nervana

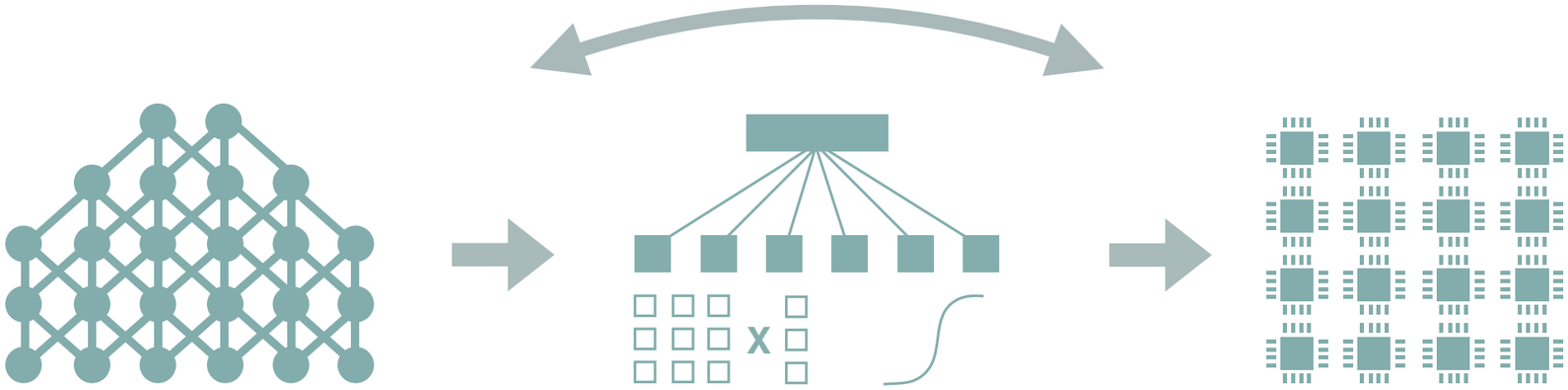
MAKING MACHINES SMARTER.™

Rethinking computation:
A processor architecture for machine intelligence

Amir Khosrowshahi
Co-founder and CTO, Nervana

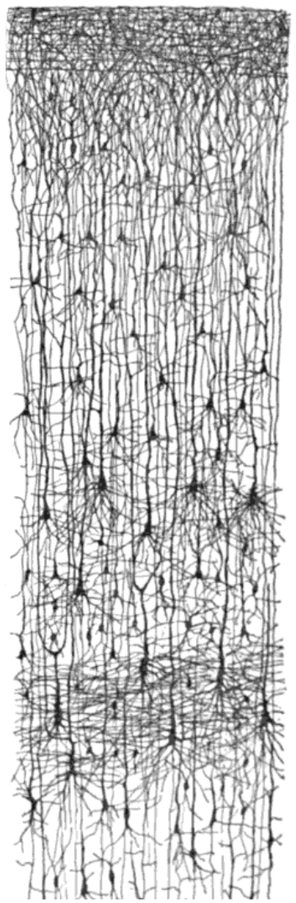
About nervana

- A platform for machine intelligence
  - enable deep learning at scale
  - optimized from algorithms to silicon



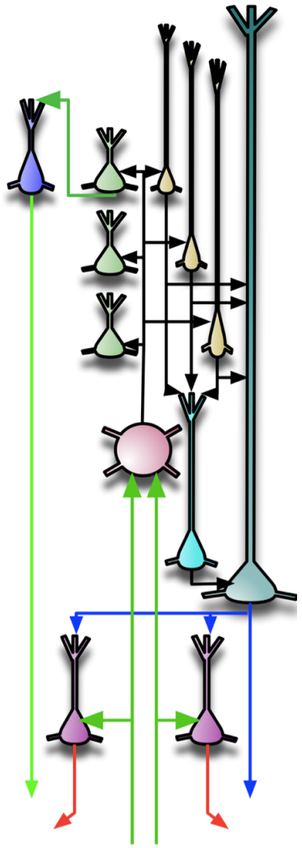
Model and substrate for computation

Hard!



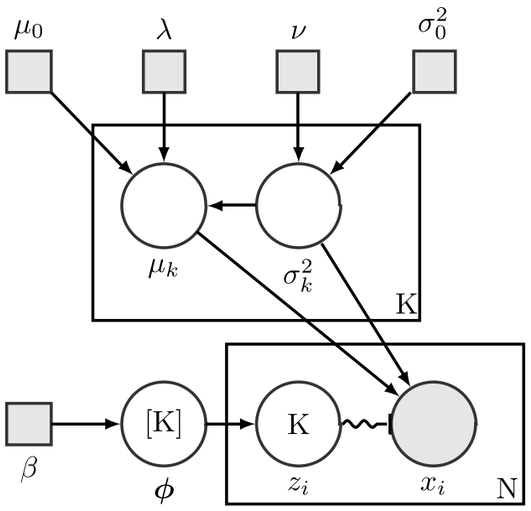
Mammalian cortex

?



Functional model

?



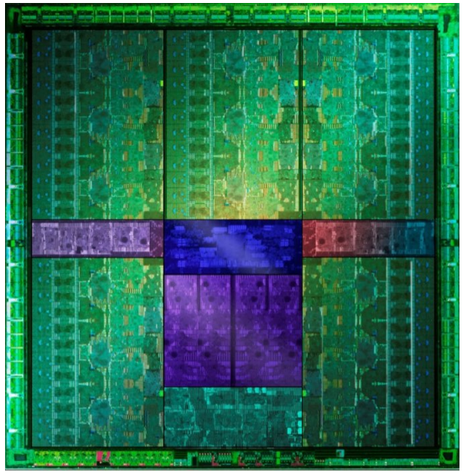
Machine learning model

nervana

Model and substrate for computation

Do this instead:

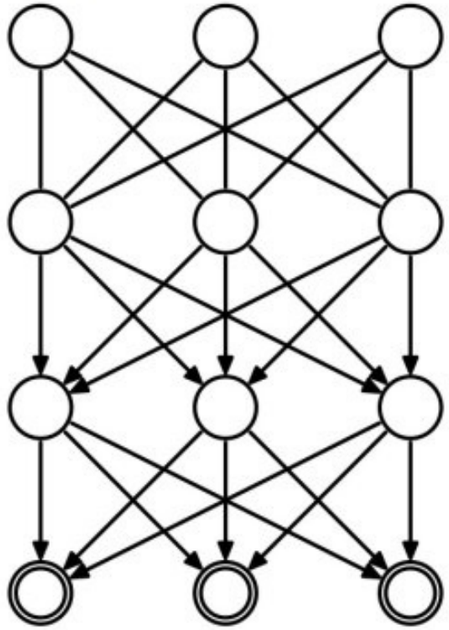
Feasible, but still hard.



Custom ASIC

```
model: !obj:models.MLP {
  num_epochs: 30,
  batch_size: &bs 100,
  layers: [
    &datalayer !obj:layers.DataLayer {
      name: d0,
      nout: 784,
    },
    !obj:layers.FCLayer {
      name: h0,
      nout: 100,
      lrule_init: *gdm,
      weight_init: *wt_init,
      activation: !obj:transforms.RectLin {},
    },
    &lastlayer !obj:layers.FCLayer {
      name: output,
      nout: 10,
      lrule_init: *gdm,
      weight_init: *wt_initb,
      activation: !obj:transforms.Logistic {},
    },
    &costlayer !obj:layers.CostLayer {
      name: cost,
      ref_layer: *datalayer,
      cost: !obj:transforms.CrossEntropy {},
    },
  ],
},
```

- Model description language
- Hardware abstraction layer
- Distributed primitives
- Compilers, drivers



Deep learning model

Application areas

Healthcare

Agriculture

Finance

Online Services

Automotive

Energy

nervana cloud

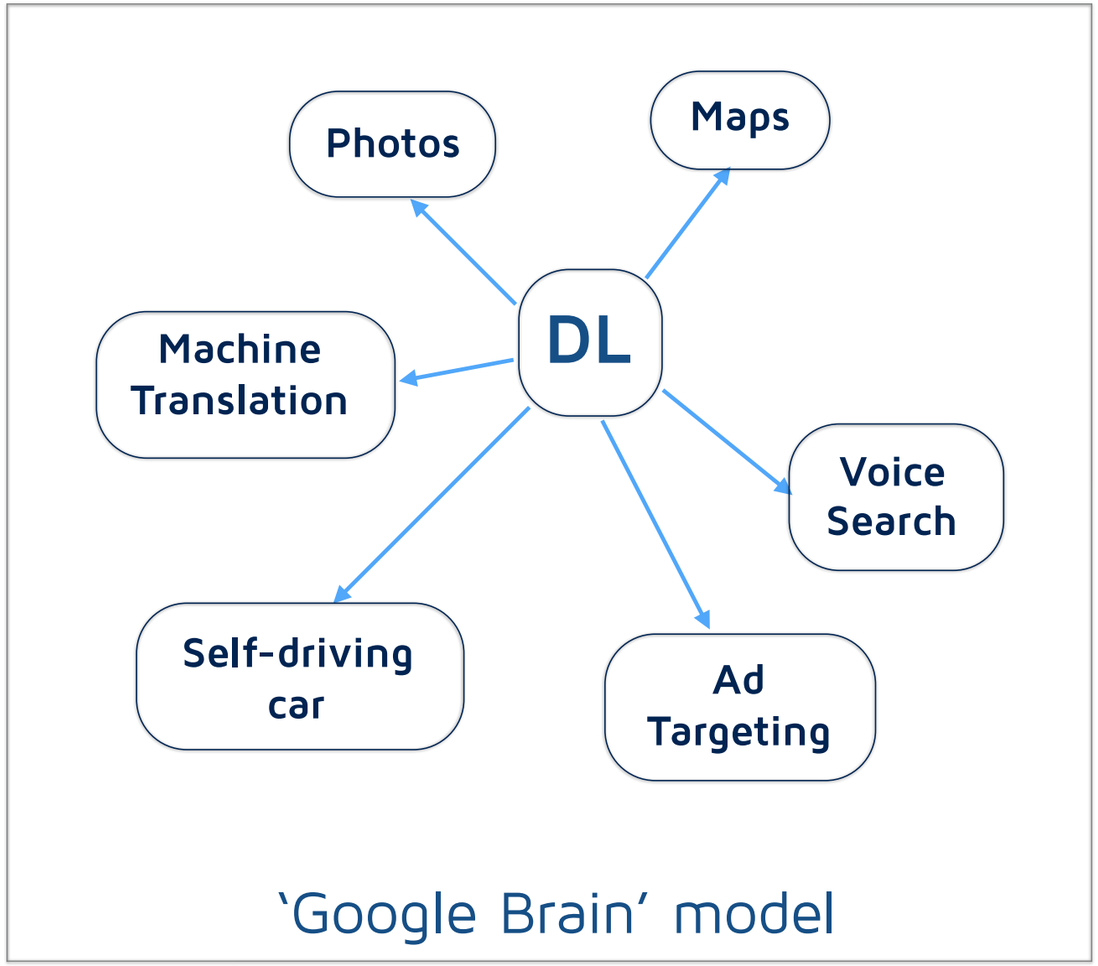
Data

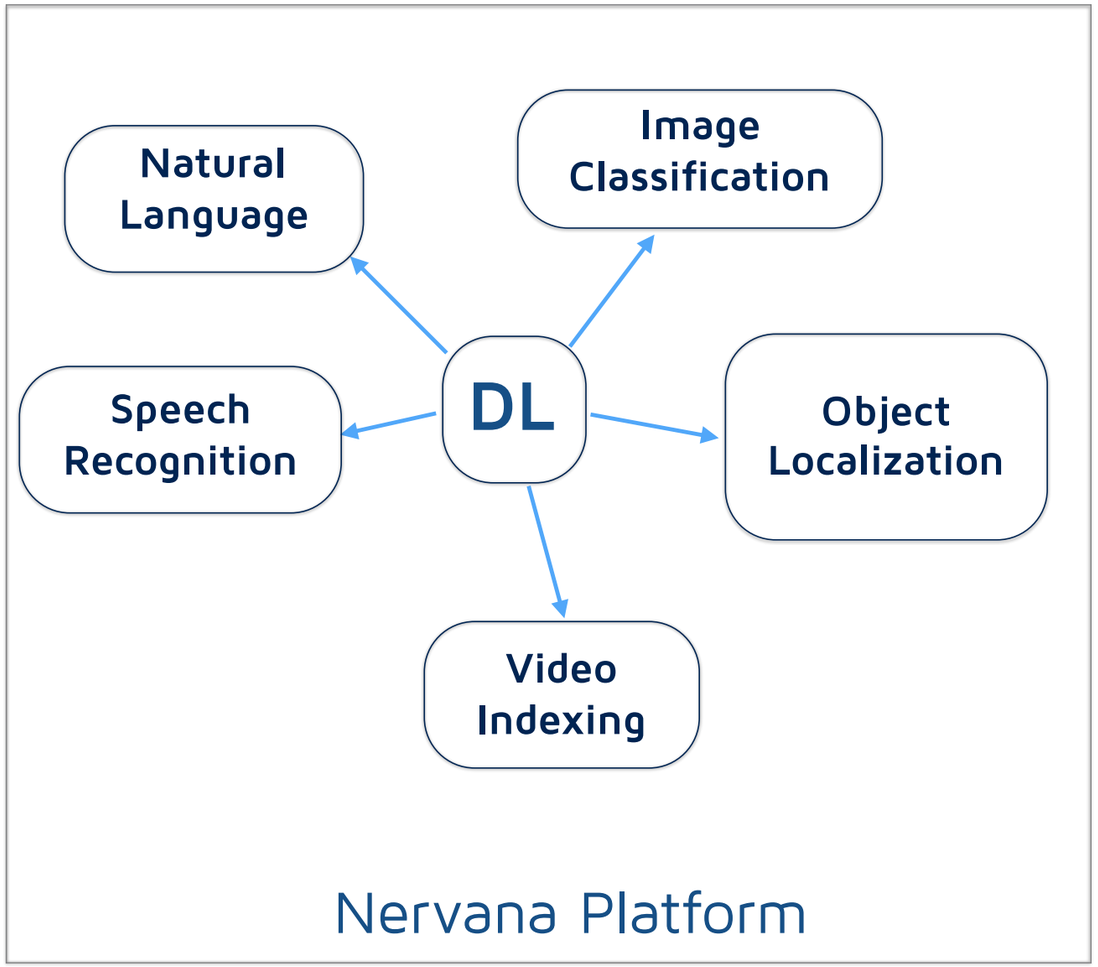
- Images
- Video
- Text
- Speech
- Tabular
- Time series

Cloud

import build train deploy

Deep learning as a core technology





nervana

nervana neon

- Fastest library

GoogleNet V1 - Input 128x3x224x224

Library	Class	Time (ms)	forward (ms)	backward (ms)
Nervana-neon-fp16	ConvLayer	230	72	157
Nervana-neon-fp32	ConvLayer	270	84	186
CuDNN[R4]-fp16 (Torch)	cudnn.SpatialConvolution	462	112	349
CuDNN[R4]-fp32 (Torch)	cudnn.SpatialConvolution	470	130	340
Chainer	Convolution2D	687	189	497
TensorFlow	conv2d	905	187	718
Caffe	ConvolutionLayer	1935	786	1148
CL-nn (Torch)	SpatialConvolutionMM	7016	3027	3988
Caffe-CLGreenTea	ConvolutionLayer	9462	746	8716

nervana neon

- Fastest library

GoogleNet V1 - Input 128x3x224x224

Library	Class	Time (ms)	forward (ms)	backward (ms)
Nervana-neon-fp16	ConvLayer	230	72	157
Nervana-neon-fp32	ConvLayer	270	84	186
NVIDIA / Facebook	cudnn.SpatialConvolution	462	112	349
NVIDIA / Facebook	cudnn.SpatialConvolution	470	130	340
Chainer	Convolution2D	687	189	497
Google	conv2d	905	187	718
Berkeley	ConvolutionLayer	1935	786	1148
CL-nn (Torch)	SpatialConvolutionMM	7016	3027	3988
Caffe-CLGreenTea	ConvolutionLayer	9462	746	8716

nervana neon

- Fastest library
- Model support

Models	Domains
• Convnet • RNN, LSTM • MLP • DQN • NTM	• Images • Video • Speech • Text • Time series

nervana neon

- Fastest library
- Model support
- Cloud integration

Running locally:

```
% python rnn.py                                  # or neon rnn.yaml
```

Running in nervana cloud:

```
% ncloud submit -py rnn.py                       # or -yaml rnn.yaml

% ncloud show <model_id>

% ncloud list

% ncloud deploy <model_id>

% ncloud predict <model_id> <data>               # or use REST api
```

nervana neon

- Fastest library
- Model support
- Cloud integration
- Multiple backends

Backends

- CPU
- GPU
- Multiple GPUs
- Parameter server
- (Xeon Phi)
- nervana TPU

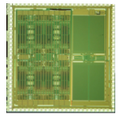
nervana neon

- Fastest library
- Model support
- Cloud integration
- Multiple backends
- Optimized at assembler level

```
j1c17 => "--:-:6:-:1  \@P1 LDS.U.64 sliceIF, [lutOffset + addr_lut];\n",

j1c40 => "02:2:-:-:1  \@P0 STS.128 [writeS], loadF;\n",

j2c10 => "--:-:-:-:1  \@P1 XMAD     offsetFc, channel, param_KRST, RZ;\n",
j2c15 => "--:-:-:-:1  \@P1 XMAD     offsetIc, channel, param_DHWN,    RZ;\n",
j2c20 => "--:-:-:-:1  \@P1 XMAD.PSL offsetIc, channel, param_DHWN.H1, offsetIc;\n",
j2c22 => "--:-:-:-:1      IADD posCRST, posCRST, -8;\n",

j2c29 => "20:-:-:-:1  \@P1 IADD3    offsetF, offsetFk, offsetFc, sliceF;\n",
j2c34 => "--:-:-:-:1  \@P1 LEA      trackF0.CC, offsetF, param_F[0],     2;\n",
j2c36 => "--:-:-:-:1  \@P1 IADD3    offsetI, offsetIn, offsetIc, sliceI;\n",
j2c38 => "--:-:-:-:1  \@P1 LEA.HI.X trackF1,    offsetF, param_F[1], RZ, 2;\n",

j2c40 => "02:-:2:-:1  \@P1 LDG.E.CI.128 loadF, [trackF];\n",


j6c8  => "04:3:-:-:1  \@P0 STS.128 [writeS + 4x<szShareF>], loadI;\n",
```

nervana tensor processing unit (TPU)

- Unprecedented compute density



1
nervana
engine
@200 watts

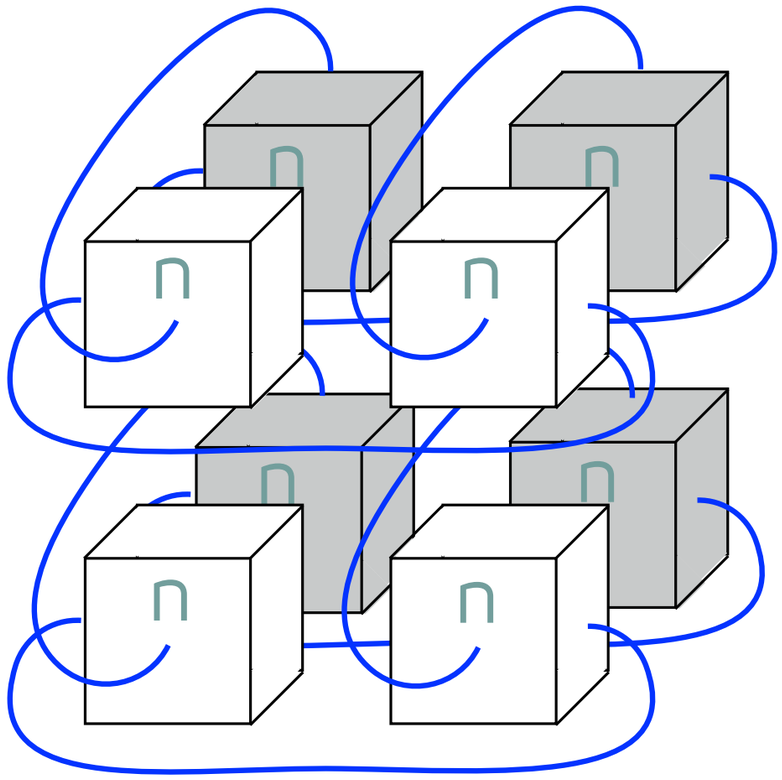
=

200 CPUs
@20,000 watts

10 GPUs
@2000 watts

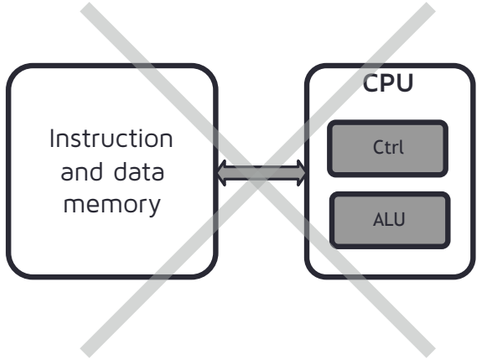
nervana tensor processing unit (TPU)

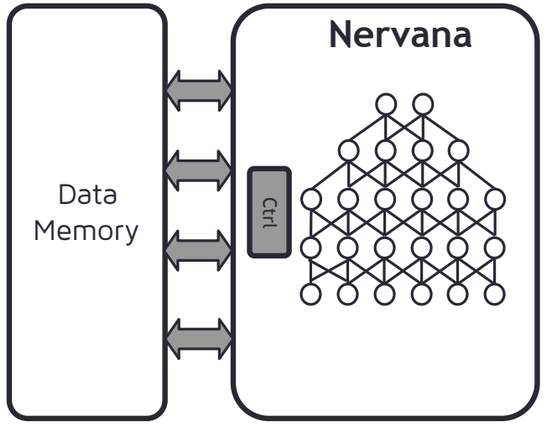
- Unprecedented compute density
- Scalable distributed architecture



nervana tensor processing unit (TPU)

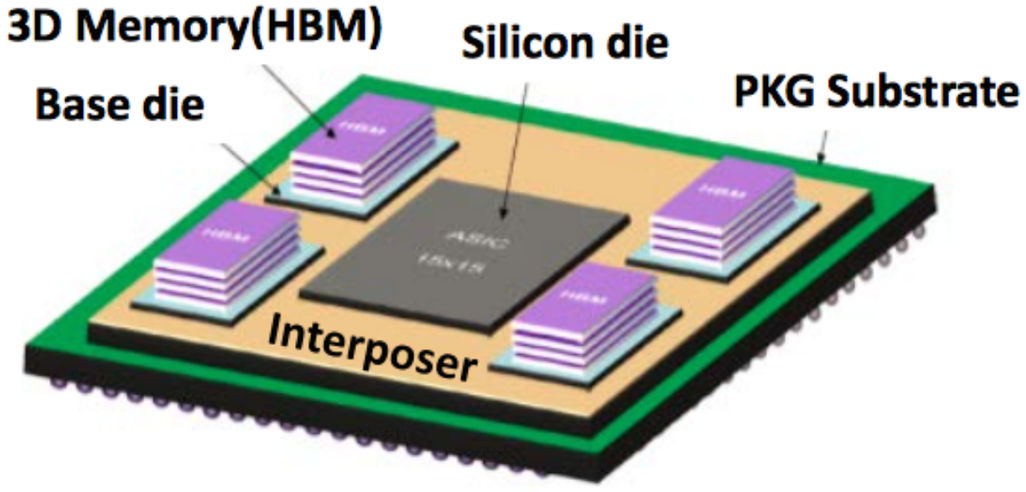
- Unprecedented compute density
- Scalable distributed architecture
- Memory near computation





nervana tensor processing unit (TPU)

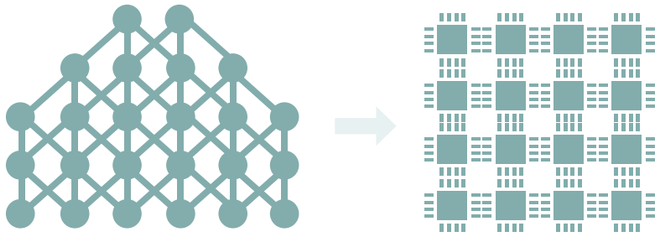
- Unprecedented compute density
- Scalable distributed architecture
- Memory near computation
- Learning and inference
- Exploit limited precision
- Incorporate latest advances
- Power efficiency



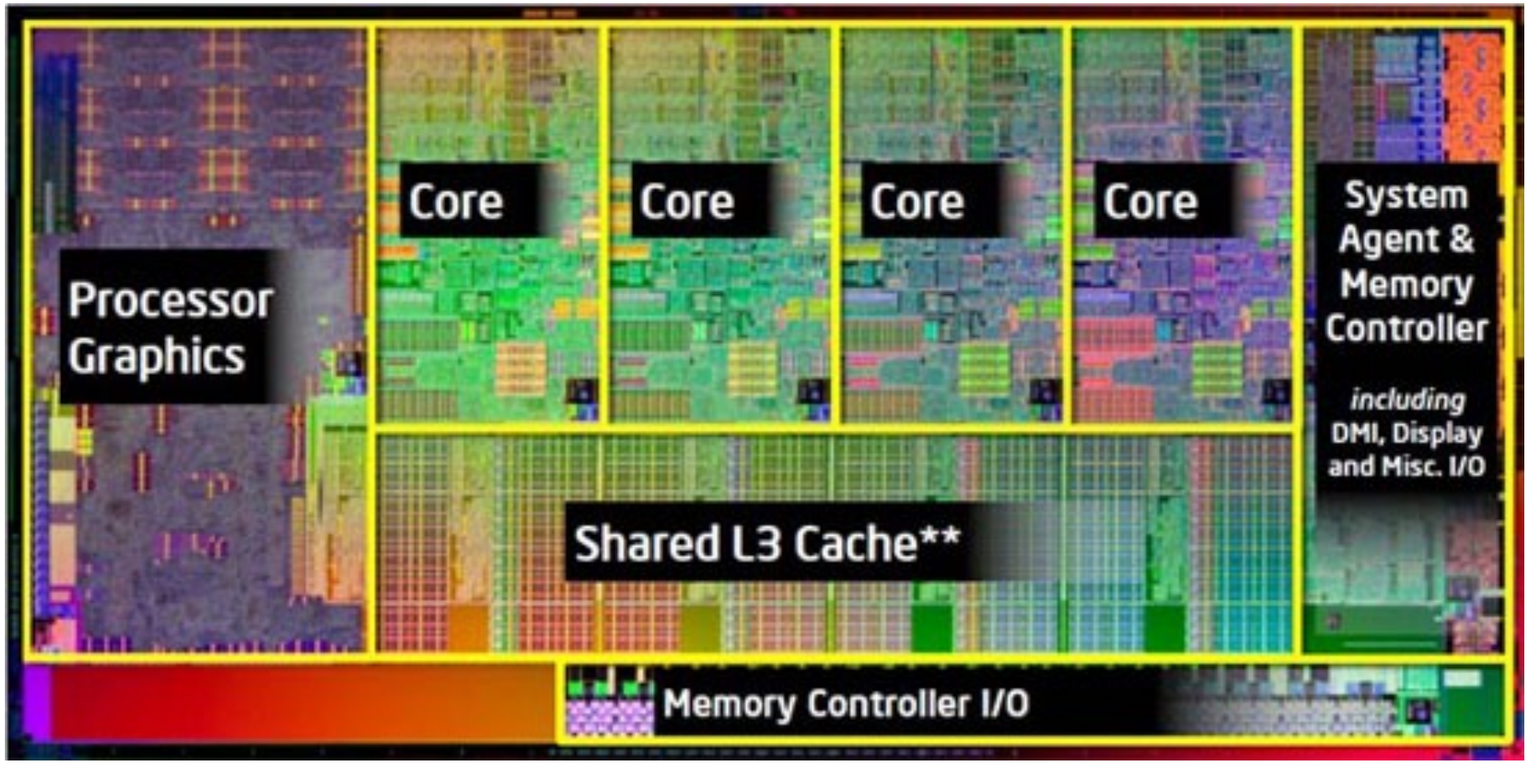
nervana tensor processing unit (TPU)

- Unprecedented compute density
- Scalable distributed architecture
- Memory near computation
- Learning and inference
- Exploit limited precision
- Incorporate latest advances
- Power efficiency

- **10-100x** gain
- Architecture optimized for algorithm



General purpose computation

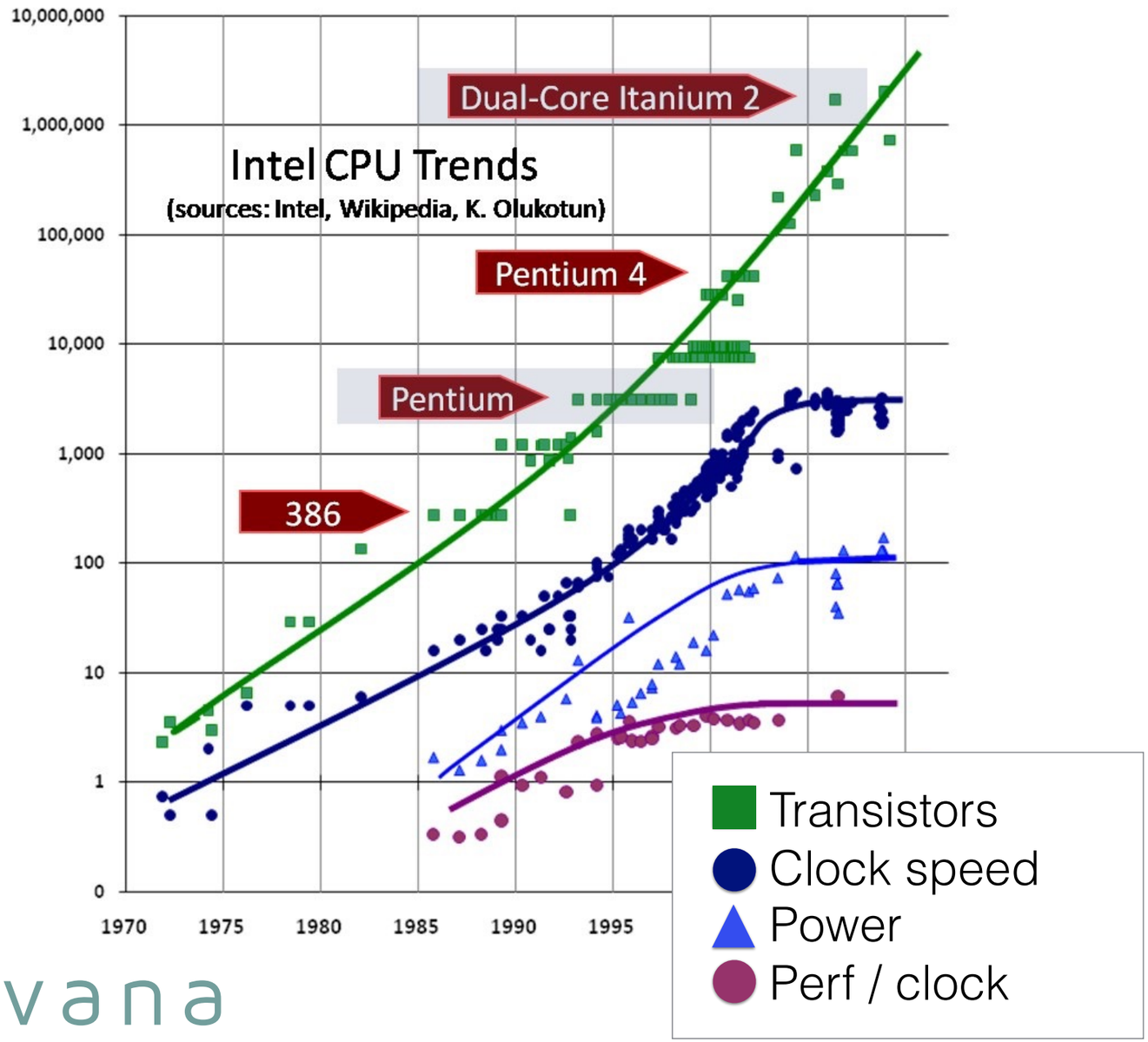


2000s: SoC

Motivation: reduce power and cost, fungible computing.

Enabled inexpensive mobile devices.

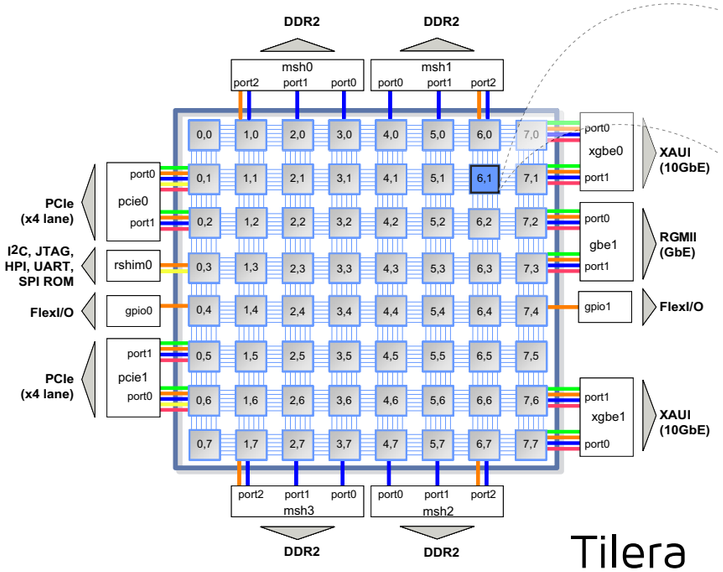
Dennard scaling has ended



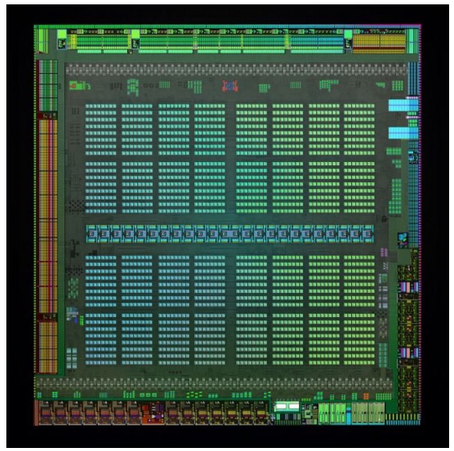
What's next?

nervana

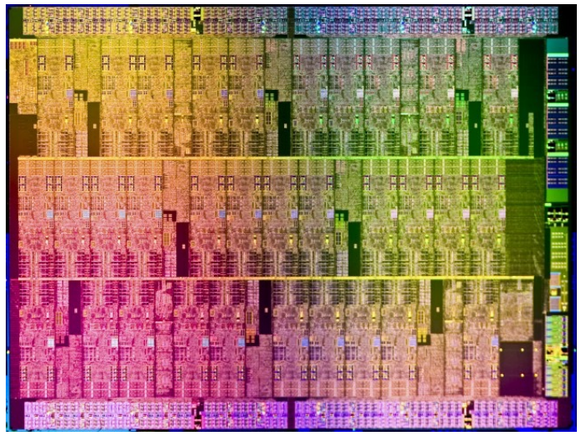
Many-core tiled architectures



Tilera



NVIDIA GM204



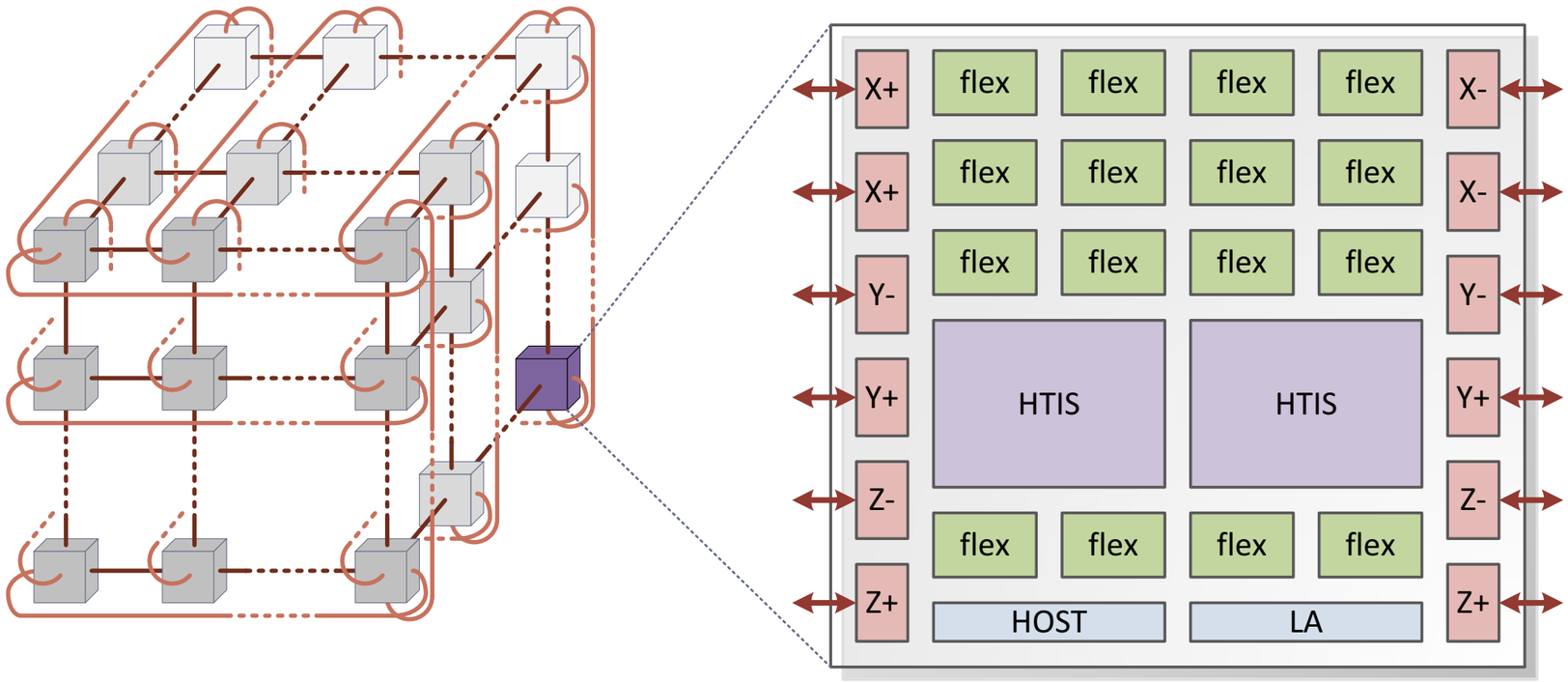
Intel Xeon Phi
Knight's landing

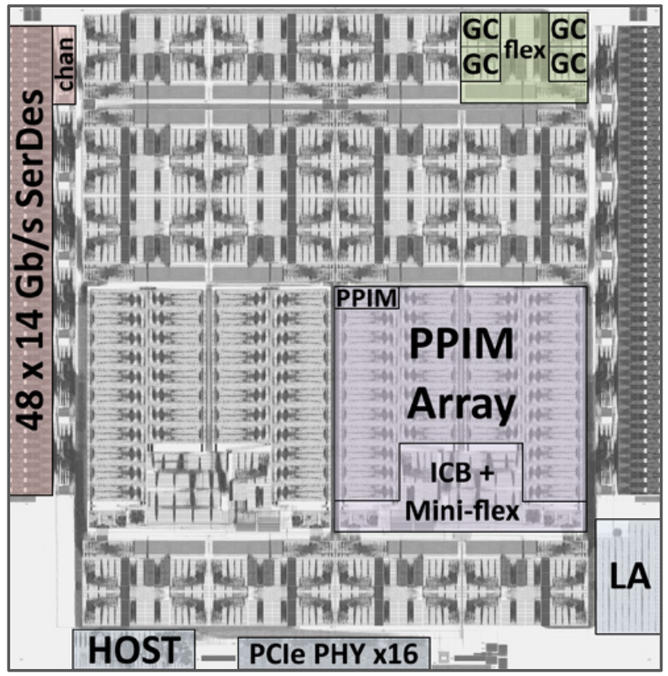
Motivation: increased performance without clock rate increase or smaller devices.

Requires changes in programming paradigm.

2010s: multi-core, GPGPU

Special purpose computation: Anton





(Shaw et al., 2014)

nervana

Computational motifs

Can be implemented using:

- Silicon
- Software
- Neural network architectures!

	Motif	Examples
1	**Dense linear algebra**	Matrix multiply (GEMM)
2	**Sparse linear algebra**	SpMV
3	**Spectral methods**	FFT
4	**N-Body methods**	Molecular dynamics
5	**Structured grids**	Lattice Boltzmann
6	**Unstructured grids**	CFD
7	**Map-Reduce**	Expectation
8	**Combinational logic**	Encryption, hashing
9	**Graph traversal**	Decision trees, quicksort
10	**Dynamic programming**	Forward-backward
11	**Bactrack, branch and bound**	Constraint satisfaction
12	**Graphical models**	HMM, Bayesian
13	**Finite state machines**	Compilers

(Asanovic et al., 2006)

Summary

- Computers are tools for solving problems of their time
  - Was: Coding, calculation, graphics, web
  - Today: Learning and Inference on data
- Deep learning as a computational paradigm
  - Custom architecture can do vastly better
- We are hiring! Summer interns and full time.