HARDWARE FOR MACHINE LEARNING & MACHINE LEARNING AS A SERVICE

Miguel Ángel Martínez del Amor

Department of Computer Science and Artificial Intelligence
Universidad de Sevilla
https://www.cs.us.es/~mdelamor
mdelamor@us.es @miguelamda

I Workshop de Computing y Software de la Red Española de LHC

Outline

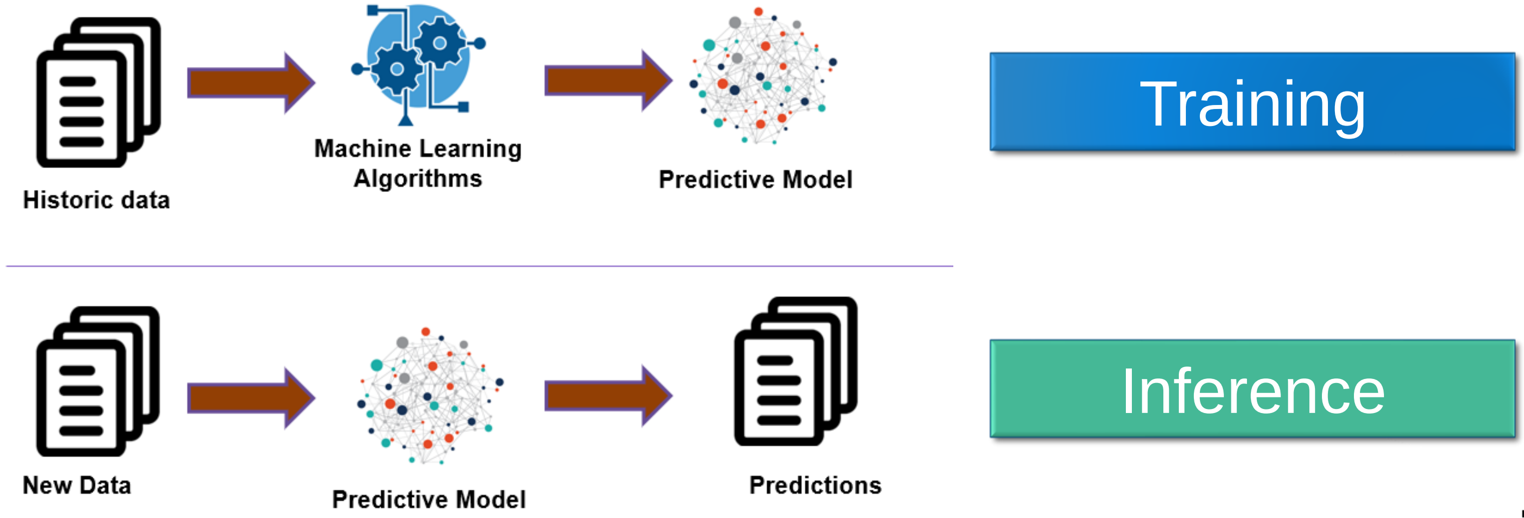
- Machine Learning: brief fundamentals
- Hardware for Machine Learning: CPU/GPU/FPGA/ASIC
- Accelerated frameworks for Machine Learning
- Machine Learning as a Service (MLAAS)

MACHINE LEARNING

Brief fundamentals

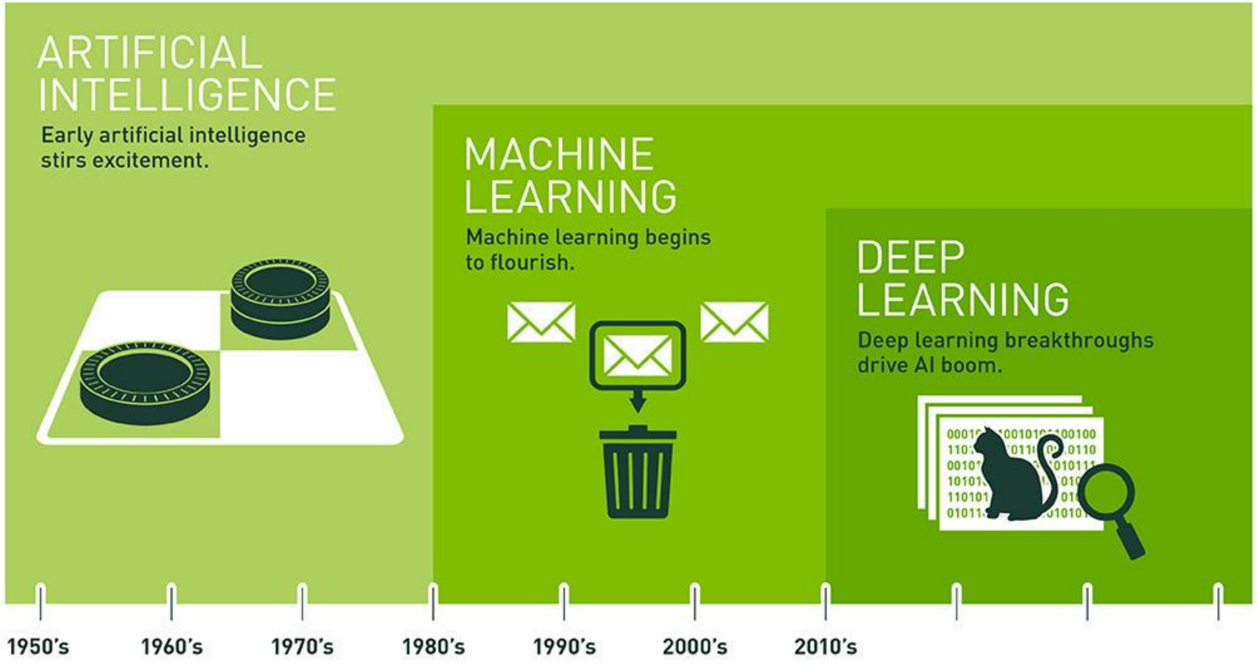
Machine Learning

- Learn by exampe:



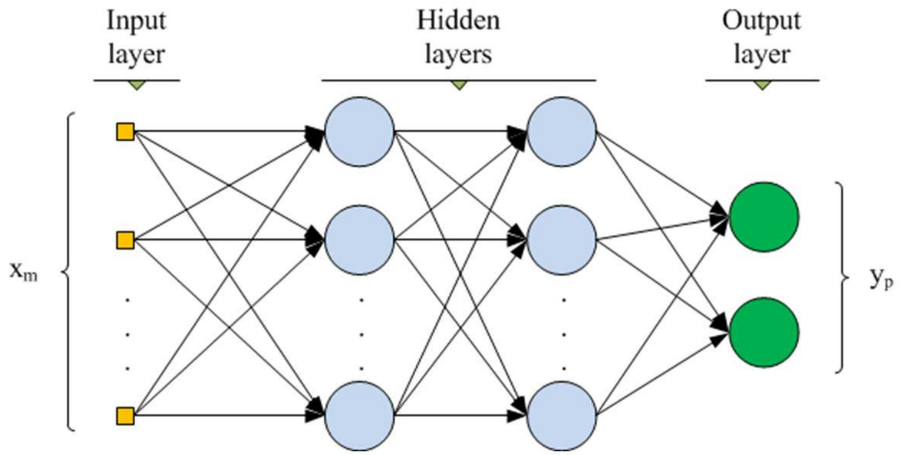
Machine Learning

- Machine learning evolution


ARTIFICIAL INTELLIGENCE
Early artificial intelligence stirs excitement.
MACHINE LEARNING
Machine learning begins to flourish.
DEEP LEARNING
Deep learning breakthroughs drive AI boom.
1950's
1960's
1970's
1980's
1990's
2000's
2010's

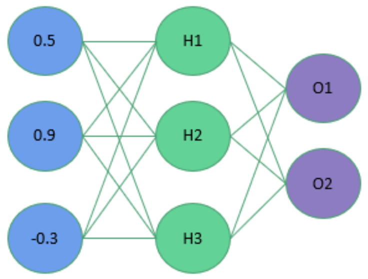
Machine Learning

- Deep Learning
  - Mainly based on (deep) multilayer neural networks (DNN)



Machine Learning

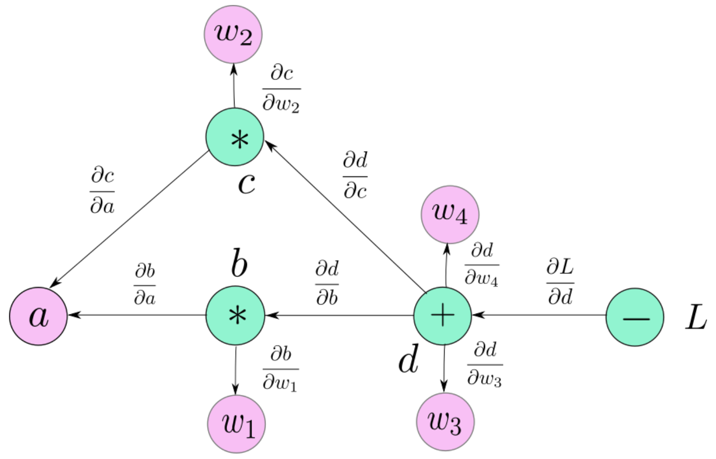
- Deep Learning
  - DNN can be interpreted as matrices and computational graphs



H1 Weights = (1.0, -2.0, 2.0)
H2 Weights = (2.0, 1.0, -4.0)
H3 Weights = (1.0, -1.0, 0.0)

Hidden Layer Weights, Inputs, Hidden Layer Outputs

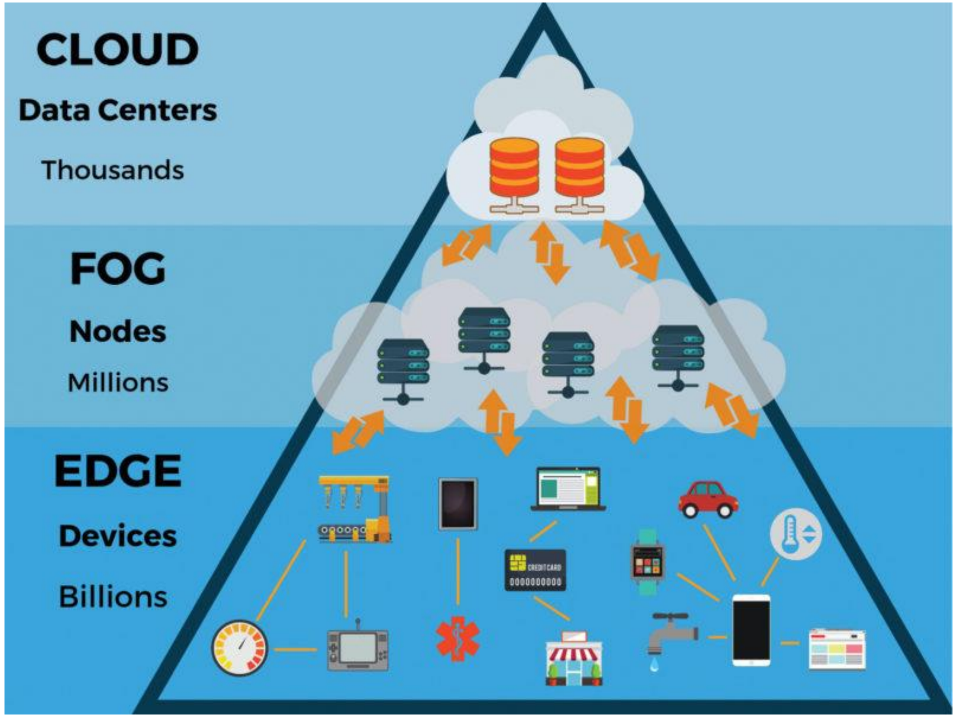
$$S\left(\underbrace{\begin{bmatrix} 1.0 & -2.0 & 2.0 \\ 2.0 & 1.0 & -4.0 \\ 1.0 & -1.0 & 0.0 \end{bmatrix}}_{\text{Hidden Layer Weights}} * \underbrace{\begin{bmatrix} 0.5 \\ 0.9 \\ -0.3 \end{bmatrix}}_{\text{Inputs}}\right) = S\left(\begin{bmatrix} -1.9 & 3.1 & -0.4 \end{bmatrix}\right) = \underbrace{\begin{bmatrix} 0.13 & 0.96 & 0.4 \end{bmatrix}}_{\text{Hidden Layer Outputs}}$$



w_2
$\frac{\partial c}{\partial w_2}$
$*$
c
$\frac{\partial c}{\partial a}$
$\frac{\partial d}{\partial c}$
w_4
$\frac{\partial d}{\partial w_4}$
b
$\frac{\partial b}{\partial a}$
$\frac{\partial d}{\partial b}$
$\frac{\partial L}{\partial d}$
a
$*$
$+$
$-$
L
$\frac{\partial b}{\partial w_1}$
d
$\frac{\partial d}{\partial w_3}$
w_1
w_3

Machine Learning

- Where?



Source: https://iot.electronicsforu.com/content/tech-trends/edge-and-fog-computing-practical-uses/

HARDWARE
FOR MACHINE LEARNING

How to make it real

Hardware

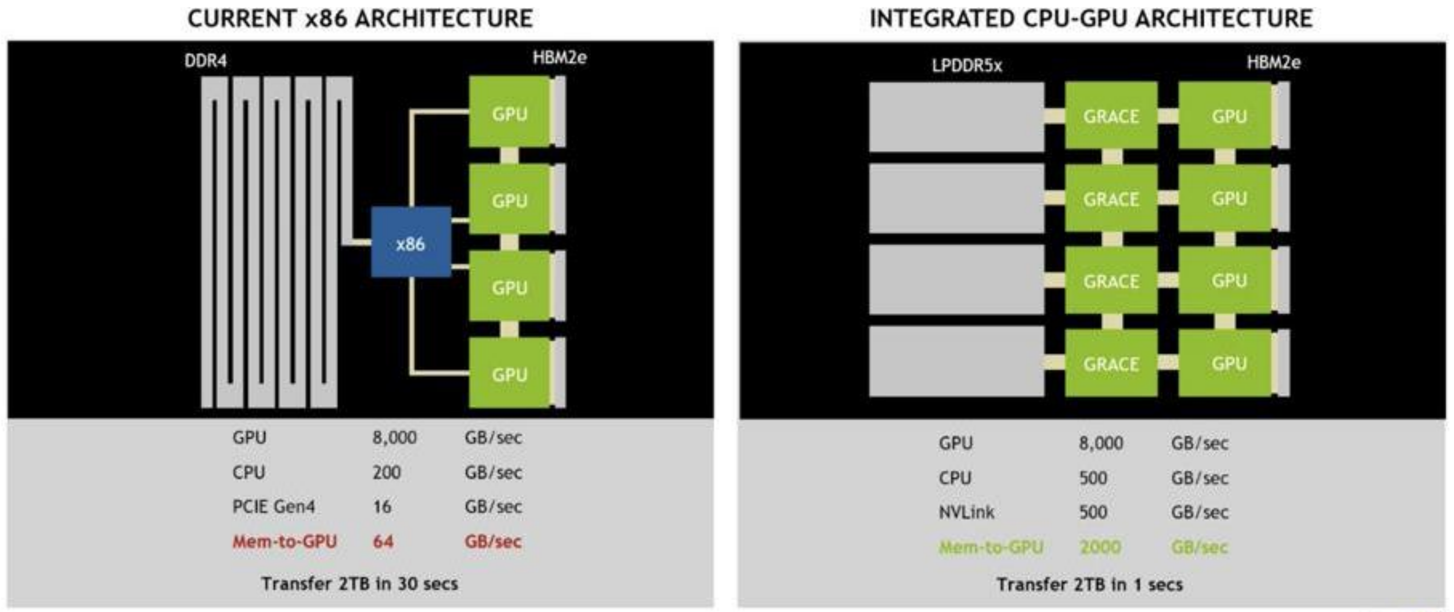
- Requirements (from ML perspective):
  - **Reduced-precision** computations
    - 8-bit integers are enough for inference.
  - Small set of **operations**: some linear algebra operations such as matrix/vector multiplication, convolution, etc.
    - Dense vs **sparse** operations
  - No need for branch prediction, deep cache memories, etc.
  - High **memory** bandwidth, low latency and large storage

Hardware (CPU)

- Usually used for inference
- **x86 family** (Intel, AMD)
  - Similar features: **Intel Xeon** (~56 cores) vs **AMD EPYC** (~64 cores)
  - Memory bus: PCI-express (PCIe v4, x16, ~16GB/s)
  - Intel's **SIMD extensions** for ML (fused mult-add vector operations):
    - **AVX512**, AVX512 VNNI (for CNN), **DL Boost** AVX512-VNNI + bfloat16
- **IBM**: currently with POWER9 arch.
  - 96 threads with 12 or 24 cores.
  - PCIe v4 vs **NVLink** 2.0 CPU-to-GPU (~ 500GB/s)
- **ARM**: currently v8 with Cortex-A (48 cores) and Neoverse (128 cores)
  - **Low energy** consumption with lightweight cores
  - Mobile, server, laptops, single boards (Raspberry Pi, Jetson, Google Coral Dev Board)

Hardware (CPU)

- ARM is about to be acquired by NVIDIA
- **NVIDIA Grace** announced at GTC2021



CURRENT x86 ARCHITECTURE

GPU	8,000	GB/sec
CPU	200	GB/sec
PCIE Gen4	16	GB/sec
Mem-to-GPU	64	GB/sec

Transfer 2TB in 30 secs

INTEGRATED CPU-GPU ARCHITECTURE

GPU	8,000	GB/sec
CPU	500	GB/sec
NVLink	500	GB/sec
Mem-to-GPU	2000	GB/sec

Transfer 2TB in 1 secs

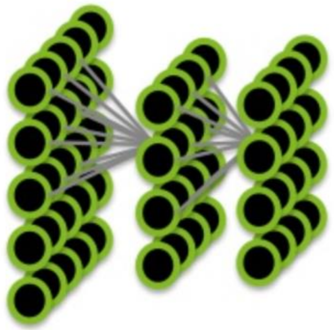
Hardware (GPU)

- **Key** device in Deep Learning

Big Data

facebook	350 millions images uploaded per day
Walmart	2.5 Petabytes of customer data hourly
YouTube	300 hours of video uploaded every minute

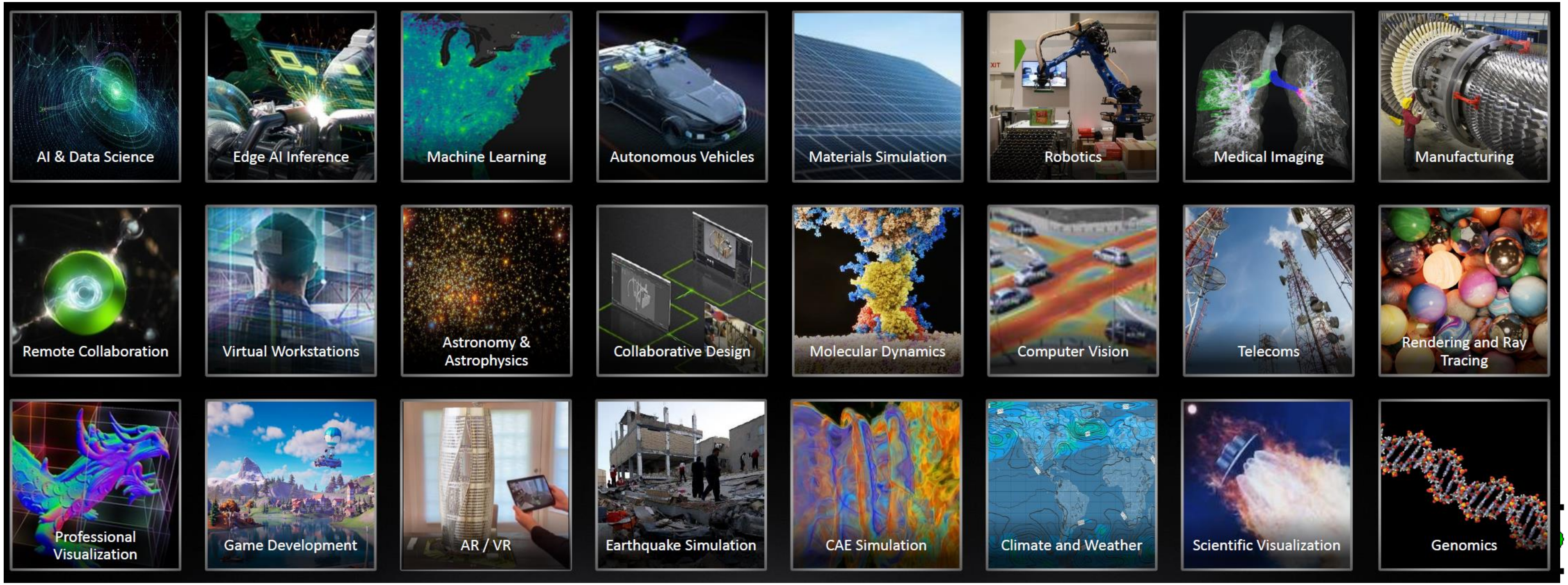
Better Algorithms



GPU Acceleration

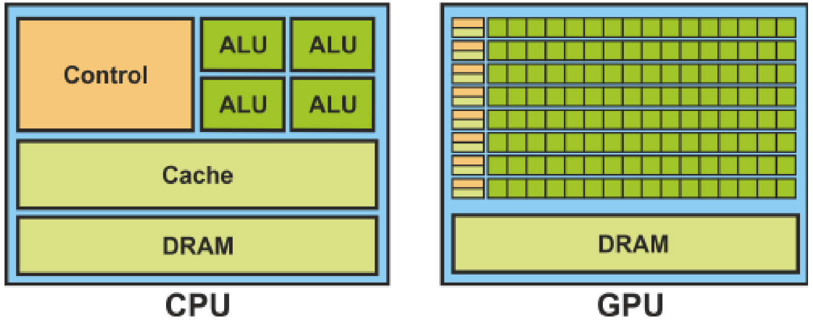
Hardware (GPU)

- And in many other applications



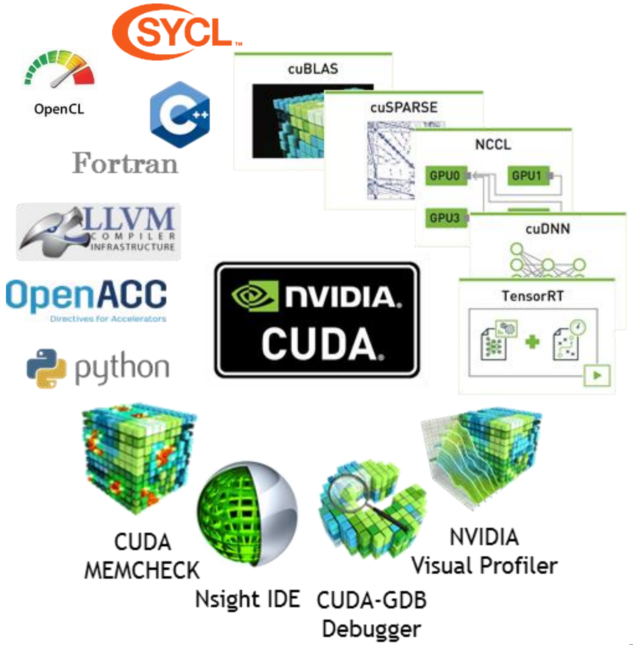
Hardware (GPU)

- **GPU computing**: general purpose computations on GPUs
- Good on **data parallelism** (e.g. matrix operations)
- Main vendors: NVIDIA, AMD, Intel, ARM



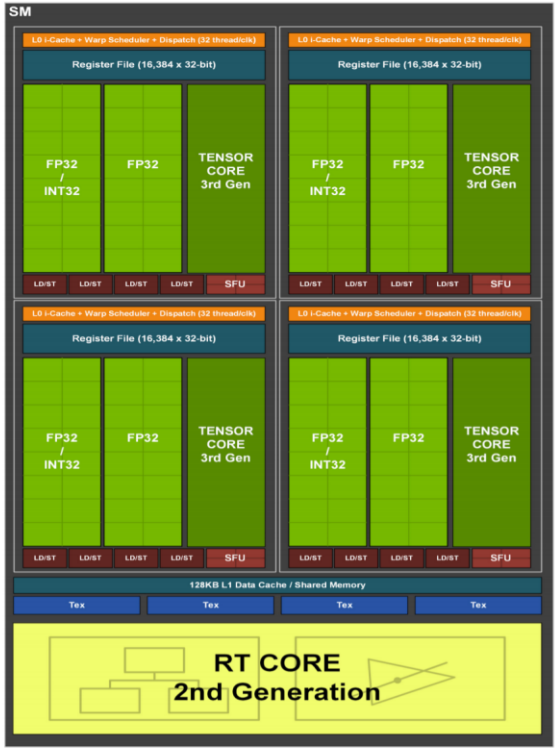
Hardware (GPU)

- **NVIDIA**:
  - Proprietary of **CUDA**
  - **GPUs**:
    - GTX/RTX (~~GeForce~~, videogames)
    - RTX Axxxx (~~Quadro~~, rendering)
    - Axxx (~~Tesla~~, supercomputing)
    - Jetson (for robotics)
    - DGX (workstation)



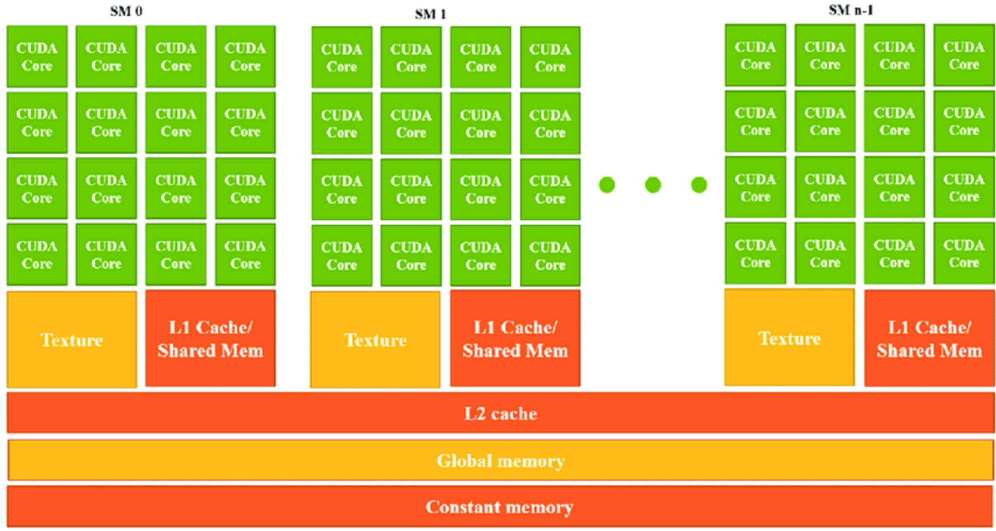
Hardware (GPU)

- **NVIDIA**:
  - **Architectures** (since 2007):
    - ~~Tesla, Fermi~~, Kepler, Pascal, Maxwell, Volta, Turing, Ampere
  - **Ampere**:
    - RTX3xxx, Axxxx (A6000, A40…)
    - GPU Mig (multi-instance / partitions of one GPU)
    - Example, RTX3080:
      - 8704 cores, 10GB, 272 Tensor cores, 68 RT cores


SM
L0 i-Cache + Warp Scheduler + Dispatch (32 thread/clk)
Register File (16,384 x 32-bit)
FP32 / INT32
FP32
TENSOR CORE 3rd Gen
LD/ST
SFU
128KB L1 Data Cache / Shared Memory
Tex
RT CORE 2nd Generation

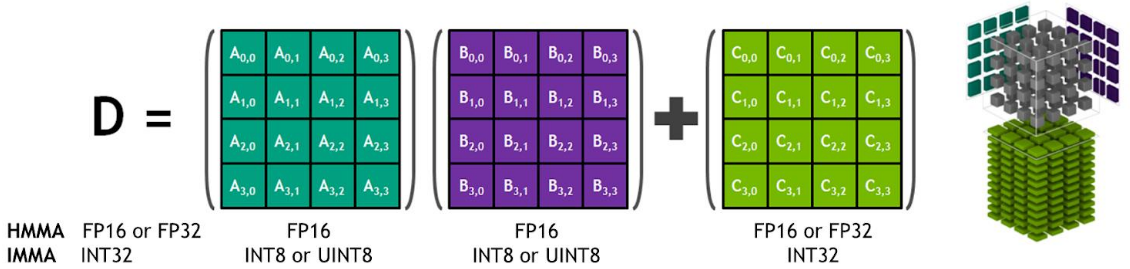
Hardware (GPU)

- **NVIDIA**:
  - **CUDA cores:**
    - SP distributed in SMs
    - FP64, FP32, FP16/Mixed precision, INT32, INT16, INT8
  - **Memories**:
    - Global (device)
    - Texture
    - L2 cache
    - Shared
    - L1 cache
    - Registers



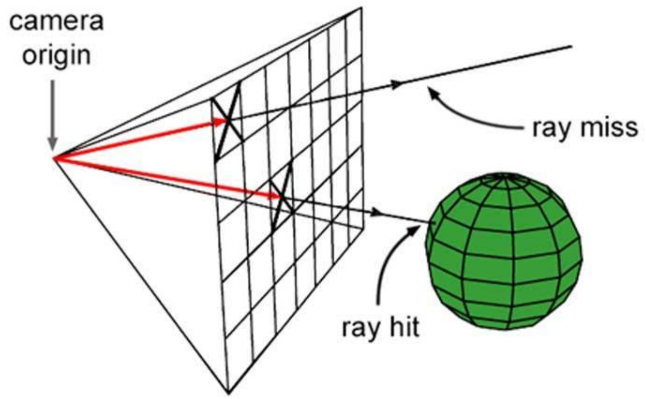
Hardware (GPU)

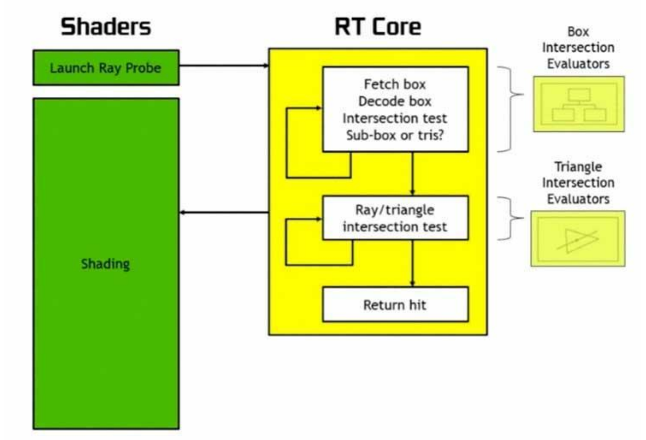
- **NVIDIA**:
  - **Tensor cores:**
    - Since Volta
    - For tensor operations (fused mult-add matrix operation), for CNN
    - Support of Half Precision (float16, int8).
    - Currently optimized for sparse operations


D =
$A_{0,0}$ $A_{0,1}$ $A_{0,2}$ $A_{0,3}$
$A_{1,0}$ $A_{1,1}$ $A_{1,2}$ $A_{1,3}$
$A_{2,0}$ $A_{2,1}$ $A_{2,2}$ $A_{2,3}$
$A_{3,0}$ $A_{3,1}$ $A_{3,2}$ $A_{3,3}$
$B_{0,0}$ $B_{0,1}$ $B_{0,2}$ $B_{0,3}$
$B_{1,0}$ $B_{1,1}$ $B_{1,2}$ $B_{1,3}$
$B_{2,0}$ $B_{2,1}$ $B_{2,2}$ $B_{2,3}$
$B_{3,0}$ $B_{3,1}$ $B_{3,2}$ $B_{3,3}$
$C_{0,0}$ $C_{0,1}$ $C_{0,2}$ $C_{0,3}$
$C_{1,0}$ $C_{1,1}$ $C_{1,2}$ $C_{1,3}$
$C_{2,0}$ $C_{2,1}$ $C_{2,2}$ $C_{2,3}$
$C_{3,0}$ $C_{3,1}$ $C_{3,2}$ $C_{3,3}$
HMMA FP16 or FP32
IMMA INT32
FP16
INT8 or UINT8
FP16
INT8 or UINT8
FP16 or FP32
INT32

Hardware (GPU)

- **NVIDIA**:
  - **RT cores:**
    - Since Turing (reason for GTX → RTX)
    - For real time Ray Tracing (track objects hit by a ray)

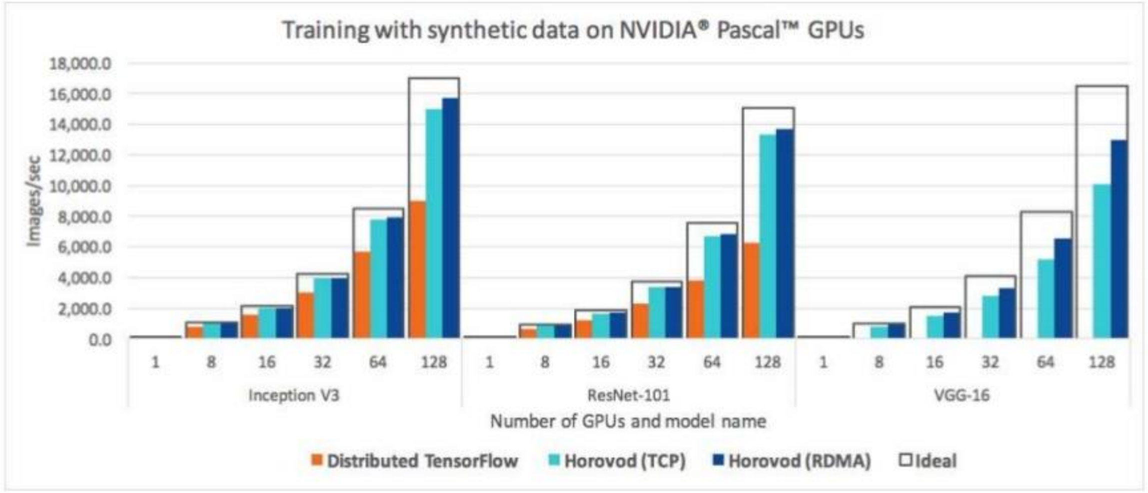




https://itigic.com/rt-cores-how-they-work-on-a-gpu-for-ray-tracing/

Hardware (GPU)

- **NVIDIA**:
  - Distributed training is possible, but scaling is sublinear



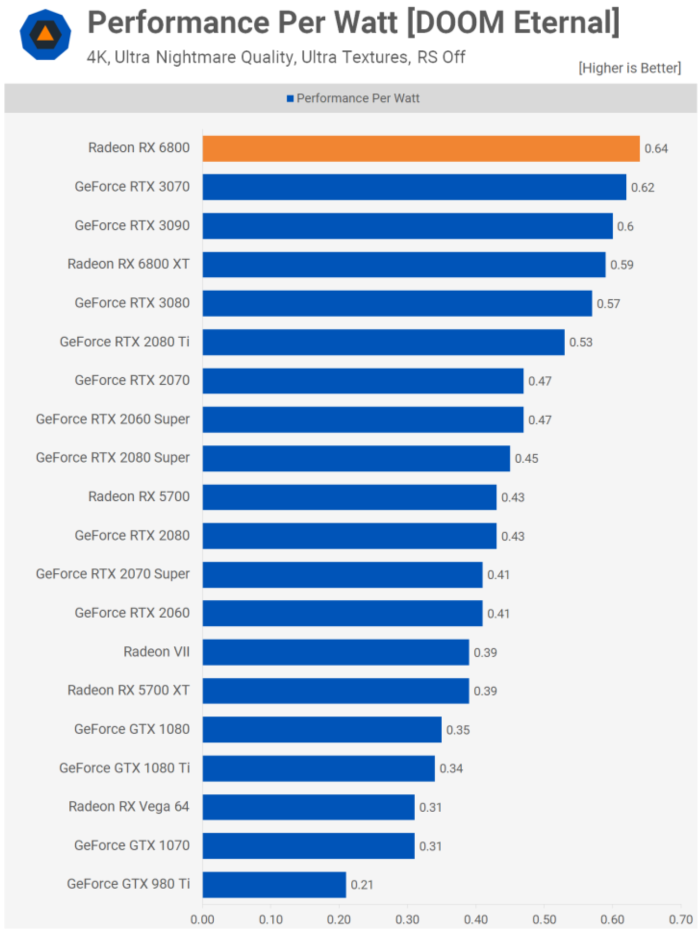
Hardware (GPU)

- **AMD**:
  - Main competitor of NVIDIA in GPUs. Main market: game consoles
  - Creator of **ROCm** project:
    - Open source, following the lead of CUDA
    - **HIP**: CUDA → ROCm translation
  - Infinity Fabric (like NVLink)

OpenCL

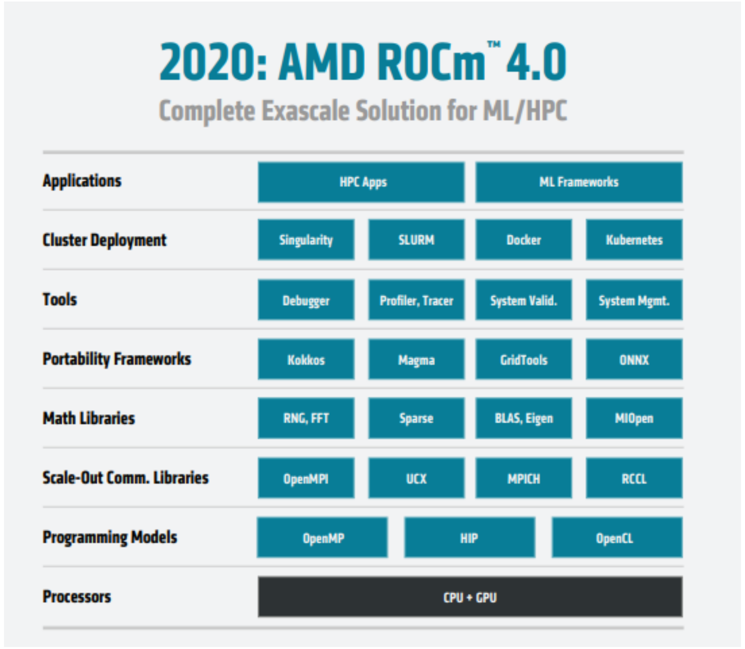
Hardware (GPU)

- **AMD**:
  - **Architectures**:
    - Graphics Core Next (GCN)
    - Radeon DNA (RDNA)
      - Changes in how the code is scheduled
  - Radeon **RX 6000** (Big navy, RDNA2)
    - Include RA (ray accelerator)
    - ML done with shader units using packed data formats (32-bit vectors)
    - Example, RX6900XT:
      - 80 CUs (5120 cores), 16GB, 80 RAs



Hardware (GPU)

- **AMD**:
  - Experimental support of ML frameworks
  - Miopen: library for ML primitives
  - ROCm is supported by SYCL LLVM and TVMlang.



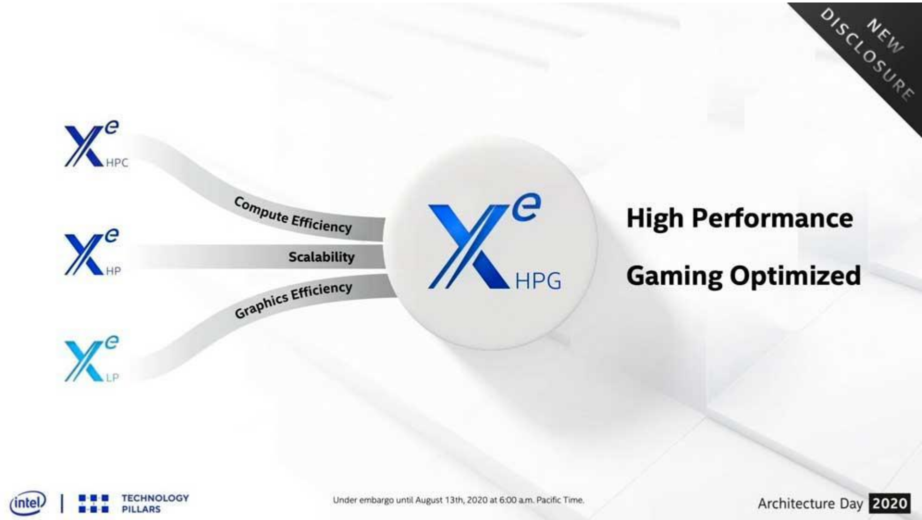
2020: AMD ROCm™ 4.0

Complete Exascale Solution for ML/HPC

Applications	HPC Apps		ML Frameworks	
Cluster Deployment	Singularity	SLURM	Docker	Kubernetes
Tools	Debugger	Profiler, Tracer	System Valid.	System Mgmt.
Portability Frameworks	Kokkos	Magma	GridTools	ONNX
Math Libraries	RNG, FFT	Sparse	BLAS, Eigen	MIOpen
Scale-Out Comm. Libraries	OpenMPI	UCX	MPICH	RCCL
Programming Models	OpenMP	HIP	OpenCL	
Processors	CPU + GPU			

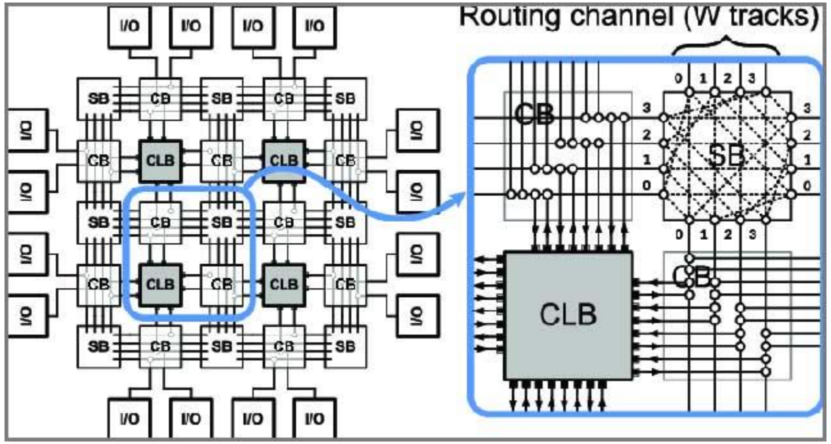
Hardware (GPU)

- **Intel**:
  - Mainly pushing SYCL
  - So far, small integrated GPUs: HD Graphics
  - Coming in 2021 (?): **Intel Xe HPG**
    - GPU computing with CPU+GPU
    - ~Expected RX6800 – RTX3070



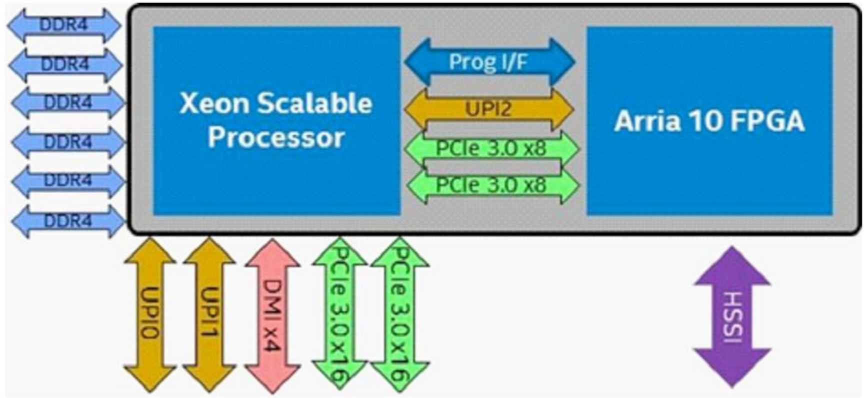
Hardware (FPGA)

- **FPGA**: field-programmable gate array
- GPUs are energy-inefficient, FPGAs can be **energy-efficient** alternative
- Usually used for **inference**
- **OpenCL** can be used for development
- Opportunities on the cloud
- Very steep learning curve



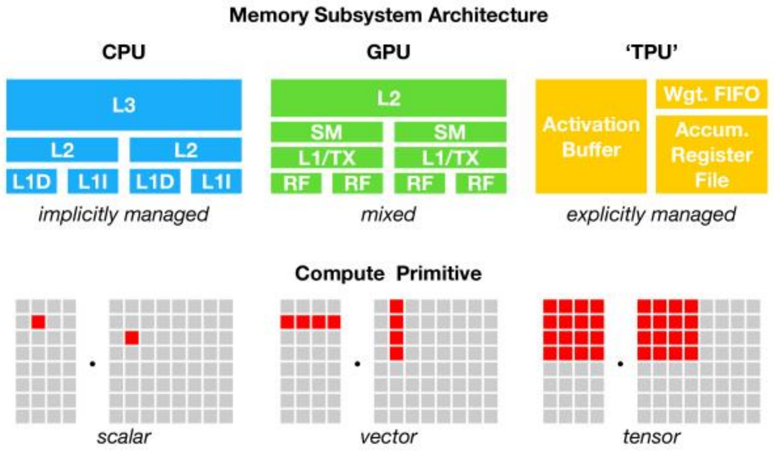
Hardware (FPGA)

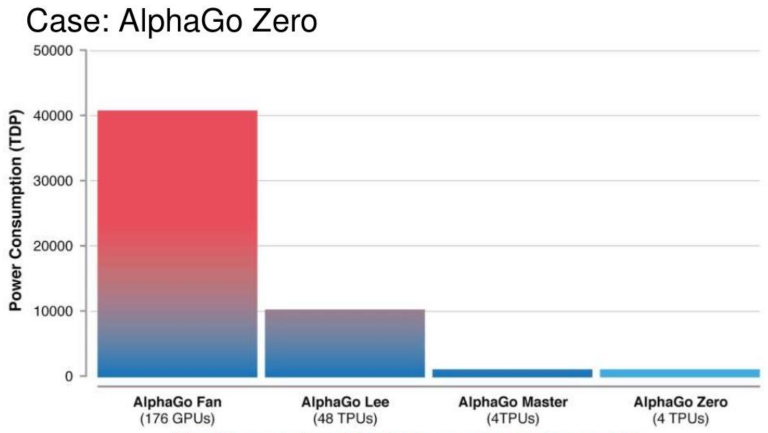
- Main manufacturers: Intel (**Altera**) and **Xilinx**
  - Intel Stratix 10GX: 10.2 million logic cells
  - Xilinx Virtex UltraScale+: 9 million logic cells
- Intel has hybrid Xeon+FPGA chips
- AMD interested on acquiring Xilinx



Hardware (ASIC)

- **ASIC**: application-specific integrated circuit
- Specialized chips **for ANN** by matrix (tensor) operations
- Very **low energy** consumption
- Cannot be re-programmed like FPGAs
- Separate ASICs for **inference** and **training**
- Plethora of options: Google, Habana, AWS, Alibaba, Huawei…
  - TPU, NNP, NPU, IPU, VPU, G(raph)PU

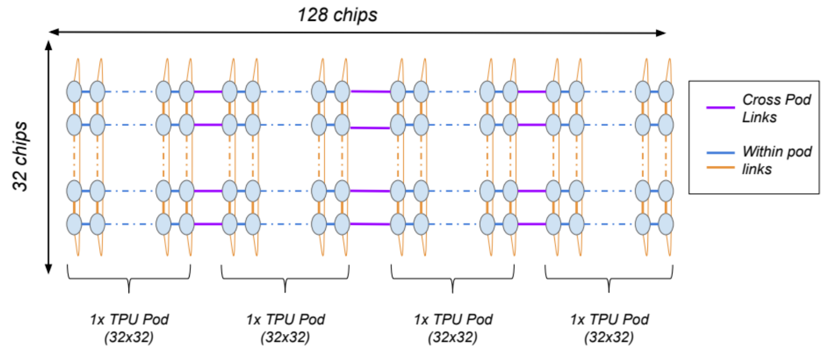




https://deepmind.com/blog/alphago-zero-learning-scratch/

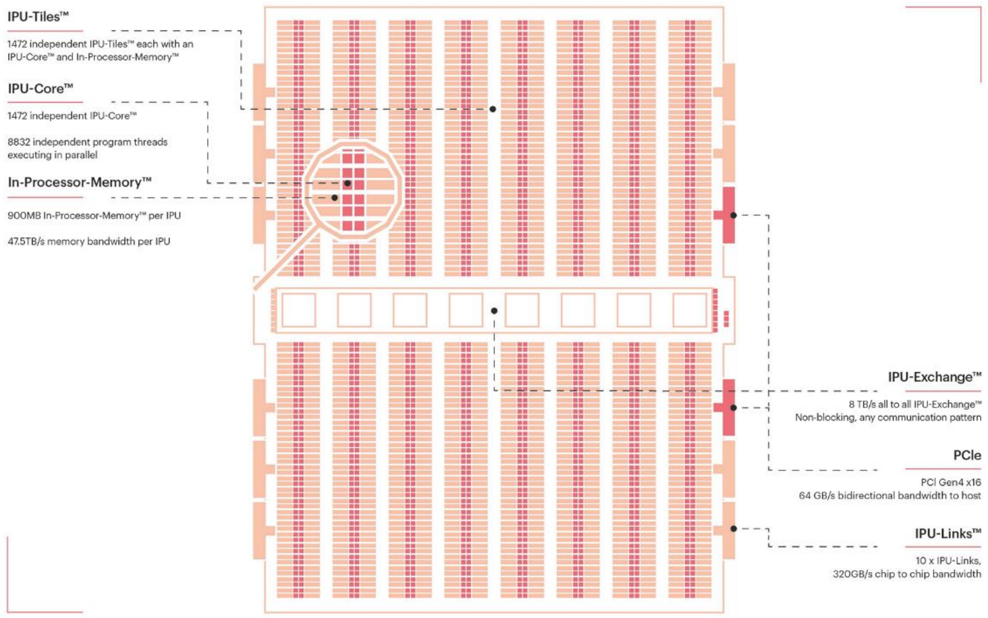
Hardware (ASIC)

- Google’s TPU:
  - Training and inference
  - Today, only available on the cloud:
    - **TPUv2**: 180 TFLOPS (bfloat16), 64GB HBM, 4.5$/h
    - **TPUv3**: 420 TFLOPS (bfloat16), 128GB HBM, 8$/h
    - **TPUv4** (dec 2020): 2xTFLOPS versus TPUv3
  - They can work together in a network (distributed ML): **POD**



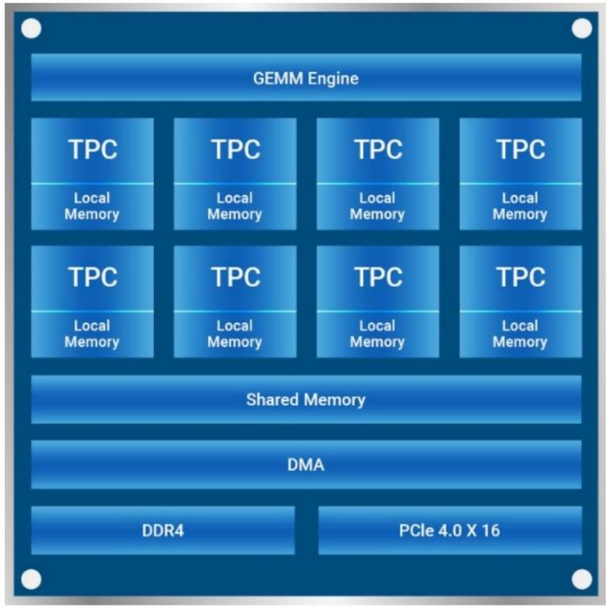
Hardware (ASIC)

- Graphcore's IPU:
  - Training and Inference.
  - GC2 (Colossus MK1 GC2 IPU):
    - 7296 threads running on 1216 IPU tiles
    - 31.1 TFLOPS FP32, 124.5 TFLOPS mixed precision
  - **GC200** (Colossus MK2 GC200 IPU):
    - 8832 threads running on 1472 IPU tiles
    - 250 TFLOPS at FP16
  - **Poplar** SDK: integration with TF2/pyTorch
  - Ready for **Bulk Synchronous Parallelism**
    - Graph based computation with high-dimensional variables
    - All tiles run code independently with their own local memory. They must synchronise to exchange data.
  - **Price**:
    - IPU-M2000 (GC200, 3.6GB on chip, 448GB outside, 32,450$ (can be used to construct a POD, up to 64,000 working as one)


IPU-Tiles™
1472 independent IPU-Tiles™ each with an IPU-Core™ and In-Processor-Memory™
IPU-Core™
1472 independent IPU-Core™
8832 independent program threads executing in parallel
In-Processor-Memory™
900MB In-Processor-Memory™ per IPU
47.5TB/s memory bandwidth per IPU
IPU-Exchange™
8 TB/s all to all IPU-Exchange™
Non-blocking, any communication pattern
PCIe
PCI Gen4 x16
64 GB/s bidirectional bandwidth to host
IPU-Links™
10 x IPU-Links,
320GB/s chip to chip bandwidth

Hardware (ASIC)

- Intel's Habana:
  - **Gaudi**: training chip
    - PCIe v4 x16, 32GB HBM2, FP32, BF16, INT32-16-8
  - **Goya**: inference chip
    - PCIe v4 x16, 4/8/16GB DDR4, FP32, BF16, INT32-16-8
  - SynapseAI compiler, supporting TF 2.2 and ONNX



Available

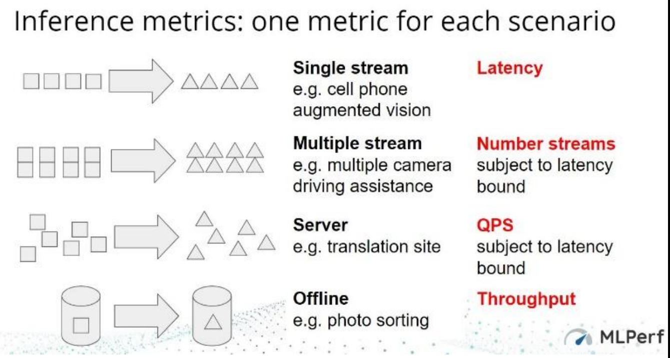
Sampling

Hardware (ASIC)

- Others:
  - **Cerebras:** WSE with 400,000 cores and 18 GB.
  - **AWS**
    - **Inferential:** 64 TFLOPS with FP16, 128TOPs INT8.
    - **Trainium**: available in 2021 (support for TF, PT, MXNet)
  - **Huawei**:
    - **Ascend 310:** 22TOPS INT8, 11 TFLOPS FP16, 8W power. In **Atlas 300I inference**
    - **Ascend 910:** 640 TOPS INT8, 320TFLOPS FP16 (close to A100). In **Atlas 300T Training card**
  - **Alibaba Hanguang 800 (AI-inference chip)**
  - **Baidu Kunlun, Groq TSP, Bitmain Sophon, Qualcomm Coud, Wave Computing (DPU), ARM ML inference NPU, SambaNova RDA, Mythic…**

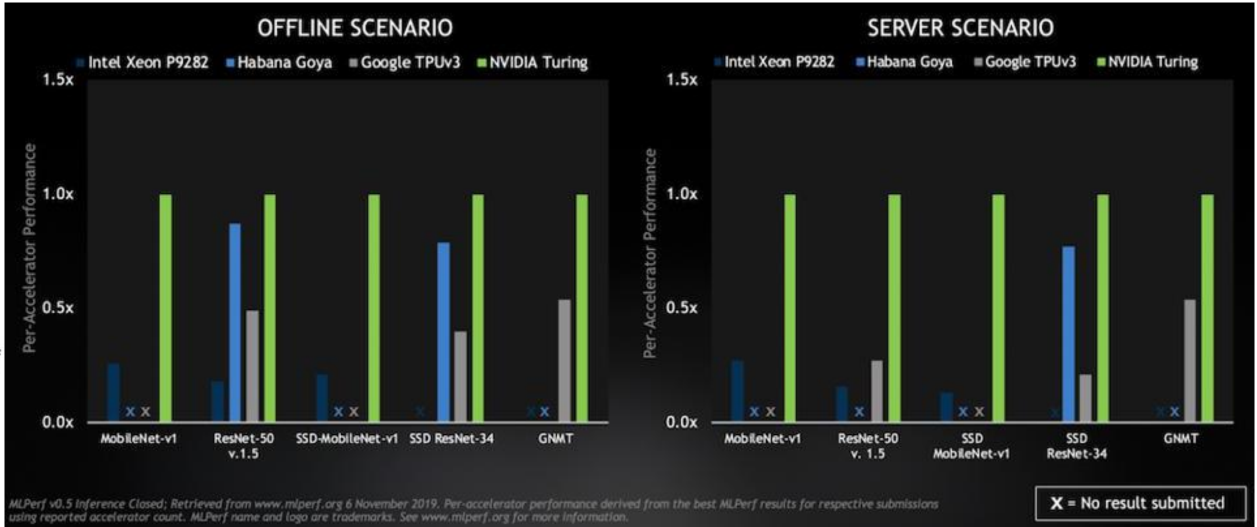
Hardware (comparison)

- How to compare? have a look to MLPerf:
  - SoTA benchmarks on different tasks (image, NLP, …), models, HW
  - Training vs Inference, Cloud vs on-premise



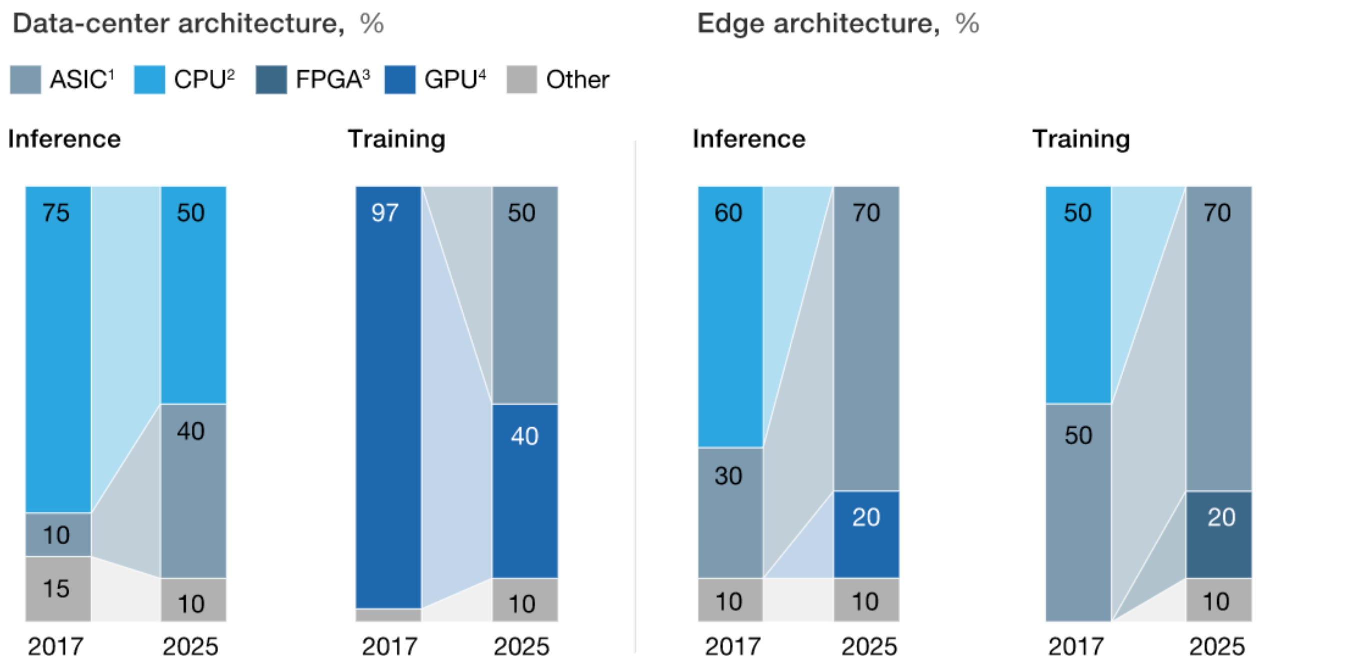
Inference metrics: one metric for each scenario

Single stream e.g. cell phone augmented vision	Latency	
Multiple stream e.g. multiple camera driving assistance	Number streams subject to latency bound	
Server e.g. translation site	QPS subject to latency bound	
Offline e.g. photo sorting	Throughput	



MLPerf v0.5 Inference Closed; Retrieved from www.mlperf.org 6 November 2019. Per-accelerator performance derived from the best MLPerf results for respective submissions using reported accelerator count. MLPerf name and logo are trademarks. See www.mlperf.org for more information.

Hardware (trends)



[1]Application-specific integrated circuit.
[2]Central processing unit.
[3]Field programmable gate array.
[4]Graphics-processing unit.

Source: https://www.mckinsey.com/industries/semiconductors/our-insights/artificial-intelligence-hardware-new-opportunities-for-semiconductor-companies

ACCELERATED FRAMEWORKS FOR MACHINE LEARNING

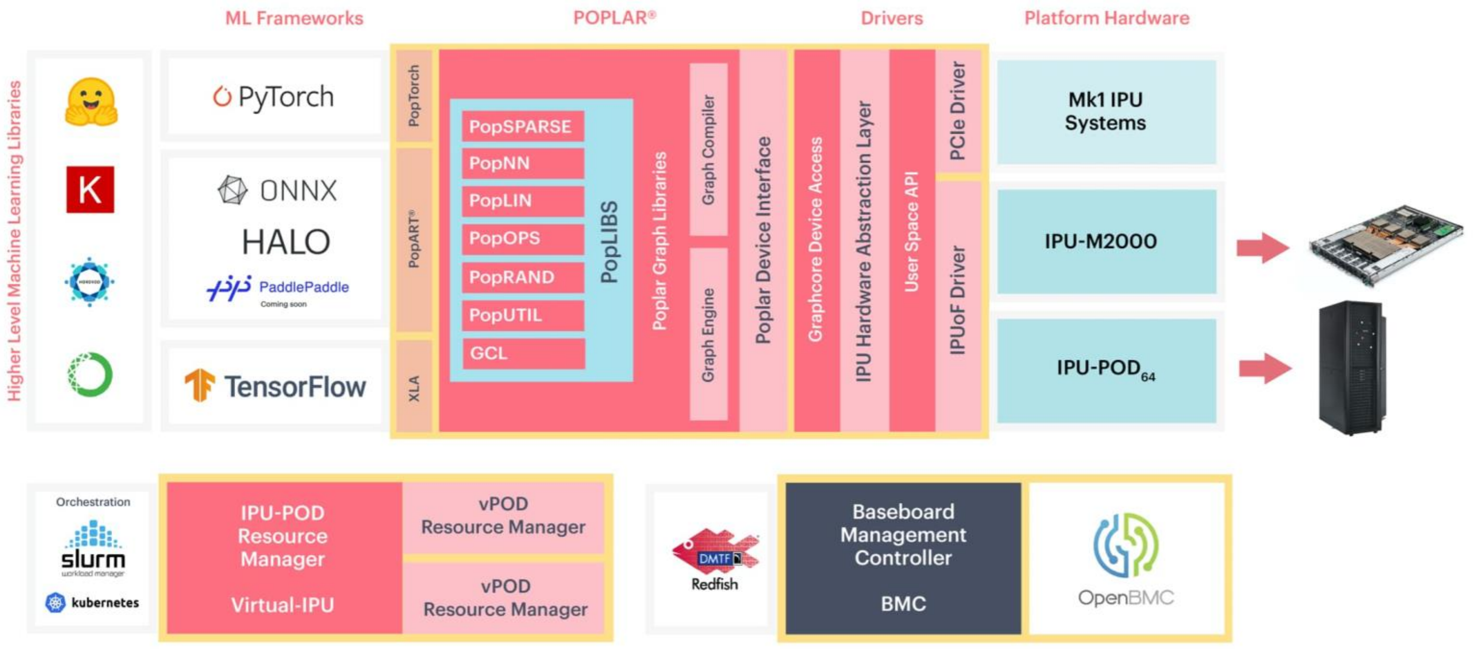
What we can use

Accelerated Frameworks (CPU)

- **BigDL** (distributed DL library for Apache Spark)
- **DNNL**: Intel’s open source library for DL applications
- **OpenVINO** toolkit: for computer vision
- **Intel DL Boost**: for running popular DL frameworks on CPU
- **Intel Caffe** (for Xeon)
- **nGraph**: open source C++ library, compiler and runtime for DL frameworks.
- **PlaidML**: tensor compiler for DL on OpenCL
  - Also for GPUs…

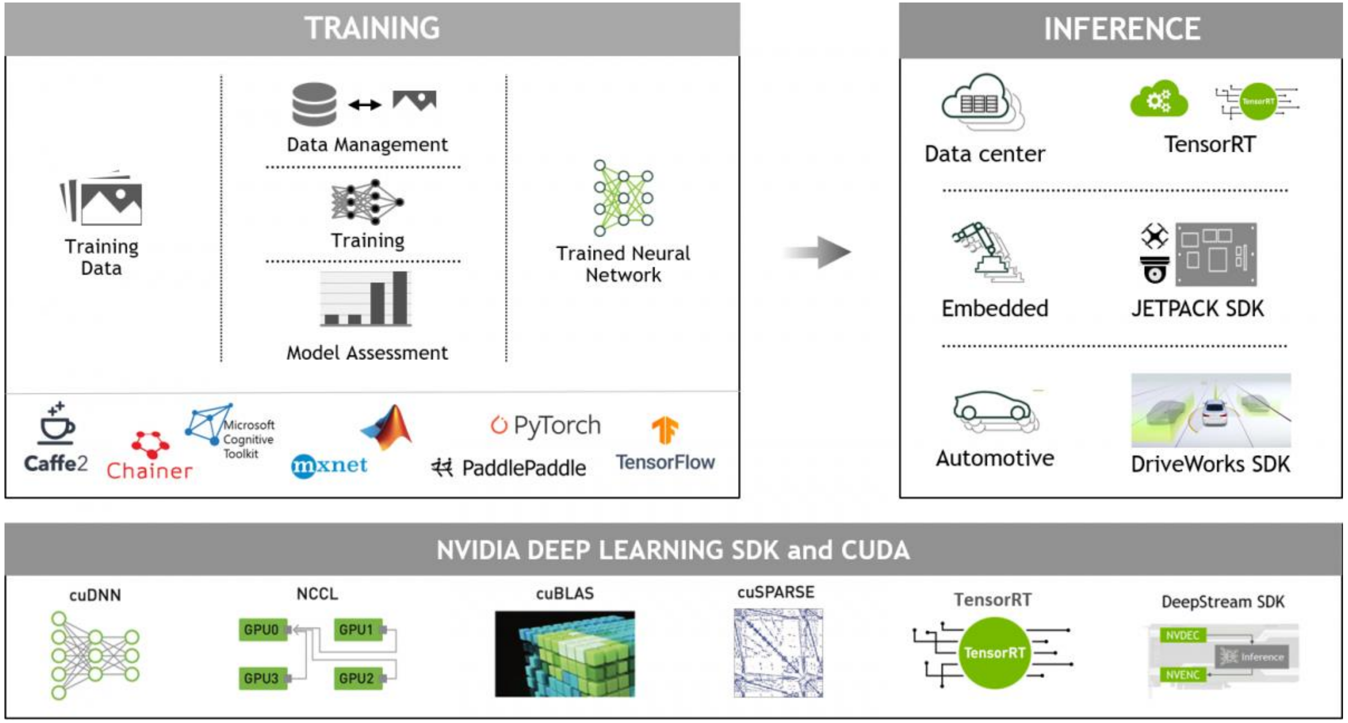
Accelerated Frameworks (IPU)

- Poplar SDK



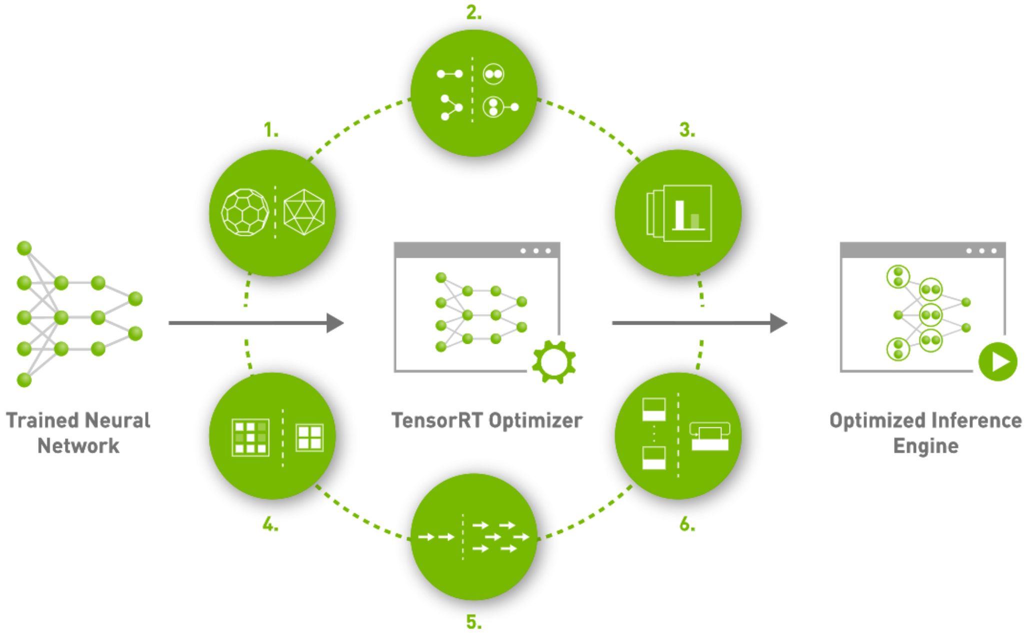
Accelerated Frameworks (GPU)

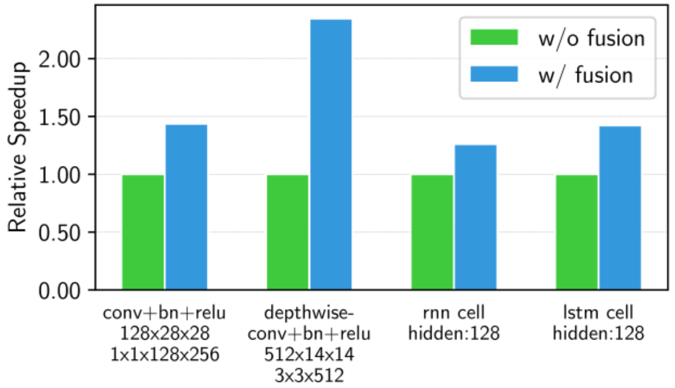
- NVIDIA



Accelerated Frameworks (Inference)

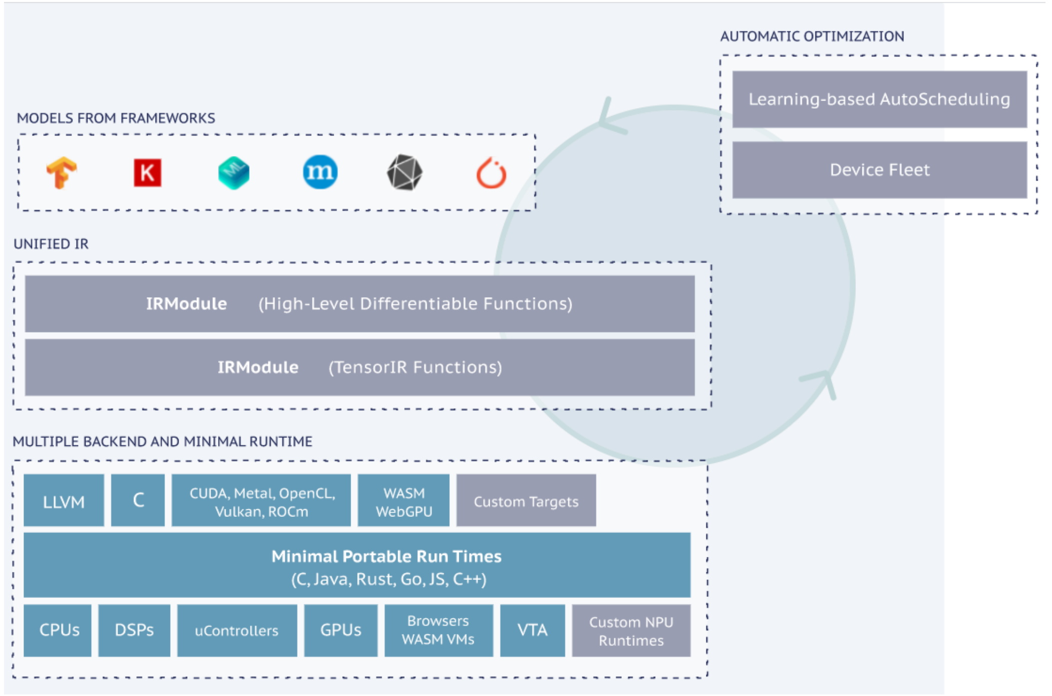
- Optimization of computational graphs for inference is key.
- E.g. Int8 can be used only for inference





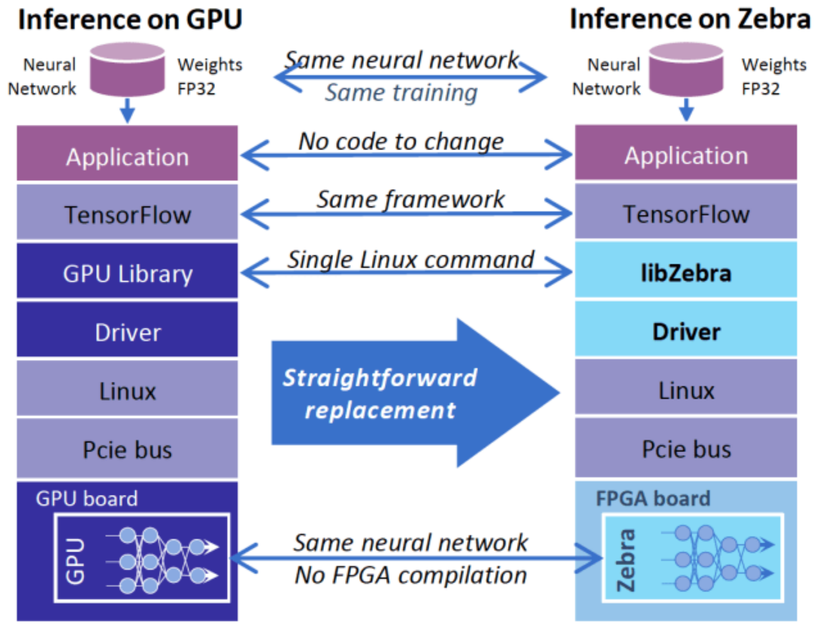
Accelerated Frameworks (Inference)

- **Apache TVM**
  - Compilation of DL models
  - Generate and optimize models for better performance
  - CPUs, GPUs, FPGAs…


AUTOMATIC OPTIMIZATION
Learning-based AutoScheduling
Device Fleet
MODELS FROM FRAMEWORKS
UNIFIED IR
IRModule (High-Level Differentiable Functions)
IRModule (TensorIR Functions)
MULTIPLE BACKEND AND MINIMAL RUNTIME
LLVM
C
CUDA, Metal, OpenCL, Vulkan, ROCm
WASM WebGPU
Custom Targets
Minimal Portable Run Times (C, Java, Rust, Go, JS, C++)
CPUs
DSPs
uControllers
GPUs
Browsers WASM VMs
VTA
Custom NPU Runtimes

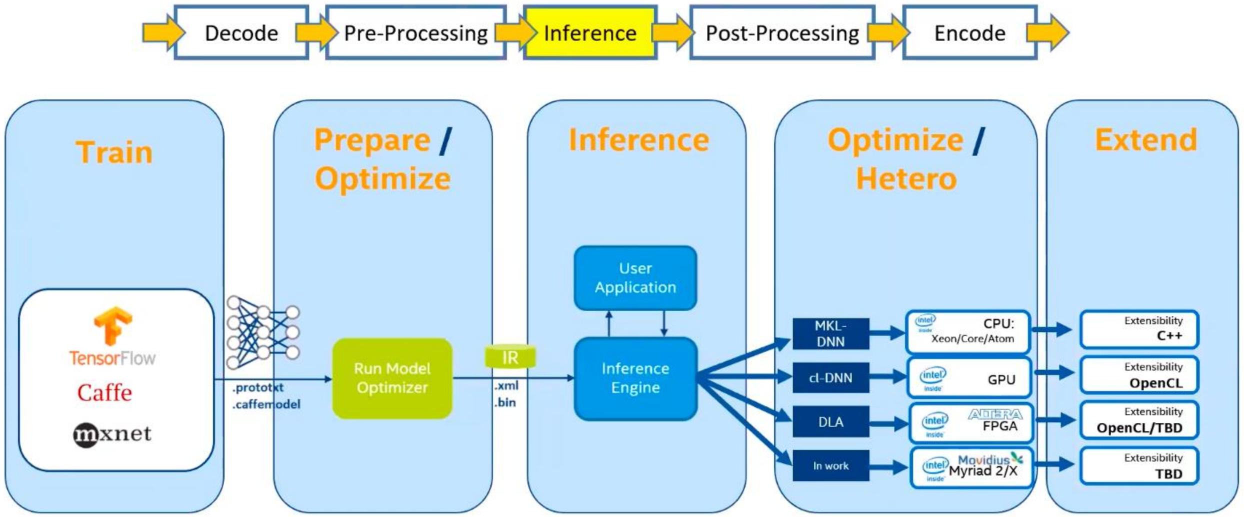
Accelerated Frameworks (Inference)

- Libraries like Zebra can help to use FPGAs for inference



Accelerated Frameworks (OpenVINO)

- Intel’s framework for computer vision: CPU, NPU, FPGAs…

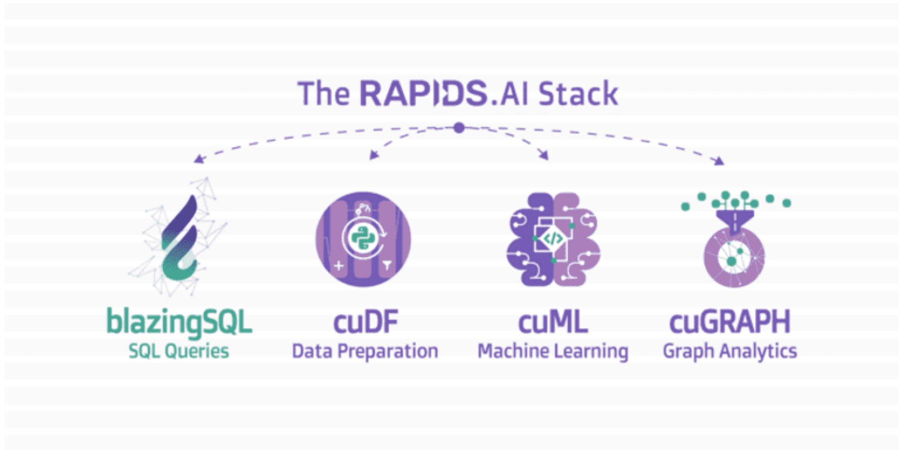


Accelerated Frameworks (GPU)

- What about other **Machine Learning algorithms and models** rather than Deep Learning?
  - KNN, K-means, Random Forest, Gradient Boosting, etc.
  - ASIC are only good for DNN, cannot be re-programmed
  - FPGAs can be adapted, but programming is hard
  - CPUs can have large delays for other ML models
  - GPUs are parallel devices that are flexible to execute a wide range of applications!

Accelerated Frameworks (GPU)

- NV-legate: replacement of Numpy and Pandas on scalable multi-GPU systems
- RAPIDS: set of Data Science libraries written for the GPU
  - **cuPy**: clone of Numpy
  - **cuDF**: clone of Pandas
  - **cuML**: clone of scikitlearn
  - **cuGraph**: NetworkX
  - **cuXfilter**: plotly, matplotlib
  - **cuSpatial, cuSignal, cuStreamz**
  - **BlazingSQL**

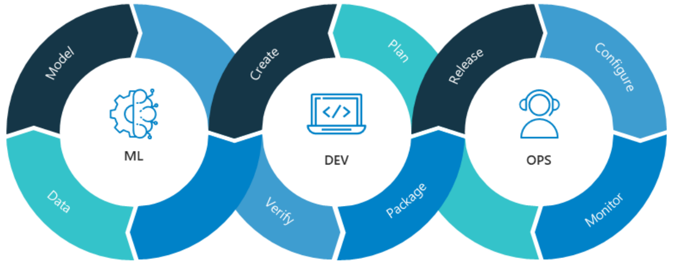


MACHINE LEARNING AS A SERVICE

Let’s go to the cloud…

MLAAS

- **Machine Learning as a Service**:
  - Cloud-based full stack AI platforms
- **MLOps**: approach to ML lifecycle management:
  - Data gathering, model creation (software development lifecycle, continuous integration/continuous delivery), orchestration, deployment, health, diagnostics, governance, and business metrics.
- MLOps can be offered as MLaaS



MLAAS (services)

- MLaaS services and key players:
  - **Natural language processing**: Amazon Comprehend, Azure Web Language Model API, Google Cloud Natural Language API
  - **Speech recognition**: Amazon Transcribe, Azure Custom Speech Service, Google Dialogflow Enterprise Edition
  - **Computer vision**: Amazon Rekognition, Azure Custom Vision Service, Google Cloud Vision API
  - **AI platforms**: Amazon Sagemaker, Azure Machine Learning Studio, Google Cloud Machine Learning Engine

https://analyticsindiamag.com/what-is-machine-learning-as-a-service-mlaas/

MLAAS

- **AWS** will offer their ASICs in EC2 instances
- **GCP** has MLOps services.
- GCP offers AutoML, and an API in Keras
    - AutoML covers hyperparameter tuning, algorithm selection, feature engineering.
- **Azure** also offers AutoML and MLOps services
- Others: **IBM Watson ML, bigML**

MLAAS (Free Environments)

- **Free Jupyter Notebook environments**:
  - Google Colaboratory:
    - up to 12 hours, K80, T4, P100 GPUs
    - With pro version, up to 24 hours and more RAM
  - Kaggle kernels:
    - Up to 9 hours, P100 GPUs. For R and Python
  - BlazingSQL:
    - Free for one GPU, paid for multi-GPUs
    - For RAPIDS
  - Gradient Community:
    - Up to 6 hours, public your notebook (up to 5).
    - M4000 and P5000 GPUs.

THE END

~~Packing~~ Wrapping up

Conclusions

- Hardware at a glance (approximately)…

	CPU	GPU	FPGA	ASIC
Flexibility	V. GOOD	GOOD	MEDIUM?	BAD
Software stack	V. GOOD	V. GOOD	MEDIUM	MEDIUM?
Power	MEDIUM	MEDIUM	V. GOOD	V. GOOD
Delay	GOOD	MEDIUM	GOOD	V. GOOD
Inference	GOOD	GOOD	V. GOOD	V. GOOD
Training	BAD	V. GOOD	GOOD?	GOOD
More than DNN	GOOD	V. GOOD	MEDIUM	BAD

Conclusions

- Standarization of ML workflows with MLOps
- Services on the cloud with MLaaS: pay as you use
  - At different levels: application, model, code
- Data on the cloud
- Trends to AutoML

References

- J. Dean. The Deep Learning Revolution and Its Implications for Computer Architecture and Chip Design (2019) https://arxiv.org/abs/1911.05289
- G. Sapunov. Hardware for Deep Learning series of posts (2020): https://blog.inten.to/hardware-for-deep-learning-current-state-and-trends-51c01ebbb6dc
- https://deepai.org/publication/tvm-end-to-end-optimization-stack-for-deep-learning
- https://neptune.ai/blog/best-machine-learning-as-a-service-platforms-mlaas

Thank you very much

- Let’s discuss
- mdelamor@us.es
- www.cs.us.es/~mdelamor