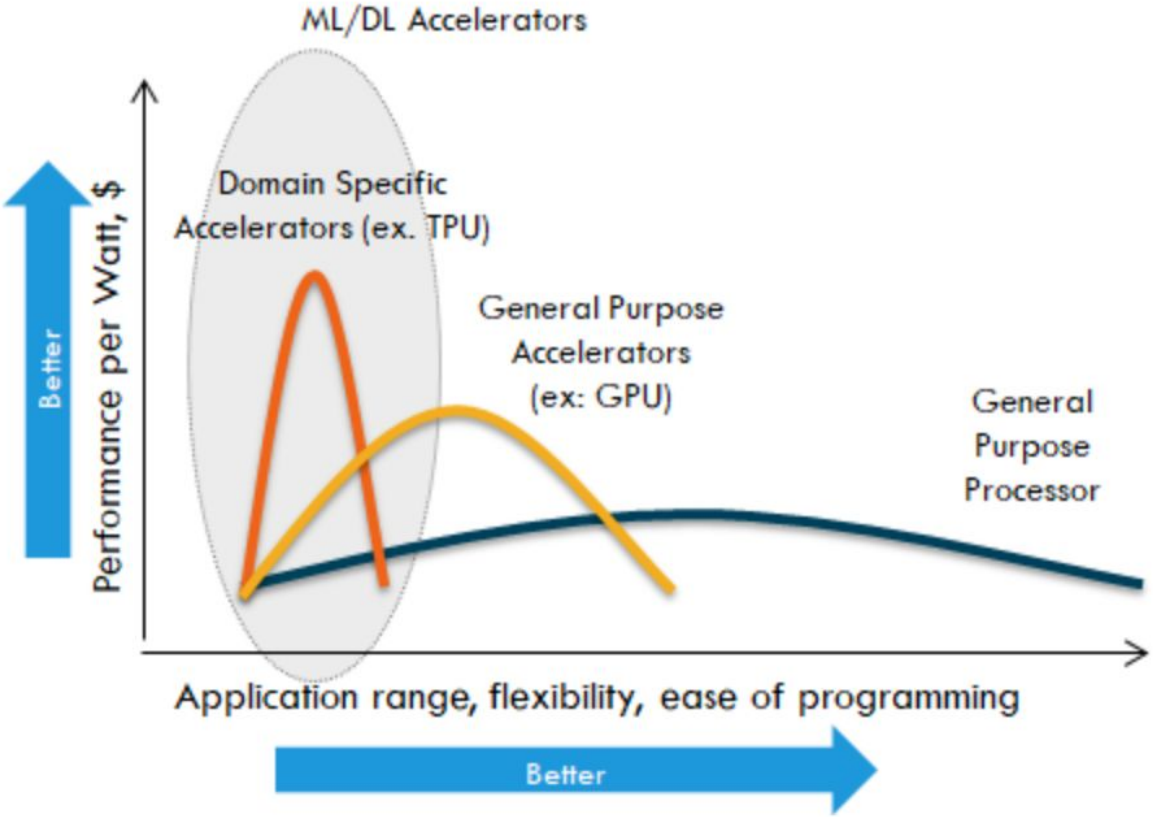
Tissue vs. Silicon

Musings on the Future of Deep Learning Hardware and Software

Nir Shavit
MIT
&
Neural Magic Inc.

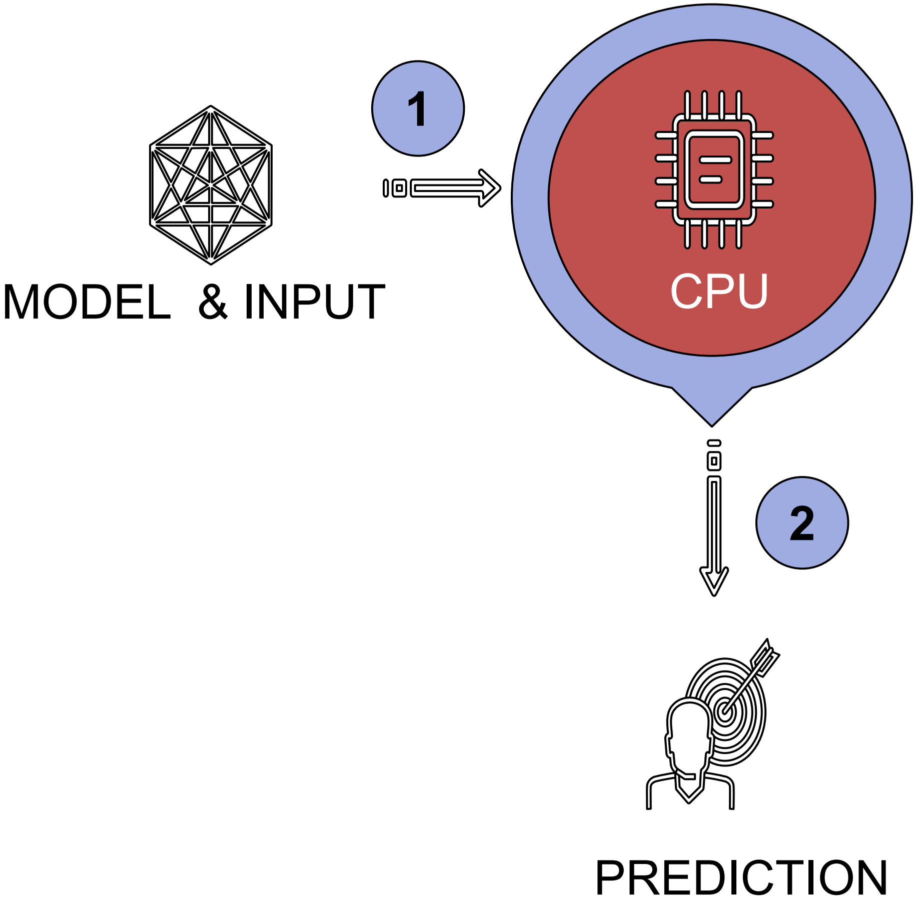
** Disclaimer: all calculations in this talk are "back of the envelope" and should be taken with a grain of salt. Sources available upon request.*

Moore's Law Dead => Long Live Domain Specific Hardware ?



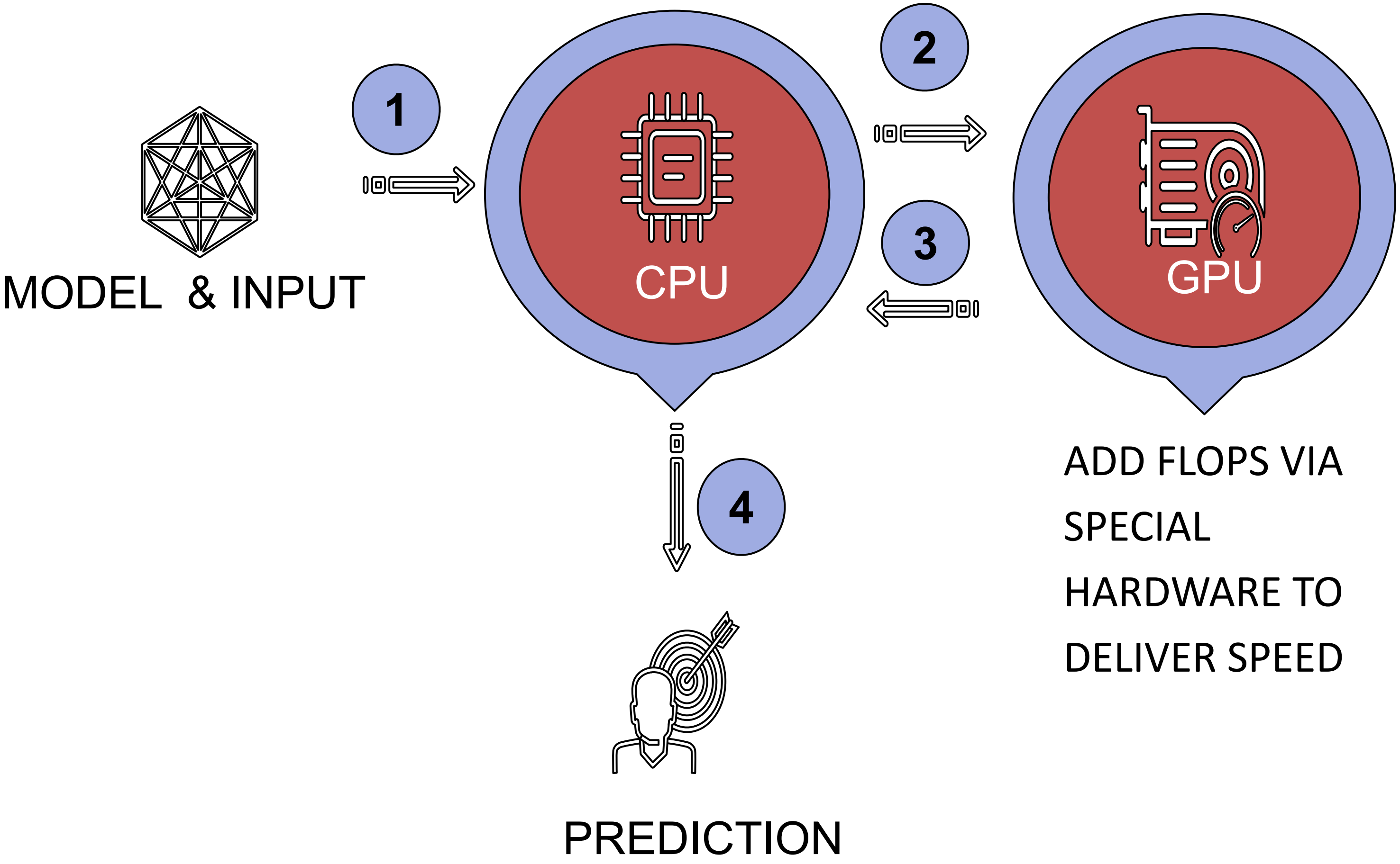
The Story of ML Inferencing

Speed is an enabler, not just a cost saver



The Story of ML Inferencing

Speed is an enabler, not just a cost saver

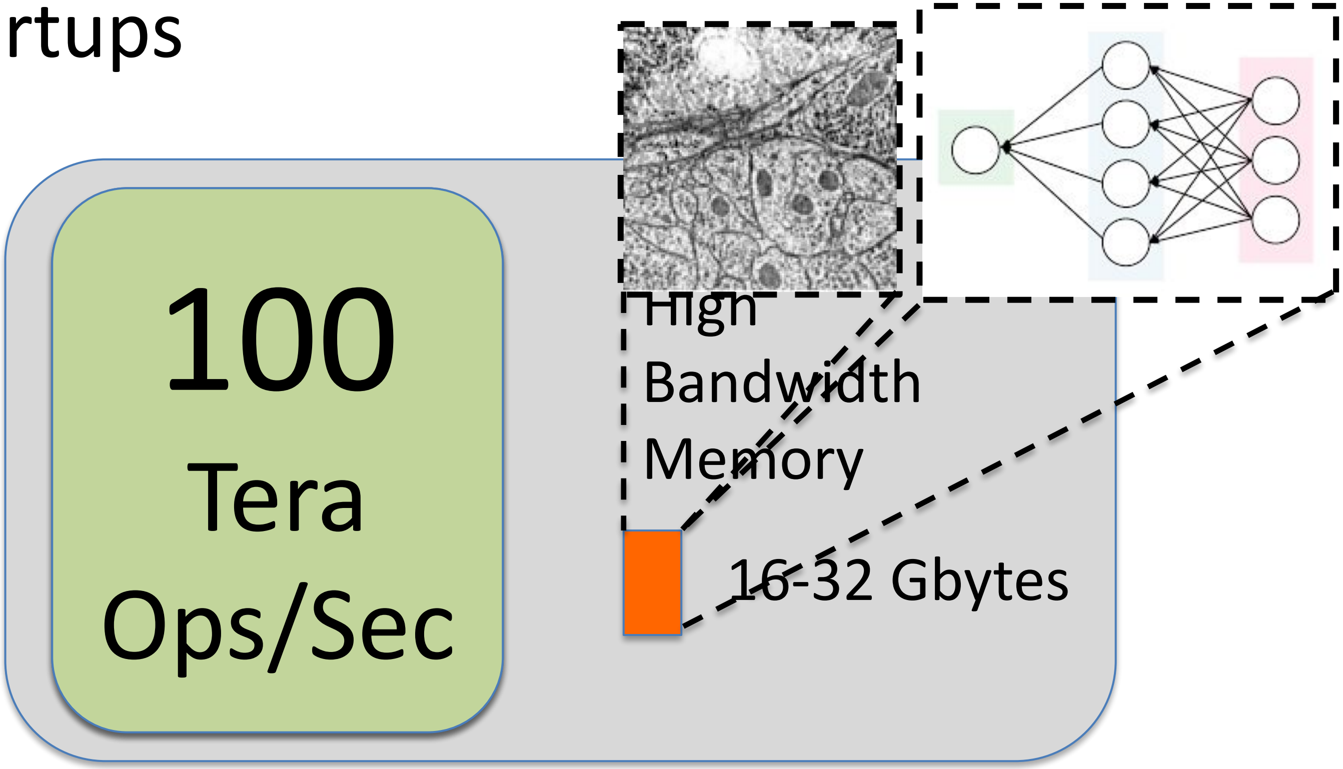


ADD FLOPS VIA SPECIAL HARDWARE TO DELIVER SPEED

Neuromorphic ML Hardware

- “Throughput Computing” hardware for ML (> 100 Billion Market)
- Nvidia GPU / Google TPU / Intel Habana and over 70 Startups

GPU =



100 Tera Ops/Sec

High Bandwidth Memory

16-32 Gbytes

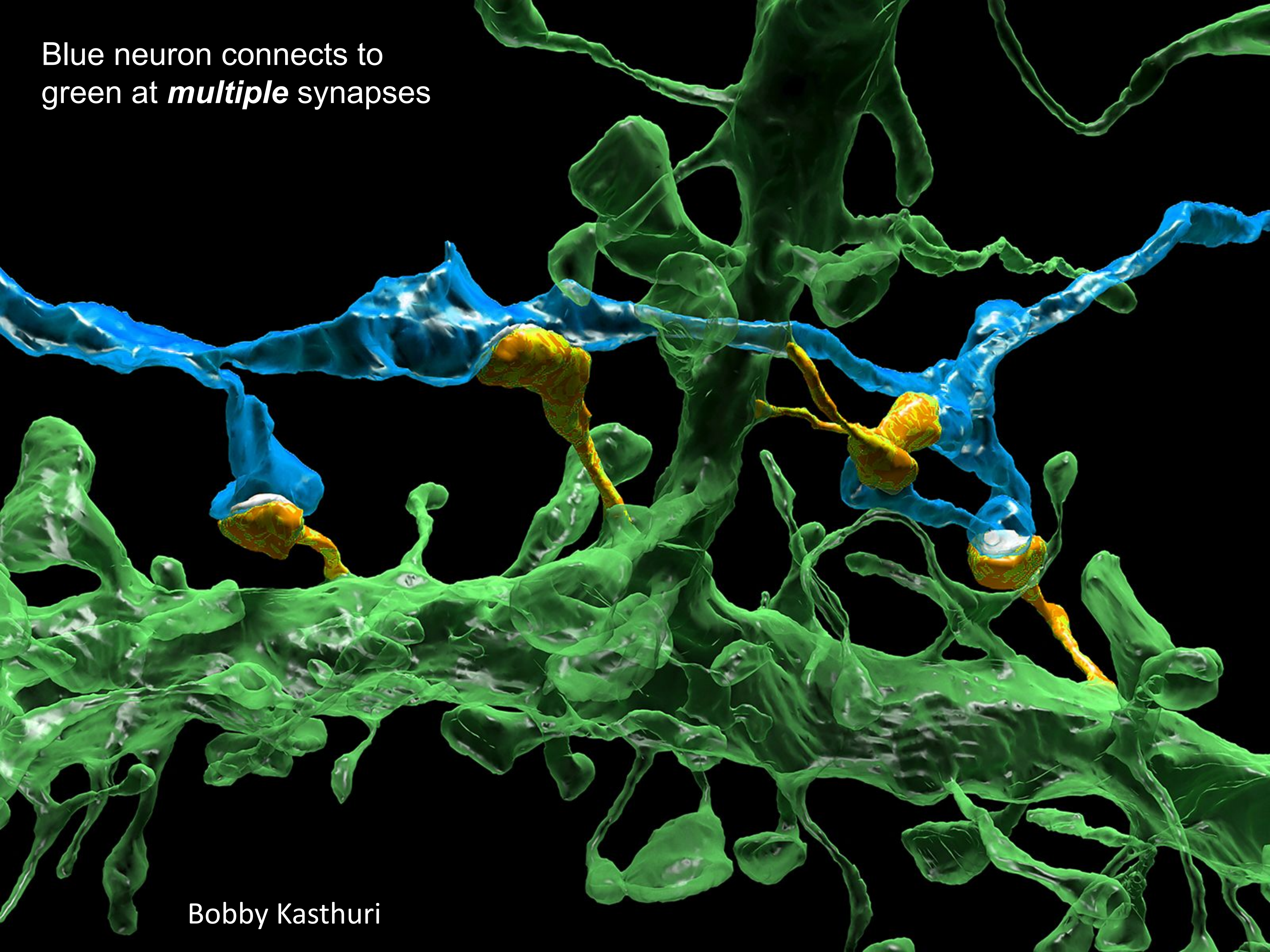
Google: The Brain as a TPU POD

“100 Peta FLOPs of machine learning power”

Compute

- Human Cortex = ~16 billion neurons
- Cortical neurons spike ~0.16 times per second
- ~7000 synapses each = 700 or 70 connections per neuron?

Blue neuron connects to green at ***multiple*** synapses



Bobby Kasthuri

Compute

- Human Cortex = ~16 billion neurons
- Cortical neurons spike ~0.16 times per second
- ~7000 synapses each = 700 or 70 connections per neuron?

  16 B x 0.16 x 700 = ~2 Trillion ops/sec
- iPhone = ~5 Trillion ops/sec

Cortex is 5-6 orders of magnitude less compute than TPU pod

Image Recognition

- A 224x224 = .05 million pixel image takes ~20-30 billion ops to compute on popular NNets
- Human Iris = ~100 million pixels (2,000x more pixels) □

NNets would take at least 40 trillion ops/image

- We can recognize an image in 13ms so even if use whole cortex …

~2 Trillion * .013 = ~20 billion ops/image

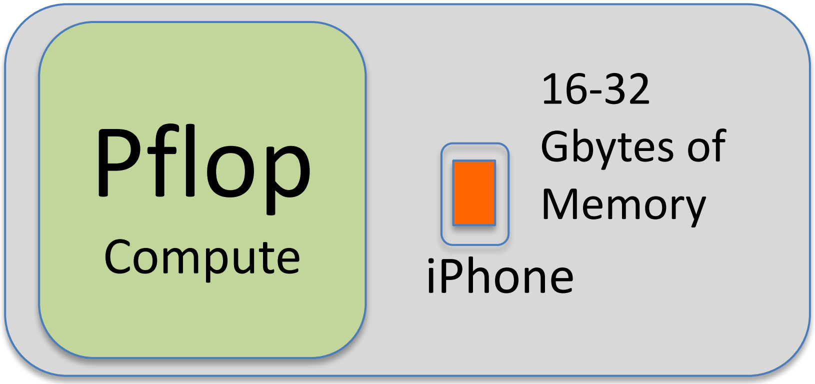
Brain 3-4 orders of magnitude more efficient

Memory Size

- Human Corex = ~300 trillion synapses
- Connectome Graph size? 300 x 4bytes = 1.2Pb
- GPU/TPU typical 16-32Gb HBM2 memory

GPU/TPU pod memory is ~4-5 orders of magnitude too small

Brain in Silicon



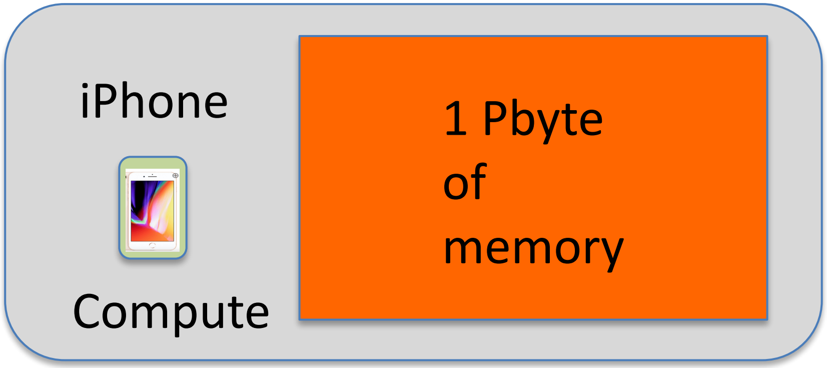
Pflop

Compute

16-32 Gbytes of Memory

iPhone

What we are building



iPhone

Compute

1 Pbyte of memory

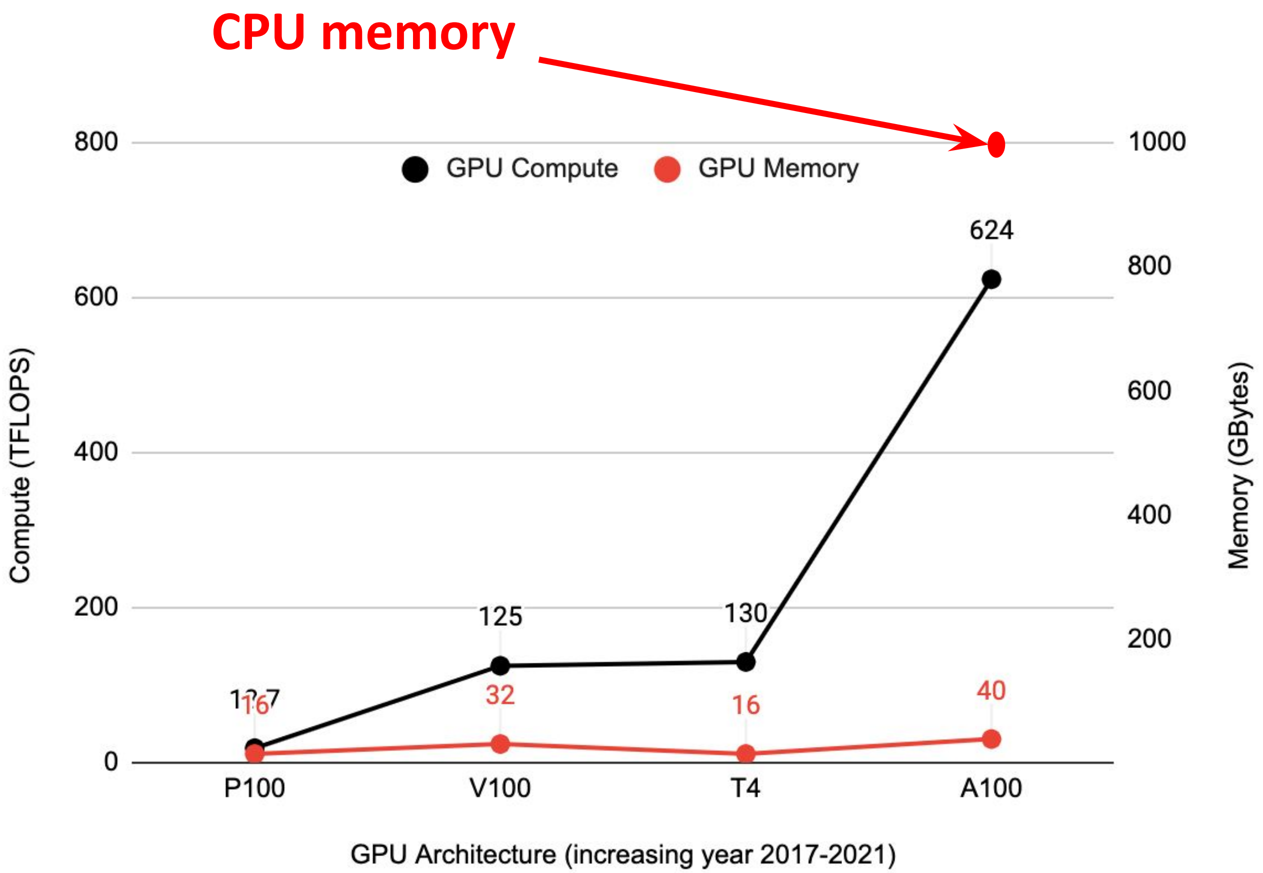
What we need

Why? Because we don’t know the graph…

Future of Neural Hardware/Software

- Silicon need not imitate neural parallelism to reproduce function (flops are flops are flops)
- Neural Tissue is
  - Sparse
  - And has “locality of reference”
- Can we mimic this in hardware/software?
- Yes… and for now perhaps we can best do this is on a CPU

GPU vs. CPU Memory

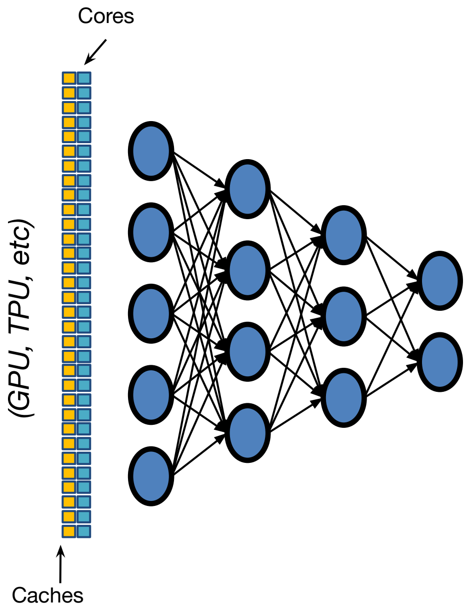

CPU memory

Accelerators have a "Big Model" Problem

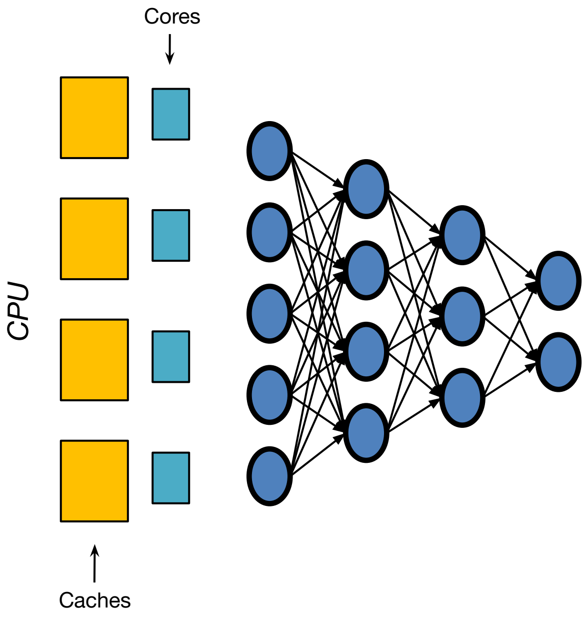
In 3 GPU generations, compute grew 15x. Memory grew only 1.5x!

CPU vs. GPU Compute

HARDWARE ACCELERATORS

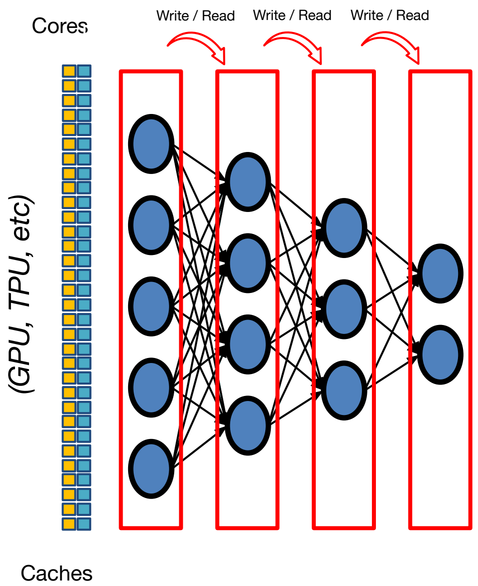


CPU ALONE



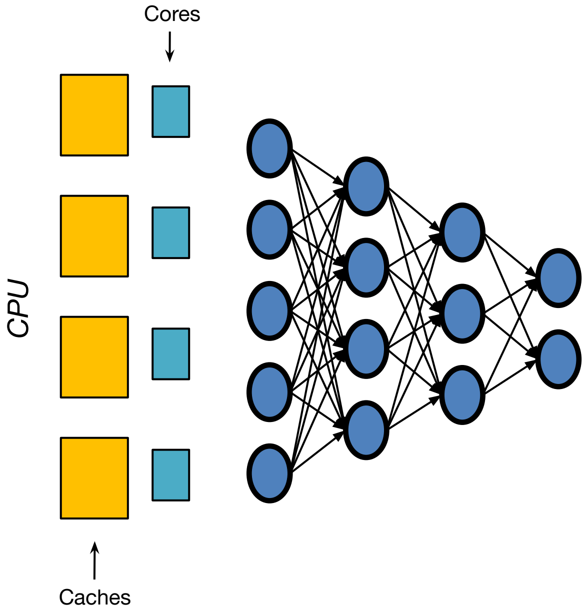
CPU vs. GPU Compute

HARDWARE ACCELERATORS



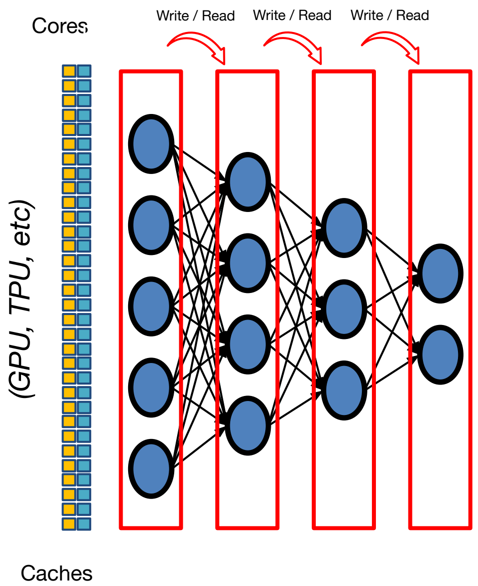
Execute synchronously layer by layer

CPU ALONE



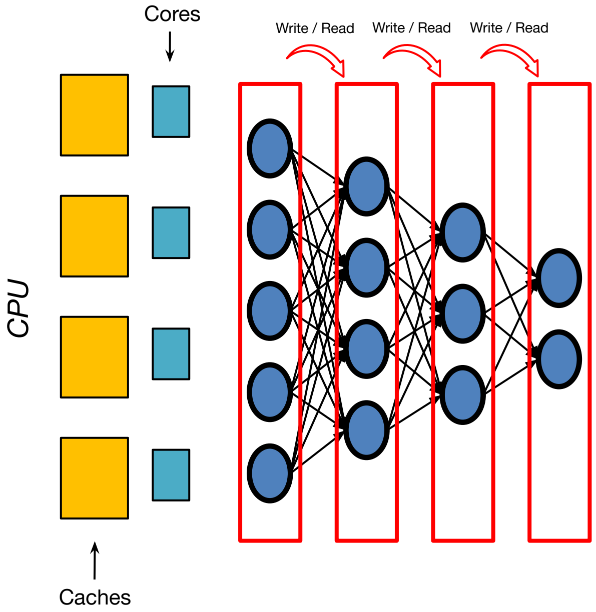
CPU vs. GPU Compute

HARDWARE ACCELERATORS



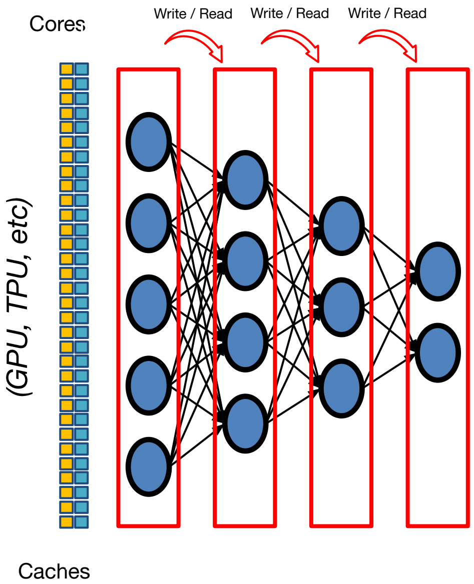
Execute synchronously layer by layer

CPU ALONE



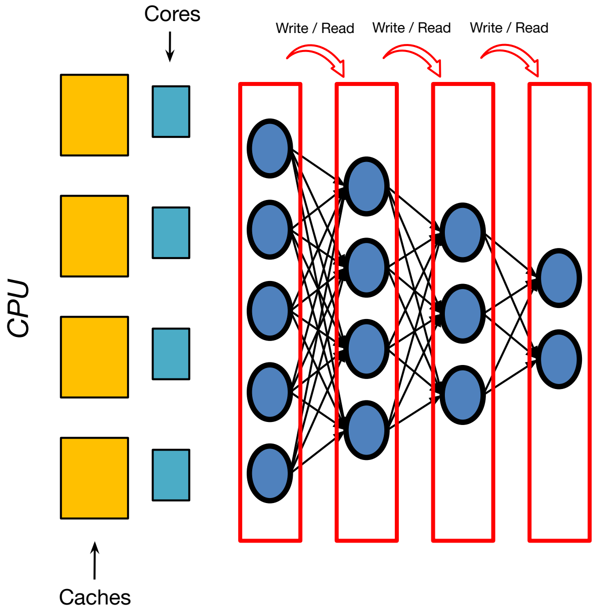
CPU vs. GPU Compute

HARDWARE ACCELERATORS



Execute synchronously layer by layer

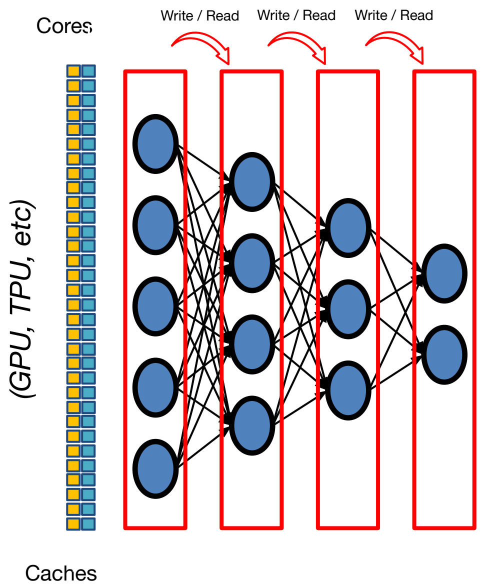
CPU ALONE



Works poorly!

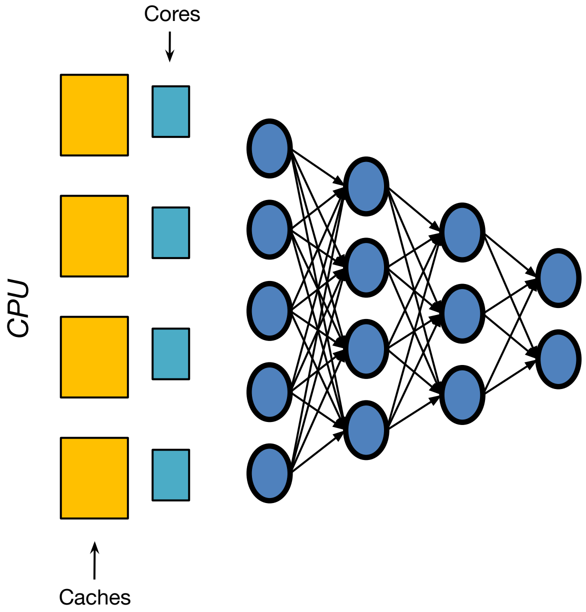
CPU vs. GPU Compute

HARDWARE ACCELERATORS



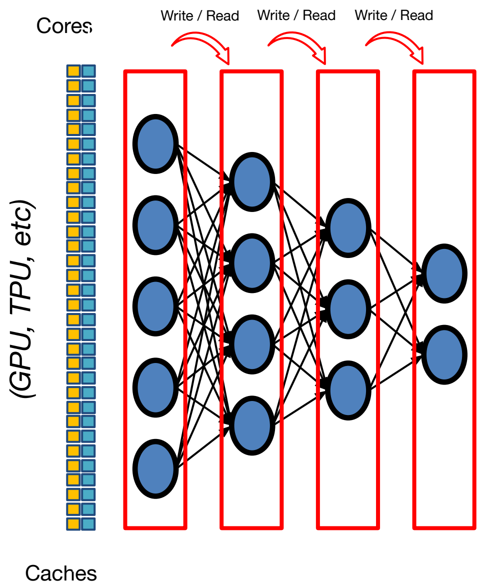
Execute synchronously layer by layer

CPU ALONE



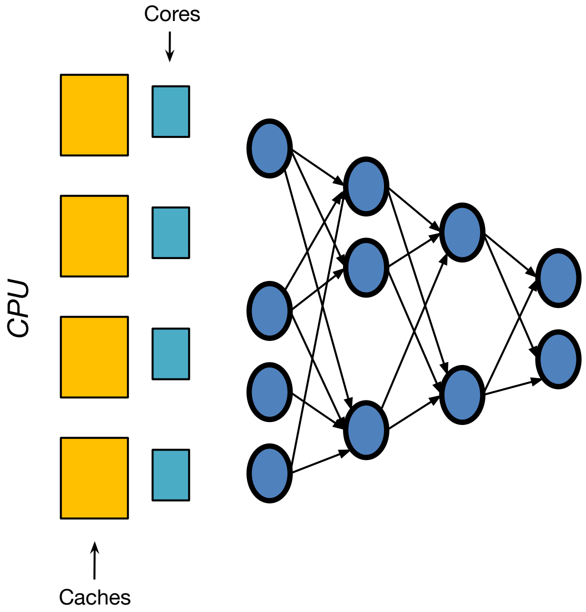
CPU vs. GPU Compute

HARDWARE ACCELERATORS



Execute synchronously layer by layer

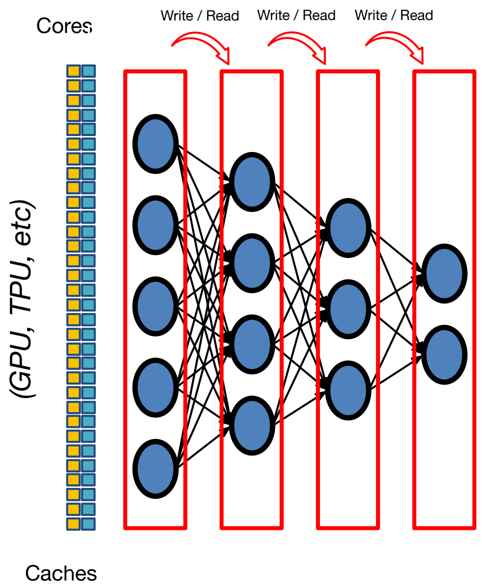
CPU ALONE



Prune the network to reduce compute

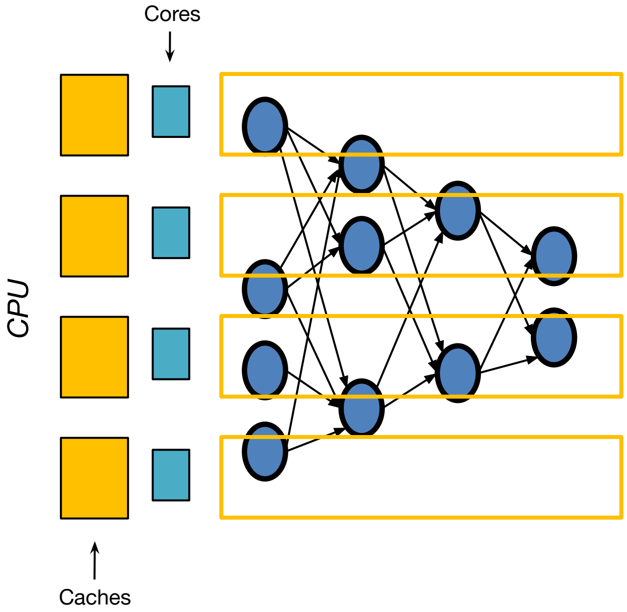
CPU vs. GPU Compute

HARDWARE ACCELERATORS



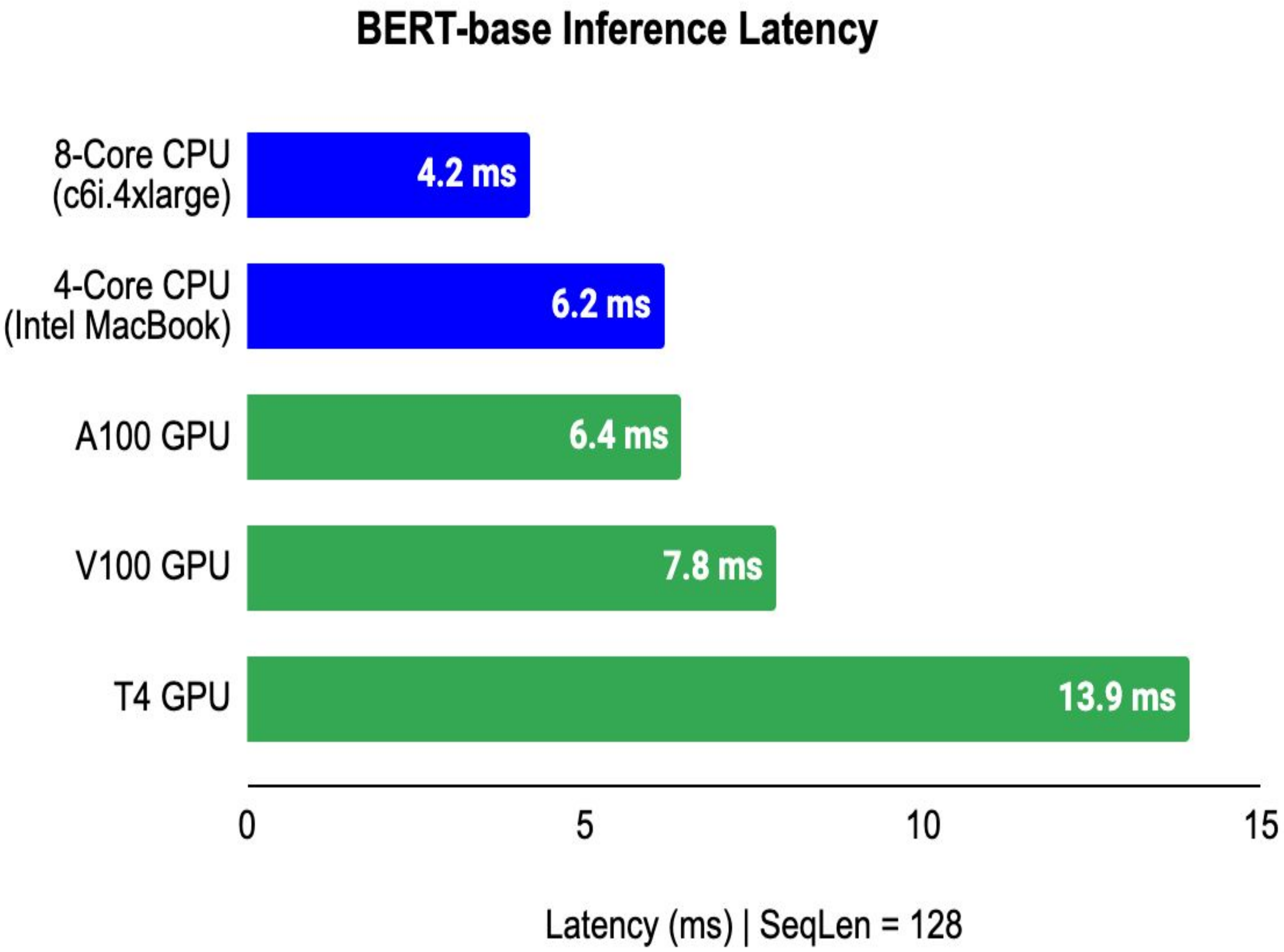
Execute synchronously layer by layer

CPU ALONE



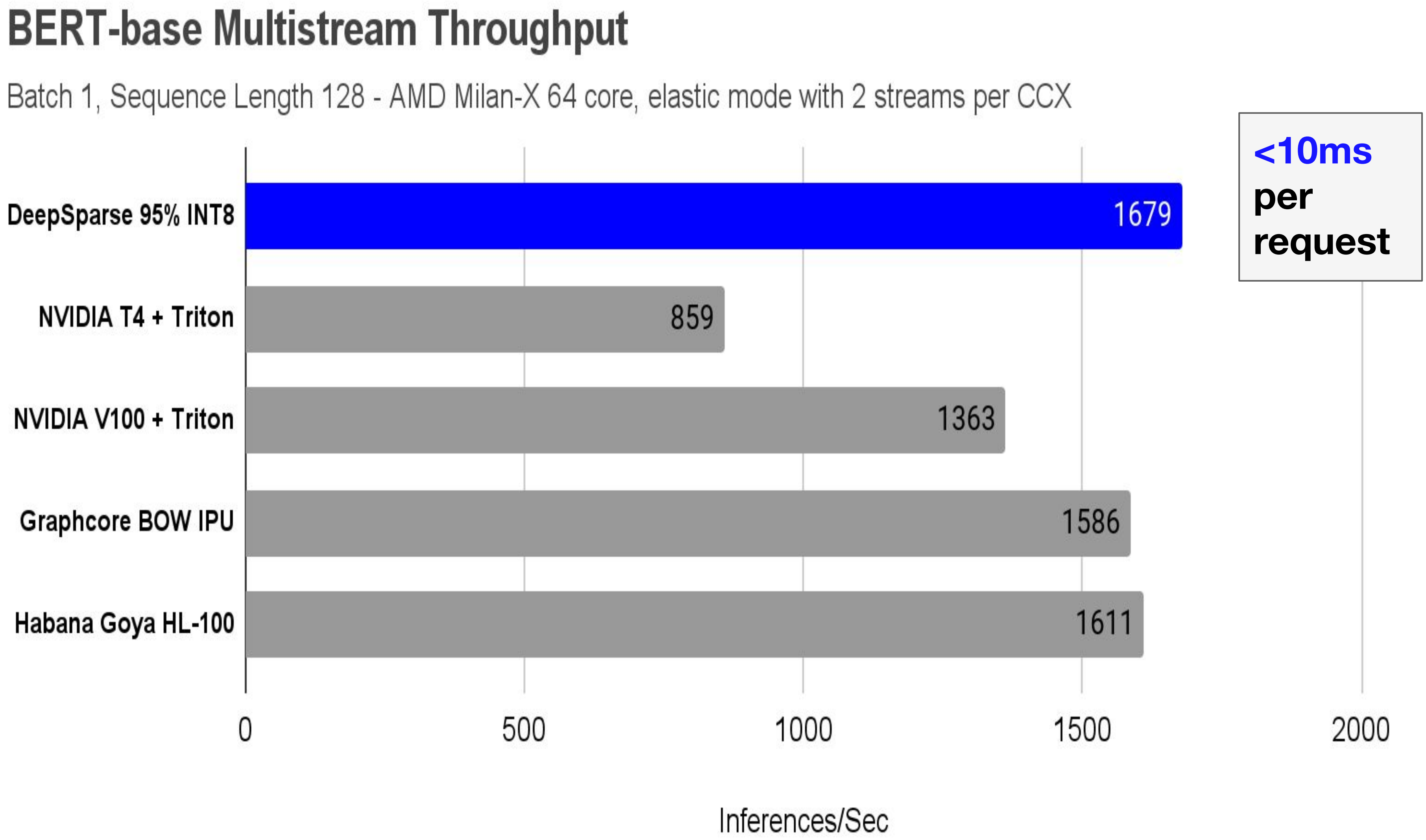
Execute asycnhronously in cache

CPU with Sparsity & Caching Aware Runtime = GPU



* CPU running 95% sparse model on Neural Magic DeepSparse^{TM} Runtime Software

And also at Batch=1 throughput



BERT-base Multistream Throughput

Batch 1, Sequence Length 128 - AMD Milan-X 64 core, elastic mode with 2 streams per CCX

<10ms per request

* CPU running 95% sparse model on Neural Magic DeepSparse^{TM} Runtime Software

The Future of Neural Hardware

Big question is, as models grow, and our ML algorithms better mimic brains, will we need special hardware, or will it suffice to just add specialized ML support operations into existing CPU hardware…

Thank You