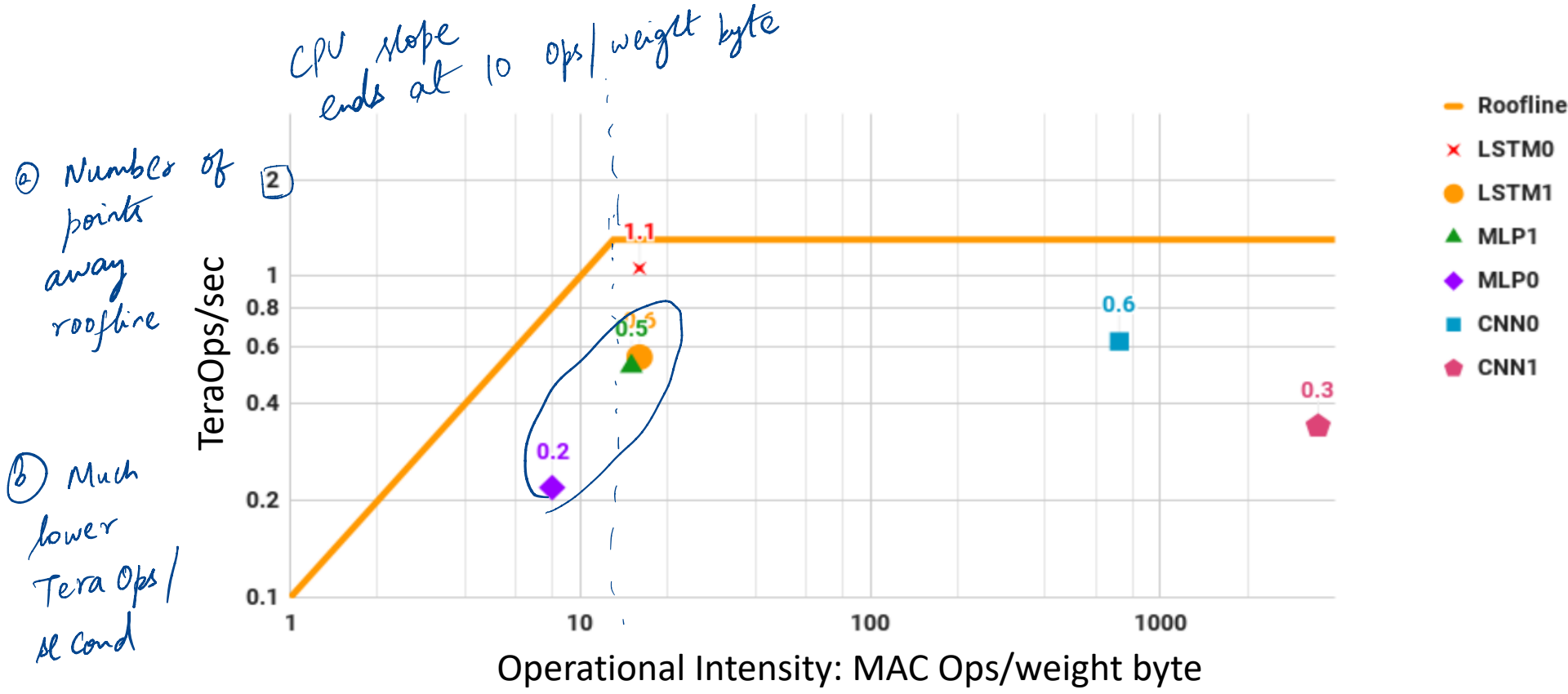
X-axis : Operational intensity
Amount of compute per byte of data read

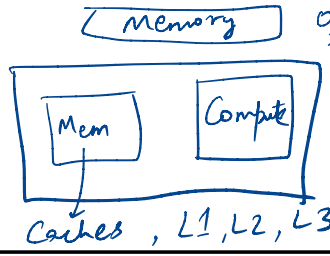
Y-axis : Operations / second

- Blue line comes from hardware spec
- CNNs are compute intensive
- LSTM & MLPs are close to peak perf of hardware



HASWELL ROOFLINE





COMPARISON WITH CPU, GPU

max power use

GPU has a higher mem bandwidth

Model	Die									
	mm^2	nm	MHz	TDP	Measured		TOPS/s		GB/s	On-Chip Memory
					Idle	Busy	8b	FP		
Haswell E5-2699 v3	662	22	2300	145W	41W	145W	2.6	1.3	51	51 MiB
NVIDIA K80 (2 dies/card)	561	28	560	150W	25W	98W	--	2.8	160	8 MiB
TPU v1	<331*	28	700	75W	28W	40W	92	--	34	28 MiB

2x lower compared to CPU and GPU

- GPUs bring down power used when idle
- TPUs not so much

SELECTED LESSONS

- Latency more important than throughput for inference
- LSTMs and MLPs are more common than CNNs
- Performance counters are helpful → to also improve compilers of DNN models
- Remember architecture history

SUMMARY

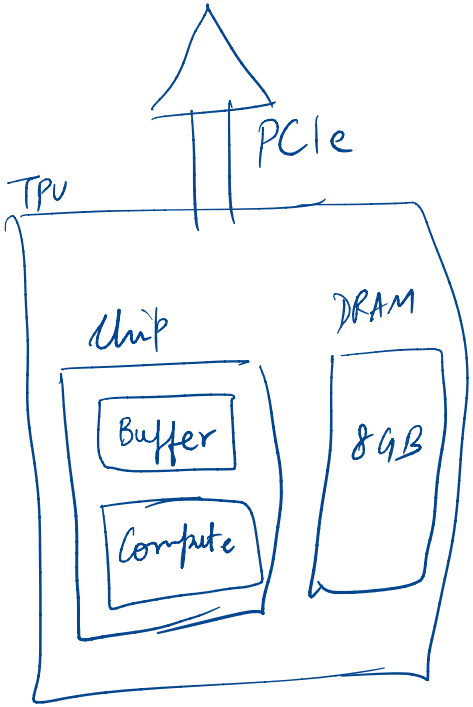
New workloads → new hardware requirements

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/tss99VSCMeMjZx7P6

① Larger batches have higher tput but also higher tail latency

Inferences per sec

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

② TPU tputs are much higher

③ TPU can be at higher util while meeting 7ms latency target

④ GPU has higher IPS at some batch size compared to CPU
→ relates to avg. latency

How would TPUs impact serving frameworks like Clipper? Discuss what specific effects it could have on distributed serving systems architecture

1. TPUs have 8 GB to store many models
   - ↳ but this might break containers in Clipper
2. Stragglers are less frequent
3. Batching (Auto batching) can be very helpful
4.

NEXT STEPS

No class Thursday! Happy Thanksgiving!

Next week schedule:

- Tue: Fairness in ML, Summary
- Thu: Midterm 2

ENERGY PROPORTIONALITY