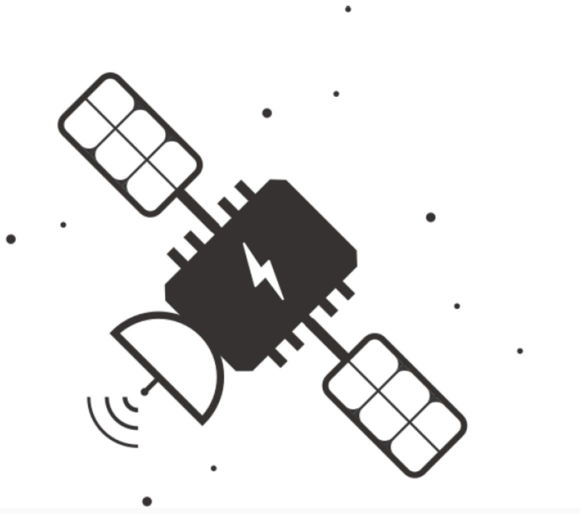


Accelerators for Cyber-Physical Systems

Sam Green, İhsan Çiçek and Çetin Kaya Koç
University of California, Santa Barbara

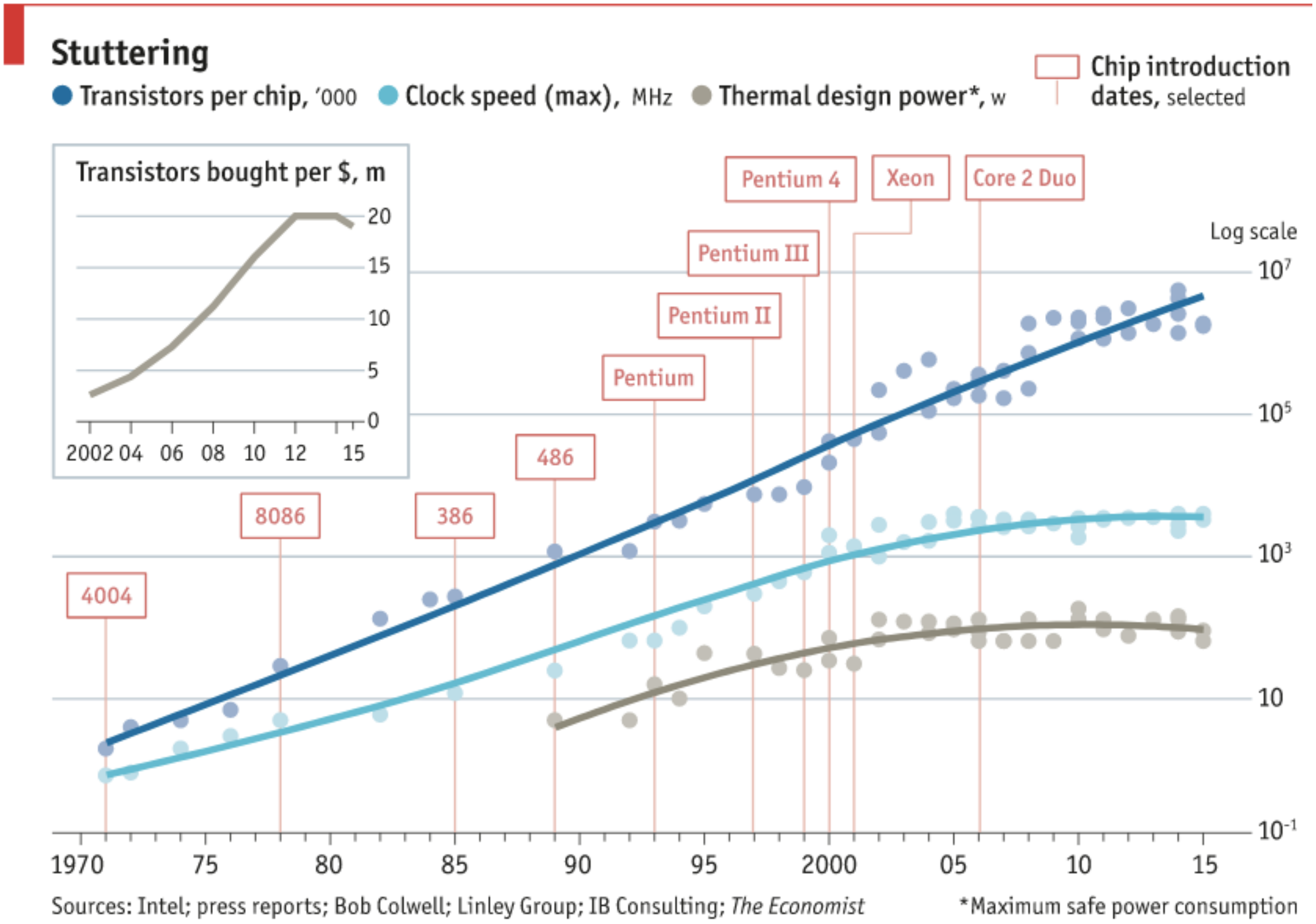
Introduction

Capabilities desired in CPS?

- Interact with physical world
- Networked
- Potentially low-power
- Resistant to environment
- Perform safety-critical tasks
- Cryptographically secure
- Autonomous
- Inexpensive

Benefits from Moore’s Law are over

- Since about 1970, could safely assume the number of transistors/$ would exponentially increase every 2 years
  - What can be done today for $X will be doable in 2 years for $X/2 dollars
- Accelerators (aka ASICs) existed during this time, but CPU/μcontroller/DSP-based approaches dominated
- No longer the case…



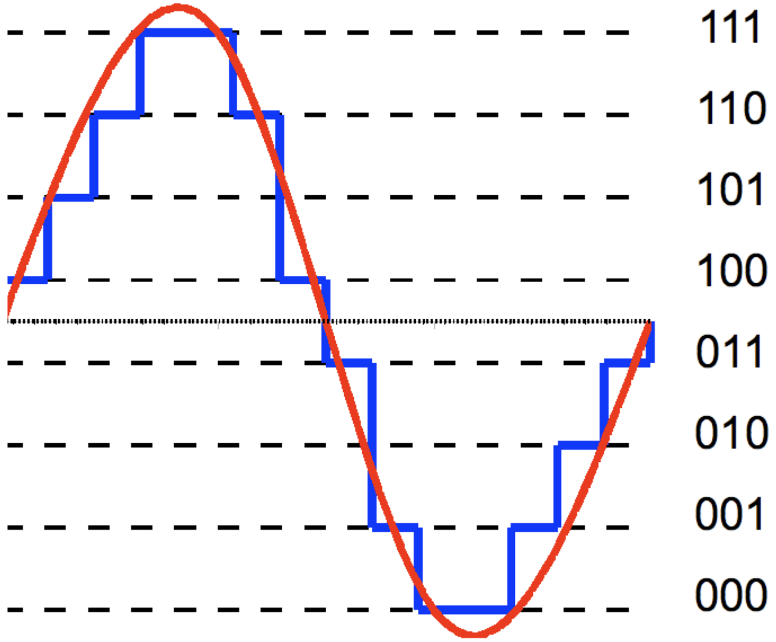
[http://www.economist.com/node/21693710/sites/all/modules/custom/ec_essay]

Other methods to increase performance/$?

- Approximate computing
- Analog computing
- Neuromorphic computing

Approximate Computing

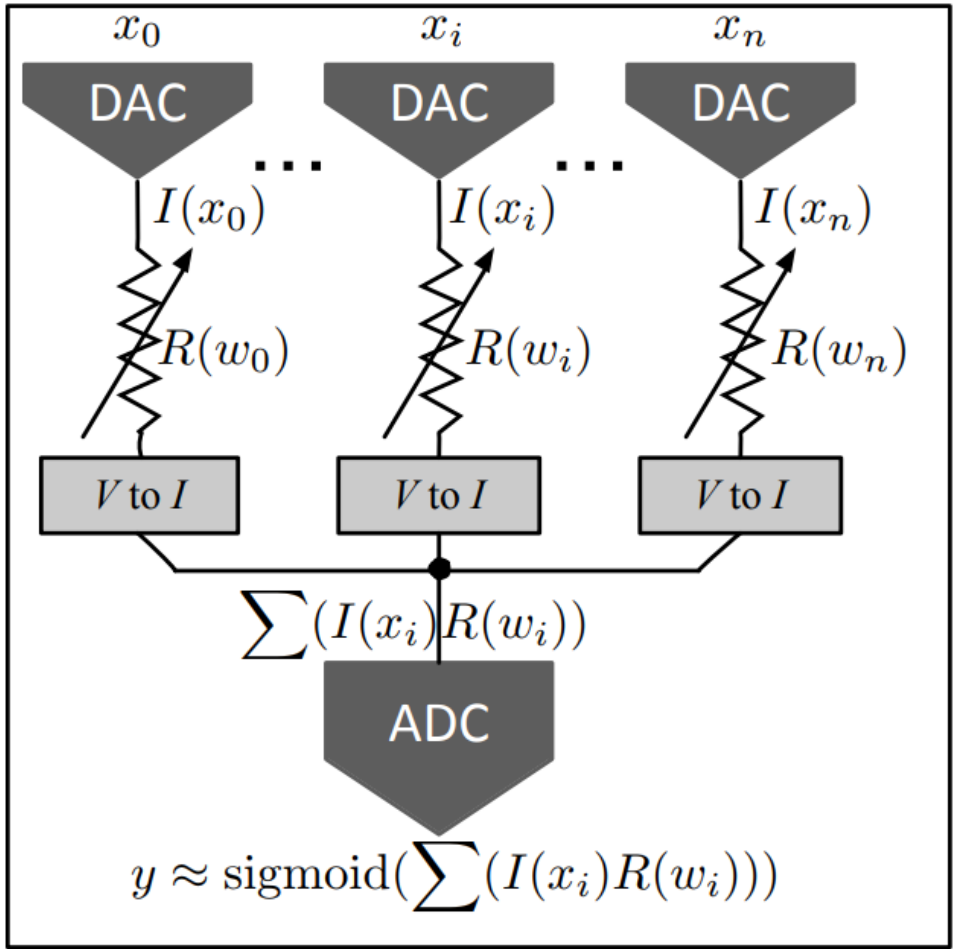
- Selective approximation can bring disproportionate gains in efficiency
- 5% accuracy loss gives
  - 50x less energy for k-means clustering
  - 26x less energy for neural network evaluation



[S. Mittal. A Survey of Techniques for Approximate Computing. *ACM Comput. Surv.*, vol. 48, no. 4, p. 62:1–62:33, Mar. 2016.]
[https://upload.wikimedia.org/wikipedia/commons/b/b7/3-bit_resolution_analog_comparison.png]

Analog Computing

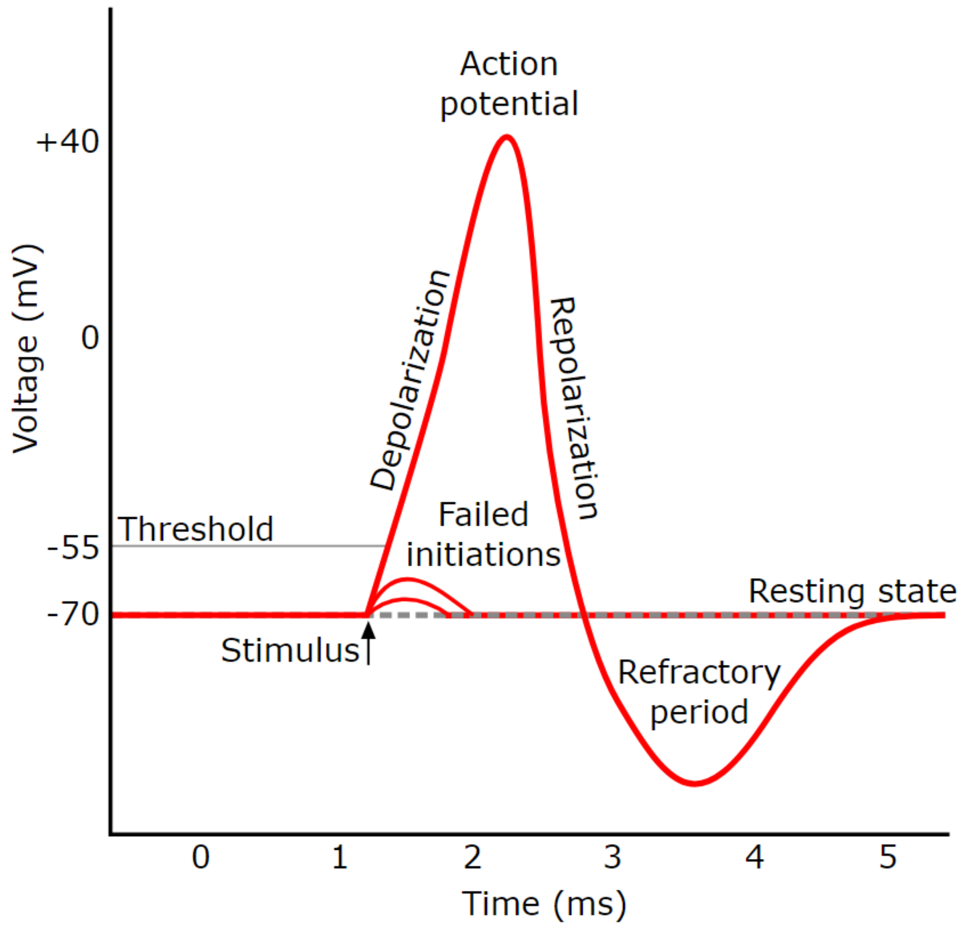
- Physical world is a computational device
- E.g. Use KVL and KCL to approximate activation function for analog neuron
- 4X speedup, 20X less energy, 2.4% higher error across benchmarks vs. approximate digital neuron



[St. Amant et al. General-purpose Code Acceleration with Limited-precision Analog Computation. *ISCA*, 2014]

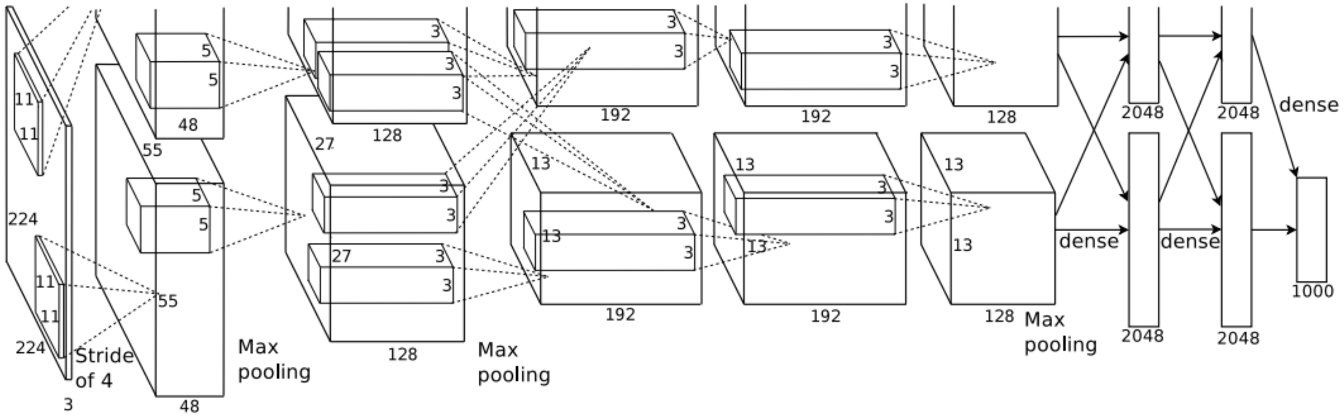
Neuromorphic Computing

- Non-von Neumann, neuro-bio inspired architectures
- Community sees biological circuits as the ultimate in efficiency



[https://upload.wikimedia.org/wikipedia/commons/4/4a/Action_potential.svg]

Accelerators for Deep Learning Inference



[A. Krizhevsky et al. ImageNet Classification with Deep Convolutional Neural Networks. *NIPS 25*, 2012, pp. 1097–1105.]

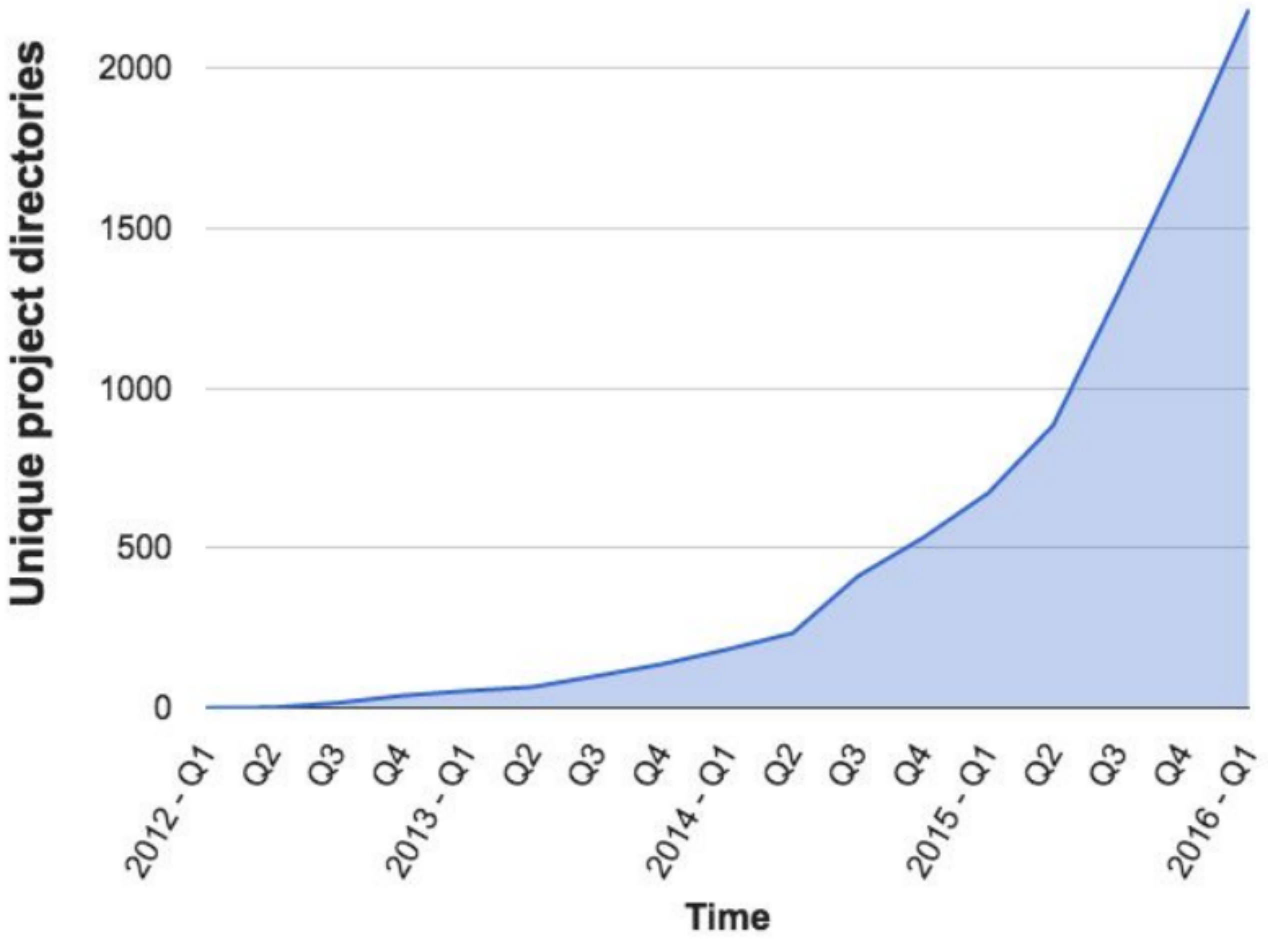
Motivation for deep learning accelerators

- Edge computing applications
  - CPS, IoT, Mobile
  - Power & compute is restriction
- Datacenter applications
  - In 2013, U.S. datacenters consumed the equivalent output of 34 large coal-fired power plants

[https://www.nrdc.org/experts/pierre-delforge/new-study-americas-data-centers-consuming-and-wasting-growing-amounts-energy]

Growing Use of Deep Learning at Google

of directories containing model description files



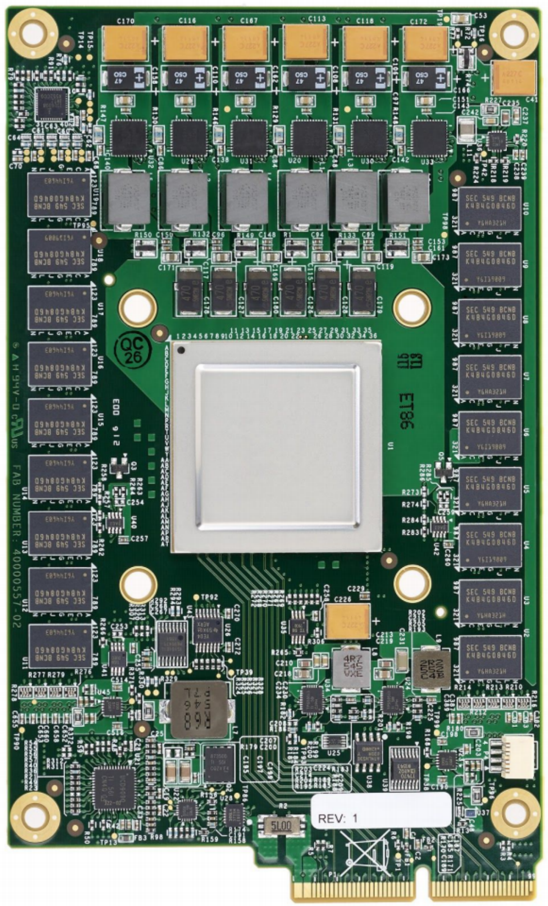
Across many products/areas:

Android
Apps
drug discovery
Gmail
Image understanding
Maps
Natural language understanding
Photos
Robotics research
Speech
Translation
YouTube
… many others ...

[https://www.slideshare.net/JenAman/large-scale-deep-learning-with-tensorflow]

Google's Tensor Processing Unit

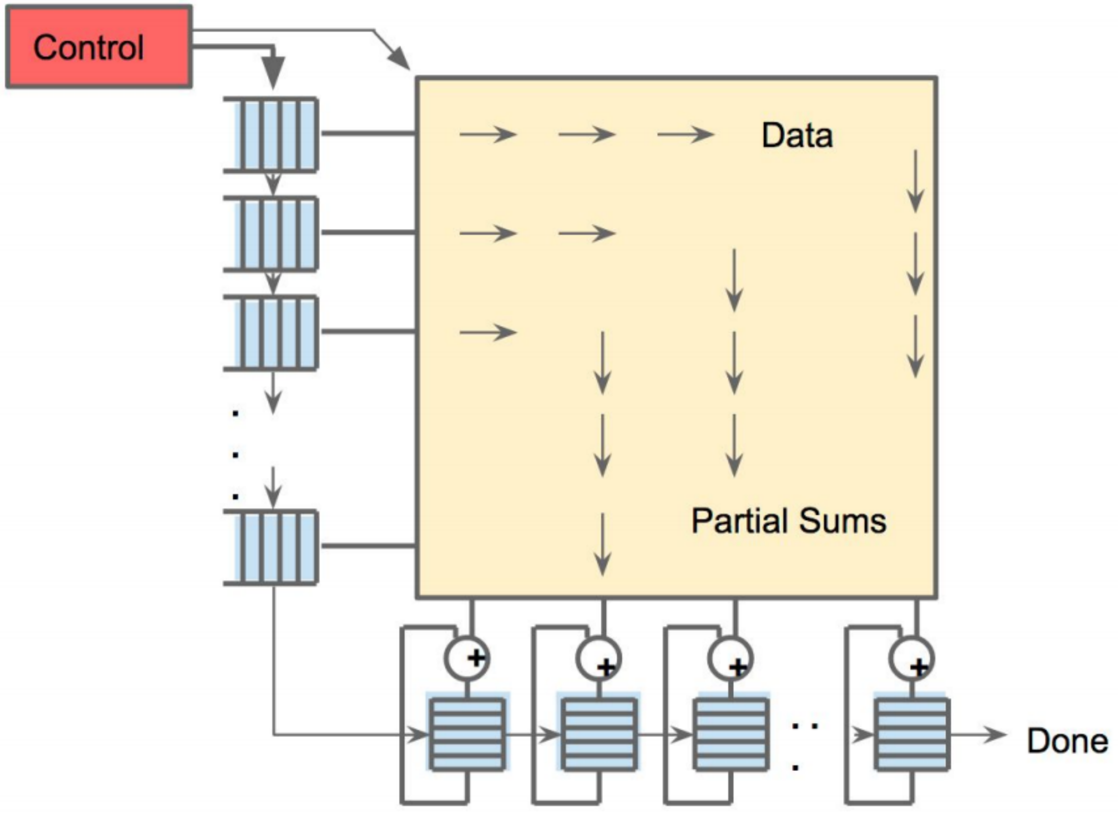
- General purpose deep neural network accelerator
  - LSTM
  - MLP
  - CNN
- 15X – 30X faster than Nvidia K80 GPU
- Performance/Watt 30X – 80X



[Jouppi et al. In-Datacenter Performance Analysis of a Tensor Processing Unit. *ISCA*, 2017.]

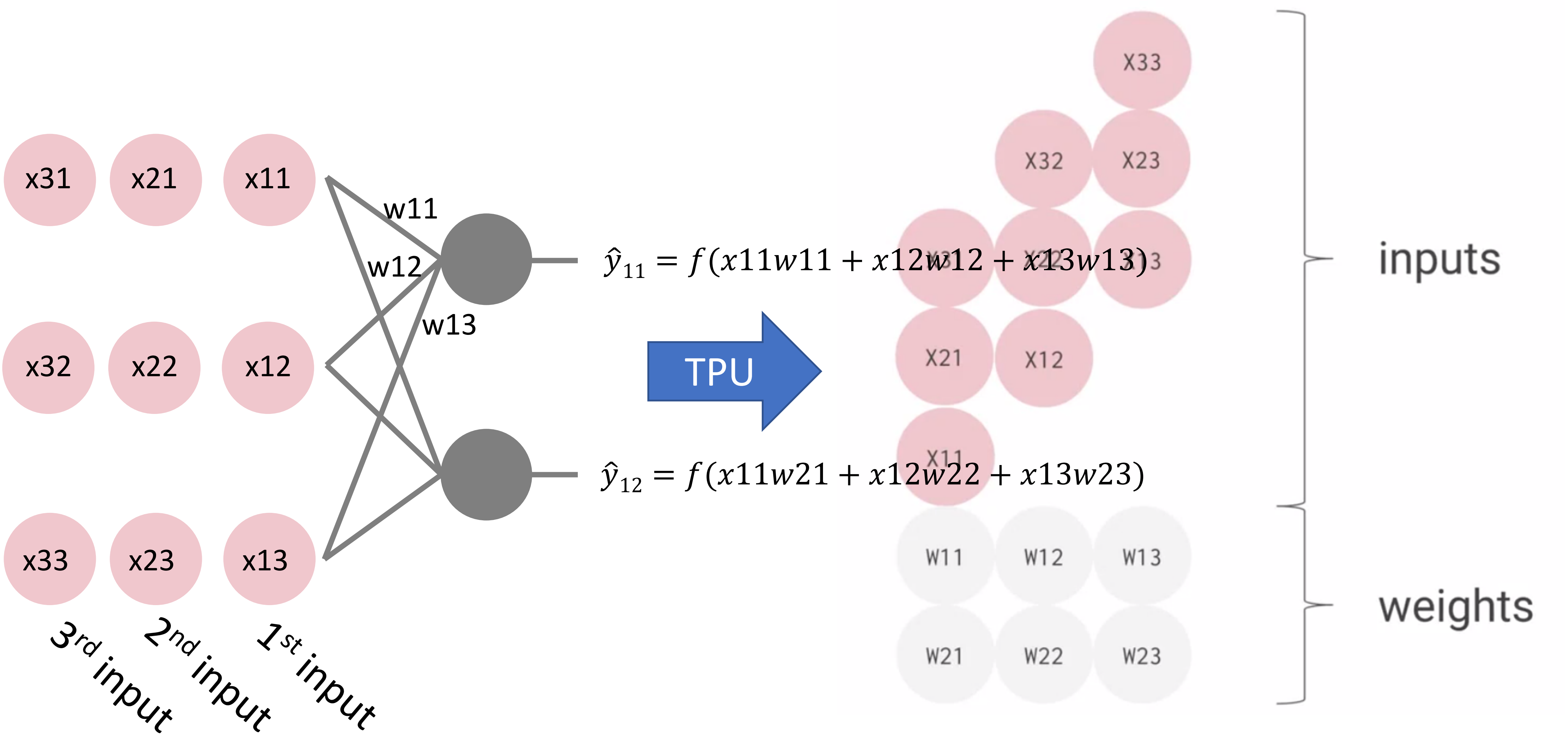
Google's Tensor Processing Unit

- Uses 8-bits of precision
- Systolic Array – 256-element multiply-accumulate operation moves through matrix as a diagonal wave front



[Jouppi et al. In-Datacenter Performance Analysis of a Tensor Processing Unit. *ISCA*, 2017.]

Example of Wave Front (2 neurons w/3 weights)

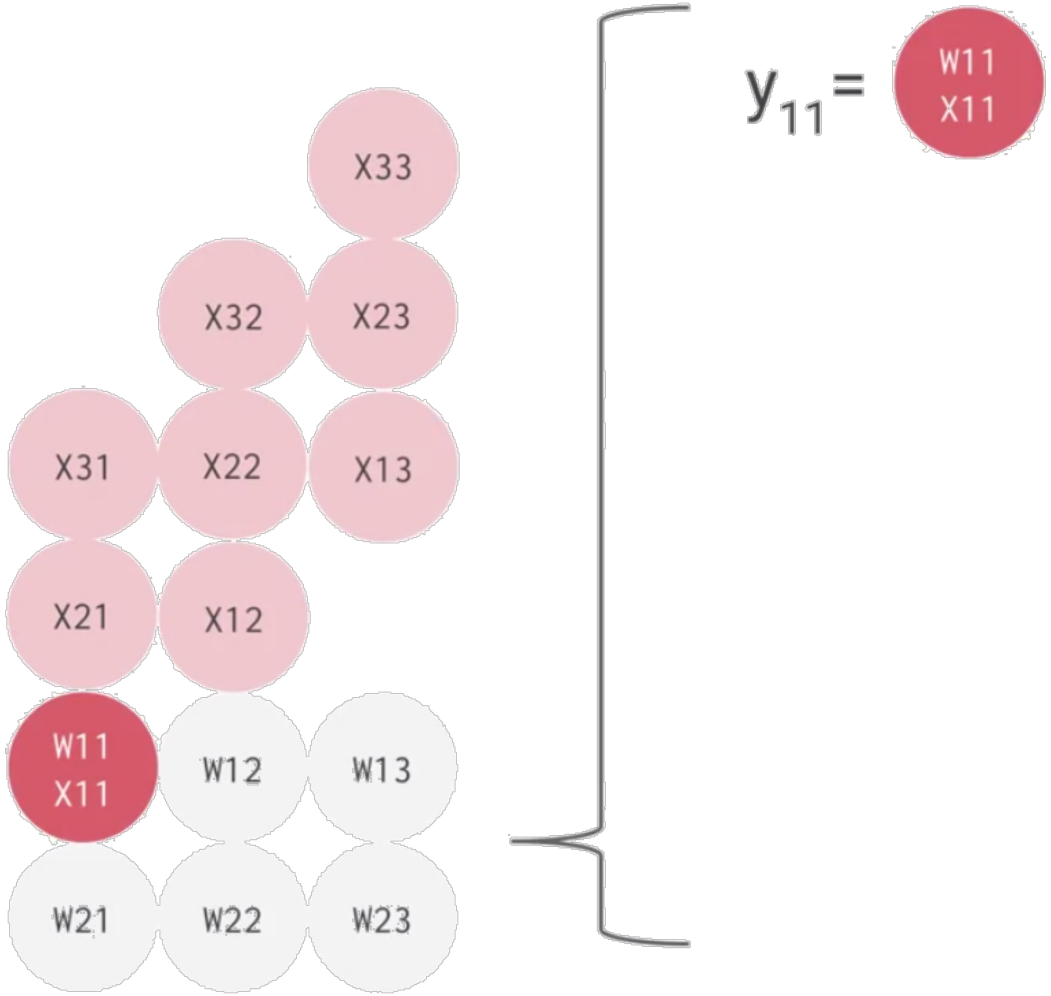


$$\hat{y}_{11} = f(x11w11 + x12w12 + x13w13)$$

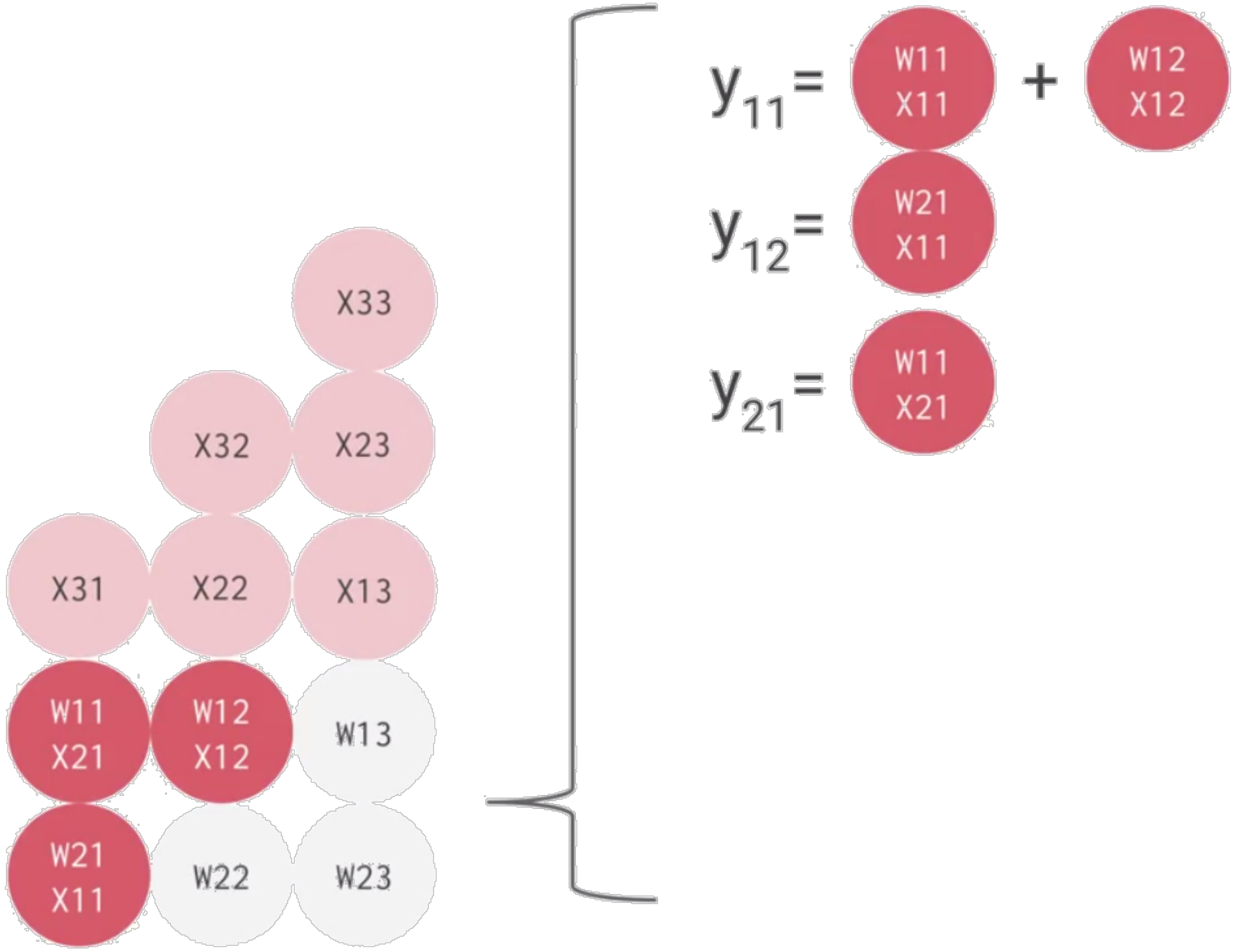
$$\hat{y}_{12} = f(x11w21 + x12w22 + x13w23)$$

[https://cloud.google.com/blog/big-data/2017/05/an-in-depth-look-at-googles-first-tensor-processing-unit-tpu]

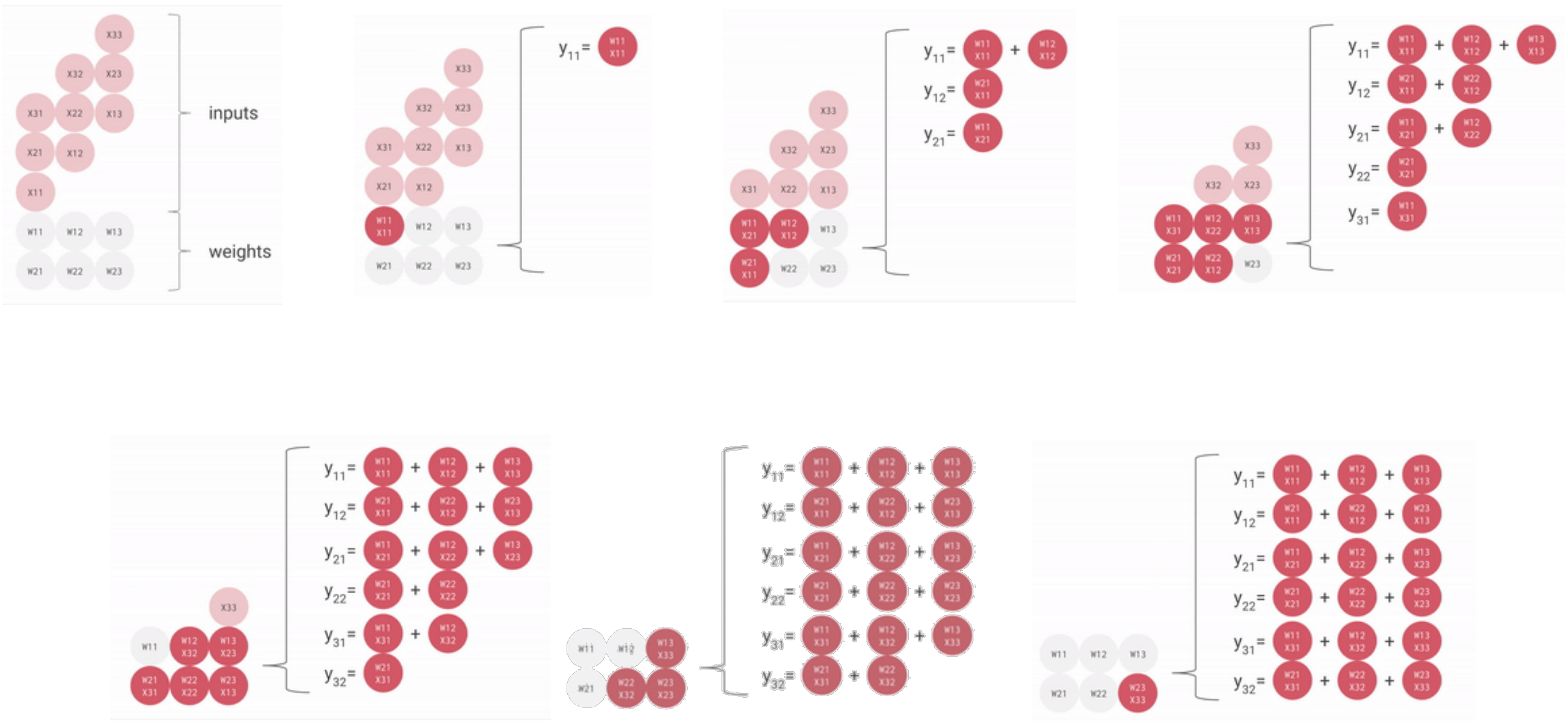
Example of Wave Front (2 neurons w/3 weights)



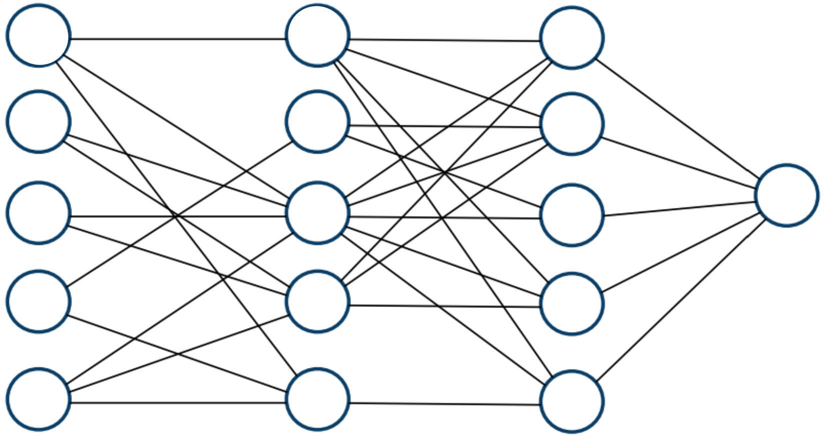
Example of Wave Front (2 neurons w/3 weights)



Example of Wave Front (2 neurons w/3 weights)



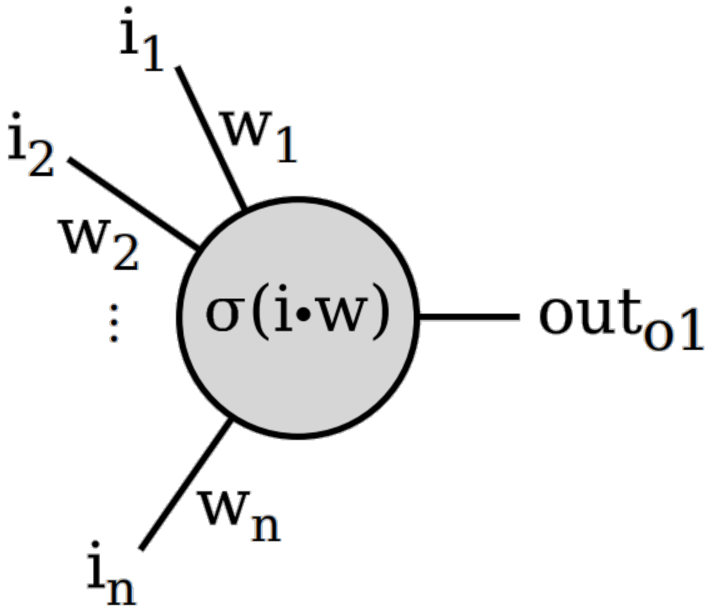
Deep Neural Network Optimizations



[M. Shafiee et al. StochasticNet: Forming Deep Neural Networks via Stochastic Connectivity. 2015]

Traditional DNN evaluation is expensive

- DNNs perform many multiply-accumulate (MAC) followed by non-linear function evaluation
- Expensive floating-point MAC traditionally used



MACs used in popular network architectures

Metrics	LeNet 5	AlexNet	Overfeat fast	VGG 16	GoogLeNet v1	ResNet 50
Top-5 error†	n/a	16.4	14.2	7.4	6.7	5.3
Top-5 error (single crop)†	n/a	19.8	17.0	8.8	10.7	7.0
Input Size	28×28	227×227	231×231	224×224	224×224	224×224
# of CONV Layers	2	5	5	13	57	53
Depth in # of CONV Layers	2	5	5	13	21	49
Filter Sizes	5	3,5,11	3,5,11	3	1,3,5,7	1,3,7
# of Channels	1, 20	3-256	3-1024	3-512	3-832	3-2048
# of Filters	20, 50	96-384	96-1024	64-512	16-384	64-2048
Stride	1	1,4	1,4	1	1,2	1,2
Weights	2.6k	2.3M	16M	14.7M	6.0M	23.5M
MACs	283k	666M	2.67G	15.3G	1.43G	3.86G
# of FC Layers	2	3	3	3	1	1
Filter Sizes	1,4	1,6	1,6,12	1,7	1	1
# of Channels	50, 500	256-4096	1024-4096	512-4096	1024	2048
# of Filters	10, 500	1000-4096	1000-4096	1000-4096	1000	1000
Weights	58k	58.6M	130M	124M	1M	2M
MACs	58k	58.6M	124M	130M	1M	2M
Total Weights	60k	61M	146M	138M	7M	25.5M
Total MACs	341k	724M	2.8G	15.5G	1.43G	3.9G

[V. Sze et al. Efficient Processing of Deep Neural Networks: A Tutorial and Survey. *ISCA*, 2017.]

Recent DNN *inference* optimizations

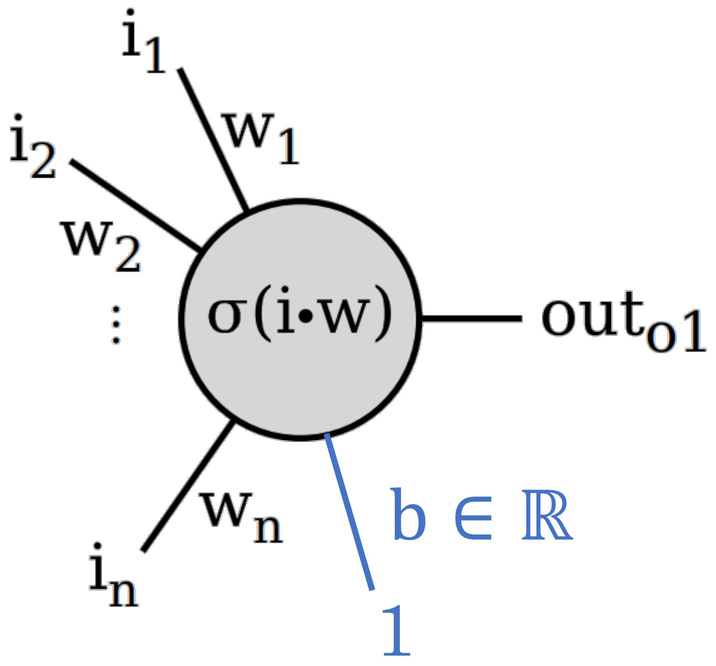
- Community focused on inference because learning the parameters is much more complicated
  - Quantization (16, 8, 4, 2 bits per value)
  - Weight binarization
  - Input and weight binarization
  - Pruning/compression

Terminology

- Forward propagation = forward pass = evaluation = inference = running the network
  - DNN is a non-linear function approximator $\hat{y} = f(\boldsymbol{x})$
- Backpropagation algorithm
  - $f(\boldsymbol{x})$ is a differentiable multivariate function
  - Gradient descent is used to locate local minima. Requires forward propagation, repeated application of chain-rule, and book keeping

Example: BinaryConnect Algorithm

- 2015 – One of first efforts to apply approximate computing to DNN
- Applies only to forward pass
- Eliminates all multiplication
  - Still requires F.P. addition and F.P. activation



[M.Courbariaux et al. BinaryConnect: Training Deep Neural Networks with binary weights during propagations. 2015.]

Example: BinaryConnect Algorithm

Intuition: Temporarily binarize weights during forward propagation, keep track of full-precision weights during backpropagation.

Data: Full-precision weight $w_i \in \mathbb{R}$

Result: Binarized weight $w_{ib} \in \{-1,1\}$

if $w_i < 0$ then

$w_{ib} = -1$

else

$w_{ib} = 1$

Example: BinaryConnect Algorithm

Data: (inputs, targets), previous parameters w_{t-1} (weights) and b_{t-1} (biases), and learning rate η

Result: Updated (full-precision) parameters w_t and b_t

1. Forward propagation

$w_b = \text{binarize}\ (w_{t-1})$
For $k = 1$ to L, compute a_k knowing a_{k-1}, w_b and b_{t-1}

2. Backward propagation

Initialize output layer's activations gradient $\frac{\partial E}{\partial a_L}$

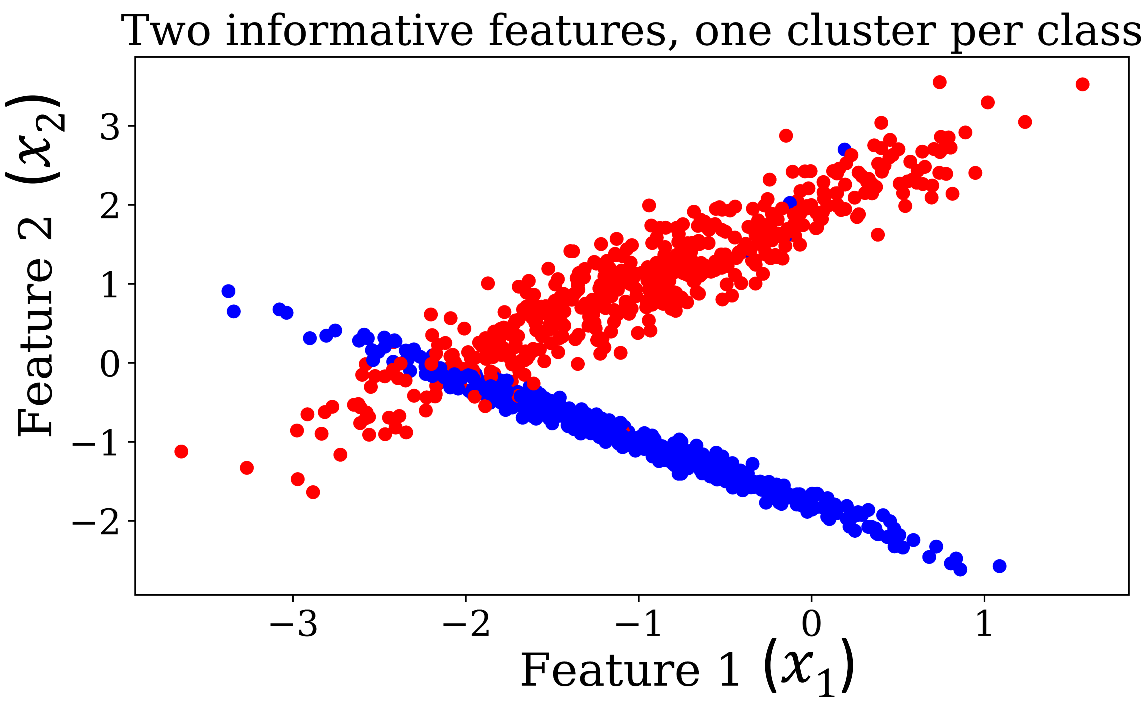
For $k = L$ to 2, compute $\frac{\partial E}{\partial a_{k-1}}$ knowing $\frac{\partial E}{\partial a_k}$ and w_b

3. Parameter update

Compute $\frac{\partial E}{\partial w_b}$ and $\frac{\partial E}{\partial b_{t-1}}$

$w_t = w_{t-1} - \eta \frac{\partial E}{\partial w_b}$ and $b_t = b_{t-1} - \eta \frac{\partial E}{\partial b_{t-1}}$

BinaryConnect Toy Example

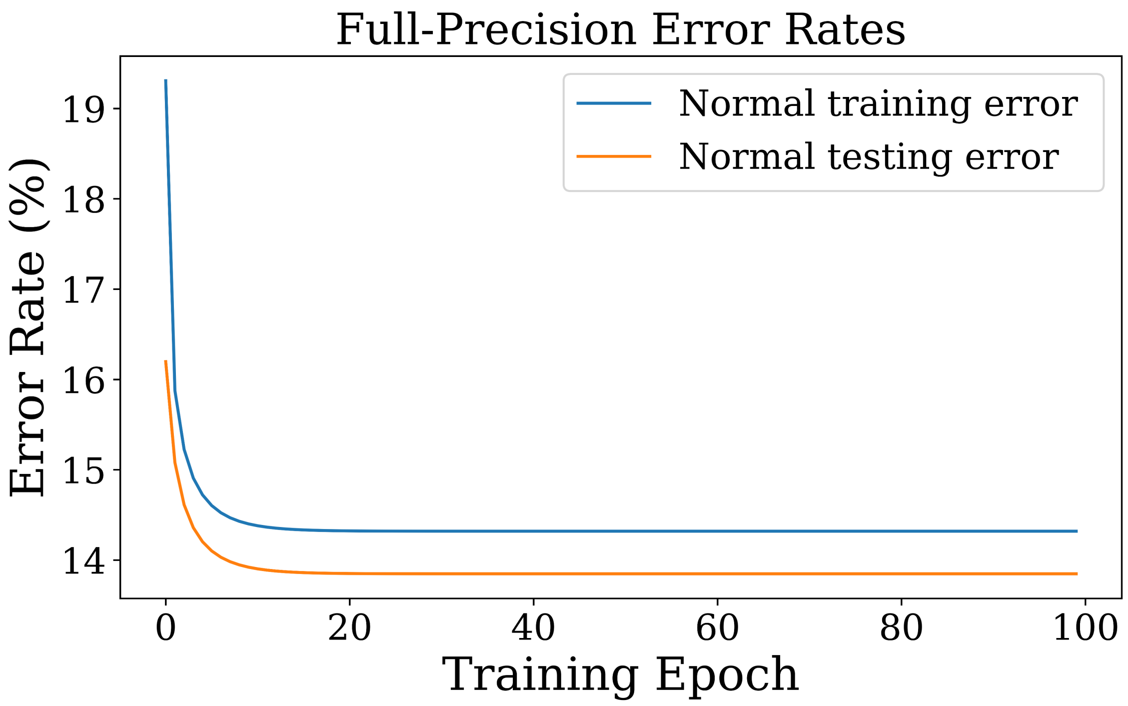


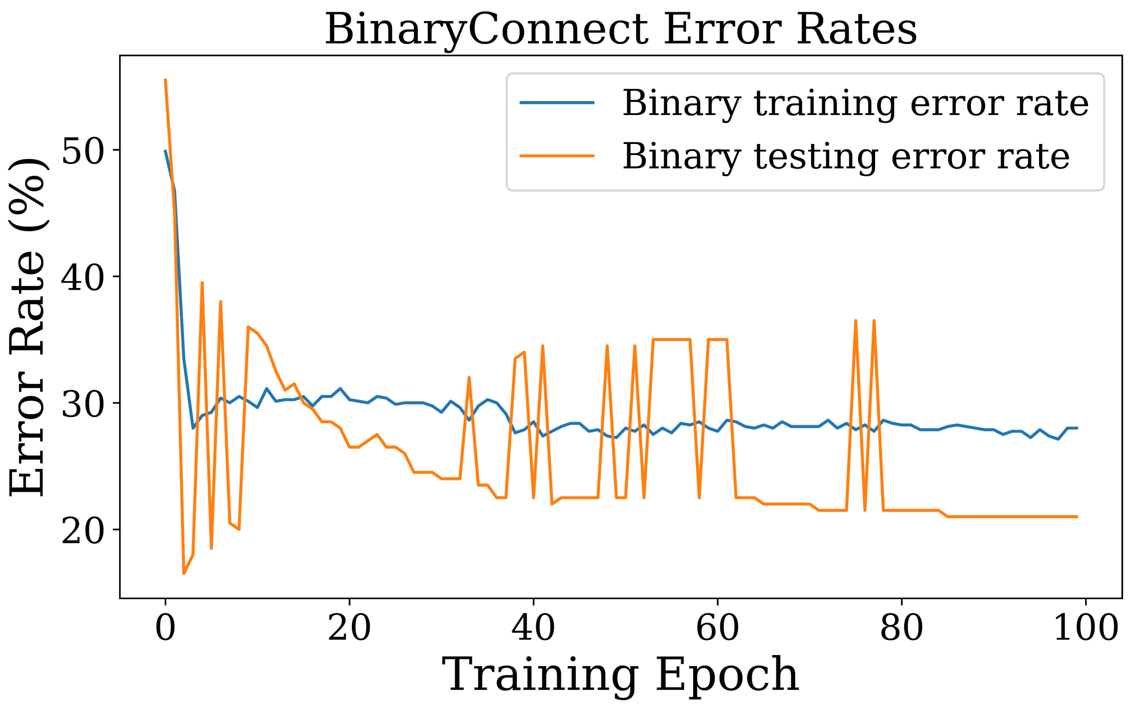
Task: Learn to predict class (blue or red) using binary weights:

$$\hat{y} = \sigma(x_1 w_1 + x_2 w_2 + b)$$

BinaryConnect Toy Example

BinaryConnect approaches <10% of full-precision method



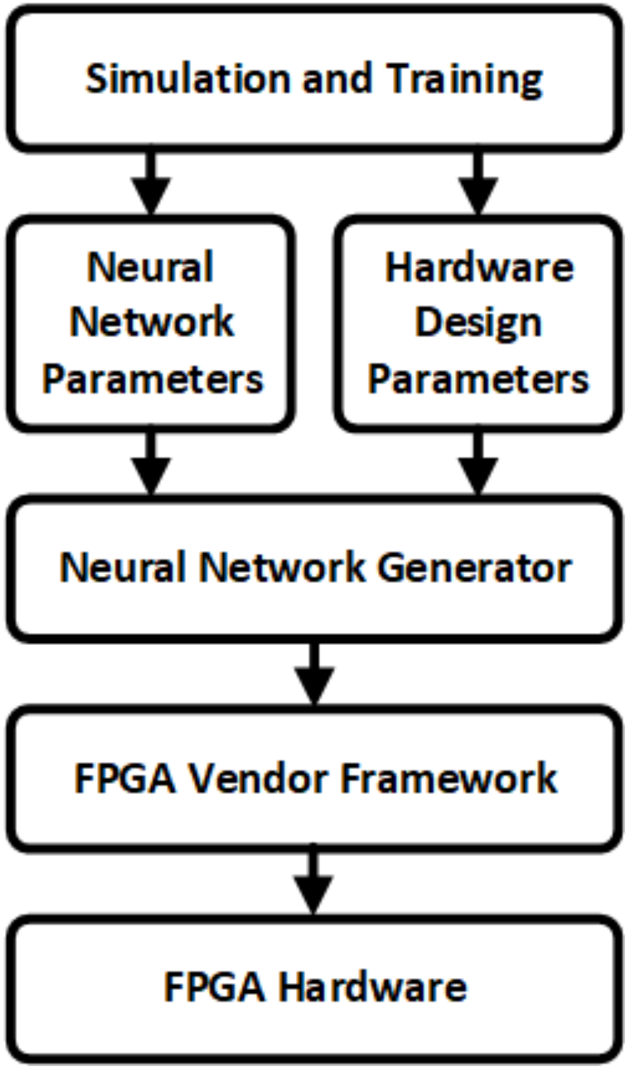


Hardware Implementations of Deep Neural Networks

A framework approach to HW DNNs

- Quantization is attractive for efficiency reasons
  - How much quantization will problem tolerate?
- Optimal DNN architecture discovery is compute-intensive
  - Experiment with different DNN architectures (MLP, LSTM, CNN)
- Performance requirements needed ahead of implementation
  - Min. inference/sec, max clock speed, power budget, area constraints
- Custom software is required to build synthesizable HDL
  - Based on the DNN architecture and performance requirements
- Once we have the HDL code, the rest is standard vendor HW flow

A framework approach to HW DNNs



Accelerators for Cyber-Physical Systems

Opportunities for research and education

- Analog computing still has many contributions

- Need research on failure modes of DNNs

Historical applications of analog computing

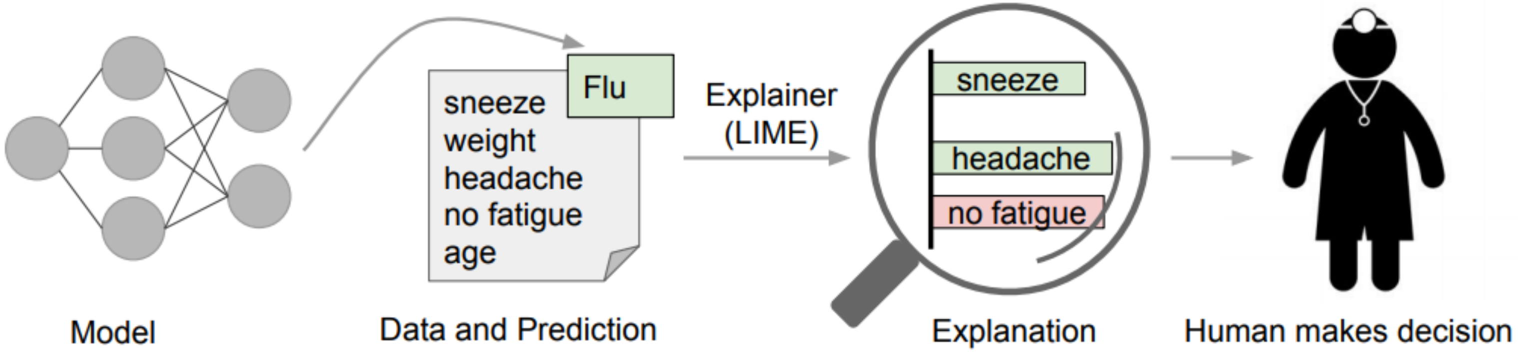
- **Power engineering:** Network simulation, power plant development
- **Automation:** Closed loop control, servo systems
- **Process control:** Mixing tanks, evaporators, distillation columns
- **Transport systems:** Steering systems, traffic-flow simulation, ship simulation
- **Aeronautical engineering:** Rotor blades, guidance and control
- **Rocketry:** Rocket motor simulation, craft maneuvers, craft simulation

Potential for hybrid systems with digital and analog components

[B. Ulmann. *Analog Computing*. 2013]

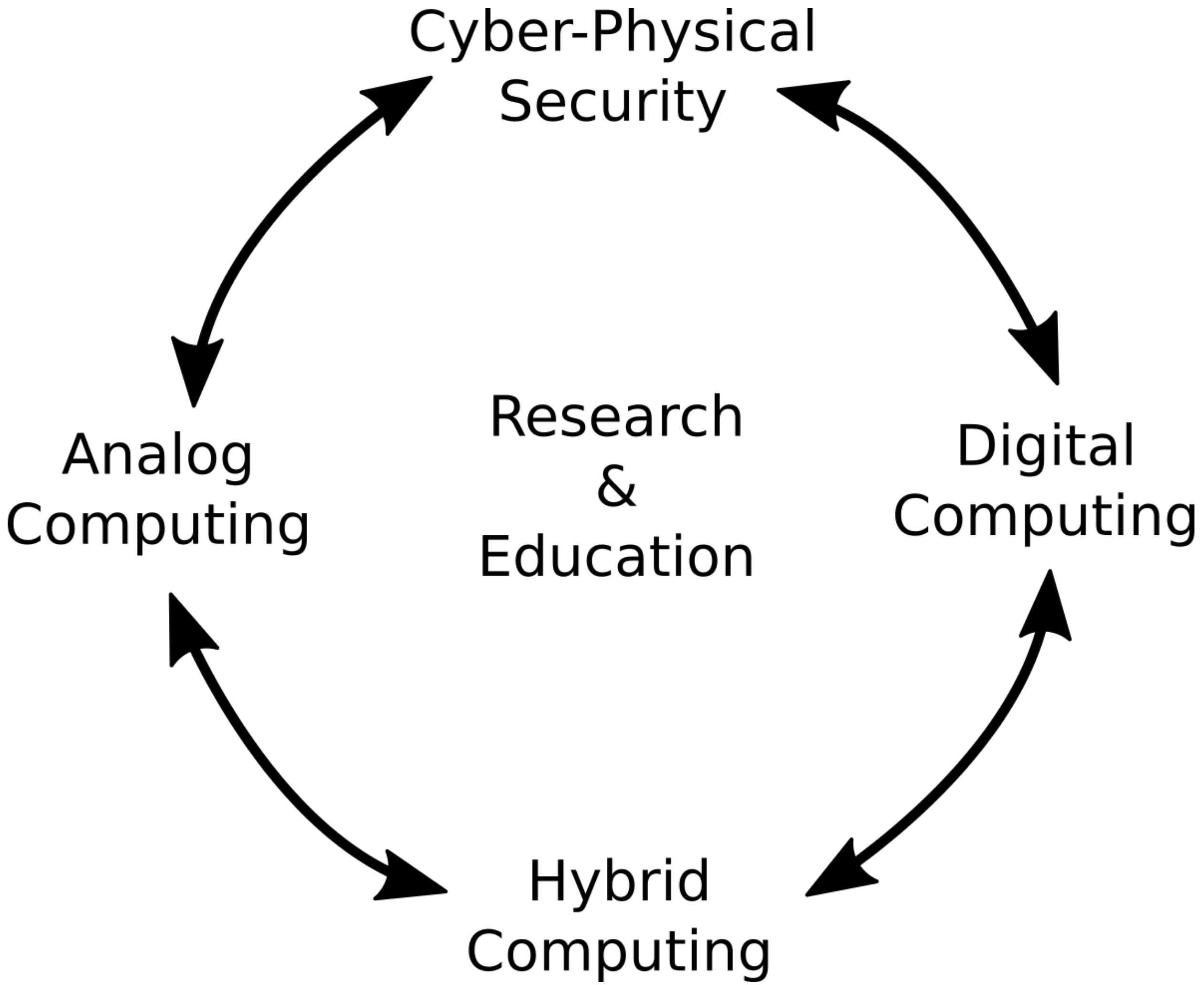
Model interpretation research

- Aim is to understand why a model makes the decision
- Example: a doctor would not blindly operate because of model prediction



- Example: “Why did the car swerve at this moment in time?”

[Ribeiro et al. “’Why Should I Trust You?’ Explaining the Predictions of Any Classifier”, KDD2016]


Cyber-Physical
Security
Analog
Computing
Research
&
Education
Digital
Computing
Hybrid
Computing

Questions?