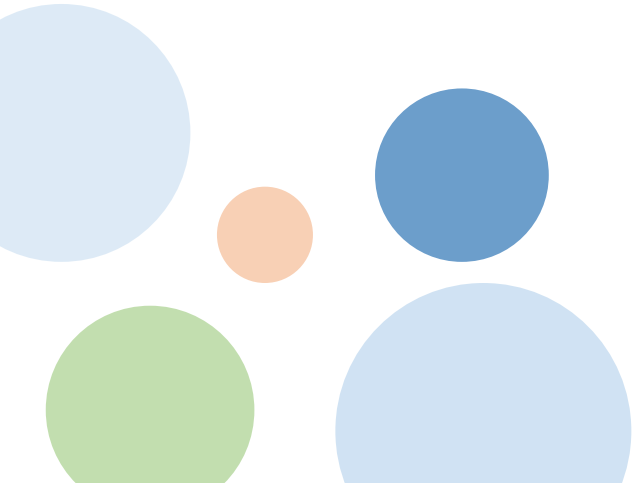
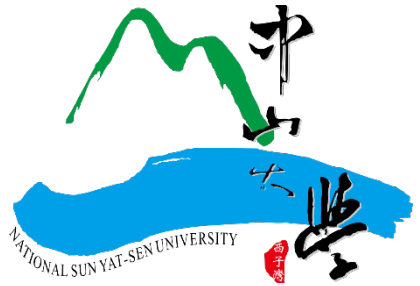
CNN Accelerators



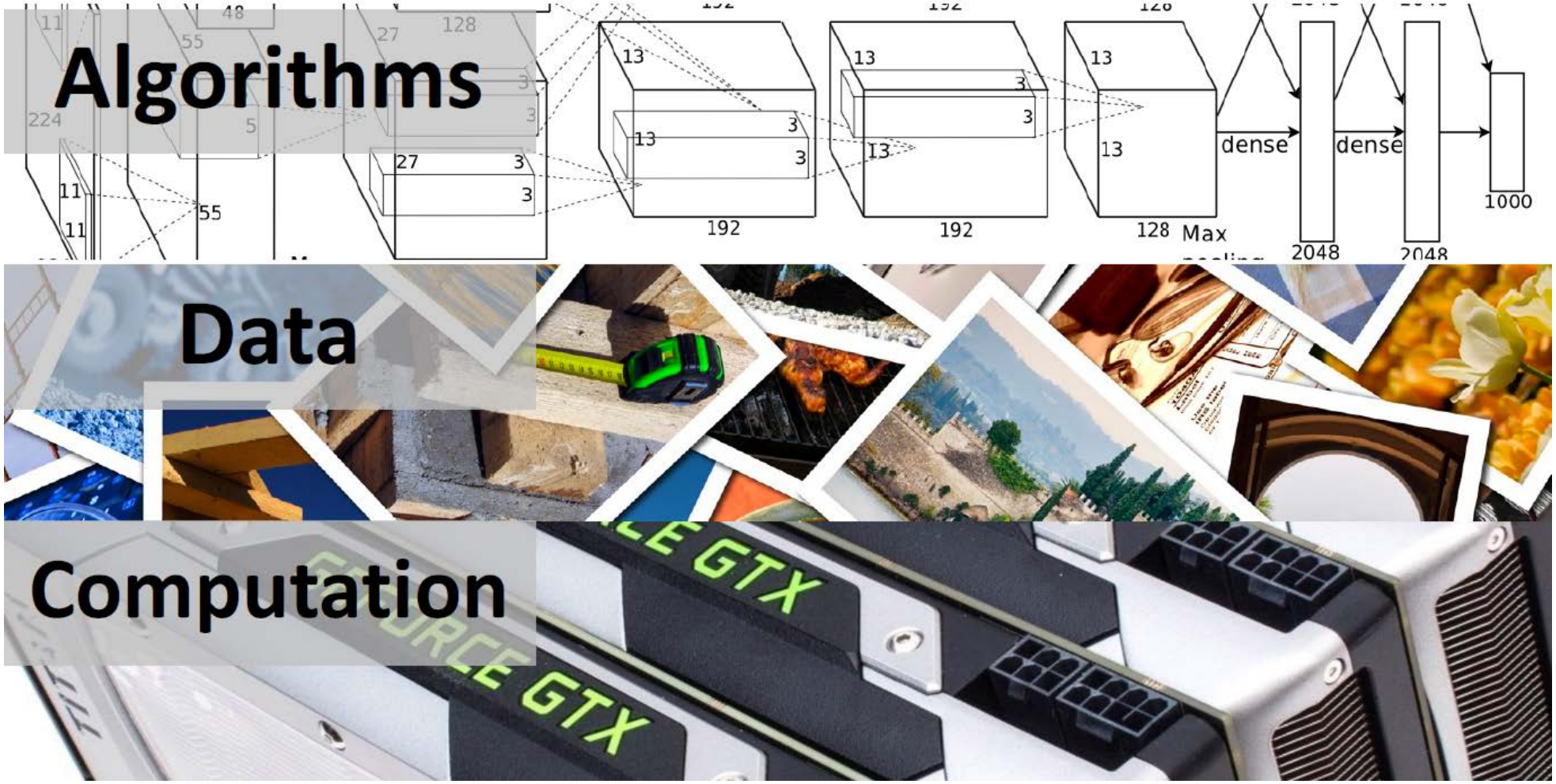


References and Credits

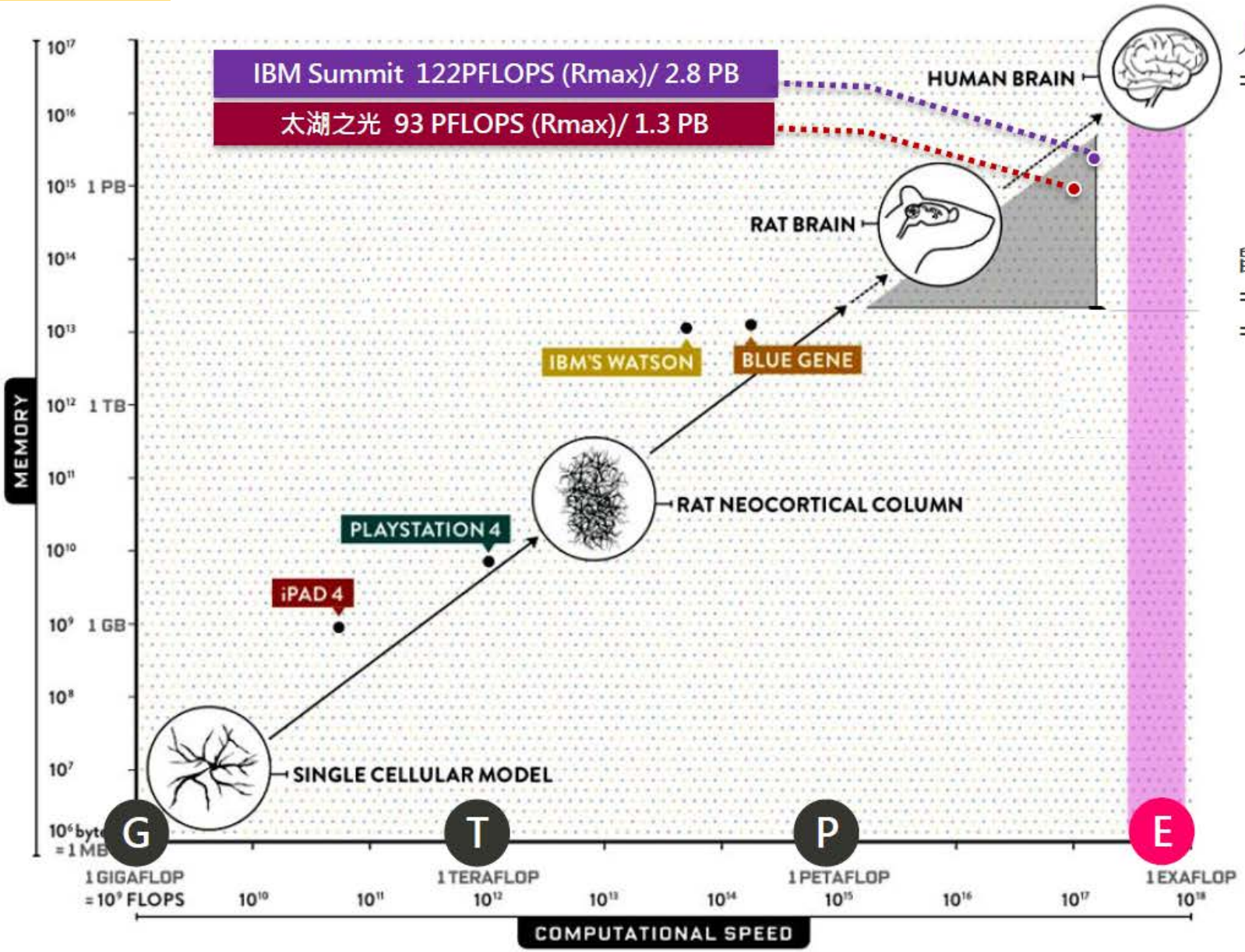
- Computing Trend for AI, 梁伯嵩博士（聯發科）, 2018
- Stanford CS231n, “Convolutional Neural Networks for Visual Recognition”
  - by Fei-Fei Li, Justin Johnson, and Serena Yeung
- MIT 6.S191, “Deep Learning”
  - by Alexander Amini, and Ava Soleimany
- UVA Deep Learning , Univ. of Amsterdam
  - by Efstratios Gavves
- CMSC 35264 Deep Learning, Univ. of Chicago
  - by Shubhendu Trivedi and Risi Kondor
- Deep Learning for Computer Vision
  - by 台大資工系 莊永裕 教授

Ingredients in Deep Learning

- Algorithm (Model) + Training Data + Computation
  - CNN models + Big Data + Accelerators



Computation Complexity



人類大腦 ~1 Exaflops
= ~1000 鼠類大腦

鼠類大腦
= ~100 mesocircuits 皮質中型迴路
= ~10000 neocortical columns 新皮質柱

鼠類 新皮質柱
= ~10000 neurons

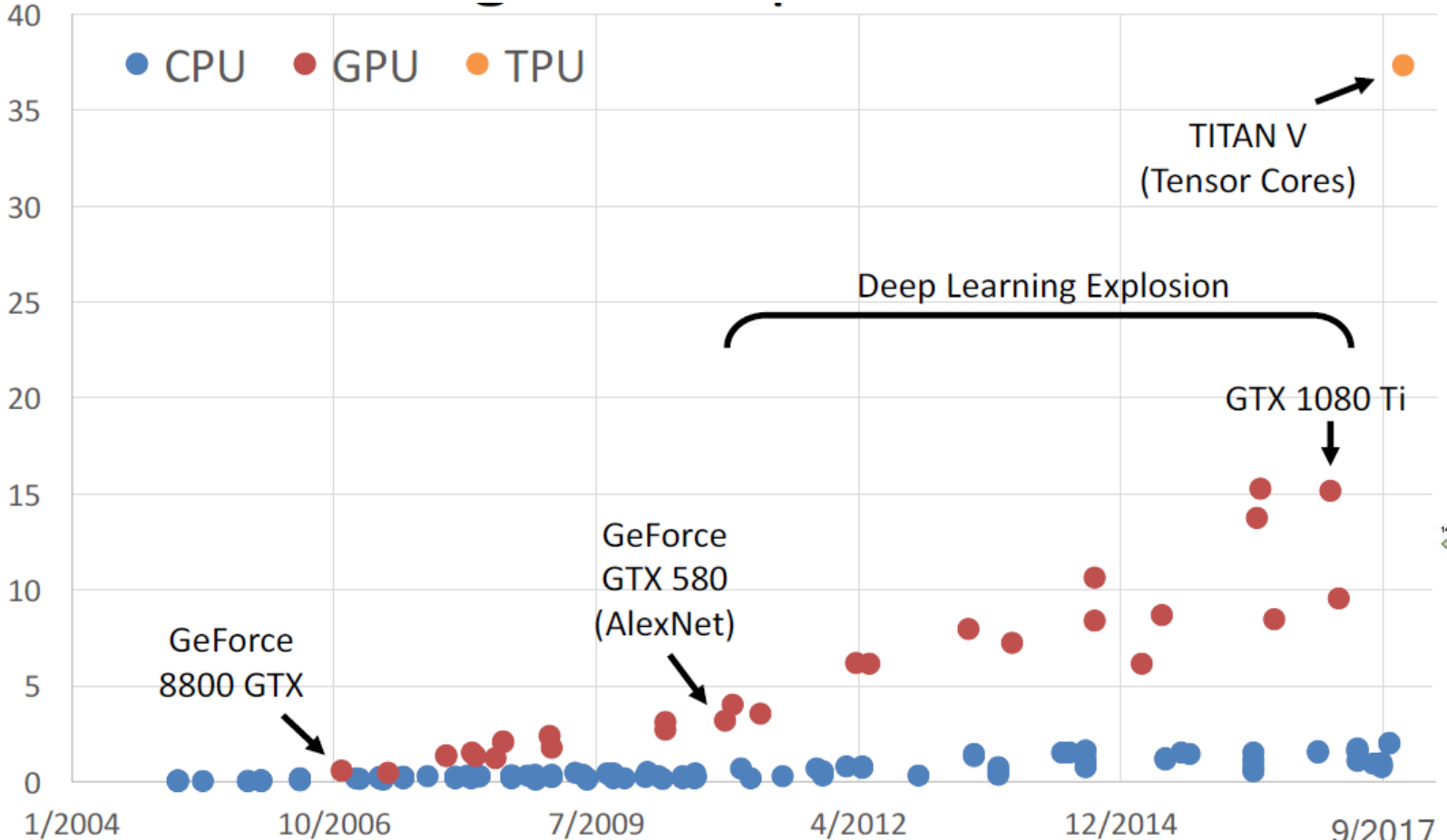
Source:
(1) Wired, 2013
(2) "Computer modelling: Brain in a box", Nature, 482, 456–458 , 23 Feb 2012
(3) Top500.org
Illistrated/Modified by Bor-Sung Liang, 2018.10

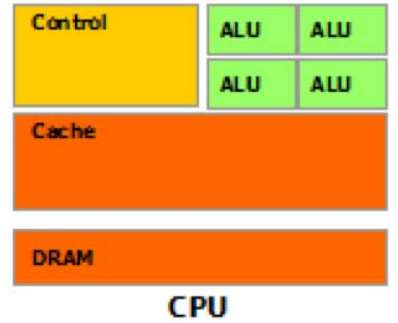
Inference vs. Training Time

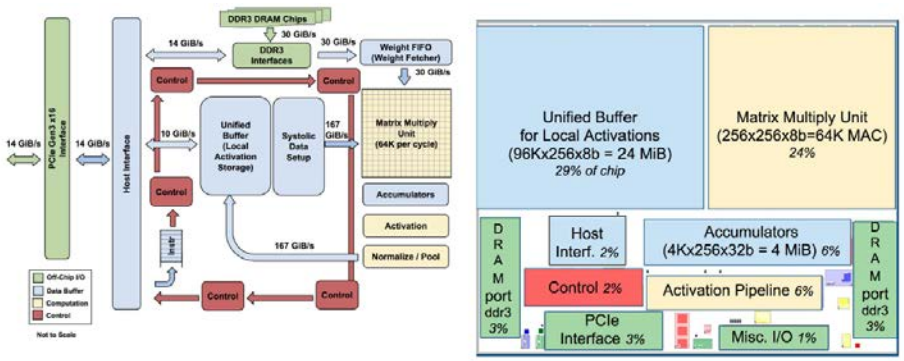
- Training takes much more time than inference
- Computer Vision
  - inference: ~ 7.5 Giga (10^9) ops
  - training: ~ 1 Exa (10^{18}) ops,
- AlphaZero
  - inference: ~ 368,000 Giga ops
  - training: ~56,304 Exa ops

GFLOP/Dollars in CPU, GPU, TPU

- CPU: multi-core (~8) with 128-bit floating-point operations(FLOPs)/core
- GPU: many-core (~5,000) with 32-bit FLOPs / core
- TPU: ultra-many-core (~100,000) with 8/16-bit fixed-point operations / core







Google TPU

CPU vs. GPU vs. TPU

	Cores	Clock Speed	Speed (TFLOPs)	Power
CPU (Intel Core i7-7700k)	4 (8 threads with hyperthreading)	4.2 GHz	~540 FP32	91 W
GPU (NVIDIA GTX 1080 Ti)	3584 CUDA	1.6 GHz	~11.4 FP32	250 W
GPU (NVIDIA GTX 2080 Ti)	4352 CUDA, 544 Tensor	1.55 GHz	~12 FP32	250 W
GPU (NVIDIA TITAN V)	5120 CUDA, 640 Tensor	1.5 GHz	~14 FP32 ~112 FP16	250 W
GPU (NVIDIA V100)	5120 CUDA, 640 Tensor	1.53 GHz	~16 FP32 ~125 for ML	300 W
TPU (Google Cloud TPU v3)	8 Tensor cores (8 x 128 x 128 x 2 = 262,144 FP16 MAC)	0.7 GHz	~91.8 for FP8 ~180 for ML	75 W

CPU：Fewer cores, but each core is much faster and much more capable; great at sequential tasks

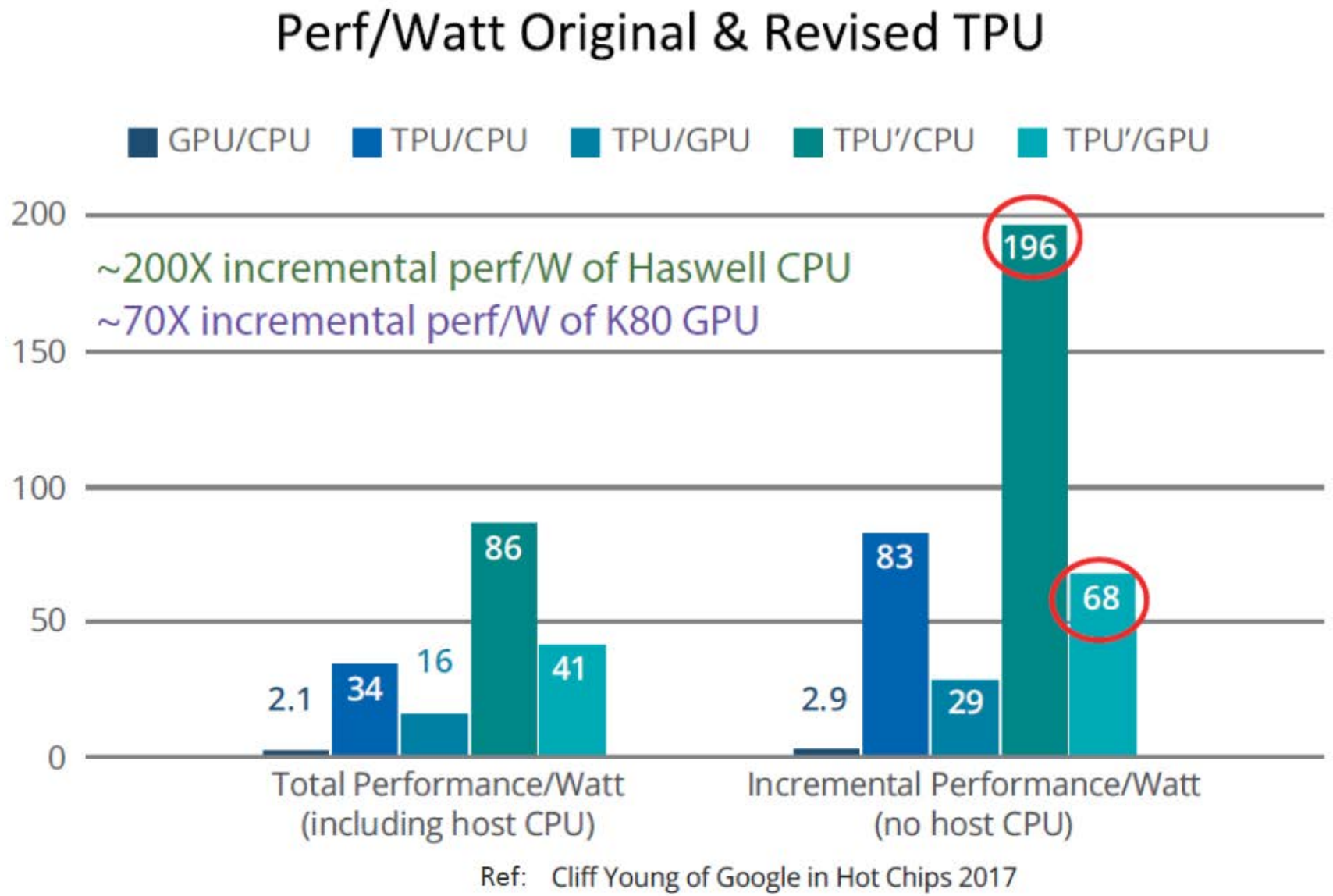
GPU：More cores, but each core is much slower and "dumber"; great for parallel tasks

TPU：Specialized hardware for deep learning

fastest supercomputers: 1. IBM Summit: 125,000 TFLOPs(10^{12}), 15,000,000 W !!!

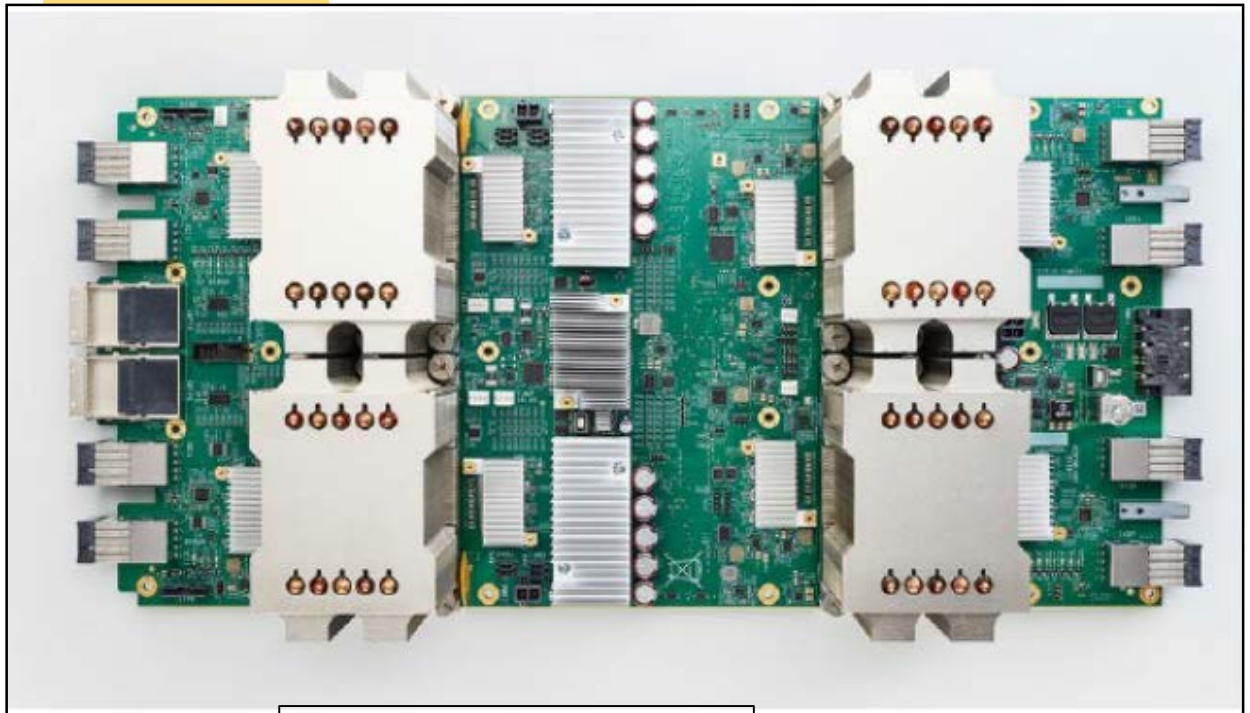
2. 神威·太湖之光: 92,000 TFLOPs, 15,371,000 W

Google TPU vs. CPU/GPU



Ref: Cliff Young of Google in Hot Chips 2017

TensorFlow: Tensor Processing Units



Google Cloud TPU
= 180 TFLOPs!

Power: 40 W

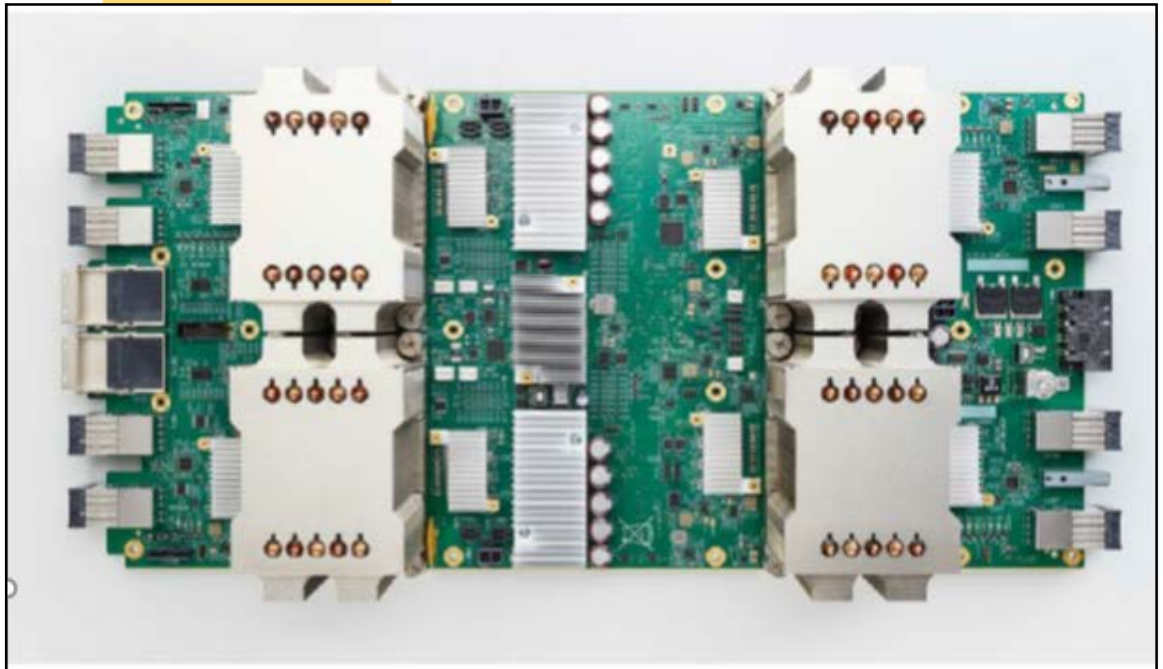
NVIDIA Tesla V100
= 125 TFLOPs

Power: 300 W

NVIDIA Tesla P100 = 11 TFLOPs
NVIDIA GTX 580 = 0.2 TFLOPs

But Power in mobile devices: <1 W !!!

TPU in Google Cloud



Google Cloud TPU
= 180 TFLOPs of compute!

Google Cloud TPU Pod
= 64 Cloud TPUs
= 11.5 PFLOPs of compute!

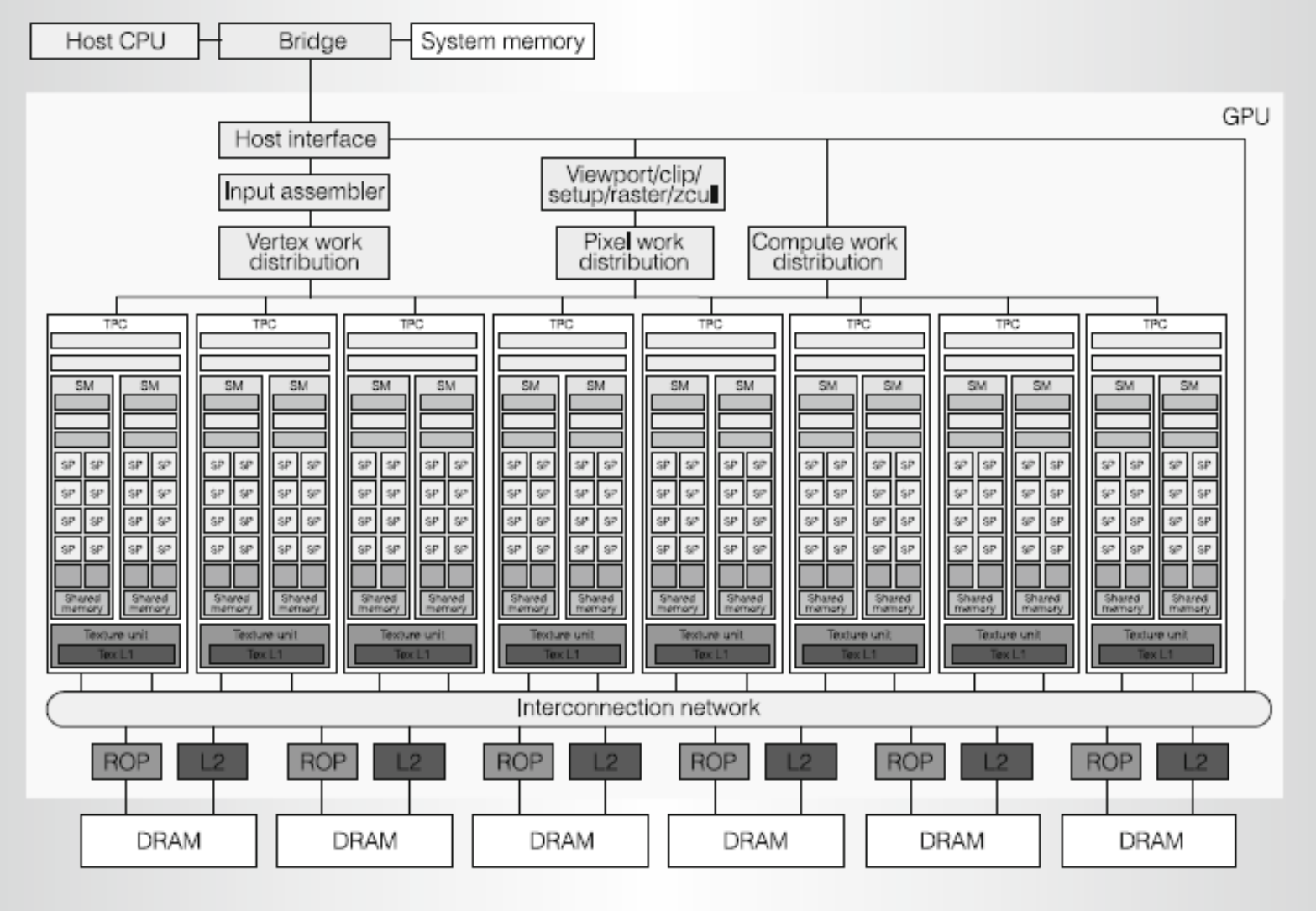
the fastest supercomputer of the world in 2018/11

Rank	Site	System	Cores	Rmax (TFlop/s)	Rpeak (TFlop/s)	Power (kW)
1	DOE/SC/Oak Ridge National Laboratory United States	**Summit** - IBM Power System AC922, IBM POWER9 22C 3.07GHz, NVIDIA Volta GV100, Dual-rail Mellanox EDR Infiniband IBM	2,397,824	143,500.0	200,794.9	9,783

https://www.tensorflow.org/versions/master/programmers_guide/using_tpu

Nvidia Tesla GPU

- Scalable unified architecture based on GeForce 8-series

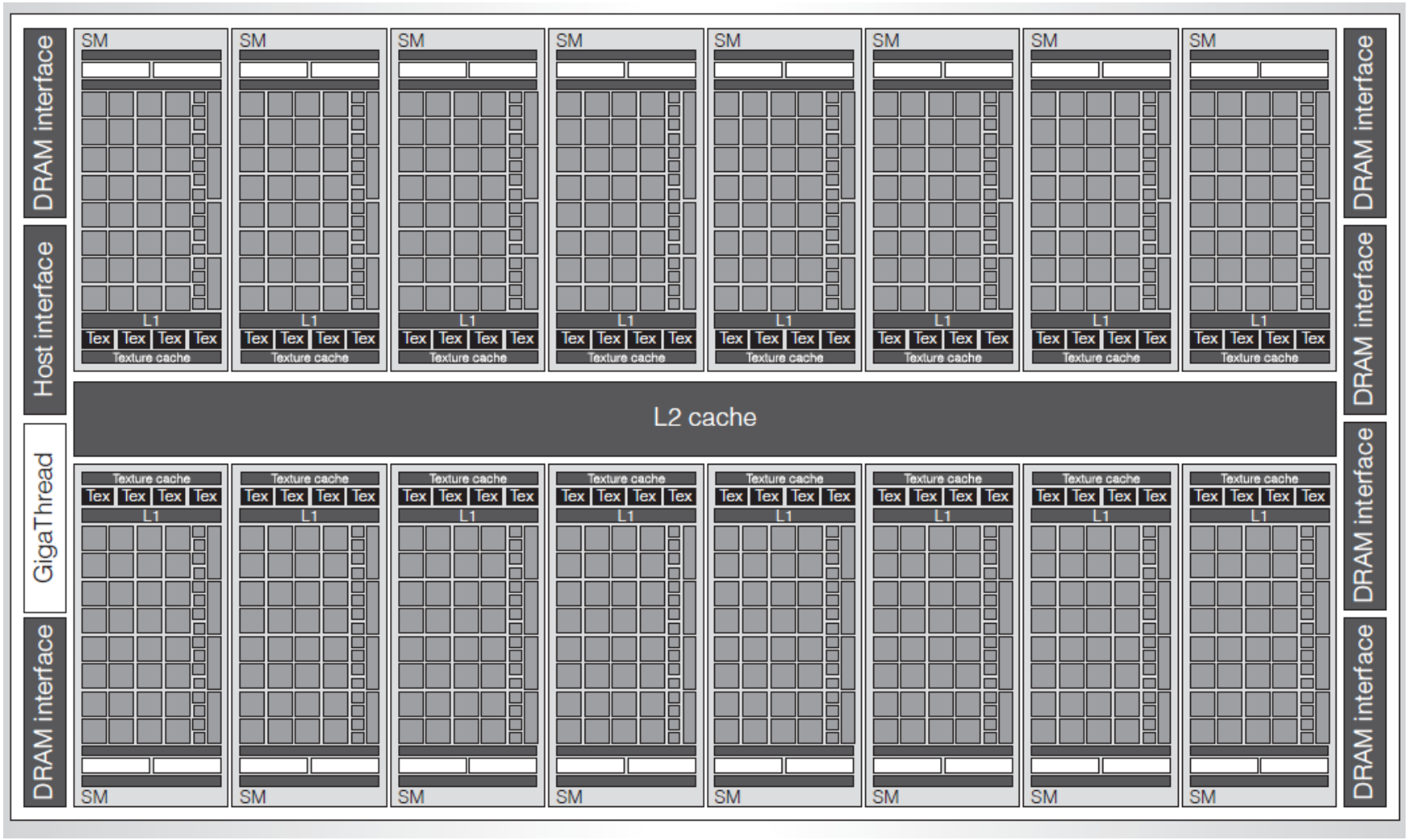


NVIDIA Telsa, A Unified Graphics and Computing Architecture, *IEEE Micro*, Mar./Apr. 2008.

Patterson and Hennessy, *Computer Organization and Design, The Hardware/Software Interface*, 4th ed., Appendix A, 2009.

NVIDIA Fermi GPU

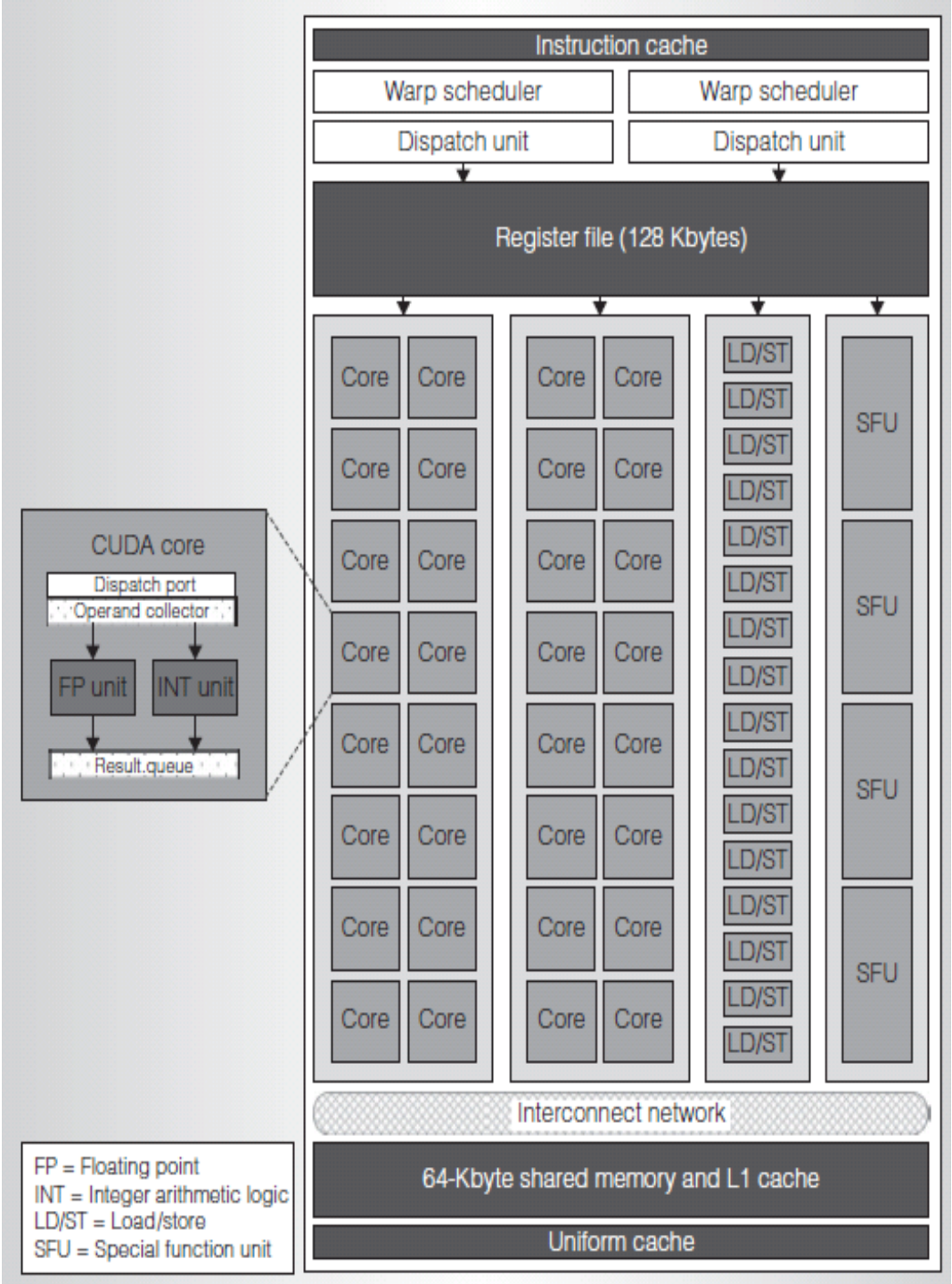
- 16 Streaming Multiprocessors (SM)
- 32 CUDA cores in each SM



The GPU Computing Era, *IEEE Micro*, Mar.-Apr. 2010.

Fermi Streaming Multiprocessor

- 32 CUDA processor cores
- 16 load/store units
- 4 special function units
- 64KB shared memory/L1 cache
- 128KB register file
- Up to 1536 concurrent threads



Nvidia V100 GPU

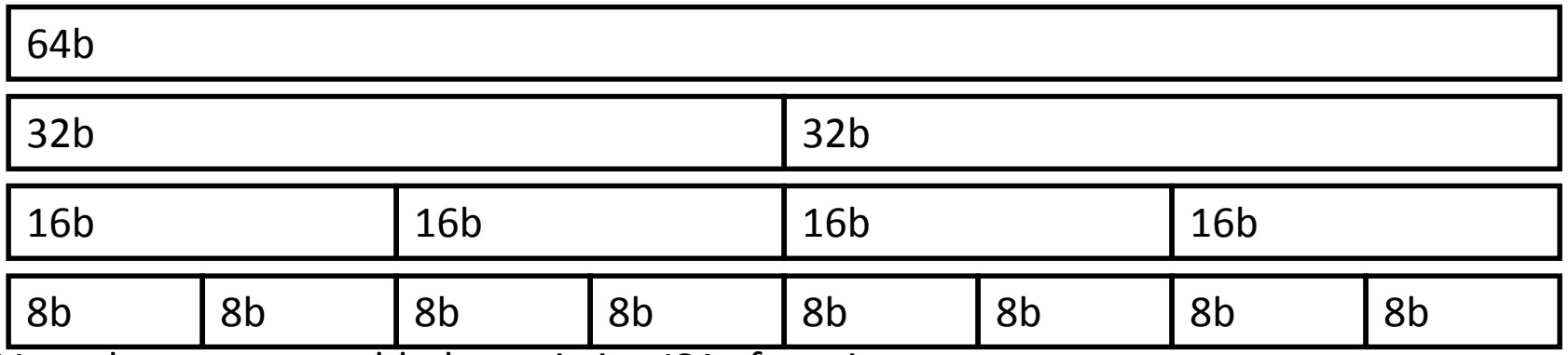
- 640 Tensor Cores (TC)
  - each TC operates on a 4x4 matrix at 64 ops/clock
- 120 TFLOPS for training and inference

$$D = \begin{pmatrix} A_{0,0} & A_{0,1} & A_{0,2} & A_{0,3} \\ A_{1,0} & A_{1,1} & A_{1,2} & A_{1,3} \\ A_{2,0} & A_{2,1} & A_{2,2} & A_{2,3} \\ A_{3,0} & A_{3,1} & A_{3,2} & A_{3,3} \end{pmatrix} \begin{pmatrix} B_{0,0} & B_{0,1} & B_{0,2} & B_{0,3} \\ B_{1,0} & B_{1,1} & B_{1,2} & B_{1,3} \\ B_{2,0} & B_{2,1} & B_{2,2} & B_{2,3} \\ B_{3,0} & B_{3,1} & B_{3,2} & B_{3,3} \end{pmatrix} + \begin{pmatrix} C_{0,0} & C_{0,1} & C_{0,2} & C_{0,3} \\ C_{1,0} & C_{1,1} & C_{1,2} & C_{1,3} \\ C_{2,0} & C_{2,1} & C_{2,2} & C_{2,3} \\ C_{3,0} & C_{3,1} & C_{3,2} & C_{3,3} \end{pmatrix}$$

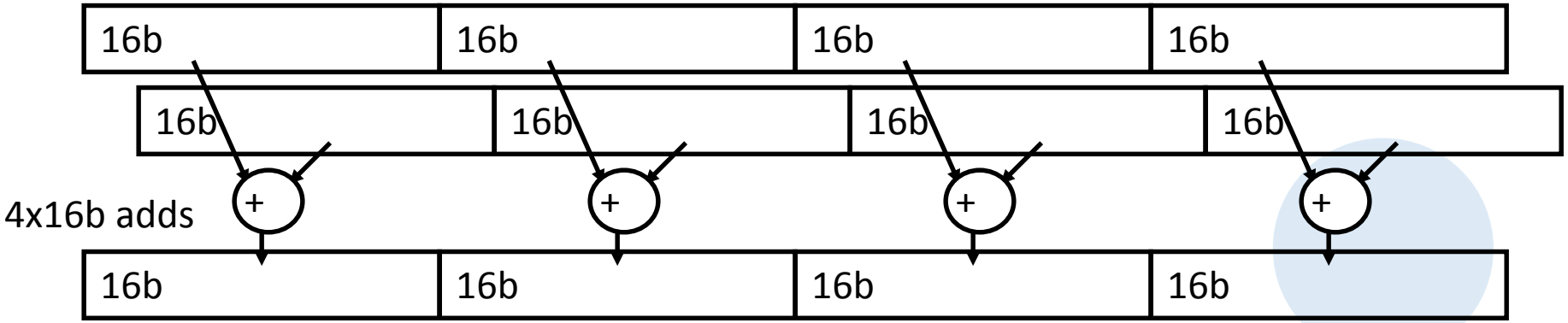
FP16 or FP32 FP16 FP16 FP16 or FP32

SIMD (Single Instruction Multiple Data)

Nvidia Cudnn INT8

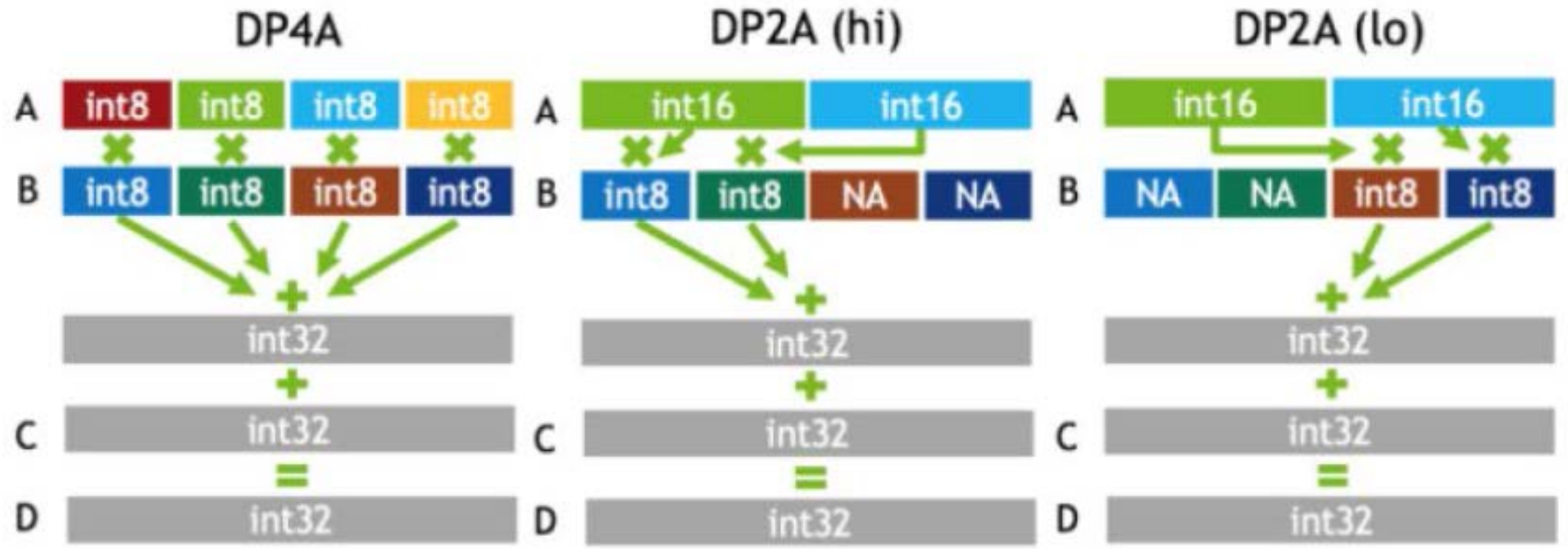


- Very short vectors added to existing ISAs for microprocessors
- Use existing 64-bit registers split into 2x32-b or 4x16-b or 8x8-b
  - Lincoln Labs TX-2 from 1957 had 36b datapath split into 2x18b or 4x9b
  - Newer designs have wider registers
    - 128b for PowerPC Altivec, Intel SSE2/3/4
    - 256b for Intel AVX
- Single instruction operates on all elements within register



Nvidia Int8

- Low-bit accuracy (<16-b) in most deep learning applications
  - sometimes, even binary (1-b) is enough, e.g., BWN
- one Nvidia CUDA core has hardware of 32-b x 32-b MAC hardware
  - one 32-b x 32-b MAC (Multiply-ACcumulate)
  - two 16-b x 16-b MAC
  - four 8-b x 8-b MAC



ASIC (Application Specific IC) DNN HW Accelerators

- Deep learning hardware accelerator for “edge” devices
  - power < 1W
- Several benchmarks
  - DianNao series (CAS中國科學院, 2014~2016)
  - Angel-Eye (Tsinghua北京清大, 2016~2018)
  - DNA, GNA, RNA, Thinker (Tsinghua北京清大, 2017~2019)
  - Eyeriss v1, v2 (MIT 麻省理工學院, 2017~2019)
  - EIE, ESE (Stanford史丹福大學, 2016~2017)
  - TPU (Google谷歌, 2017~2018)
  - DNPU, UNPU (KAIST南韓科大, 2017~2019)
  - …

Comparison of HW DNN

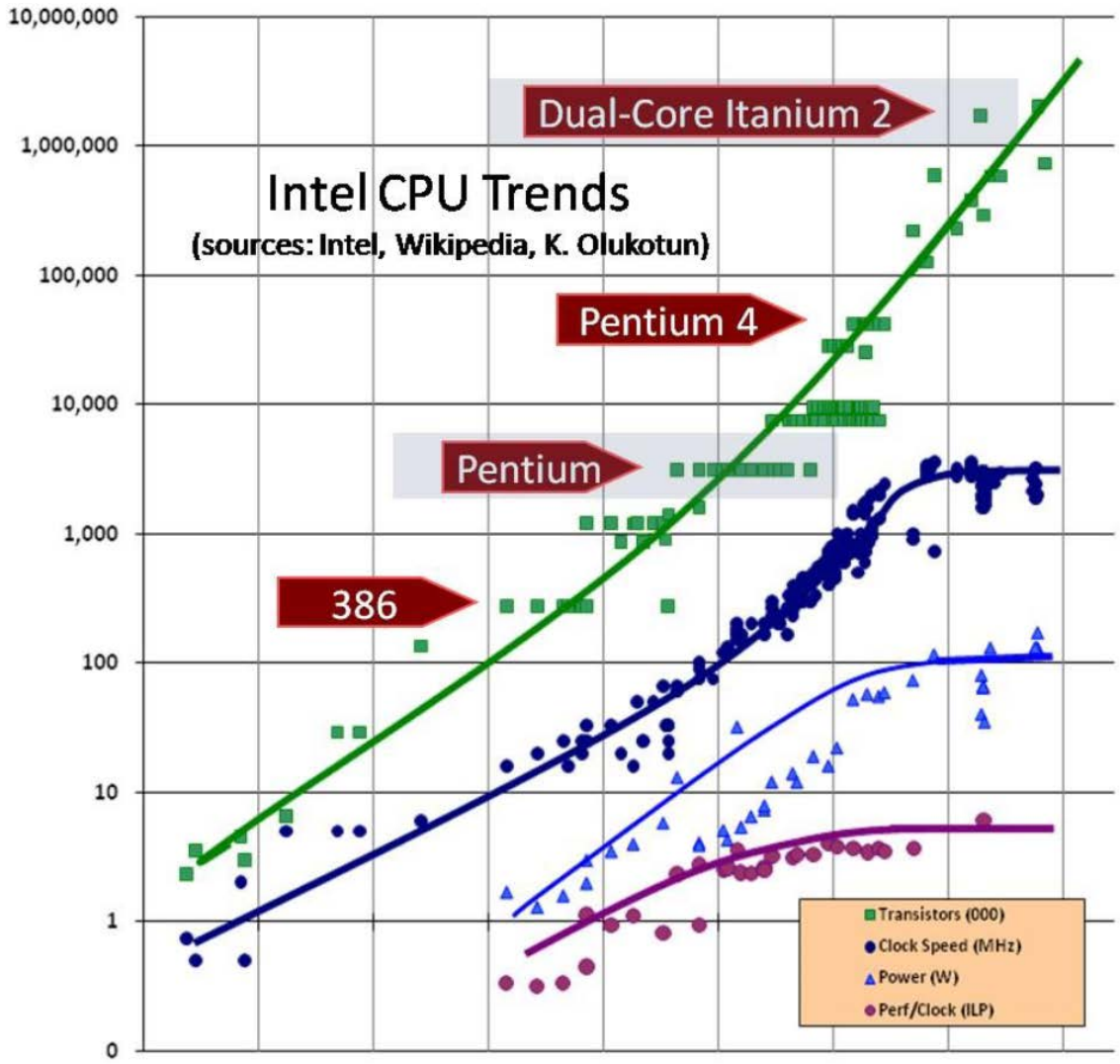
	Tech. (nm)	bits	f(MHz)	Power(mW)	speed (GOP/s)	pwr.eff. (GOP/s/W)	SRAM (KB)	Multi.	layer types	Parallelism Types(註一)	features
DianNao (2014)	65	16	1,000	0.485	452	932	44	256	CNN	OCP (no data reuse)	MAT
DaDianNao (2014)	28	16	606	16	5,580	349	36,000	4,096	CNN	OCP (no data reuse)	NoC, MAT
ShiDianNao (2015)	65	16	1,000	0.32	606	1,893	288	-	CNN	OCP (input data transfer inter PE)	2D, MAT
CambriconX (2016)	65	16	1,000	0.95	544	573	56	-	CNN FC	OCP (no data reuse)	1D, MAT
EIE (2016)	45	16	800	0.6	102	170	10,368	64	FC	OCP	1D, MAC
Eyeriss (2016, 2017)	65	16	200	0.28	60	23.1	83	336	CNN FC	WP 其餘平行不明	2D, MAC
Origami (2017)	65	12	500	0.5	196	437	43	196	CNN	WP、OCP	1D, MAT
TPU (2017)	28	8	700	40,000	92,000	2,300	28,000	65,536	CNN FC LSTM	平行不明	2D, MAC
DNPU (2017)	65	4~16	50~200	0.063	300@16-b	4,200	280	768@16-b	CNN FC LSTM	ICP、WP、OCP	2D, MAT
PS-ConvNet (2017)	40	4,8,12,16	204	0.287@16-b	74	270@16-b	148	256	CNN	WP、OCP	2D, MAC
DNA (2017)	65	16	200	0.48	194	406	280	1,024	CNN FC	ICP、WP、OCP	2D, MAC

	Tech. (nm)	bits	f(MHz)	Power(mW)	speed (GOP/s)	pwr.eff. (GOP/s/W)	SRAM (KB)	Multi.	layer types	Parallelism Types(註一)	features
FlexFlow (2017)	65	16	1,000	-	420	-	64	256	CNN	ICP、WP、OCP	2D, MAC, MAT
DSIP (2018)	65	16	250	0.153	16	105	140	64	CNN	ICP、WP、OCP	2D, MAC
UNPU (2018)	65	1~16	200	297	345.6@16-b	3,080	256	1,152	CNN FC LSTM	ICP、WP、OCP	2D, MAC, NoC
Think1 (2018)	65	8, 16	10~200	4~386	410	1,060~5,090	348	512	CNN FC LSTM	ICP、WP、OCP	2D, MAC
Think2 (2018)	28	1, 2, 4, 8, 16	20~400	3.4~20.8	410@(16, 1)-b	95,800@(16, 1)-b	224	32	CNN	ICP、OCP	MAT
GNA (2018)	28	8, 16	200	142	409.6	2,880	404	256	CNN DeCNN	ICP、WP、OCP、cross layer	2D, MAT
proposed	**45**	**1~24**	**500**	**0.25@8-b**	**1,000@8-b**	**4,000@8b-b**	**~200**	**64*9**	**CNN FC LSTM**	**opt. weight, act.**	**1D, 2D, NoC, MAC, MAT**
GPU Titan X	28	32f	1,075	210	5,991	29					
GPU K40	28	32f	560	250	1,783	7					
mGPU Tegra K1	28	32f	852	9	68	8					
CPU Intel Xeon	22	32f	2,900	130	97	1					
FPGA XC7Z045		16	150	9.6	137	14					
FPGA XC7Z020		8	214	3.5	84.3	24					

註一： OCP(output channel parallel) 、
ICP(input channel parallel)、
WP (window parallel)、
BP(batch parallel)

Moore's Law, Quantum Computer

- Moore's Law
  - double transistors / 18 month
  - power wall in 2005
  - ends at 1nm?, 0.3nm?
  - applications push technology
    - smart phone + cloud
    - IoT + big data
    - deep learning + AI
    - …
- Quantum Computer?
  - exponential parallelism
  - information security



10 µm – 1971
6 µm – 1974
3 µm – 1977
1.5 µm – 1982
1 µm – 1985
800 nm – 1989
600 nm – 1994
350 nm – 1995
250 nm – 1997
180 nm – 1999
130 nm – 2001
90 nm – 2004
65 nm – 2006
45 nm – 2007
32 nm – 2010
22 nm – 2012
14 nm – 2014
10 nm – 2017
7 nm – 2018
5 nm – ~2020
3 nm – ~2024

Software Frameworks

Keras

- quick to develop and test new ideas
- automatically compute gradients
- efficiently run on GPU (wrap cuDNN, cuBLAS, etc.)

Caffe (UC Berkeley) → **Caffe2** (Facebook)

Torch (NYU / Facebook) → **PyTorch** (Facebook)

Theano (U Montreal) → **TensorFlow** (Google)

PaddlePaddle (Baidu)

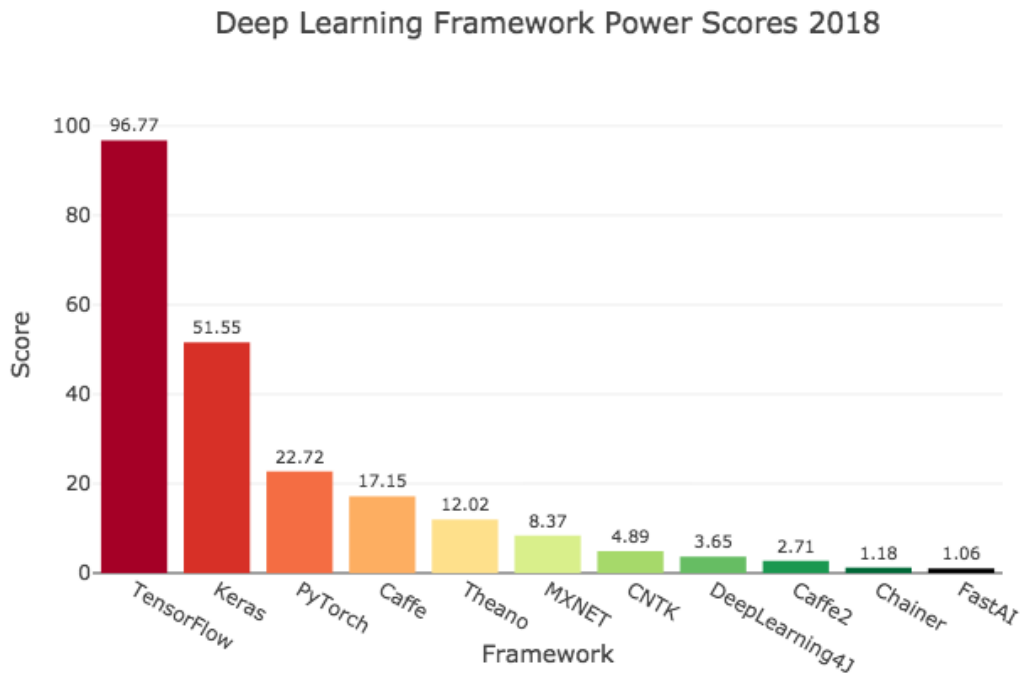
Chainer (Preferred Networks)

MXNet (Amazon)

Developed by U Washinton, CMU, MIT, Hong Kong U, etc but main framework of choice at AWS

CNTK (Microsoft)

Deeplearning4j



And others...

DL Frameworks

- Keras
  - . Keras sits on top of TensorFlow, Theano, or CNTK
- PyTorch
  - allows customization; supported by Facebook
- MXNET
  - incubated by Apache, and used by Amazon
- Chainer
  - developed by the Japanese company Preferred Networks
- FastAI
  - built on PyTorch; supported by Kaggle