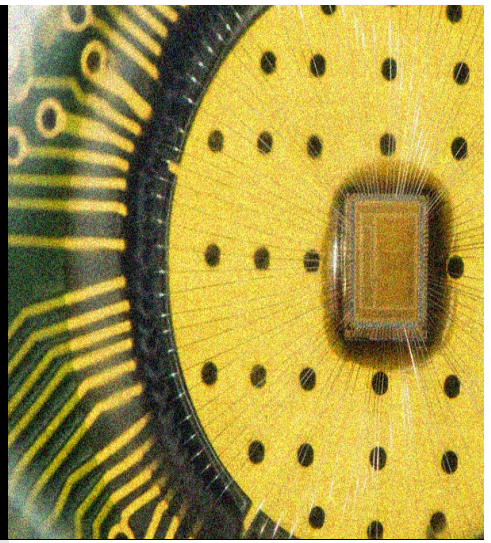
Introduction to High Performance Machine Learning

Lecture 1 09/04/24

Meet the Instructors



[Dr. Parijat Dube](#)

Senior Research Scientist,
IBM Research, NY

pd2216@nyu.edu

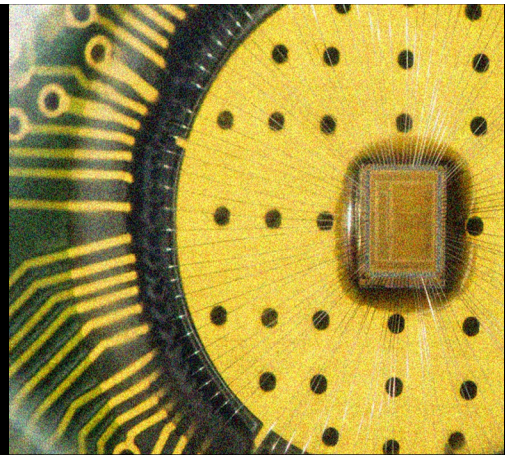
[Dr. Zehra Sura](#)

Team Lead for Recommendation Systems,

Bloomberg Law/Tax/Gov, NY

ss20018@nyu.edu

Meet the Teaching Assistants



Divya Agarwal

MS in Computer Engineering @ NYU Tandon

Dhanesh Baalaji Srinivasan

MS in Computer Engineering @ NYU Tandon

Varuni Buereddy

MS in Computer Engineering @ NYU Tandon

Yugesh Panta

MS in Computer Engineering @ NYU Tandon

Class Introduction

- Room: 238 Thompson St (GCASL) Room 261 Loc: Washington Square

- Brightspace: https://brightspace.nyu.edu/d2l/home/404117
- All information about the class will be available on Brightspace, including the syllabus, lecture slides, announcements, and assignments.

- Communication platform: Campuswire
  - Join the class Campuswire using this link: https://campuswire.com/p/G3C351A59
  - Code: 3698

Prerequisites

- General Knowledge of computer architecture
- C/C++: intermediate programming skills
- Python: intermediate programming skills.
- Understanding of Machine Learning concepts and Neural Networks architectures and algorithms

Today's Agenda

- Course Overview
  - Motivation
  - Goals
  - Organization
  - Topics
- HPC Technology Overview
- Generative AI evolution
- System Challenges with Generative AI

Course Motivation

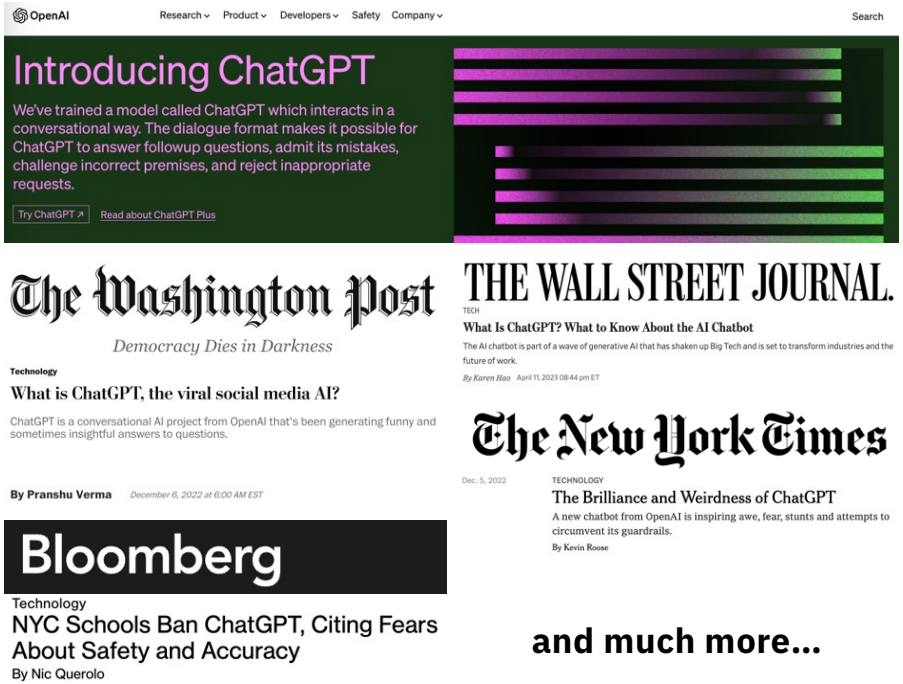
AI everywhere

HPML- Dube & Sura

Data from April 2017

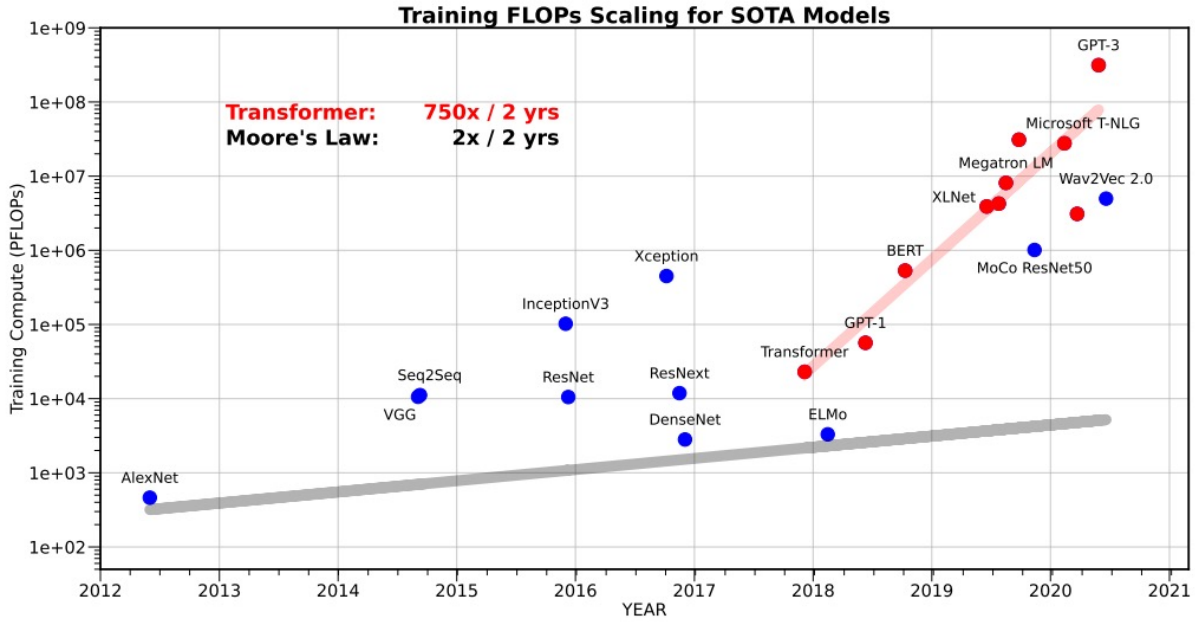
LLMs Are Capturing the Imagination…

- **Large language models + Generative AI making headlines**
  - Sparked by OpenAI's November '22 release of ChatGPT
- **Promise of AI is becoming mainstream**
  - Not just in technical circles!
  - From enterprises to social media to…student homework?



and much more…

Compute Requirements for AI Continue to Rise

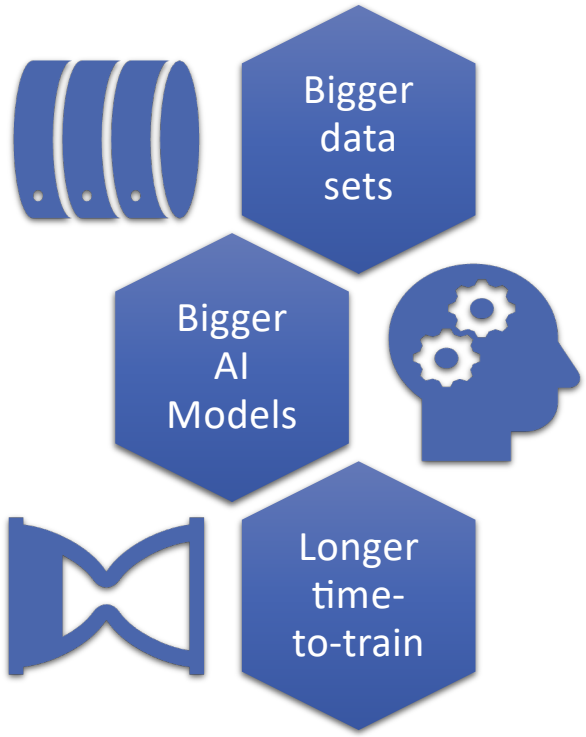


Reference: Gholami et al. AI and Memory Wall. March 2024

DL Training is a "Big Data" Problem

- Accuracy and time-to-train are all that matter when training
- Scalability is a requirement as neural network training is a "big data" problem

Go to Solution: Distributed DL training across multiple processors and nodes

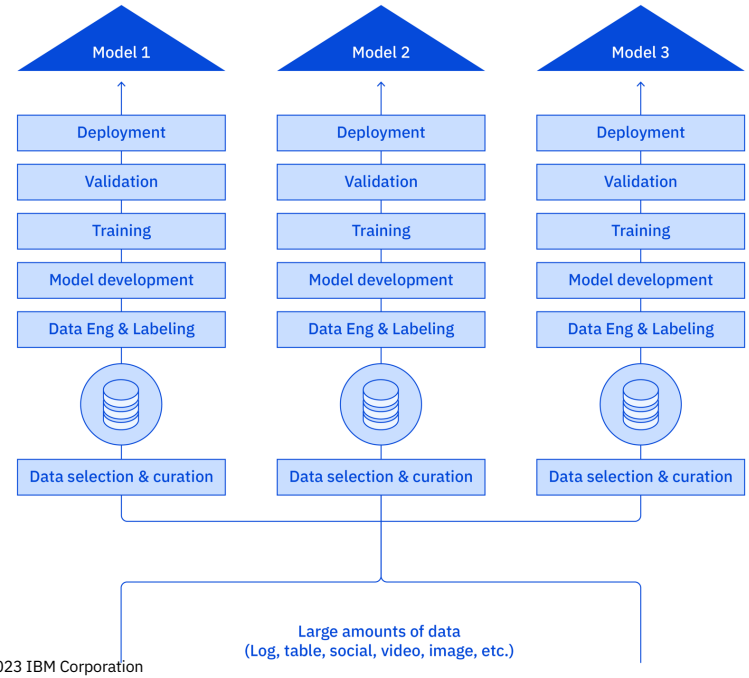


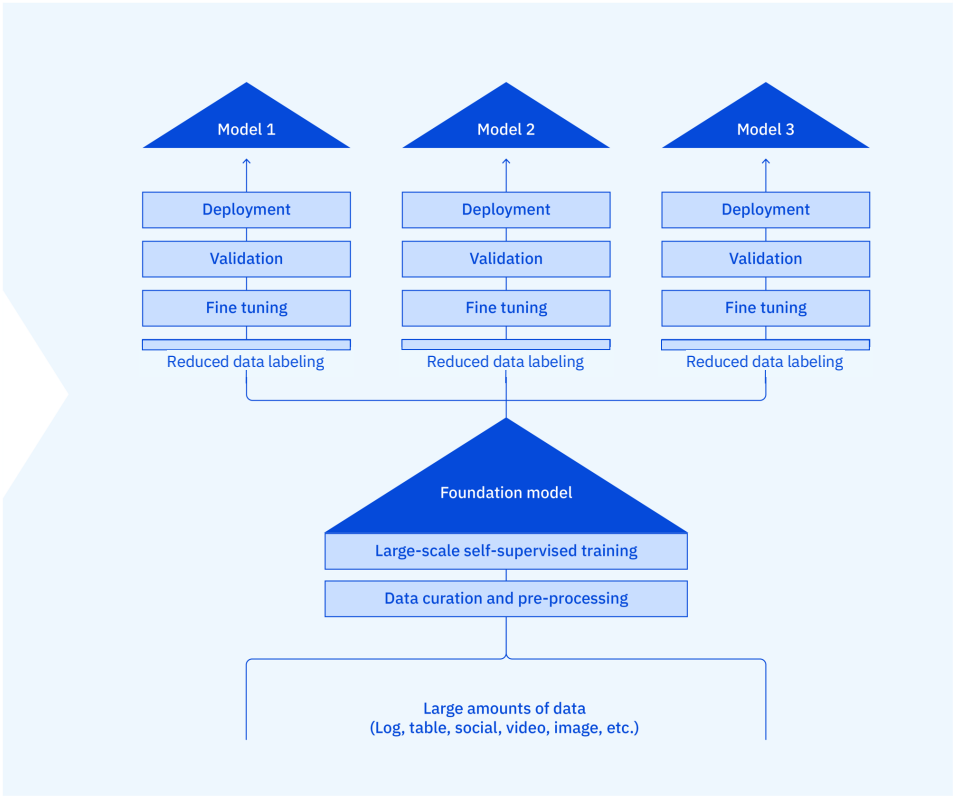
Foundation models: Shift to model adaptation and inference optimization

- Fine-tuning or prompt engineering a pretrained foundation model
- Real time inference is challenging
  - Resource (compute and memory) requirements are prohibitive for deployment on edge devices
- Inference optimization

"Foundation models are incredibly powerful and are ushering in a new age of generative AI, But to generate meaningful business outcomes, they need to be trained on high-quality data and develop domain expertise. And that's why an organization's proprietary data is the key differentiator when it comes to AI."
Shannon Miller, IBM Consulting

Foundation models are becoming an essential ingredient of a new AI workflow.





Supercomputing and Deep Learning: A perfect Match

"This is why around 2008 my group at Stanford started advocating shifting deep learning to GPUs (this was really controversial at that time; but now everyone does it); and I'm now advocating shifting to HPC (High Performance Computing/Supercomputing) tactics for scaling up deep learning. Machine learning should embrace HPC. These methods will make researchers more efficient and help accelerate the progress of our whole field."
– Andrew Ng, 2016

Extreme Scale: High Performance Computing

- **Supercomputers** are built for Extreme Scalability
- New Supercomputer cost: > $200M
- Fastest Supercomputer : 1.206 exaFLOPS
  - 2024: The Frontier Supercomputer
  - 1 EF = 10^18 FLOPS
  - FLOPS: floating point operations per second

- Scientific Simulation: 3rd scientific research paradigm
  - Magnetic Fusion
  - Nuclear Energy
  - Wind Energy
  - Cosmology
  - Astrophysics
  - …

Frontier — A supercomputer at the Department of Energy's Oak Ridge National Laboratory (ORNL)

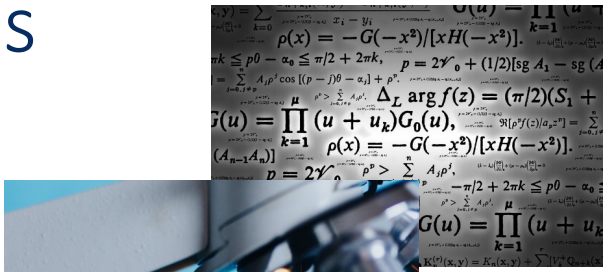
HPC and Scientific Paradigms

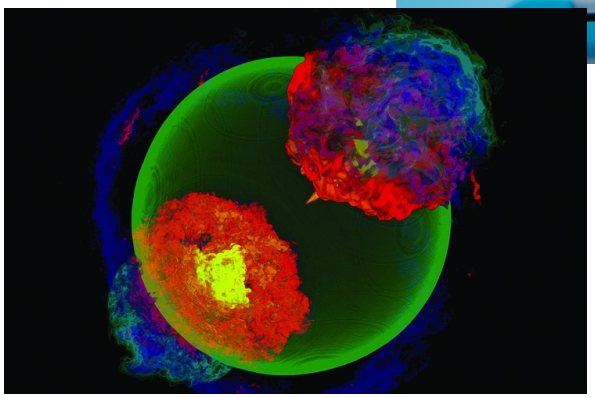
1. Theory (mathematics)

2. Experimentation (empiricism)

3. Simulation

[4. Machine Learning] ?

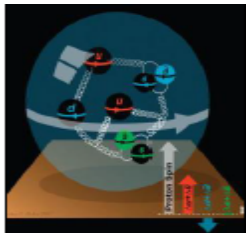


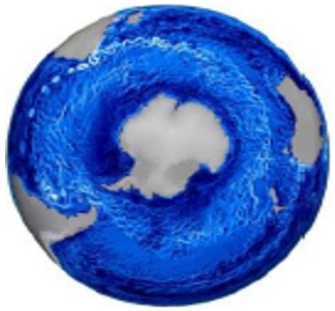


Scientific Simulation Examples

Standard Model:

Quantum Chromodynamic (QCD)-based elucidation of fundamental laws of nature: Standard Model validation and beyond



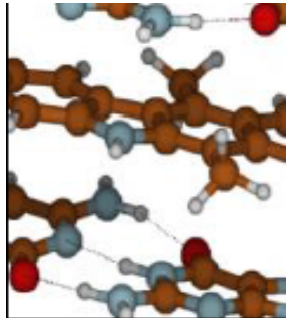


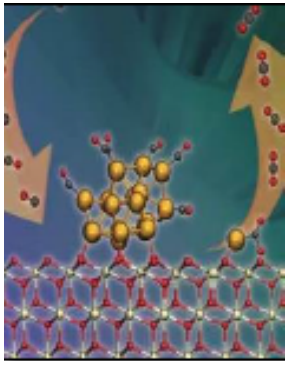
Climate:

Accurate regional impact assessment of climate change

Materials Science:

Find predict and control materials and properties



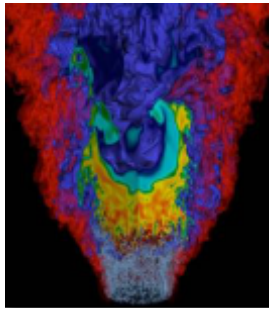


Chemical Science:

Study biofuel catalysis; protein folding

Combustion:

Design high efficiency, low emission, combustion engines and gas turbines



Traditional HPC vs. Machine Learning

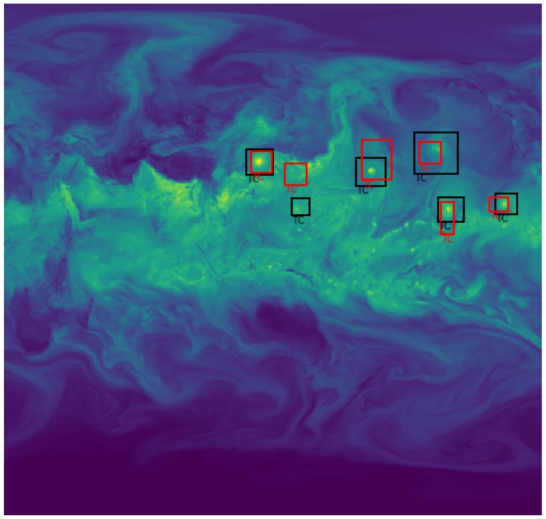
	Traditional HPC	Machine Learning
Application	Scientific and Industrial Research Scientific Modeling/Simulations	Consumer products: recognition/classification/prediction Industry: modeling/optimization
Software Environment	Custom; Low-level; Complex;	Wide-adoption; user-friendly;
Deployment	Large and very expensive Supercomputers	Cloud; Small Clusters; Single Workstations
Computation demands	Intense floating-point matrix/vector ops	same
Data demands	Tera-byte to Petabytes	same
Communication demands	Low-latency – High Bandwidth	same

HPC and Machine Learning

- Machine Learning for HPC Applications
  - Improve Scientific Simulations and other Applications with ML algos
  - Improve Software Stack using ML algorithms
    - Scheduling
    - Memory allocation
    - Reliability
    - Runtime optimization
- **HPC for Machine Learning - this course**
  - Execute ML training and inference on very large dataset (Scale)
  - Speedup and Scale ML with HPC techniques:
    - HPC Hardware
    - HPC software stack and Programming Models
    - Performance Optimization

HPC for Machine Learning

- Localizing and classifying extreme weather in climate data
- Semi-supervised bounding box regression algorithm (CNN)
- Executed on Cori at the National Energy Research Scientific Computing Center (NERSC)
  - Cray XC40 Supercomputer
  - ~9600 Xeon Phi nodes
  - 68 cores running at 1.4GHz on each node processor
  - 4 HyperThreads per core for a total of 272 threads per node
  - Cray Aries Network (low-latency, high bandwidth, dragonfly topology)
  - ~50PF peak
- 15PF peak performance
- Trained with 15TB climate dataset generated using climate simulation over 30 years
- 7205x faster than a single node



Results from plotting the network's most confident (>95%) box predictions on an image for integrated water vapor (TMQ) from the test set for the climate problem. Black bounding boxes show ground truth; Red boxes are predictions by the network.

Reference: "*Deep Learning at 15PF - Supervised and SemiSupervised Classification for Scientific Data" Kurth et al. -Supercomputing 2017*

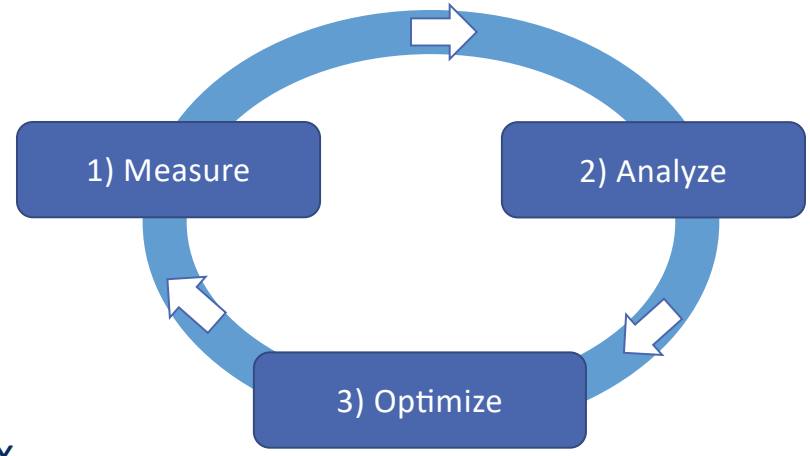
Goals of this course

- Use HPC techniques to find and solve performance bottlenecks
- Performance measurements and profiling of ML software
- Evaluate the performance of different ML software stacks and hardware systems
- High-performance distributed ML algorithms for Training
- Libraries like CuDNN, MKL
- CUDA and C++ to accelerate High-Performance ML/DL
- Efficient AI Techniques for Inference
  - Reduced precision, model compression, neural architecture search
- Efficient neural network architectures

Course Topics

Course Topic: Performance Optimization

- What does it mean?
  - System approach to performance
  - Complexity -> Methodology
  - Examples from real “life”
    - Optimizing applications
- Why is relevant?
  - Can be applied to every algorithm
  - Speedup sometimes can be very high 100x
  - Solving problems faster/Solving bigger problems



Course Topic: PyTorch Performance

- PyTorch is our **use case**
  - But also plain old C/C++ ☺
- PyTorch topics:
  - Deep learning using PyTorch
  - PyTorch multiprocessing
  - Performance aspects
  - PyTorch profiling

Course Topic: CUDA

- DL success really about GPUs!
- High Performance:
  - GPUs programming = CUDA
  - CPUs programming = Math libraries
- CUDA topics:
  - How to program
  - Performance

Course Topic: Distributed ML

- Challenges and opportunities
- Software and hardware for Distributed ML
- Distributed ML algorithms performance
- Distributed training using PyTorch:
  - Programming
  - Performance

Course Topic: Efficient DL Techniques

- Quantization, Sparsification, Knowledge Distillation
- Efficient transformers: KV caching and compression, Linear attention, flash attention
- Efficient fine-tuning for LLMs: PEFT, LORA, QLORA
- Efficient vision architectures: small filters, efficient convolution
- Neural architecture search (NAS) and hardware-aware NAS

Course Organization

Course Organization - Grading

- Quizzes
  - Based on lecture materials and assigned readings
- Homework Labs (programming assignments)
  - 5 Labs
  - Usually due in 2 weeks
  - Programming in Python/PyTorch/C/C++
- Final Project (Groups of 2)
- **Grading:** Homework (30%) + Final Project (30%) + Final Exam (30%) + Quizzes (10%)

Course Organization – Labs Rules

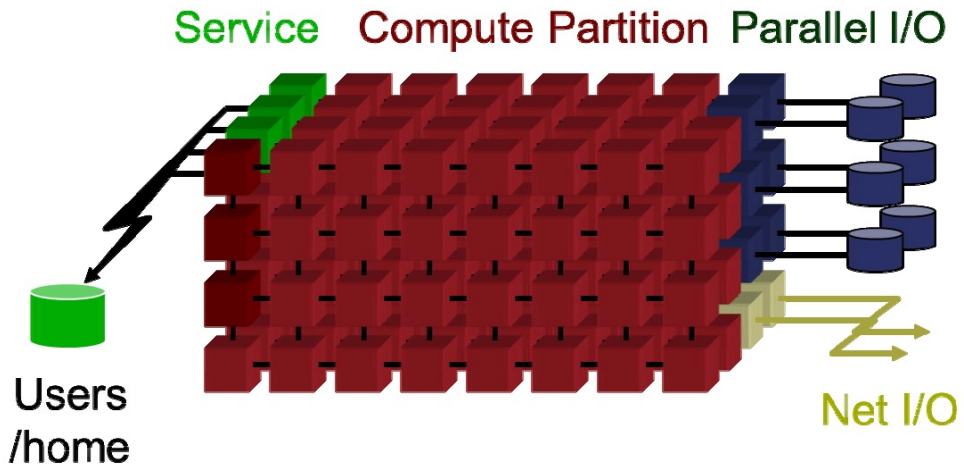
- **You must work alone on all labs**
- Questions:
  - We will be using Campuswire
  - You are encouraged to answer others' questions but refrain from explicitly giving away solutions. (counts towards your participation grade)
- Deadlines:
  - due at 11:59 pm on the due date

HPC Technology

HPC design principles

- Partition Model
- Network Topology
- Balance of Hardware Components
- Scalable System Software

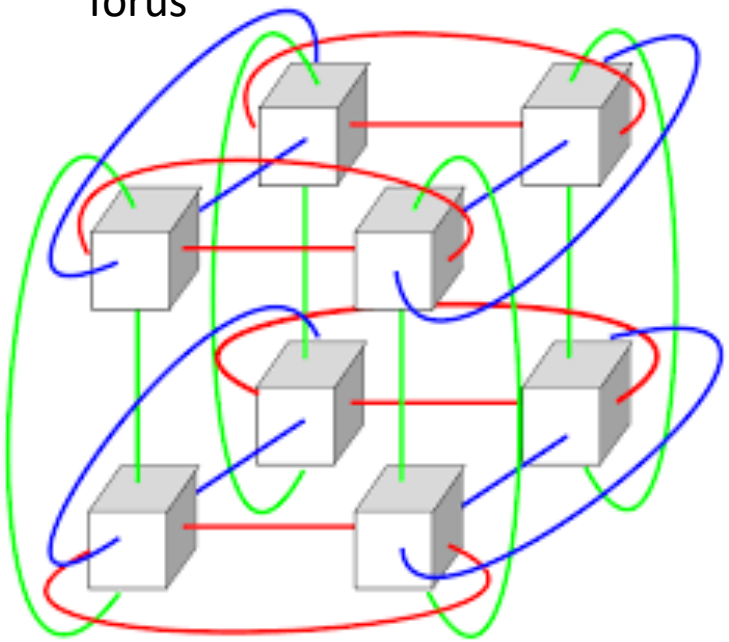
Partition model



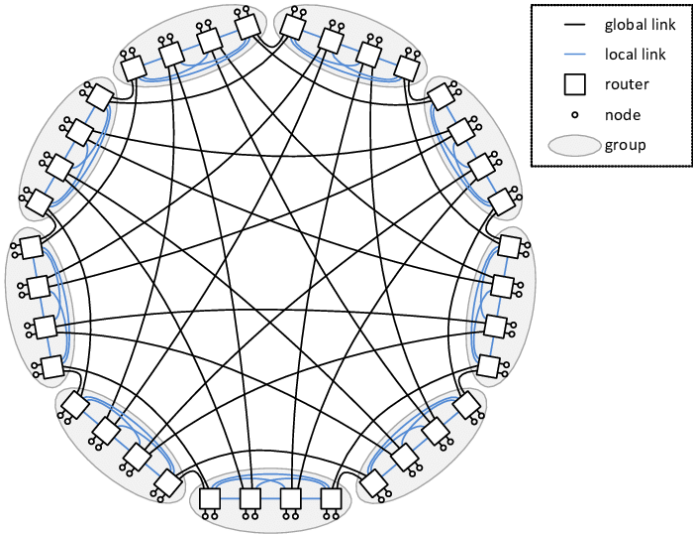
- Applies to both hardware and software
- Physically and logically divide the system into functional units
- Compute hardware different configuration than service & I/O
- Only run the necessary software to perform the function

Network Topology

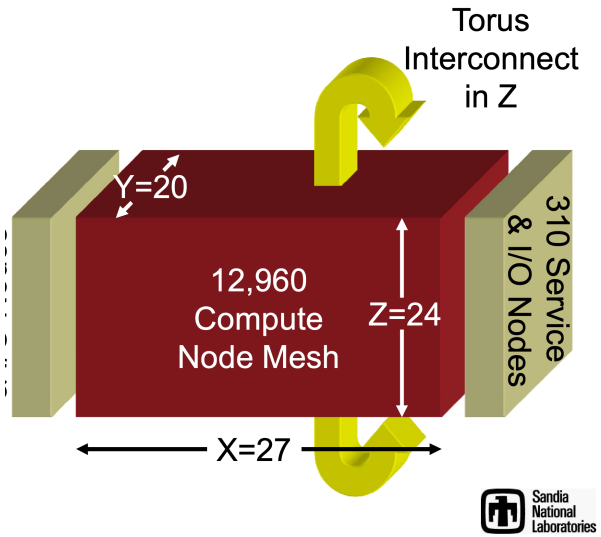
Torus

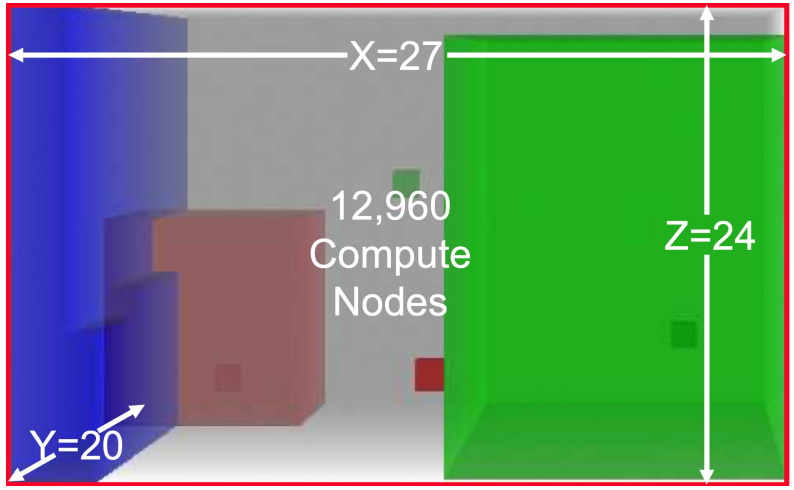


Dragonfly



Partitioning of Jobs





- Jobs occupy disjoint regions simultaneously
- Minimize communication interference

Scalable System Software

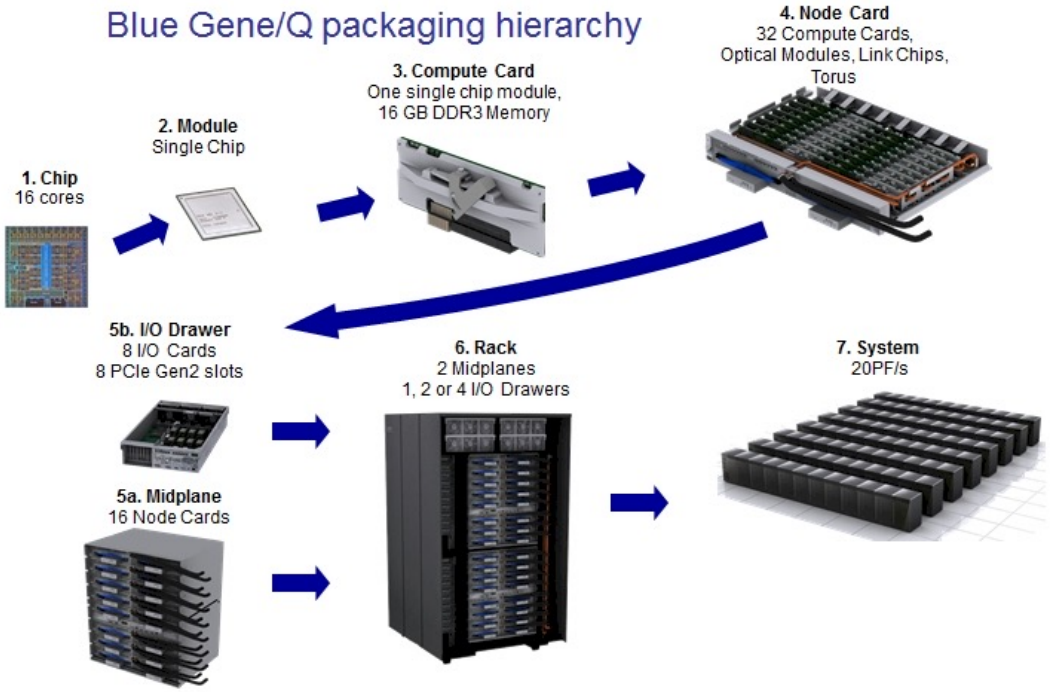
- Minimize compute node operating system overhead
- Non-invasive and out of band system monitoring
- Reduce OS interrupts by stripping down OS running on compute nodes
- Parallel File System GPFS

Key Properties of HPC architecture

- Speed
- Parallelism
- Efficiency
- Power
- Reliability
- Programmability

Dissection of a Supercomputer

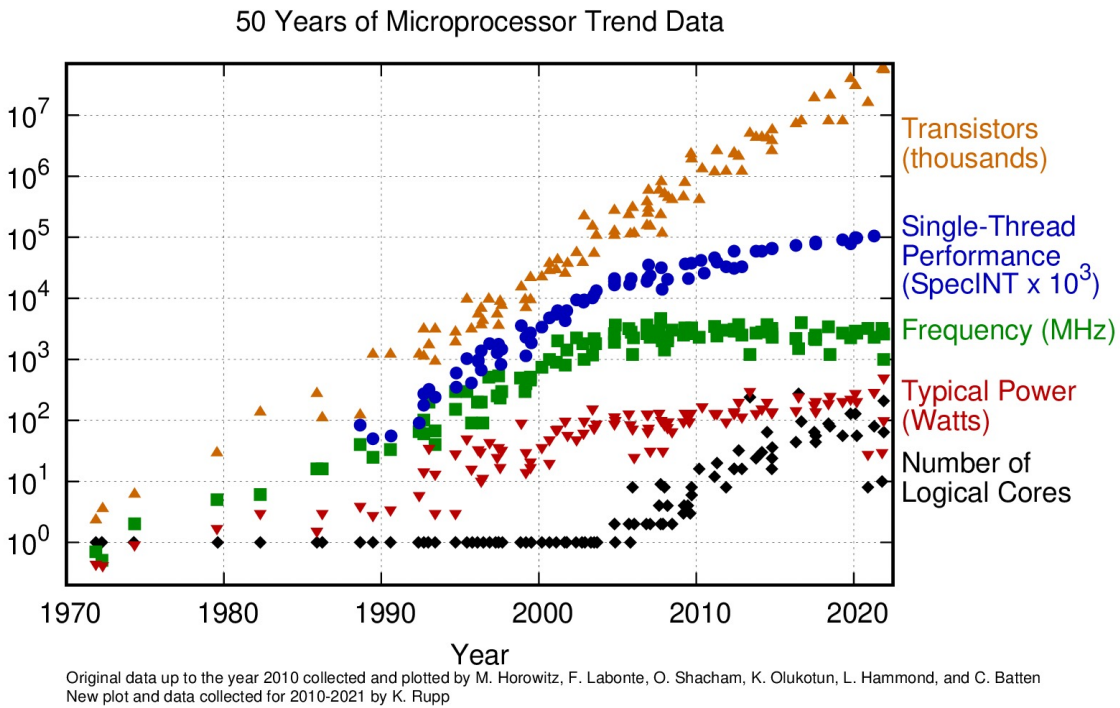
- Massive Parallelism
- Fast Floating Point
- Separate I/O and Compute
- High Performance Custom Network
- Power consumption of a small town
  - (10MW: more than 10,000 homes…)



Peripheral Component **Interconnect** (**PCI**) is a local computer bus for attaching hardware devices in a computer and is part of the **PCI** Local Bus standard.

Microprocessor Trends

- Moore's law
- Frequency (power wall)
- Single-core -> Multi-core -> GPUs

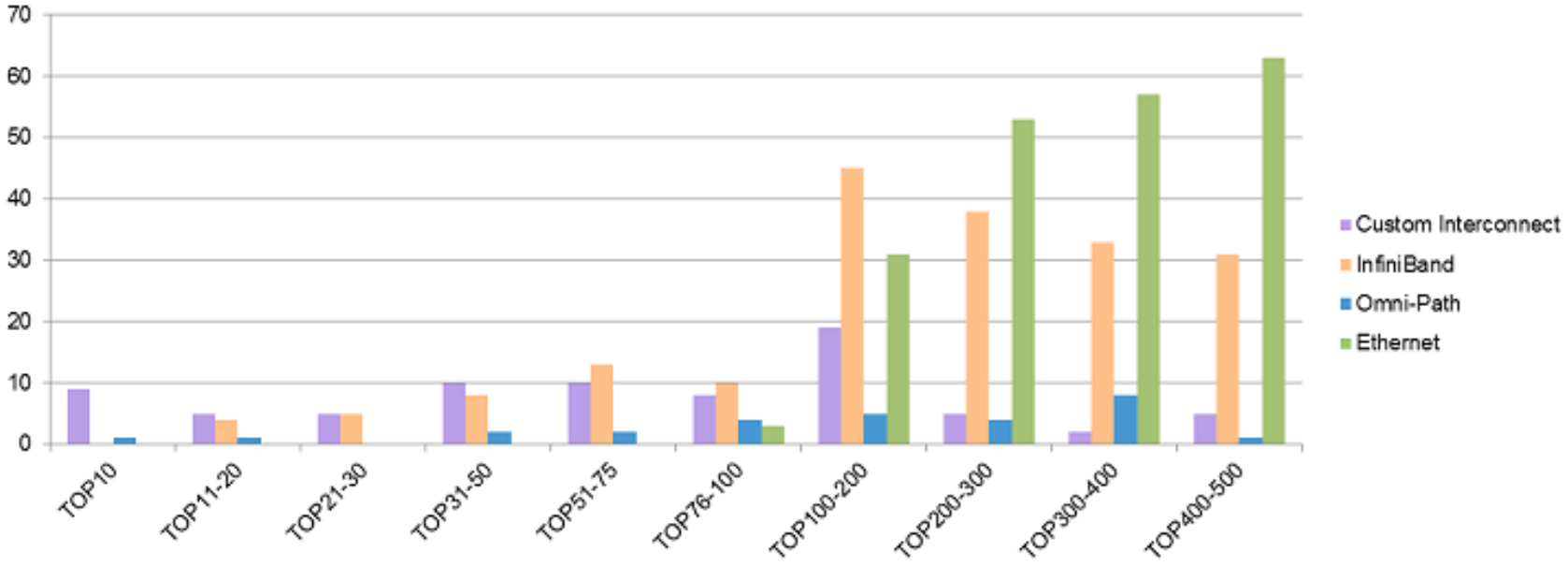


High Performance Networking (1)

- Large Scale Parallel and Deep Learning applications needs:
  - High Bandwidth
  - Low Latency
- Ethernet is not enough
- Infiniband (IB) is widely adopted
- Custom Networks are the best

Network technology	Bandwidth [Gb/s]	Latency [us]
10GigE	10	4
40GigE	40	4
IB EDR	100	1
NVLink	> 400	0.1-0.2

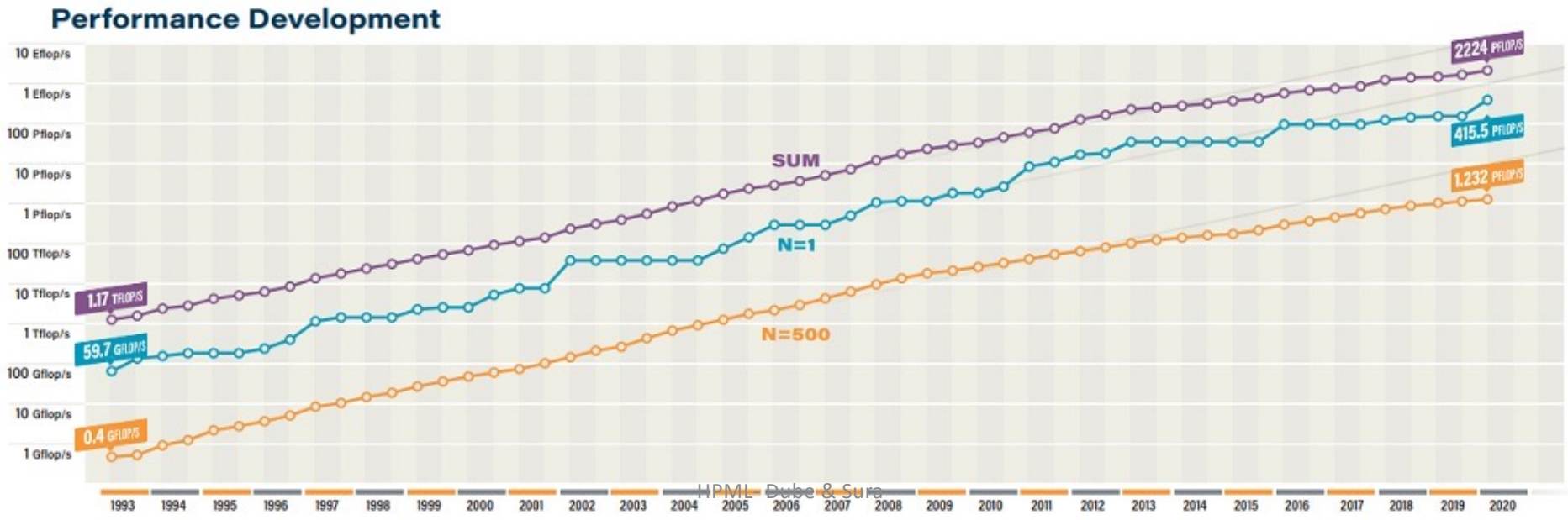
High Performance Networking (2)



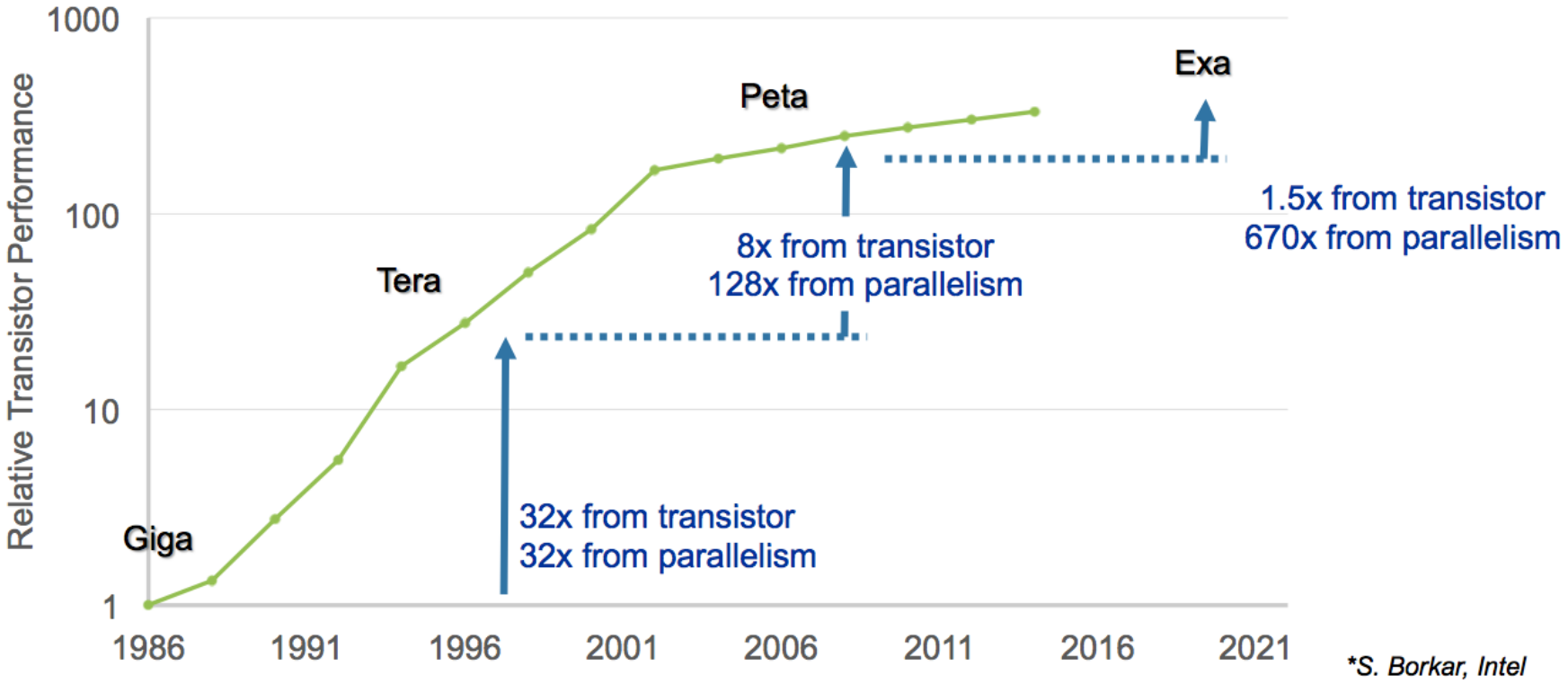
- HPC would not exist without high-bandwidth low-latency networks

Top500 Trends

- Linpack Benchmark
  - Dense linear algebra
- Exponential Performance Growth
- Announced twice a year



Performance Gain is Shifting



Top 5 positions of the TOP500 in June 2024

References:

- https://www.top500.org

Rank (previous)	Rmax Rpeak (PetaFLOPS)	Name	Model	CPU cores	Accelerator (e.g. GPU) cores	Total Cores (CPUs + Accelerators)	Interconnect	Manufacturer	Site country	Year	Operating system
1 —	1,206.00 1,714.81	Frontier	HPE Cray EX235a	561,664 (8,776 × 64-core Optimized 3rd Generation EPYC 64C @2.0 GHz)	36,992 × 220 AMD Instinct MI250X	8,699,904	Slingshot-11	HPE	Oak Ridge National Laboratory United States	2022	Linux (HPE Cray OS-SUSE)
2 ▲	1,012.00 1,980.01	Aurora	HPE Cray EX	1,104,896 (21,248 × 52-core Intel Xeon Max 9470 @2.4 GHz)	63,744 × 128 Intel Max 1550	9,264,128	Slingshot-11	HPE	Argonne National Laboratory United States	2023	Linux (HPE Cray OS-SUSE)
3 —	561.20 846.84	Eagle	Microsoft NDv5	172,800 (3,600 × 48-core Intel Xeon Platinum 8480C @2.0 GHz)	14,400 × 132 Nvidia Hopper H100	2,073,600	NVIDIA Infiniband NDR	Microsoft	Microsoft United States	2023	Linux (Ubuntu 22.04)
4 —	442.01 537.21	Fugaku	Supercomputer Fugaku	7,630,848 (158,976 × 48-core Fujitsu A64FX @2.2 GHz)	-	7,630,848	Tofu interconnect D	Fujitsu	Riken Center for Computational Science Japan	2020	Linux (RHEL)
5 —	379.70 531.51	LUMI	HPE Cray EX235a	186,624 (2,916 × 64-core Optimized 3rd Generation EPYC 64C @2.0 GHz)	11,664 × 220 AMD Instinct MI250X	2,752,704	Slingshot-11	HPE	EuroHPC JU European Union, Kajaani, Finland	2022	Linux (HPE Cray OS-SUSE)

Top 5 of the Green500 in June 2024

Rank	Performance per watt (GFLOPS/watt)	Name	Model Processors, GPU, Interconnect	Vendor	Site Country, year	Rmax (PFLOPS)
1	72.733	*JEDI*	**BullSequana XH3000** Grace Hopper Superchip (72C) 3GHz, Nvidia GH200 Superchip, Quad-Rail NVIDIA, InfiniBand NDR200,	ParTec[13]/EVIDEN (ex-Atos)	EuroHPC/FZJ, Germany, 2024	4.50
2	68.835	*Isambard-AI phase 1*	**HPE Cray EX254n** NVIDIA Grace 72C 3.1GHz, NVIDIA GH200 Superchip, Slingshot-11	Hewlett Packard Enterprise	University of Bristol, United Kingdom, 2024	7.42
3	66.948	*Helios GPU*	**HPE Cray EX254n** NVIDIA Grace 72C 3.1GHz, NVIDIA GH200 Superchip, Slingshot-11	Hewlett Packard Enterprise	Cyfronet, Poland, 2024	19.14
4	65.396	*Henri*	**Lenovo ThinkSystem SR670 V2** Intel Xeon Platinum 8362 2.8 GHz (32C), Nvidia H100 80 GB PCIe, InfiniBand HDR	Lenovo	Flatiron Institute, United States, 2022	2.88
5	64.381	*preAlps*	**HPE Cray EX254n** NVIDIA Grace 72C 3.1GHz, NVIDIA GH200 Superchip, Slingshot-11	Hewlett Packard Enterprise	Swiss National Supercomputing Centre (CSCS), Switzerland, 2024	15.47

IBM Summit – One of the Fastest in the World

- ~4600 nodes with 2 IBM POWER9™ CPUs and 6 NVIDIA Volta® GPUs
- CPUs and GPUs connected with high speed **NVLink**
- Large coherent memory: over 512 GB (HBM + DDR4)
- All memory directly addressable from the CPUs and GPUs
- Over 40 TF peak performance per node (> 150PF)
- Mellanox® EDR-IB full non-blocking fat-tree interconnect
- IBM Elastic Storage (GPFS™) - 1TB/s I/O and 120 PB disk capacity

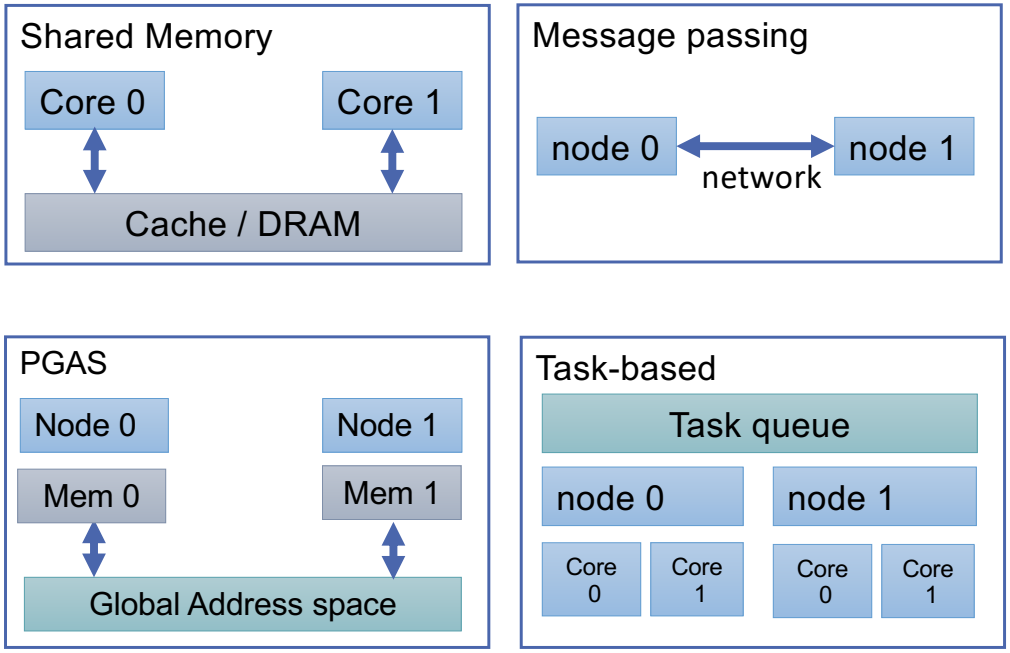
Source: IBM

Parallel Programming Models for HPC

- Developers = Domain experts/Scientists
- Node = 1 computer on the network

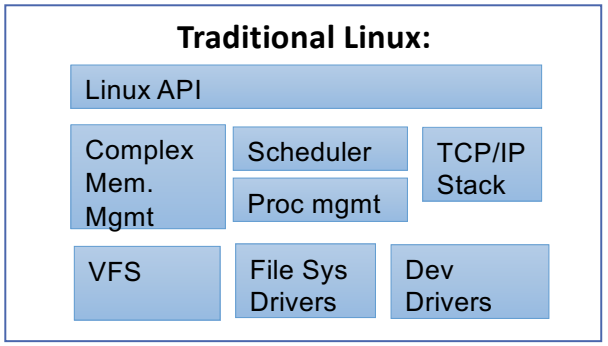
Programming models:

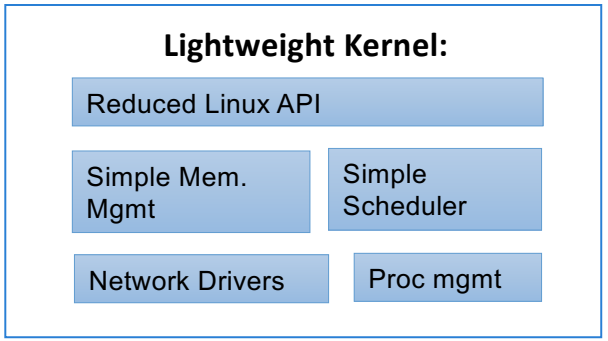
- Shared memory: OpenMP, Intel TBB (Thread building block)
- GPU Programming: CUDA, OpenMP, OpenACC (open accelerators)
- Off-load/Accelerator Approach (Copy Only Model)
- Message Passing: MPI
- Partitioned Global Address Space (PGAS)
- UPC/C++, SHMEM, CAF, X10, Chapel
- Task-based: Legion, CHARM++, HPX



Operating Systems for HPC

- Traditional Linux (Red Hat)
- Optimized Linux (Cray's Compute Node Linux)
- Lightweight Kernel (LWK, CNK)
- Hybrid: Linux + Lightweight kernel





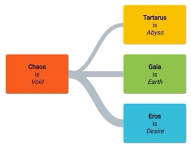
LLM Evolution

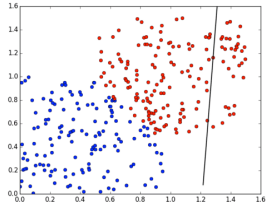
Story of AI is a story of data representations

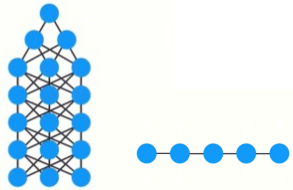
1980s | Big data

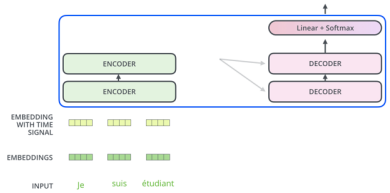
1980s to ~2010 | Massive labeled data + Compute

2017+ | Self-supervision at scale + Massive unlabeled data + Compute









Expert Systems

Hand-crafted symbolic representations

Machine Learning

Task-specific hand-crafted feature representations

Deep Learning

Task-specific learnt feature representations

Foundation Models

Generalizable & adaptable learnt representations

The paper that started it all!

The "Attention is All You Need" paper, which introduced the Transformer architecture, had a profound impact on the field of natural language processing (NLP) and deep learning in general:

- Efficiency in Training
- Parallelization
- Scalability
- State-of-the-Art Performance
- Transferability
- Pre-training and Transfer Learning
- Wider Adoption Beyond NLP
- Open-Source Implementations

Attention Is All You Need

Ashish Vaswani* Google Brain avaswani@google.com

Noam Shazeer* Google Brain noam@google.com

Niki Parmar* Google Research nikip@google.com

Jakob Uszkoreit* Google Research usz@google.com

Llion Jones* Google Research llion@google.com

Aidan N. Gomez* † University of Toronto aidan@cs.toronto.edu

Łukasz Kaiser* Google Brain lukaszkaiser@google.com

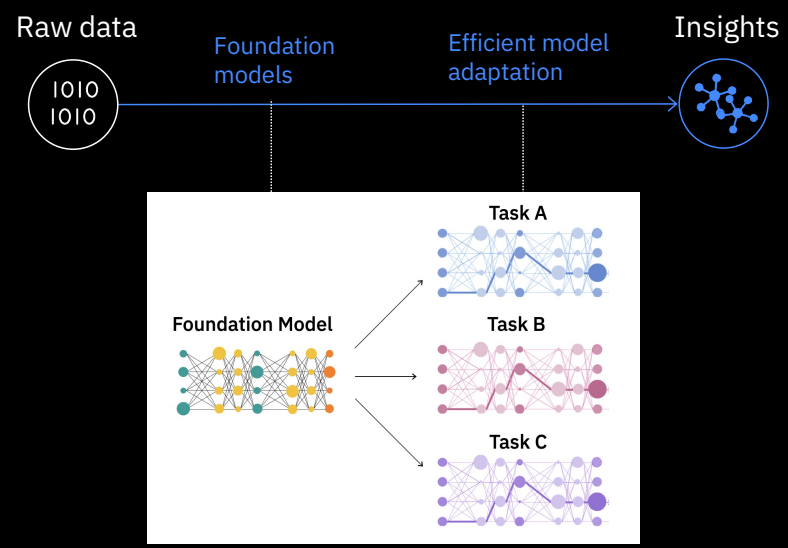
Illia Polosukhin* ‡ illia.polosukhin@gmail.com

Abstract

The dominant sequence transduction models are based on complex recurrent or convolutional neural networks that include an encoder and a decoder. The best performing models also connect the encoder and decoder through an attention mechanism. We propose a new simple network architecture, the Transformer, based solely on attention mechanisms, dispensing with recurrence and convolutions entirely. Experiments on two machine translation tasks show these models to be superior in quality while being more parallelizable and requiring significantly less time to train. Our model achieves 28.4 BLEU on the WMT 2014 English-to-German translation task, improving over the existing best results, including ensembles, by over 2 BLEU. On the WMT 2014 English-to-French translation task, our model establishes a new single-model state-of-the-art BLEU score of 41.8 after training for 3.5 days on eight GPUs, a small fraction of the training costs of the best models from the literature. We show that the Transformer generalizes well to other tasks by applying it successfully to English constituency parsing both with large and limited training data.

Foundation models

An emerging era of adaptable models



A foundation model is a large artificial intelligence model trained on a vast quantity of unlabeled data at scale (usually by self-supervised learning) resulting in a model that can be adapted to a wide range of downstream tasks.

Google	Gemini 1.5 Pro
Meta	LLaMA 3, LLaMA 3.1
OpenAI	GPT-3, GPT-4
Technology Innovation Institute	Falcon 180B
IBM	Granite
Anthropic	Claude

The Need for Efficient & Sustainable AI

The Need for Efficient & Sustainable AI

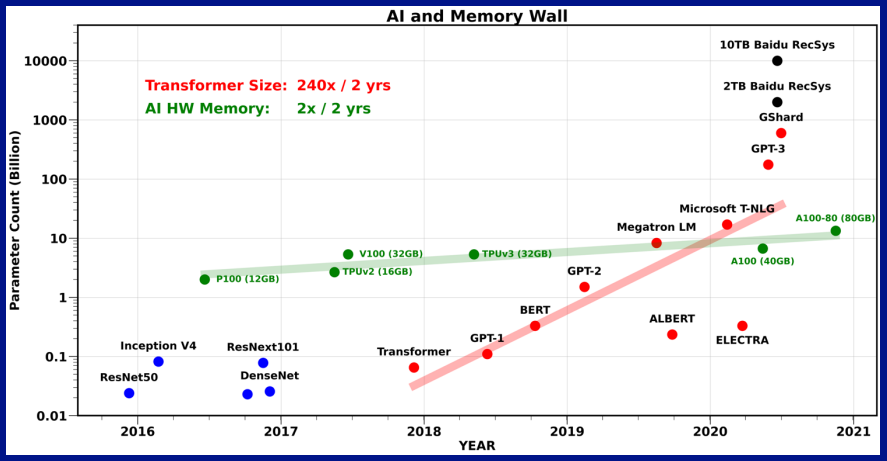
1

Increased Model Complexity

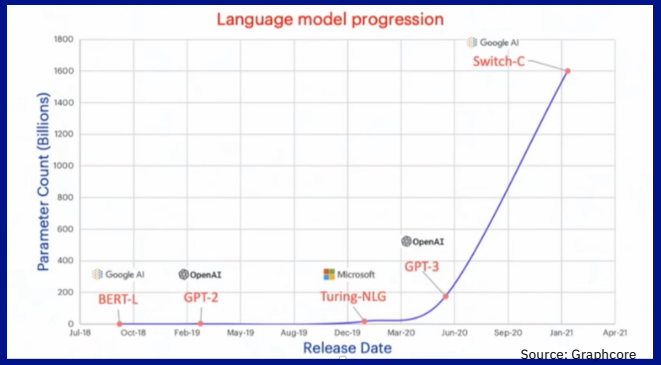
The number of parameters in neural networks models is increasing on the order of 10x year on year.

Increase of data volumes, sources, and richness

The Increasing Complexity of AI Models



https://medium.com/riselab/ai-and-memory-wall-2cb4265cb0b8/



Source: Graphcore

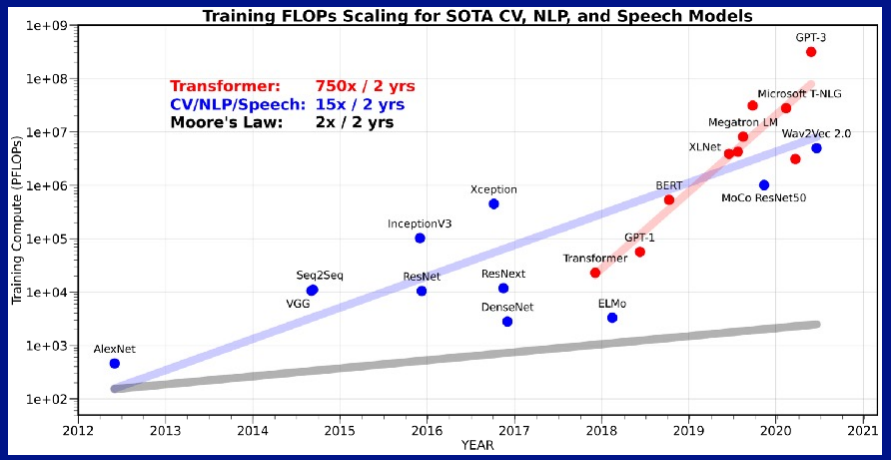
The Need for Efficient & Sustainable AI

2

Unbounded Computational Demands

Training compute requirements are doubling every 3.5 months[1]

The Accelerating Computational Demands



Reference: Gholami et al. AI and Memory Wall. March 2024

The Need for Efficient & Sustainable AI

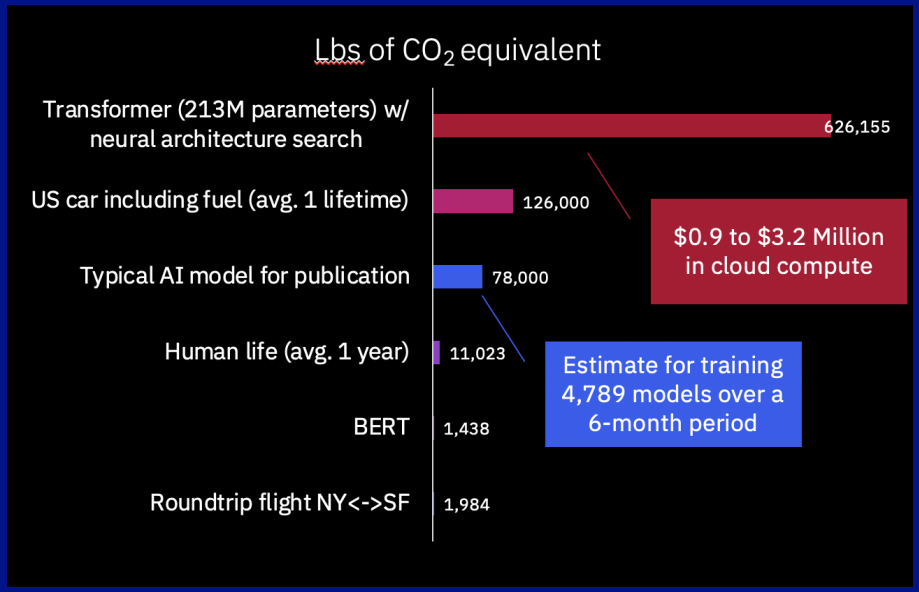
3

Increasing Carbon Footprint

Ever increasing carbon footprint and cost

Training a single model can emit as much as carbon as 5 cars in their lifetimes[2]

The Ever-increasing Carbon Footprint and Cost



Source: https://www.technologyreview.com/2019/06/06/239031/training-a-single-ai-model-can-emit-as-much-carbon-as-five-cars-in-their-lifetimes/

1. D. Amodei, D. Hernandez: https://blog.openai.com/ai-and-compute/

2: https://www.technologyreview.com/2019/06/06/239031/training-a-single-ai-model-can-emit-as-much-carbon-as-five-cars-in-their-lifetimes/

The Need for Efficient & Sustainable AI

1 Increased Model Complexity

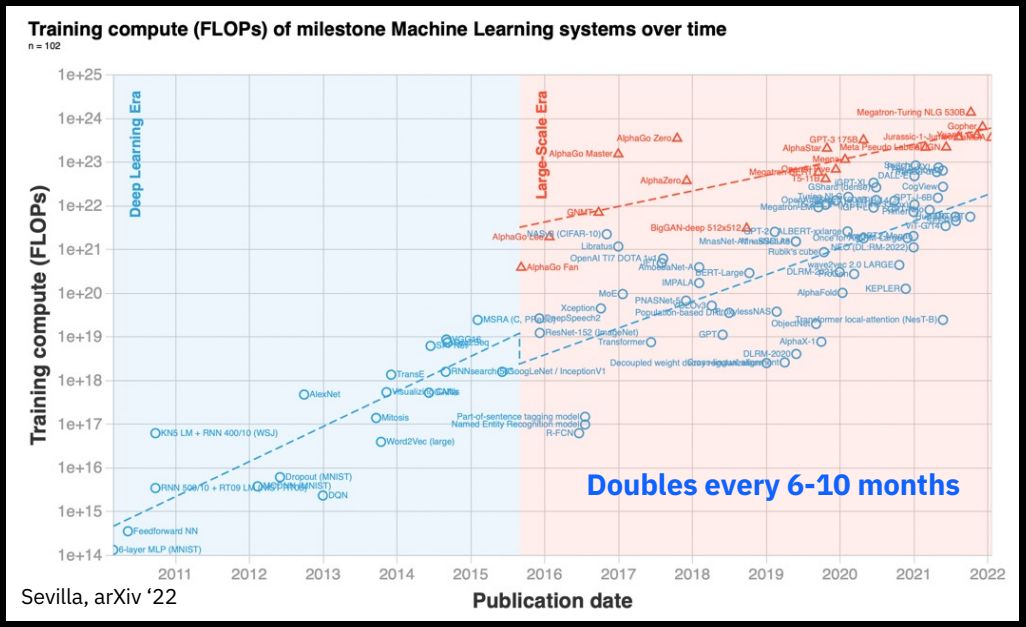
2 Unbounded Computational Demands

3 Increasing Carbon Footprint

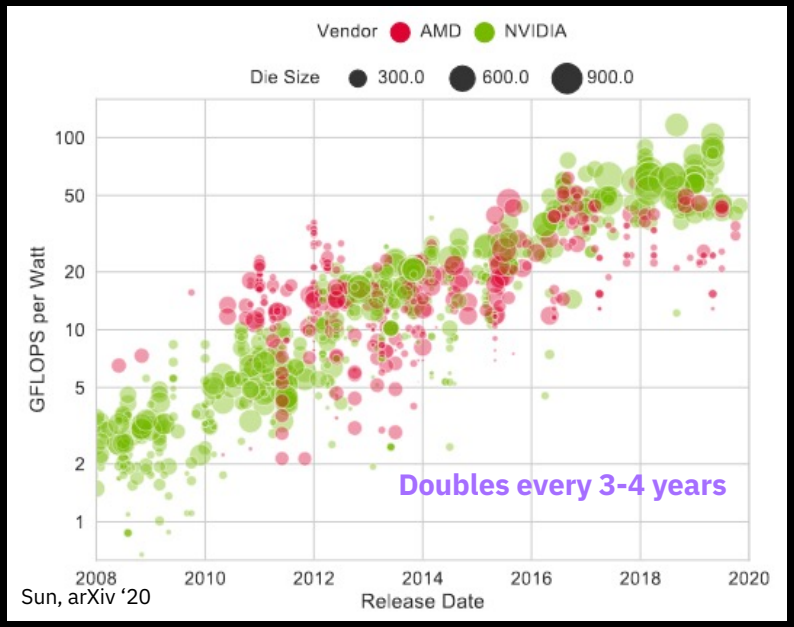
<u>Larger models need more hardware – business as usual is not sustainable</u>

- **175 Billion parameters**
- **OpenAI's GPT-3 supercomputer: 285,000 CPUs and 10,000 GPUs**
- **Open AI estimated spend of 4-12 M$ on cloud compute to train GPT-3**
- **310,000 ExaFLOP for training consume ~552.1 tons CO_2 equivalent, ~ 3 jet** (not: passenger) **round trips NY ⇔ SF**
- **"By the time they found some mistakes with GPT-3, they had already spent too much money and did not have the budget to rerun without the bugs."**

AI Compute Growth & Technology Scaling



Sevilla, arXiv '22



Sun, arXiv '20

Compute Need **GPU Efficiency**

< 1yr doubling rate > 3yr

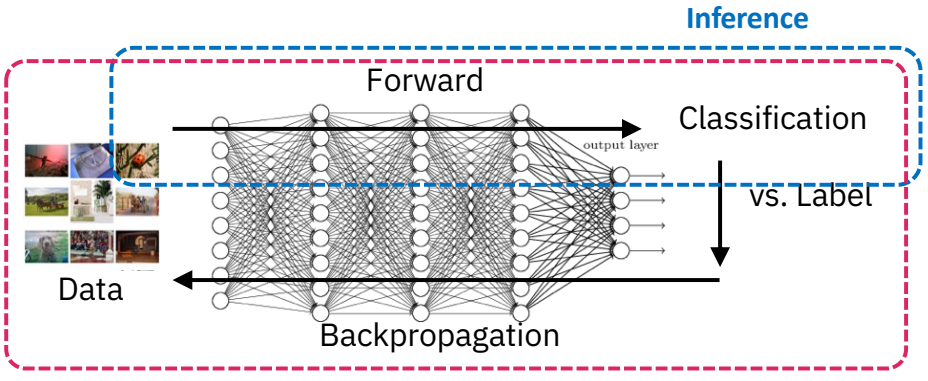
Technology Scaling Alone is Necessary but Not Sufficient

New Workloads Require New Ways to Compute

AI Training vs. Inference Performance Demands

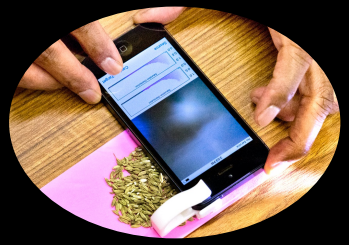
Systems optimized for inference and training may be quite different

- Large distributed clusters vs. single device
- Optimization for workload specifics (compute vs. BW, precision, metrics, ...)



	Inference	Training
Compute Phases	Forward	Forward + Backward + Update
Compute Precision	Lower	Higher
Batch size	Large or Small	Large
Performance	Latency + Throughput	Throughput
Memory footprint	Small(er)	Large (activations)
System	Down to single device	Distributed

Inference at the Edge



IoT

100 mW

(< few 10 GOps)

Single AI Core

Lower accuracy permissible

Mobile

250 mW to <2W

(< 100's of GOps)

Few AI Cores

Accuracy important

Automotive

20 - 50W

(10's – 100's of TOps)

Multiple AI Cores+
Custom Interconnect

No loss of accuracy is acceptable

For Inference, across different domains, TOp per Watt is the key metric
Larger Model => More Memory References => More Energy

Edge Intelligence on The Rise but with Many Challenges

Resource constrained devices

Deploying models on a fleet of devices is not easy

Challenges

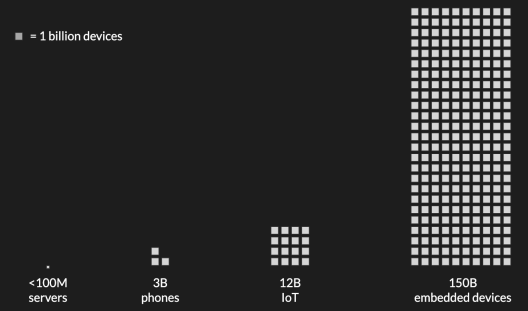
Connectivity is not stable and guaranteed

Privacy: data cannot leave the device in many cases

HPML- El Maghraoui

Need to shift from state-of-the-art accuracy to state-of-the-art efficiency

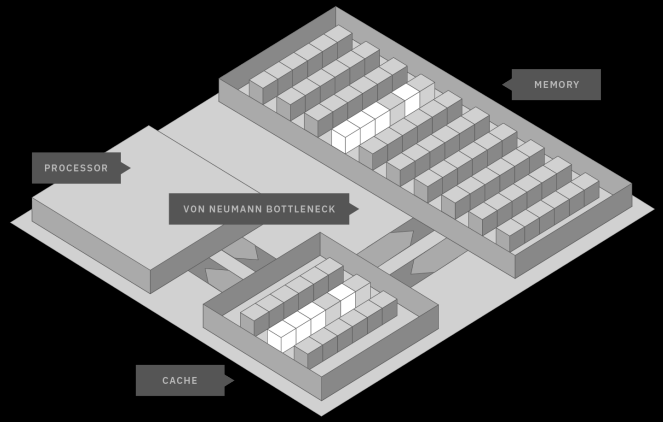
Most intelligence will be at the edge.



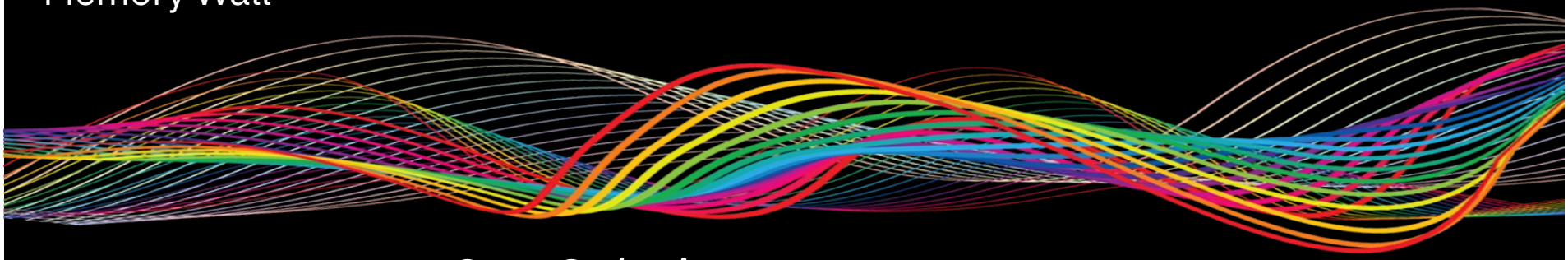
The Bottlenecks of AI Compute

Use of high-precision serial architecture

* memory access
* memory density
* energy consumption
* latency



Memory Wall

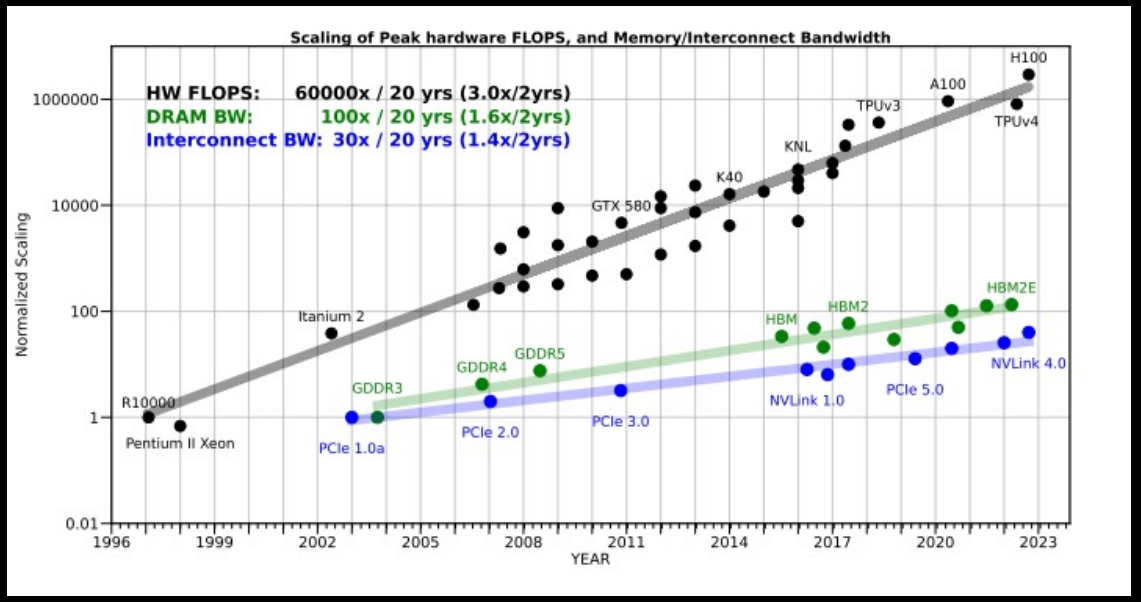


One Solution - *An approximate massively parallel pipeline*

Memory Wall

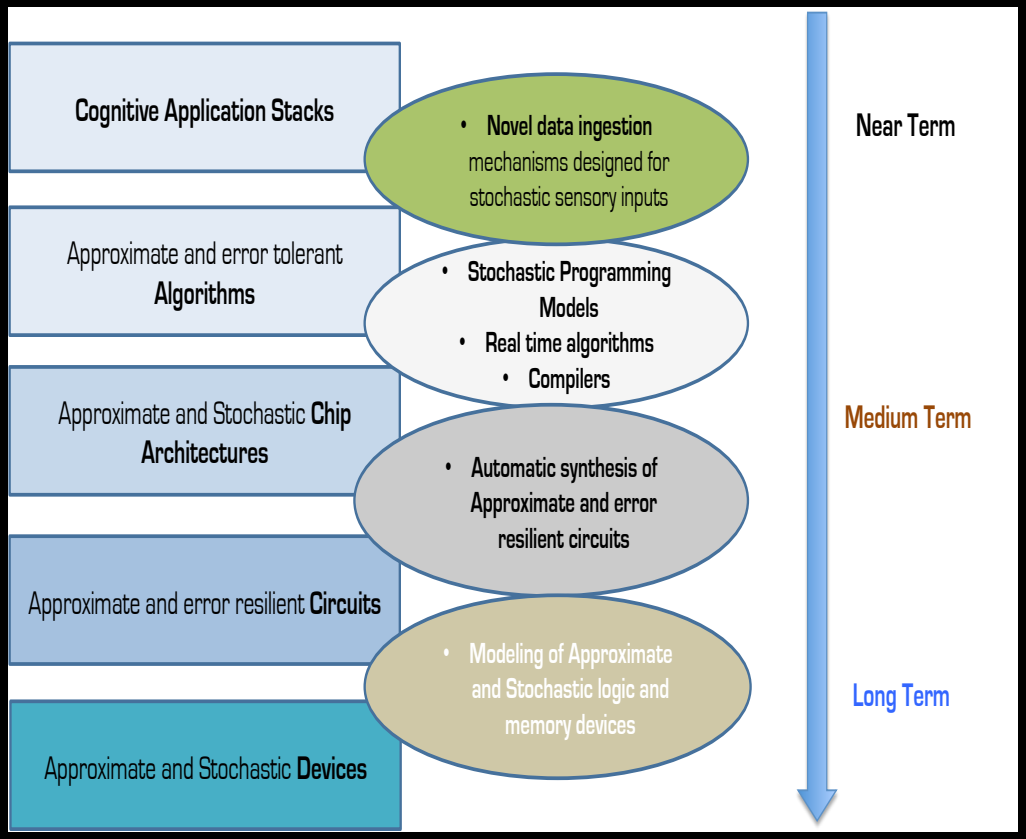
The time to complete an operation is dependent on how fast we can perform the arithmetic as well as how fast we can feed data to the arithmetic units of hardware.

The diverging speed of improvement of how fast computations can be performed versus how fast data can be fetched is going to create a “memory wall” issue.



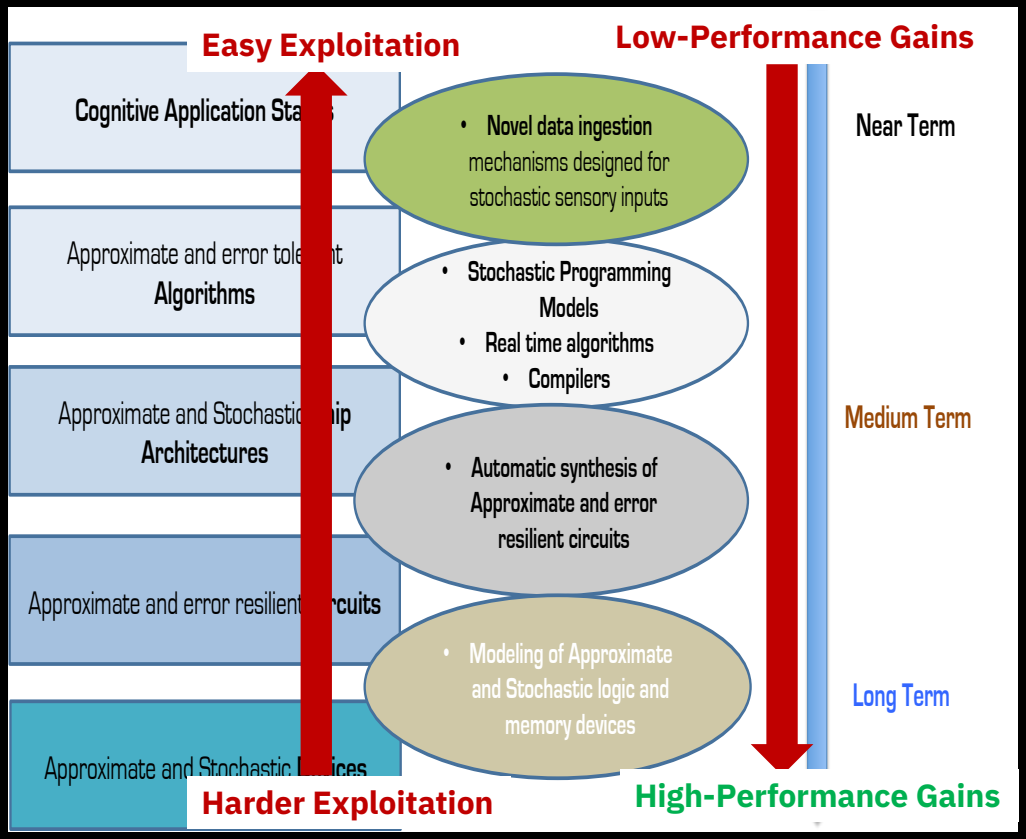
“Each is improving exponentially, but the exponent for microprocessors is substantially larger than that for DRAMs. The difference between diverging exponentials also grows exponentially”

Approximate Computing To Address These Challenges



- **Large spectrum** of cross-stack approximate computing techniques available.
- 3 Primary techniques (already) being used widely in DL
  - **Precision:**
    - **Scaled precision** for Training and Inference
    - Maximum bang for the buck (**quadratic gains in efficiency w. precision**)
  - **Compression:**
    - Lossy compression to minimize data communicated between ASICs for training.
  - **Synchronization:**
    - (Mostly) SW techniques to minimize synchronization overheads for distributed training.

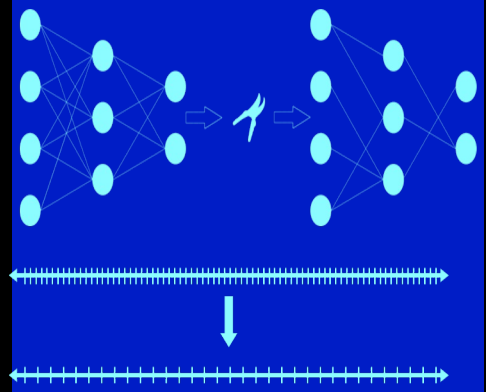
Approximate Computing Overview and Techniques



- **Large spectrum** of cross-stack approximate computing techniques available.
- 3 Primary techniques (already) being used widely in DL
  - **Precision:**
    - **Scaled precision** for Training and Inference
    - Maximum bang for the buck (**quadratic gains in efficiency w. precision**)
  - **Compression:**
    - Lossy compression to minimize data communicated between ASICs for training.
  - **Synchronization:**
    - (Mostly) SW techniques to minimize synchronization overheads for distributed training.

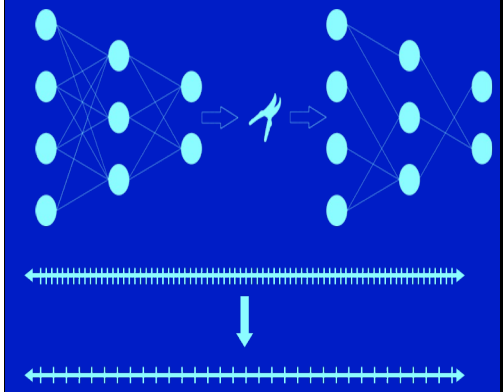
Building Efficient AI

Model Efficiency



Design accurate and efficient neural networks

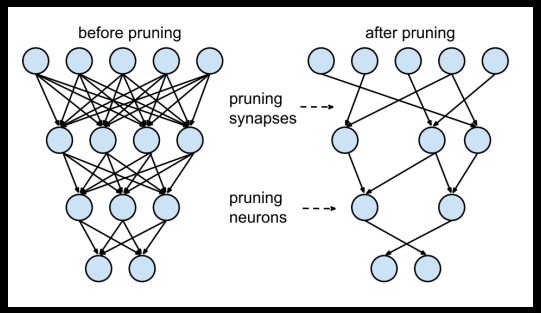
Model Efficiency



Design accurate and efficient neural networks

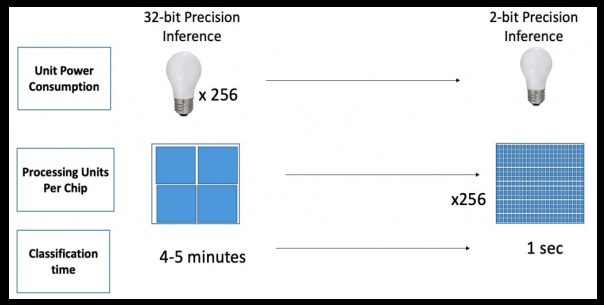
Popular Approaches

Pruning Deep Neural Networks



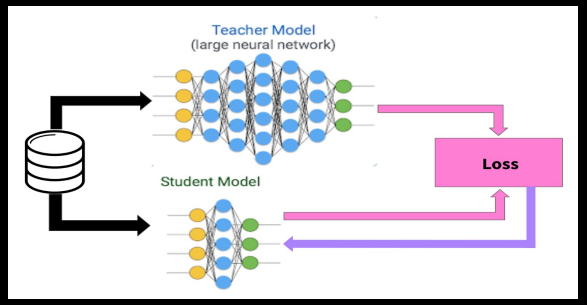
[Lecun et al. NIPS'89] [Han et al. NIPS'15]

Reduced Precision



J. Choi at al, NeurIPS 2019

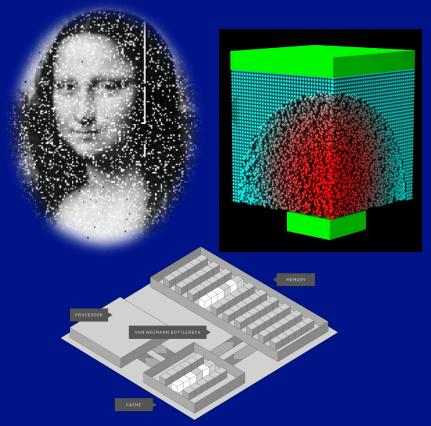
Knowledge Distillation



Hinton et al, arXiv:1503.02531

Building Efficient AI

Hardware Efficiency

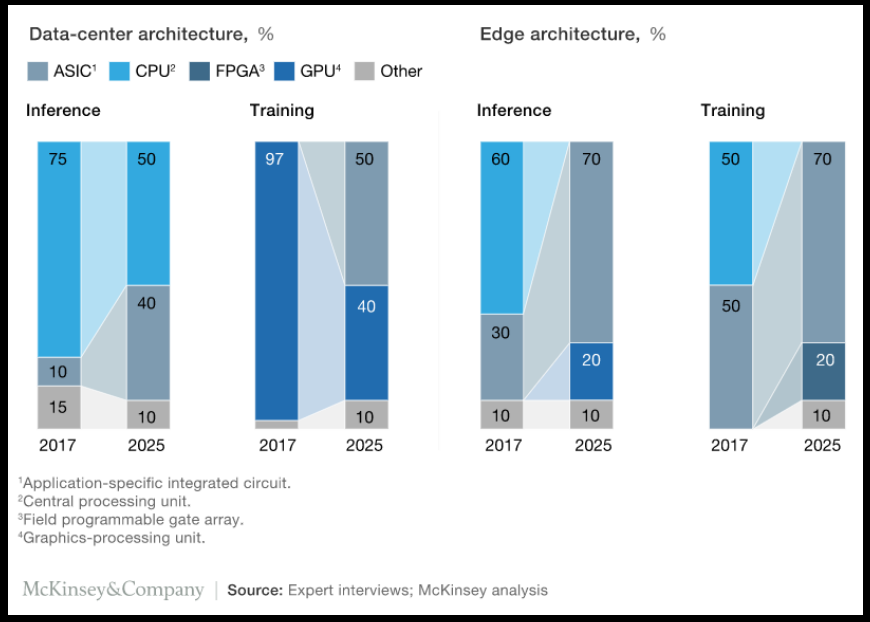


Purpose-built AI hardware

ML hardware trends: towards HPC and beyond

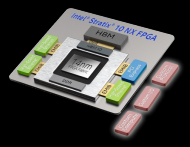
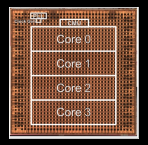
	before		today
Computing	Homogenous (CPU only)	→	Heterogenous (CPU + Accelerators)
Communication	Standard networks (Ethernet)	→	High Performance Networks (IB: low-latency & high-bandwidth)
Datasets	Small size (Gigabytes)	→	Large size (Terabytes to Petabytes)
Precision	DP and SP	→	DP, SP, HP

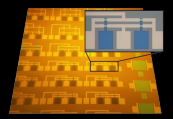
AI Hardware Trends



AI ASICs expect to have the biggest growth





Building Blocks for Deep Learning

Multiply / accumulate

$$y_i = \sum_j w_{i,j}\, x_j$$

multiply + add
multiply+ add
…

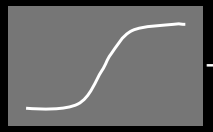
Update

$$\begin{bmatrix} w_{11} & \cdots & w_{1m} \\ \vdots & \ddots & \vdots \\ w_{n1} & \cdots & w_{nm} \end{bmatrix}$$

$$w_{ij} \leftarrow wij + \eta x_i\, \delta_j$$

multiply + add

Activation

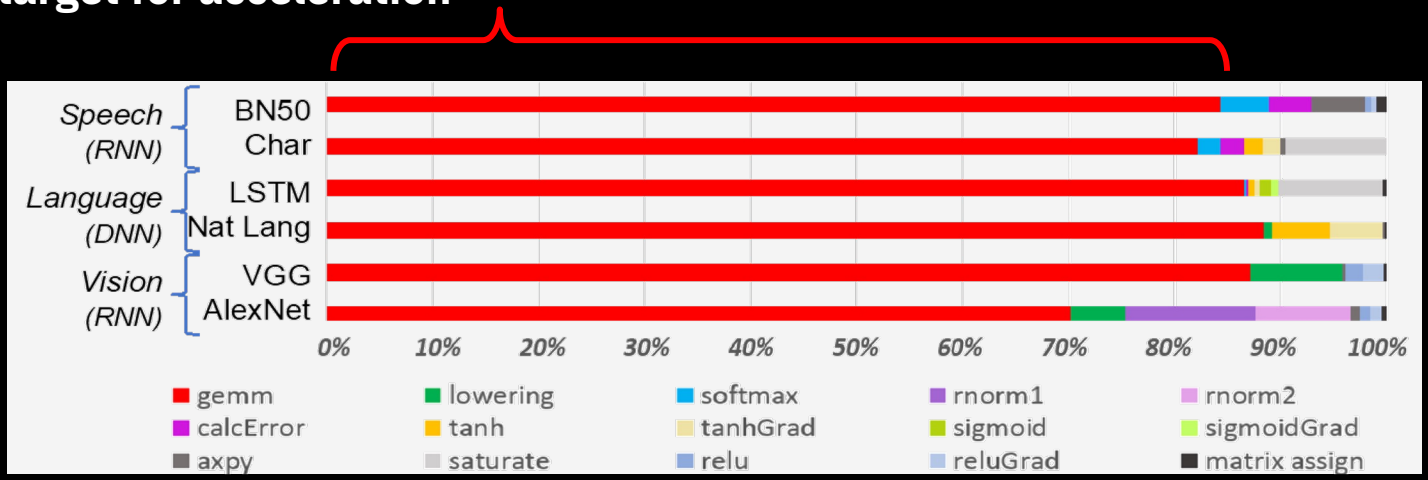
$y_j \rightarrow$  $\rightarrow f(yj)$

Sigmoid
Sofmax
reLu
….

- Matrix manipulations and non-linear activation functions are reoccurring operations in deep learning networks

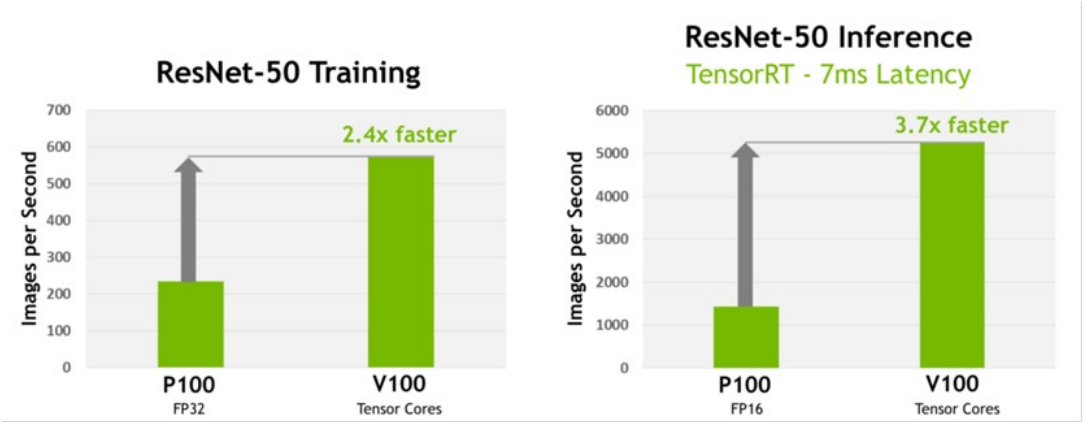
Deep Learning Workload Characteristics

Deep learning inference dominated by matrix- vector multiplication (MVM) a.k.a. general matrix multiplication (gemm) or multiply-accumulate (MAC) and a good target for acceleration



ML Hardware: Nvidia Tensor Cores

- Tensor core
  - Computes a single operation:
    $$D = A \times B + C$$
  - Where:
    - A, B are multiple of 4x4 HP matrices
    - D, C are SP (or HF) 4x4 matrices
  - Up to 8x more throughput than FP64 GPU operations



https://devblogs.nvidia.com/inside-volta/

ML Hardware: Nvidia Volta GPU

- General-purpose Accelerator + Tensor cores for Neural Nets

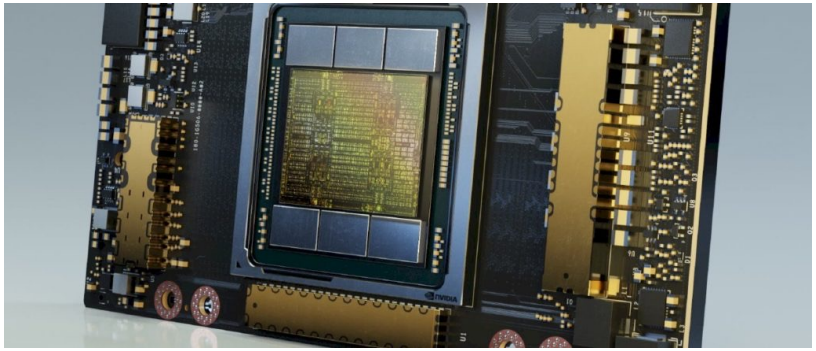
	Nvidia Tesla V100 (Volta)
FP64 performance	7.8 TFLOP/s
FP32 performance	15.7 TFLOP/s
Tensor performance	125 TFLOP/s
Clock frequency	1.53GHz
Memory BW	900GB/s
Memory capacity	16GB
High-speed Interconnect	Nvlink - proprietary

https://devblogs.nvidia.com/inside-volta/

ML Hardware: Nvidia A100 GPU

	Peak Performance
Transistor Count	54 billion
Die Size	826 mm²
FP64 CUDA Cores	3,456
FP32 CUDA Cores	6,912
Tensor Cores	432
Streaming Multiprocessors	108
FP64	9.7 teraFLOPS
FP64 Tensor Core	19.5 teraFLOPS
FP32	19.5 teraFLOPS
TF32 Tensor Core	156 teraFLOPS | 312 teraFLOPS*
BFLOAT16 Tensor Core	312 teraFLOPS | 624 teraFLOPS*
FP16 Tensor Core	312 teraFLOPS | 624 teraFLOPS*
INT8 Tensor Core	624 TOPS | 1,248 TOPS*
INT4 Tensor Core	1,248 TOPS | 2,496 TOPS*
GPU Memory	40 GB
GPU Memory Bandwidth	1.6 TB/s
Interconnect	NVLink 600 GB/s PCIe Gen4 64 GB/s
Multi-Instance GPUs	Various Instance sizes with up to 7MIGs @5GB
Form Factor	4/8 SXM GPUs in HGX A100
Max Power	400W (SXM)

*structural sparsity enabled

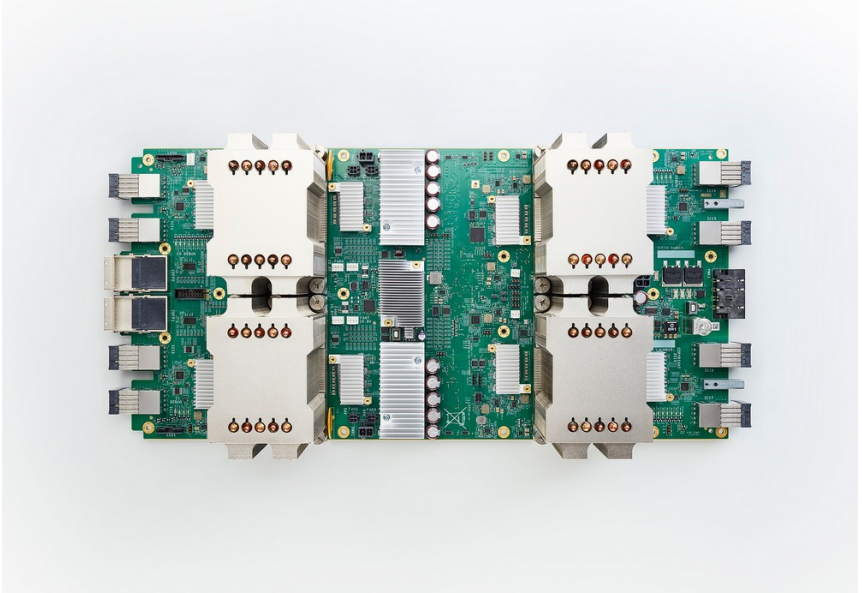


NVIDIA A100 delivers **312 teraFLOPS (TFLOPS) of deep learning performance**.
20X the Tensor floating-point operations per second (FLOPS) **for deep learning training** and **20X** the Tensor tera operations per second (TOPS) for **deep learning inference** compared to NVIDIA **Volta GPUs.**

ML Hardware: Google TPU v2

- Tensor processing unit
- TPU v1 did only inference
- Neural Nets accelerator
- 4 chips in each module

	Google TPU v2
Tensor performance	180 TFLOP/s
Clock frequency	2 GHz
Memory BW	2400 GB/s
Memory capacity	64GB
High-speed Interconnect	proprietary



From: Google

Lesson Key Points

- ML/DL Trends
- Traditional HPC Software/Hardware Technology
- Motivation for Efficient ML
- ML Software/Hardware Technology
- Differences between ML and traditional HPC

References

- Kurth et al. "*Deep Learning at 15PF - Supervised and SemiSupervised Classification for Scientific Data*". *Supercomputing 2017*
- Sue Kelly. "Principles of Scalable HPC System Design". Sandia National Laboratories. 2012 (slides available under "View Conference" tab on left margin)
- Timoth P. Morgan. HPC as a service comes full circle and will help take HPC mainstream. The Next Platform. 2022

Acknowledgements

- The lecture material is prepared by Kaoutar El Maghraoui, Parijat Dube , Giacomo Domeniconi, and Ulrich Finkler from IBM Research and Zehra Sura from Bloomberg