Deep Learning

Hardware and Software

Learning goals

- GPU training for accelerated learning
- Software for hardware support
- Deep learning software platforms

Hardware for Deep Learning

HARDWARE FOR DEEP LEARNING

- Deep NNs require special hardware to be trained efficiently.
- The training is done using **G**raphics **P**rocessing **U**nits (GPUs) and a special programming language called CUDA.
- For most NNs, training on standard CPUs takes very long.

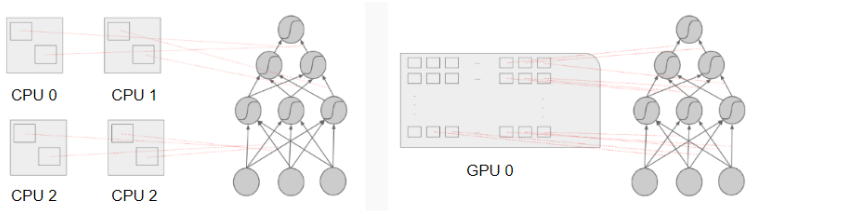


Figure: *Left:* Each CPU can do 2-8 parallel computations. *Right:* A single GPU can do thousands of simple parallel computations.

GRAPHICS PROCESSING UNITS (GPUS)

- Initially developed to accelerate the creation of graphics
- Massively parallel: identical and independent computations for every pixel
- Computer Graphics makes heavy use of linear algebra (just like neural networks)
- Less flexible than CPUs: all threads in a core concurrently execute the same instruction on different data.
- Very fast for CNNs, RNNs need more time
- Popular ones: GTX 1080 Ti, RTX 3080 / 2080 Ti, Titan RTX, Tesla V100 / A100
- Hundreds of threads per core, few thousand cores, around 10 teraFLOPS in single precision, some 10s GBs of memory
- Memory is important - some SOTA architectures do not fit GPUs with <10 GB

TENSOR PROCESSING UNITS (TPUS)

- Specialized and proprietary chip for deep learning developed by Google
- Hundreds of teraFLOPS per chip
- Can be connected together in *pods* of thousands TPUs each (result: hundreds of **peta**FLOPS per pod)
- Not a consumer product – can be used in the Google Cloud Platform (from >1.35 USD / TPU / hour) or Google Colab (free!)
- Enabled impressive progress : DeepMind's AlphaZero for Chess became world champion after just 4h of training on 5064 TPUs

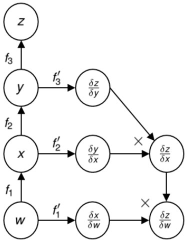
AND EVERYTHING ELSE...

- With such powerful devices, memory/disk access during training becomes the bottleneck
  - Nvidia DGX-1: Specialized solution with eight Tesla V100 GPUs, dual Intel Xeon, 512 GB of RAM, 4 SSD disks of 2TB each
- Specialized hardware for on-device inference
  - Example: Neural Engine on the Apple A11 (used for FaceID)
  - Keywords/buzzwords: *Edge computing* and *Federated learning*

Software for Deep Learning

SOFTWARE FOR DEEP LEARNING

- CUDA is a very *low level* programming language and thus writing code for deep learning requires a lot of work.
- Deep learning (software) frameworks:
  - Abstract the hardware (same code for CPU/GPU/TPU)
  - Automatically differentiate all computations
  - Distribute training among several hosts
  - Provide facilities for visualizing and debugging models
  - Can be used from several programming languages
  - Based on the concept of *computational graph*



z
f_3
y
f_3'
$\frac{\delta z}{\delta y}$
f_2
x
f_2'
$\frac{\delta y}{\delta x}$
$\times$
$\frac{\delta z}{\delta x}$
f_1
w
f_1'
$\frac{\delta x}{\delta w}$
$\times$
$\frac{\delta z}{\delta w}$

SOFTWARE FOR DEEP LEARNING

Tensorflow

- Popular in the industry
- Developed by Google and open source community
- Python, R, C++ and Javascript APIs
- Distributed training on GPUs and TPUs
- Tools for visualizing neural nets, running them efficiently on phones and embedded devices.

Keras

- Intuitive, high-level **wrapper** of Tensorflow for rapid prototyping
- Python and (unofficial) R APIs

SOFTWARE FOR DEEP LEARNING

Pytorch

- (Most) Popular in academia
- Supported by Facebook
- Python and C++ APIs
- Distributed training on GPUs

MXNet

- Open-source deep learning framework written in C++ and cuda (used by Amazon for their Amazon Web Services)
- Scalable, allowing fast model training
- Supports flexible model programming and multiple languages (C++, Python, Julia, Matlab, JavaScript, Go, R, Scala, Perl)