# Survey and Benchmarking of Machine Learning Accelerators

**Albert Reuther, Peter Michaleas, Michael Jones, Vijay Gadepally, Siddharth Samsi, and Jeremy Kepner**

**IEEE High Performance Extreme Computing Conference**

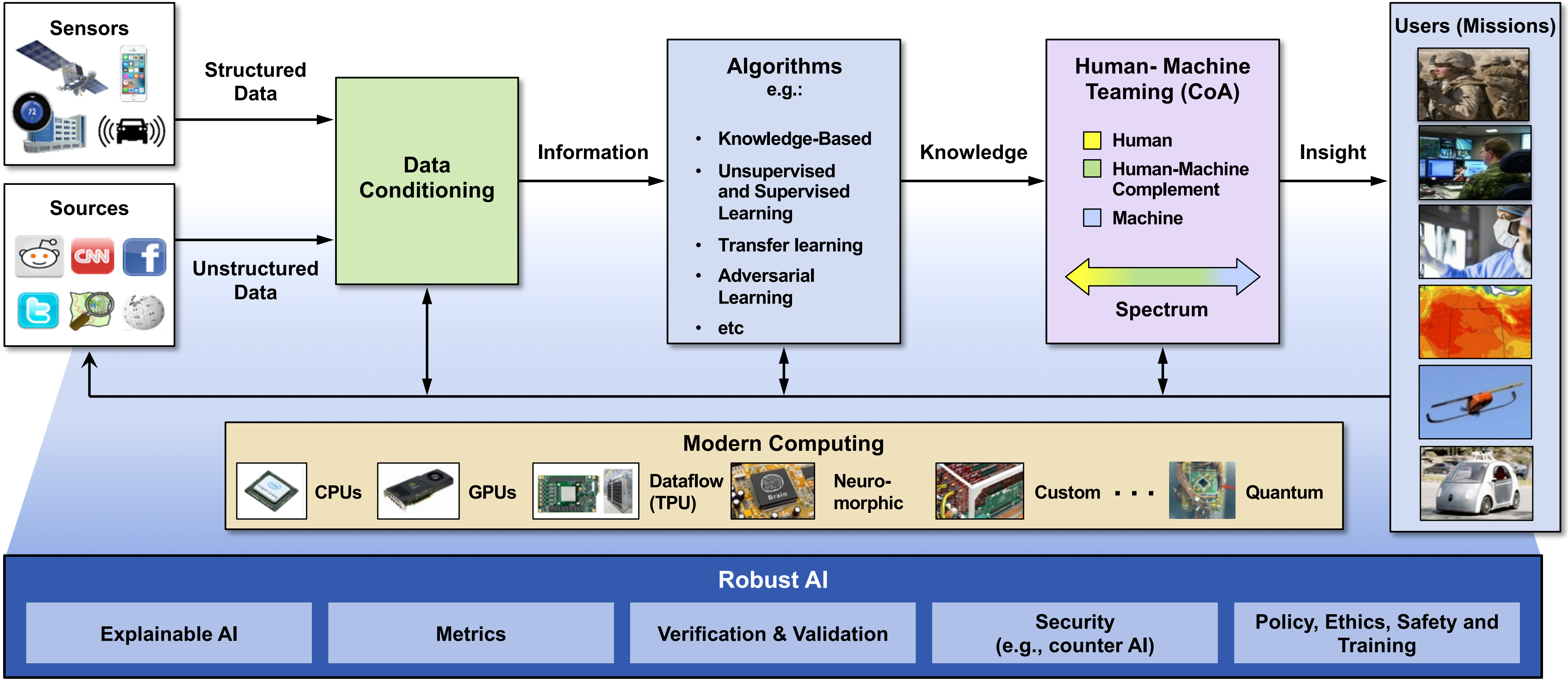
**25-Sept-2019**

LINCOLN LABORATORY
MASSACHUSETTS INSTITUTE OF TECHNOLOGY

# Outline

- Introduction
  - Neural Networks and AI Landscape
  - Training and Inference
  - Numerical Precision
  - Why create custom AI chips?
- AI Processor/Accelerator Landscape
  - Dimensions of Taxonomy
  - Technology Examples
- Embedded Accelerator Benchmarking
  - Intel Movidius and Google TPU Edge
  - Results
- Summary

# Big Data and Building Future AI Systems

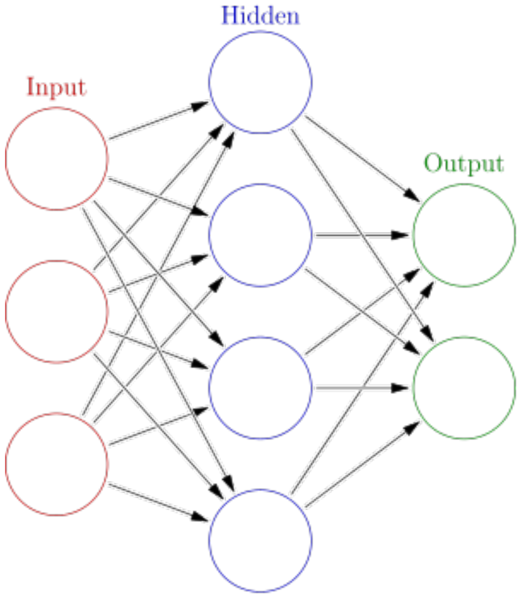





GPU = Graph Processing Unit
TPU = Tensor Processing Unit
CoA = Courses of Action

# Artificial Neural Networks

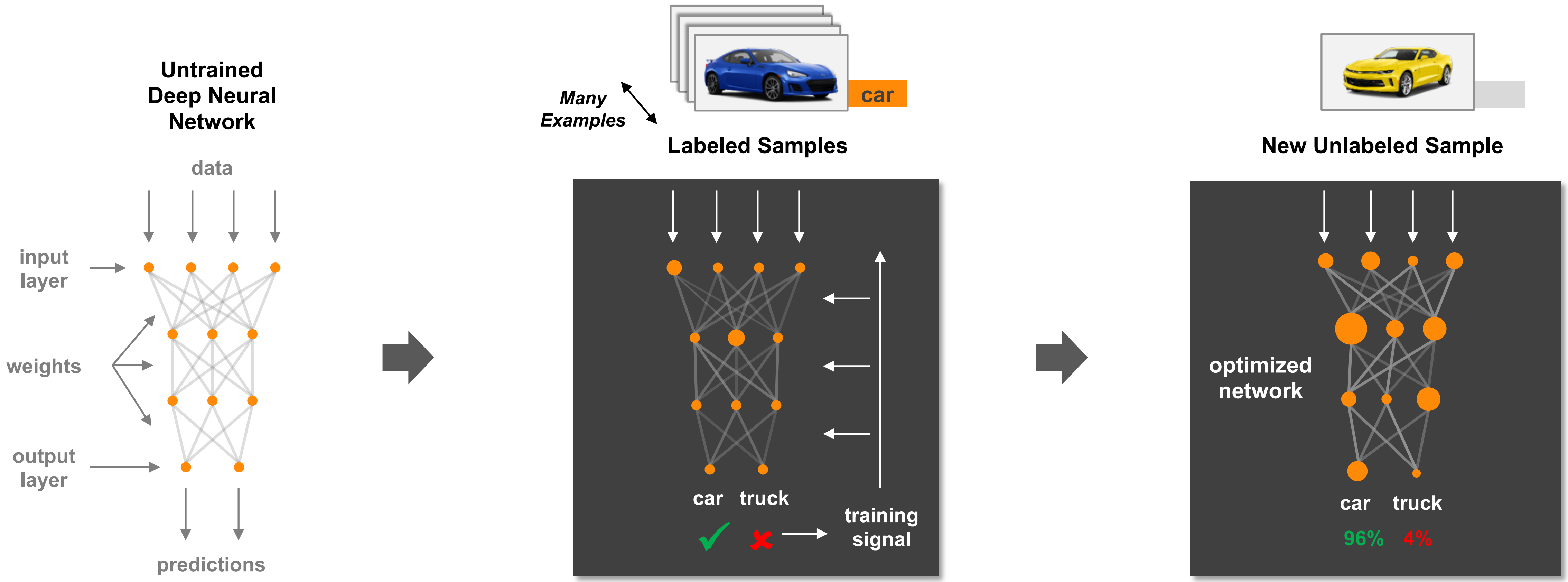
- **Computing systems inspired by biological networks**
- **Systems learn by repetitive training to do tasks based on examples**
  - **Generally a supervised learning technique (though unsupervised examples exist)**
- **Components: Inputs, Layers, Outputs, Weights**
- **Deep Neural Network: Lots of "hidden layers"**
- **Popular variants:**
  - **Convolutional Neural Nets**
  - **Recursive Neural Nets**
  - **Deep Belief Networks**
- **Very popular these days with many toolboxes and hardware support**



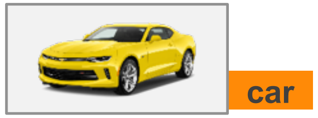
# Supervised Learning with Deep Neural Networks

## Training vs. Inference

**Training Phase** | **Deployment (Inference) Phase**




*Network weights are adjusted to correctly predict labels*

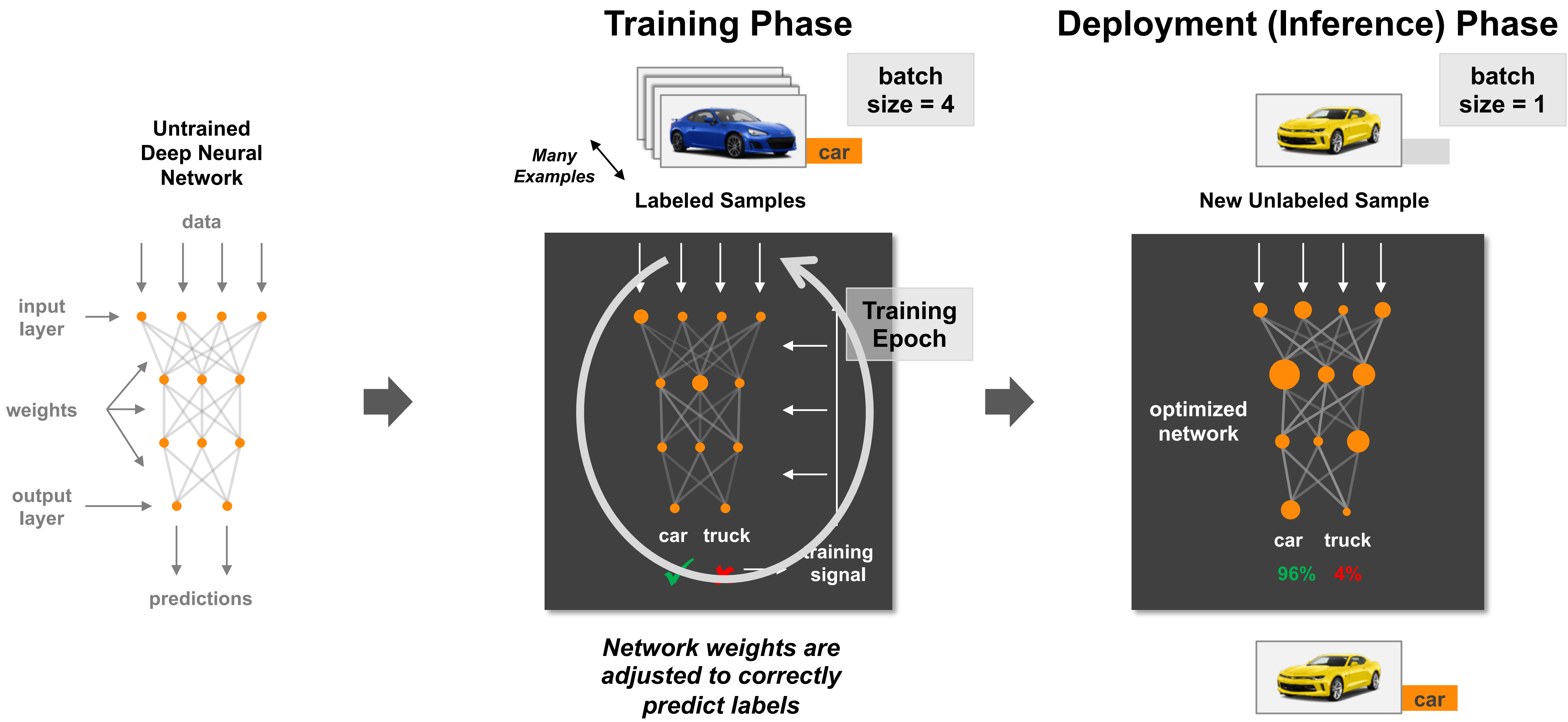





*Adapted from: https://blogs.nvidia.com/blog/2016/08/22/difference-deep-learning-training-inference-ai*

# Supervised Learning with Deep Neural Networks

## Training Epochs and Training/Inference Batches






*Adapted from: https://blogs.nvidia.com/blog/2016/08/22/difference-deep-learning-training-inference-ai*

# Neural Network Landscape

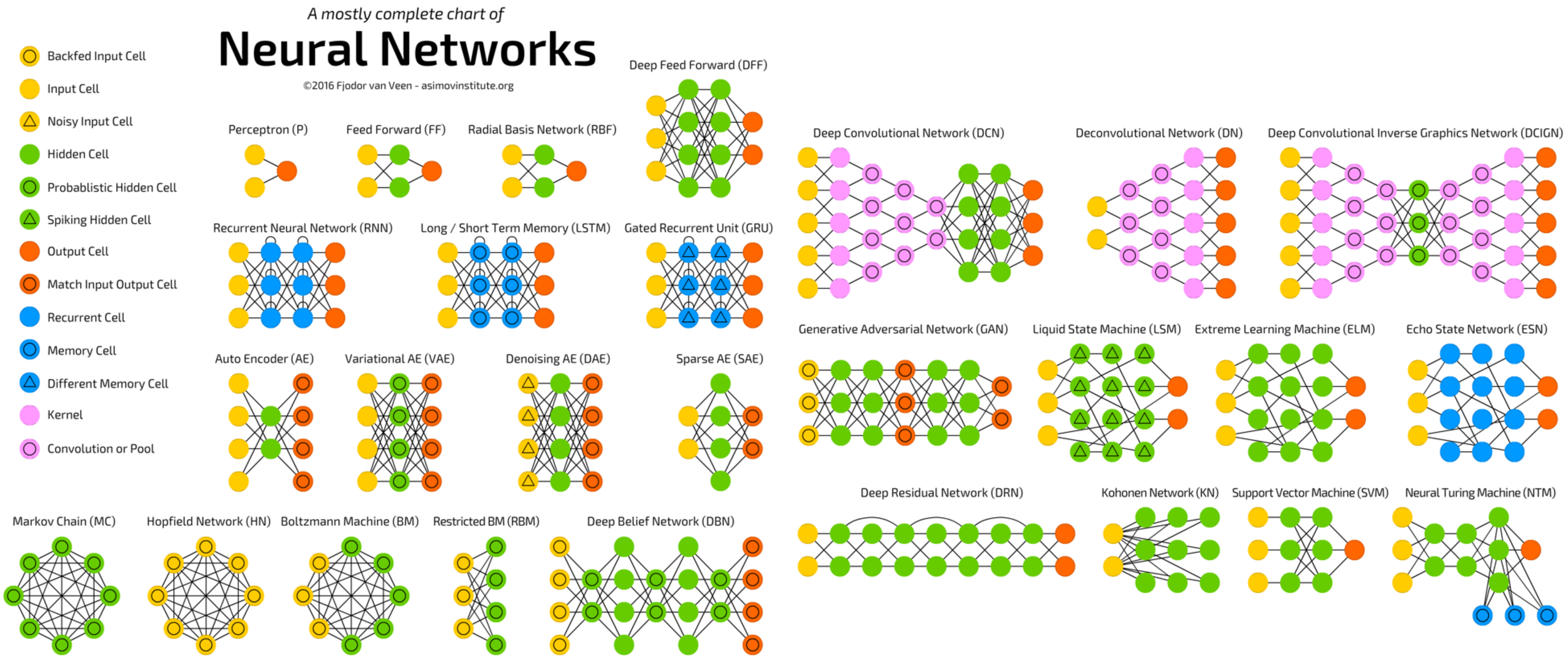


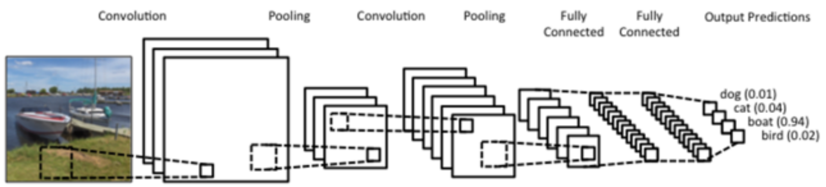

Image Source: http://www.asimovinstitute.org/neural-network-zoo/

# Primary Types of Deep Neural Networks

## Feed-Forward Neural Networks

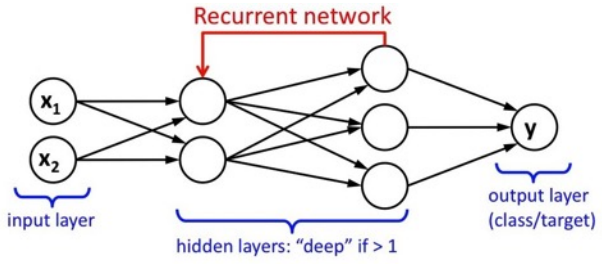

- Best for classification
- Input layer (yellow)
- Output layer (red)
- One or more hidden layers (green)
- Feed-forward weights associated with each line
- Bias weights associated with each neuron

## Convolutional Neural Networks




- Best for image processing classification
- Trainable convolution filters
- Include pooling layers after convolution layers
- Feed forward (fully-connected) layers complete classification of data

## Recursive Neural Networks

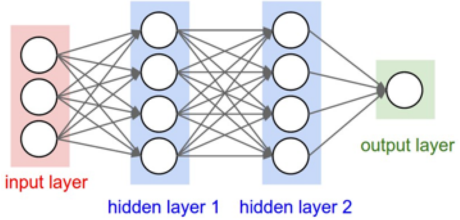



- Best for signal processing classification
- Input can be in time-domain, frequency domain (FFT), other
- Recursive weights capture time-dependent features of inputs

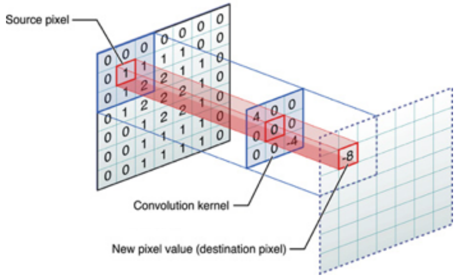
# Common NN Layers and Activation Functions

## Common Layers

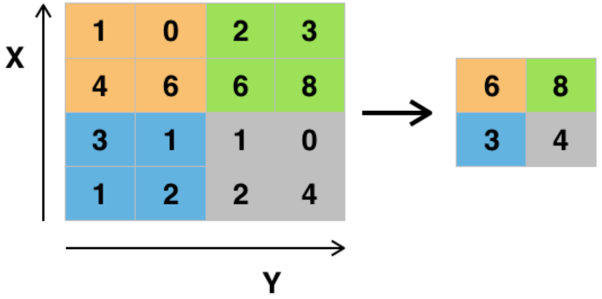
Fully Connected



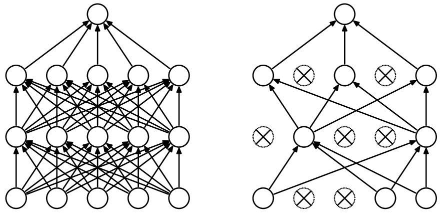

Convolutional
(Deconvolutional)




MaxPool




Dropout

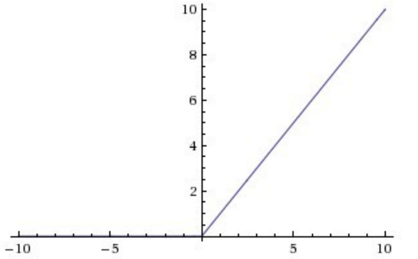


Before After

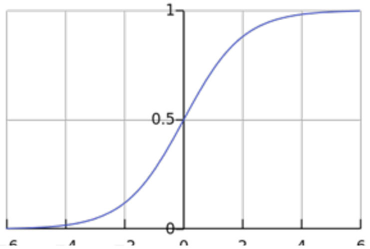
Others: Softmax, Skip Layer, etc.

## Activation Functions

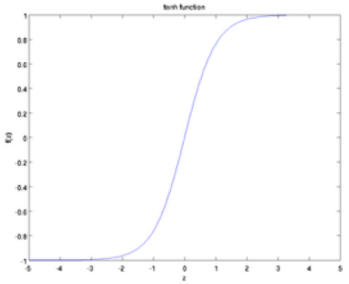
Rectified Linear Unit (ReLU):
$$f(x) = max(0, x)$$



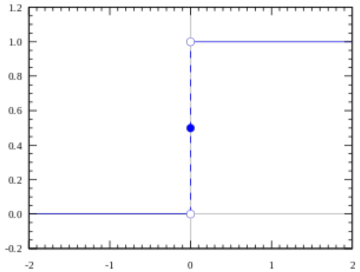
Sigmoid Function:
$$f(x) = \frac{1}{1 + e^{-x}}$$



Tanh Function:
$$f(x) = tanh(x)$$



Step Function:
$$f(x) = \begin{cases} 0, x < 0 \\ 1, x \geq 0 \end{cases}$$



# Number Representations

## Floating Point and Integers

### Common Floating Point Representations

Example: 123.45 = $1.2345 \times 10^2$ = $0.12345 \times 10^3$ = `0100 0010 1111 0110 1110 0110 0110 0110` (fp32)

**fp64: Double-precision IEEE Floating Point Format**

Range: ~1e-$^{38}$ to ~3e$^{38}$ Number of Decimal Digits: ~15.9

Exponent: 11 bits Mantissa (Significand): 52 bits

`S EEEEEEEEEEE MMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMMM`

**fp32: Single-precision IEEE Floating Point Format**

Range: ~1e-$^{38}$ to ~3e$^{38}$ Number of Decimal Digits: ~7.2

Exponent: 8 bits Mantissa (Significand): 23 bits

`S EEEEEEEE MMMMMMMMMMMMMMMMMMMMMMM`

**fp16: Half-precision IEEE Floating Point Format**

Range: ~5.96e-$^{8}$ to 65504 Number of Decimal Digits: ~3.3

Exponent: 5 bits Mantissa (Significand): 10 bits

`S EEEEE MMMMMMMMMM`

**bfloat16: Brain Floating Point Format**

Range: ~1e-$^{38}$ to ~3e$^{38}$ Number of Decimal Digits: ~2.3

Exponent: 8 bits Mantissa (Significand): 7 bits

`S EEEEEEEE MMMMMMM`

### Common Integer Representations

**int, int32_t:**

Range: -2,147,483,648 ($-2^{31}$) to 2,147,483,647 ($2^{31} - 1$)

`S bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb`

**unsigned int, uint32_t:**

Range: 0 to 4,294,967,295 ($2^{32} - 1$)

`bbbbbbbbbbbbbbbbbbbbbbbbbbbbbbbb`

**short, int16_t:**

Range: -32,768 ($-2^{15}$) to 32,767 ($2^{15} - 1$)

`S bbbbbbbbbbbbbbb`

**unsigned short, uint16_t:**

Range: 0 to 65,535 ($2^{16} - 1$)

`bbbbbbbbbbbbbbbb`

**int8_t:**

Range: -128 ($-2^{7}$) to 127 ($2^{7} - 1$)

`S bbbbbbb`

**char, uint8_t:**

Range: 0 to 255 ($2^{8} - 1$)

`bbbbbbbb`

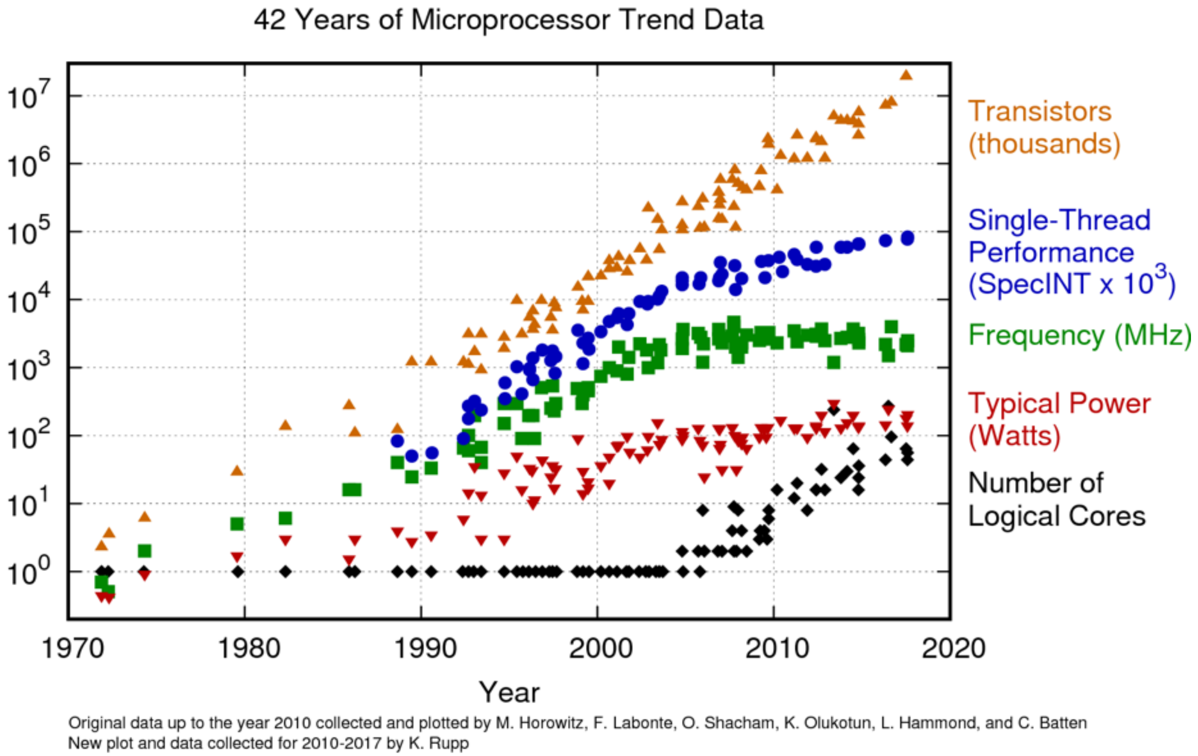
# Why Custom Processing Chips?

**In past 40 years, exhaustion of avenues**

- **Transistors**
- **Single thread performance**
- **Frequency**
- **Power**
- **Cores**

**Higher Performance will depend on**

- **Application specificity**
- **Kernel core blocks (circuit IP)**



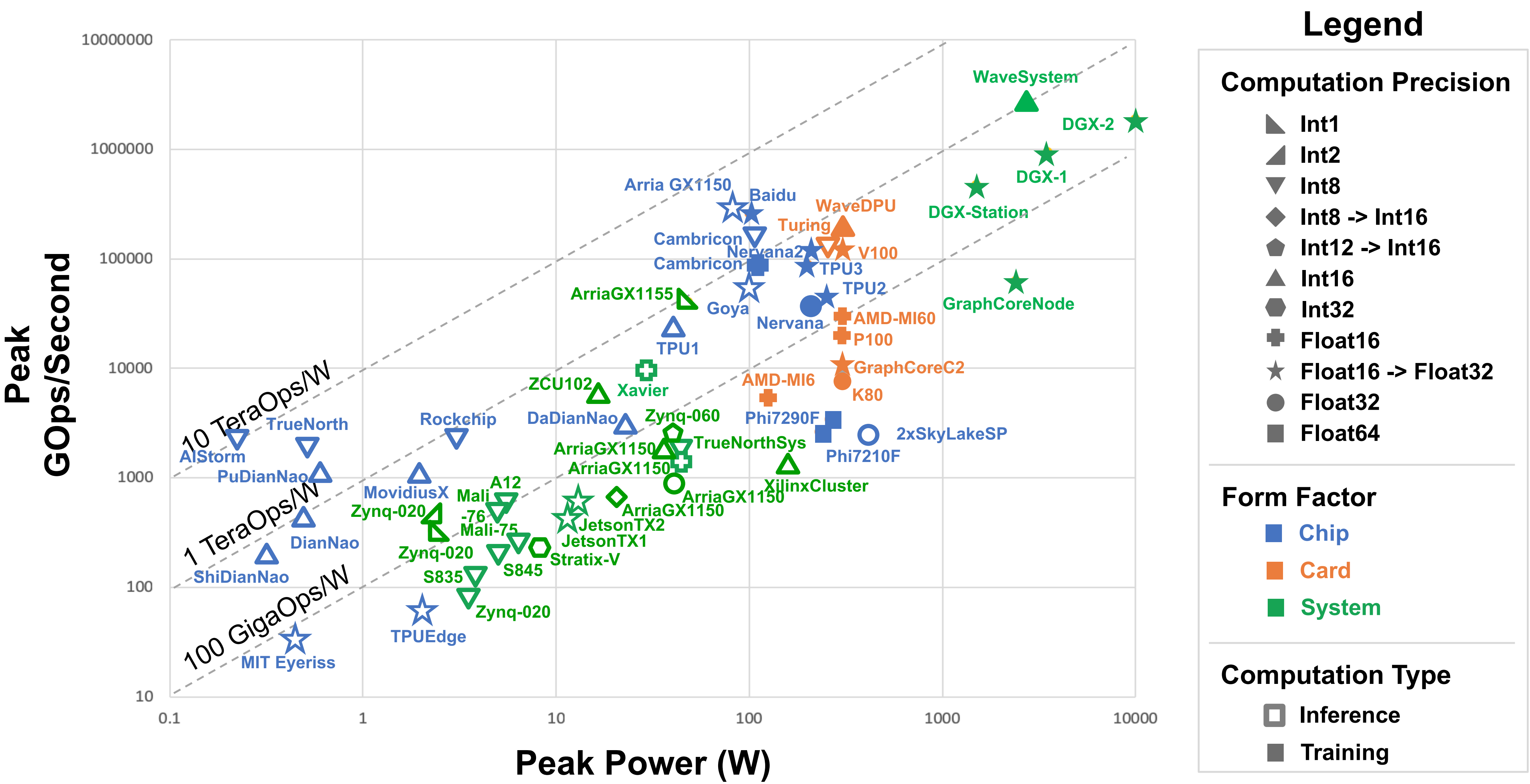

Source : https://www.karlrupp.net/2015/06/40-years-of-microprocessor-trend-data/

Mark Horowitz Computing's Energy Problem: https://ieeexplore.ieee.org/document/6757323

# Outline

- Introduction
  - Neural Networks and AI Landscape
  - Training and Inference
  - Numerical Precision
  - Why create custom AI chips?
- AI Processor/Accelerator Landscape
  - Dimensions of Taxonomy
  - Technology Examples
- Embedded Accelerator Benchmarking
  - Intel Movidius and Google TPU Edge
  - Results
- Summary

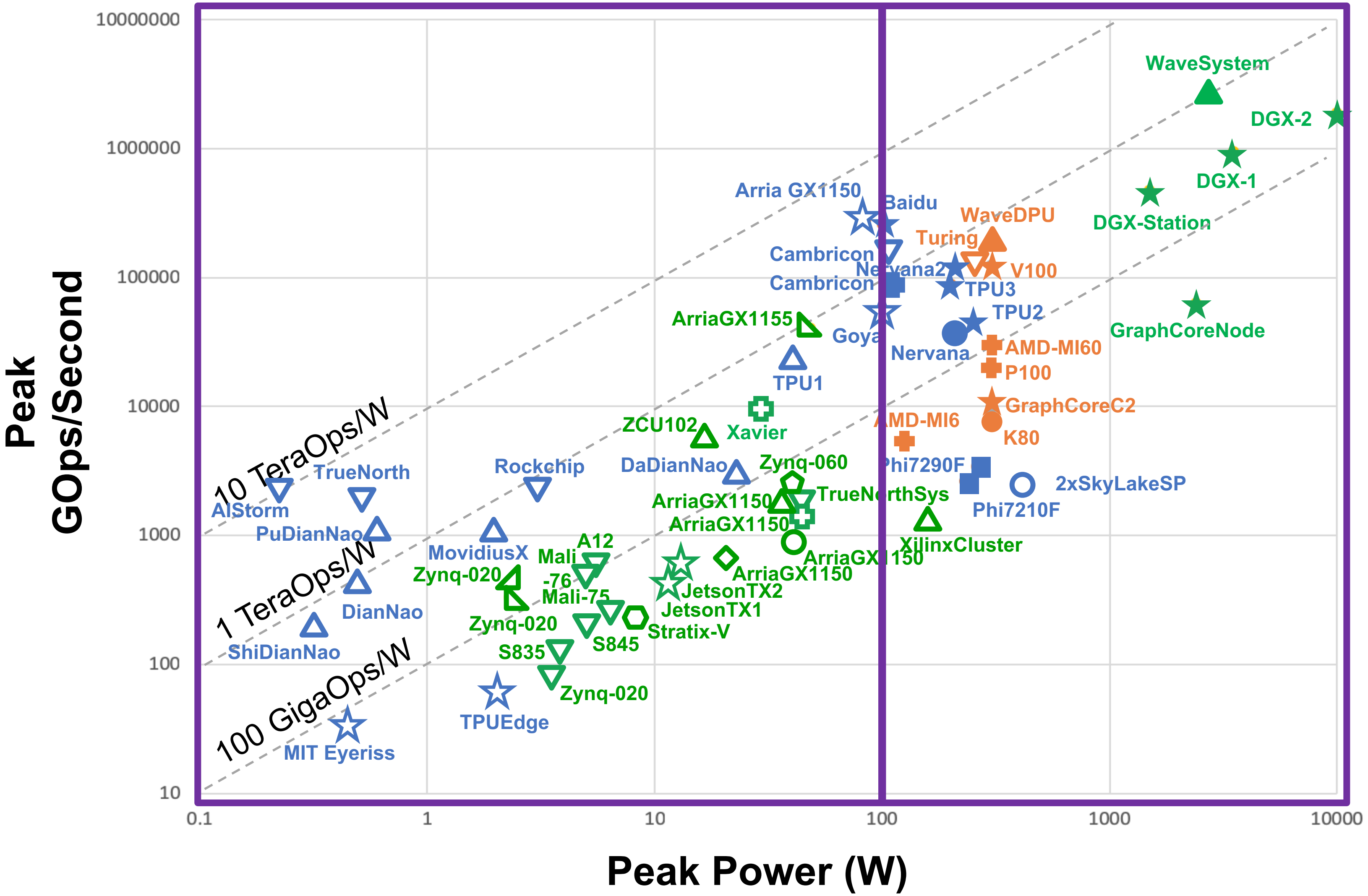
# Neural Network Processing Performance






*Slide courtesy of Albert Reuther, MIT Lincoln Laboratory Supercomputing Center*

# Neural Network Processing Performance




## Legend

**Computation Precision**
- Int1
- Int2
- Int8
- Int8 -> Int16
- Int12 -> Int16
- Int16
- Int32
- Float16
- Float16 -> Float32
- Float32
- Float64

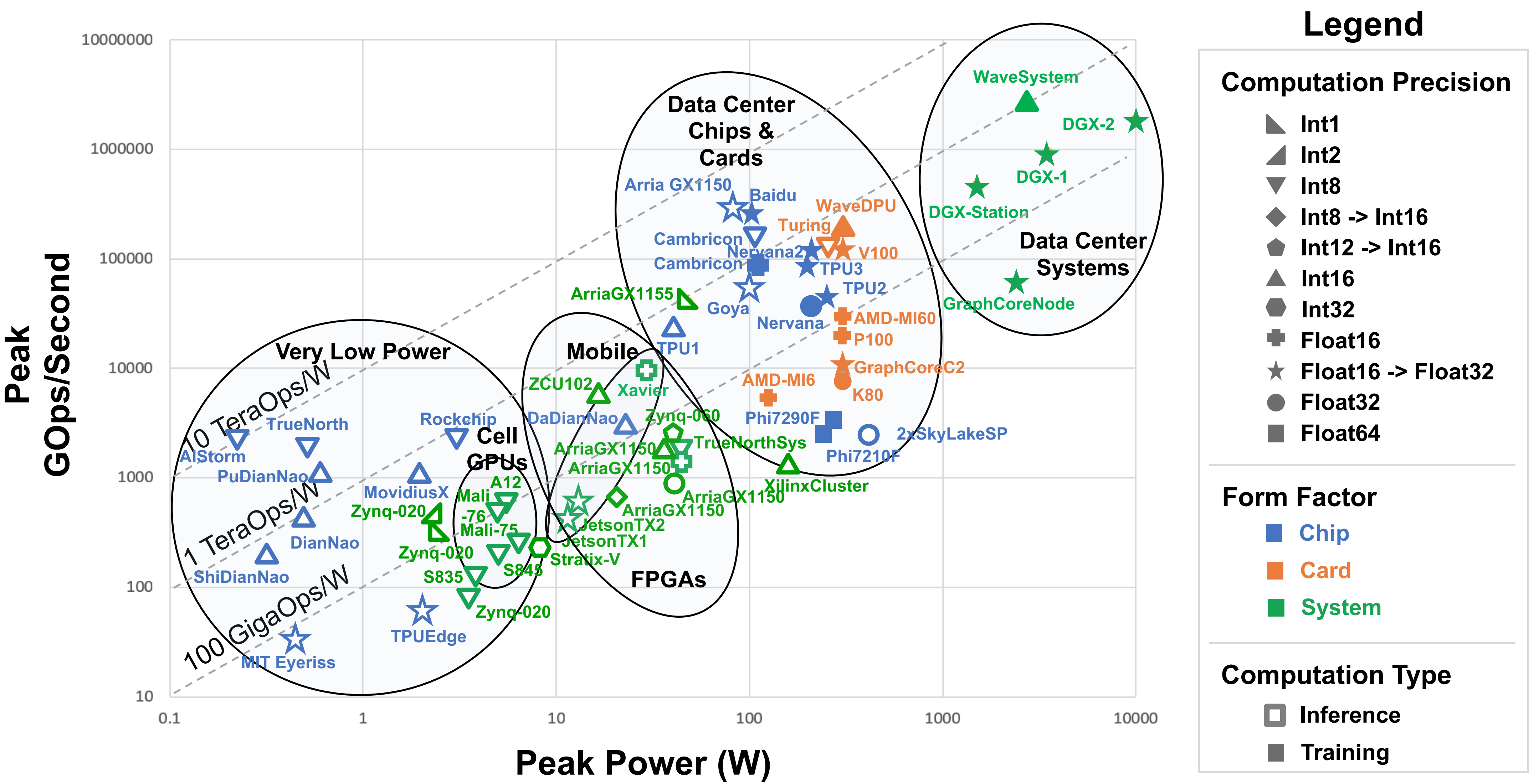
**Form Factor**
- Chip
- Card
- System

**Computation Type**
- Inference
- Training



*Slide courtesy of Albert Reuther, MIT Lincoln Laboratory Supercomputing Center*

# Neural Network Processing Performance



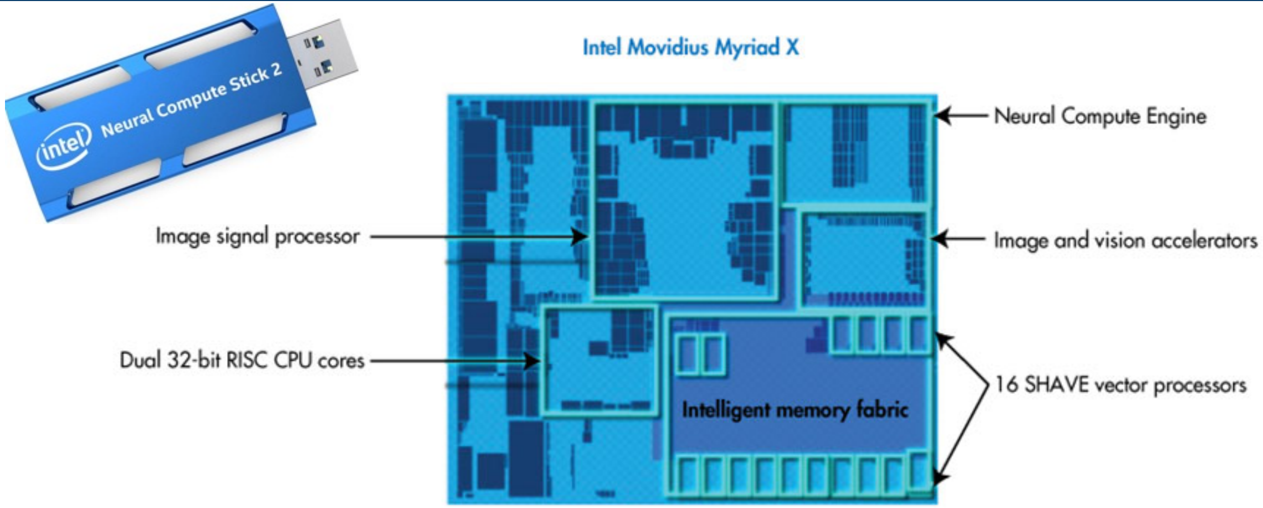


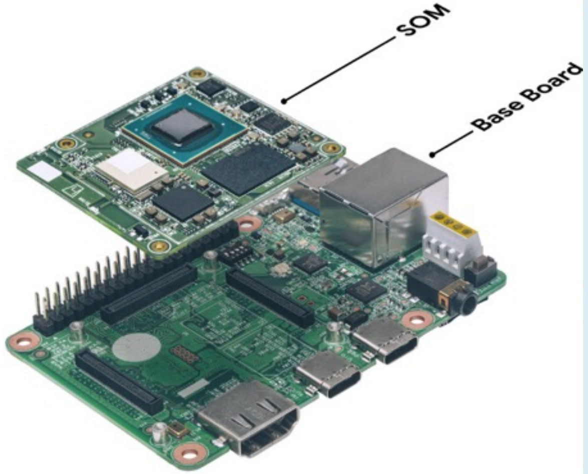

*Slide courtesy of Albert Reuther, MIT Lincoln Laboratory Supercomputing Center*

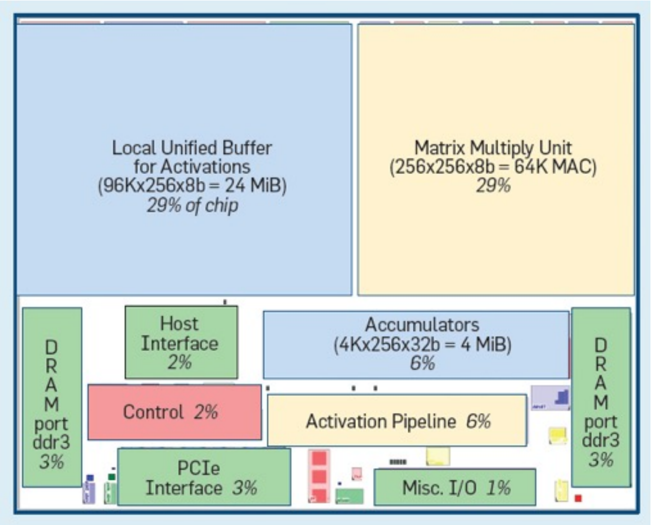
# Outline

- **Introduction**
  - **Neural Networks and AI Landscape**
  - **Training and Inference**
  - **Numerical Precision**
  - **Why create custom AI chips?**
- **AI Processor/Accelerator Landscape**
  - **Dimensions of Taxonomy**
  - **Technology Examples**
- **Embedded Accelerator Benchmarking**
  - **Intel Movidius and Google TPU Edge**
  - **Results**
- **Summary**

# Intel Movidius Myriad X and Google TPU Edge




| | Movidius Myriad X | TPU Edge |
|---|---|---|
| Use | Inference Only | Inference Only |
| Released | Nov 2018 | Dec 2018 |
| Inference Engine | Neural Compute Engine | 256x256 Matrix Multiply Units (MXU) systolic array matrix-matrix multiplier |
| Memory | 4 GB | 1 GB |
| Precision | Int8 | Int8, Int16 |
| Peak DNN Throughput | 160 GOPS | 58.5 GOPS |



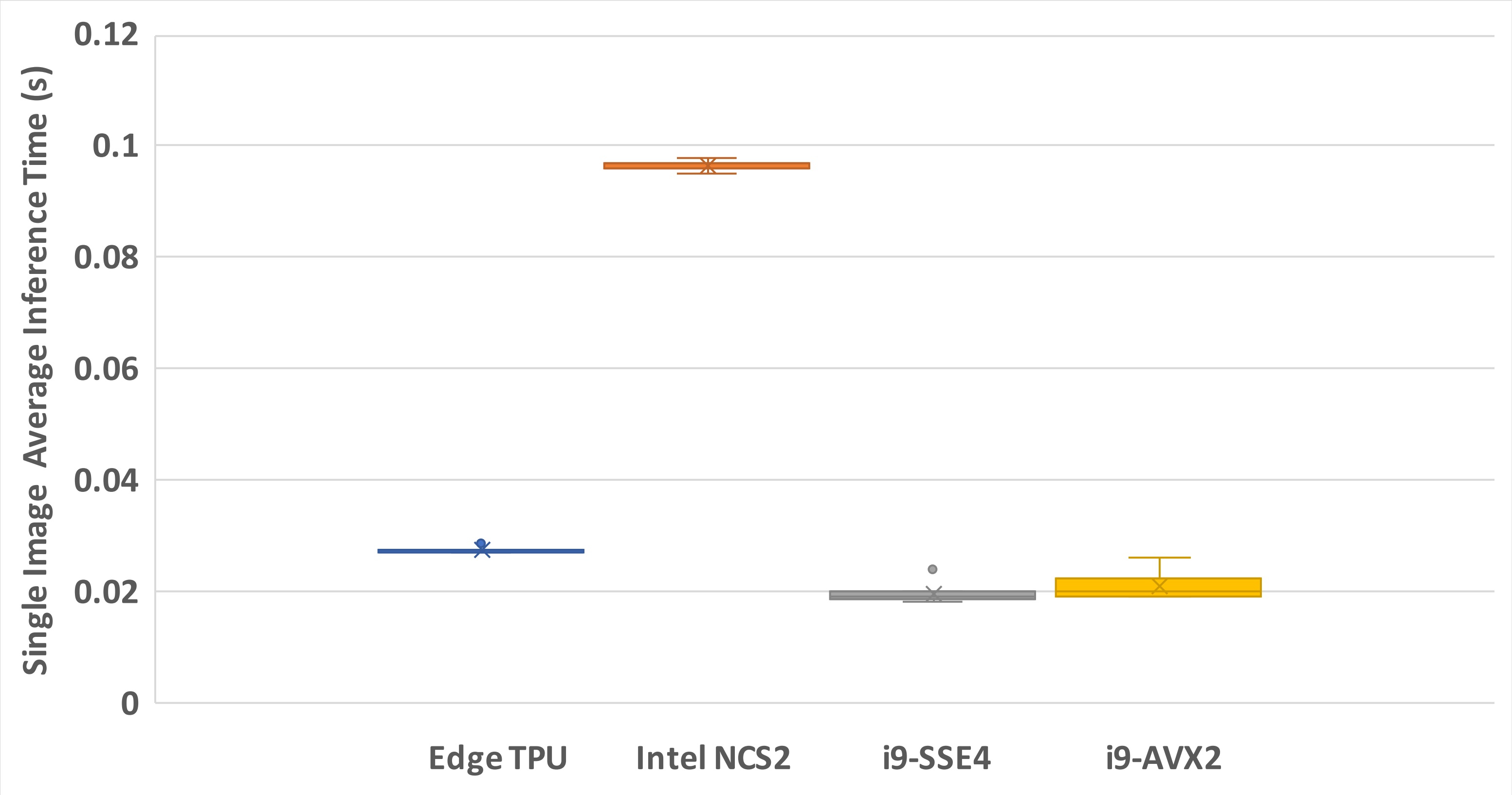
MXU = Matrix multiply unit
HBM = High bandwidth memory

# Embedded Inference Accelerator Performance

| | TPU Edge | NCS2 | i9-SSE4 | i9-AVX2 |
|---|---|---|---|---|
| NN Environment | TensorFlow | OpenVINO Lite | TensorFlow | TensorFlow |
| Mobilenet Model | v1 | v2 | v2 | v2 |
| Reported GOPS | 58.5 | 160 | | |
| **Measured GOPS** | **47.4** | **8.29** | **38.4** | **40.9** |
| Reported Power (W) | 2.0 | 2.0 | 205 | 205 |
| Measured Power (W) | 0.85 | 1.35 | | |
| Reported GOPS/W | 29.3 | 80.0 | | |
| **Measured GOPS/W** | **55.8** | **6.14** | | |
| Avg. Model Load Time (s) | 3.66 s | 5.32 s | 0.36 s | 0.36 s |
| Avg. Single Image Inference Time (ms) | 27.4 ms | 96.4 ms | 19.6 ms | 20.8 ms |

- **Much lower power on TPU Edge and NCS2**
- **Similar performance from TPU Edge and i9**
- **Slower model load time on TPU Edge and NCS2**

# Single Image Inference Times




- **Similar single image inference time from Edge TPU and i9**
- **NCS2 slower which affects GOPS/W**

# Summary

- Application customization necessary for further performance gains

- Numerical precision, NN models, and layers all influence the intensity of training and inference performance

- Many products and research projects exploring application customization for AI / ML accelerators
  – CPU / CPU mesh acceleration
  – GPU Thread-parallel acceleration
  – Dataflow accelerators

- Embedded inference accelerators approaching CPU vector performance with much lower power use