The Future of Machine Learning Acceleration

Elliott Delaye
Distinguished Engineer

- Machine Learning - Neural Networks
- Computation & Memory Requirements
- Algorithmic Optimization Techniques
- Hardware Architectures

Neural Networks

A.I. – Machine Learning - Neural Networks

Artificial Intelligence (A.I.)

Computer Vision | Pattern Recognition | **Machine Learning** | Cognitive Robotics . . .

Linear Regression | K-Means Clustering | **Neural Networks** | Decision Trees . . .

“machine mimics cognitive functions such as learning and problem solving”

“Gives computers the ability to learn without being explicitly programmed”

“Predominantly used ML algorithm Mimics the human brain”

Neural Networks (Deep Neural Networks, etc.)
Why are they so popular?

> **Requi**

> **NNs a**

> **If you**
>> C

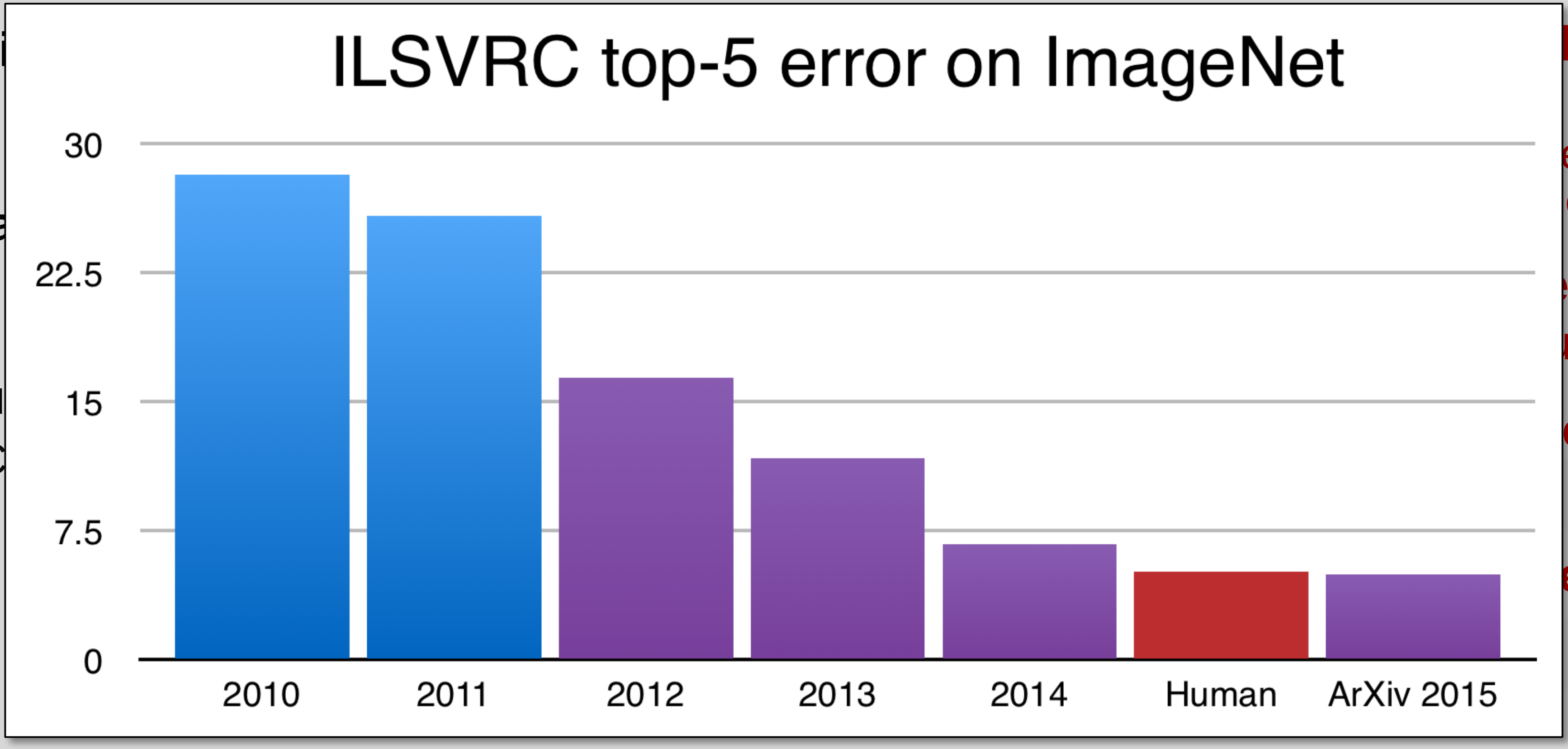
place other

e simple rules can
em

eviously
uters

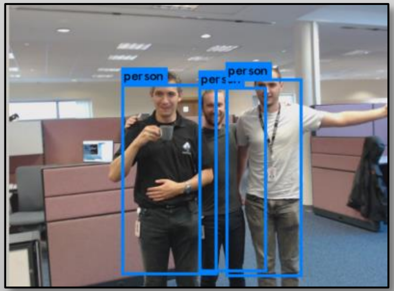
ely unsolved

etworks old?



Increasing Range of Applications


Image Classification


Object Detection


Semantic Segmentation

Computer Vision
CNNs


Speaker Diarization


Speech Recognition

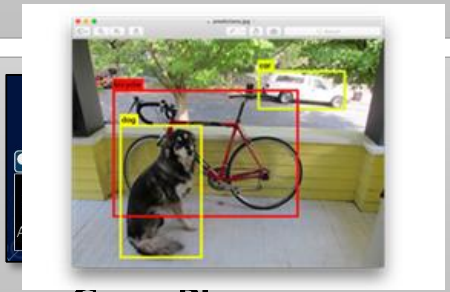
Speech Recognition
RNNs, LSTMs


Translation


Sentiment Analysis

Natural Language Processing
Sequence to sequence

Recommender


GamePlay

Many more emerging…

Others

Popular Neural Networks

Computer Vision
CNNs

ResNet50, VGG, AlexNet, InceptionV3

Image Classification

Faster R-CNN, SSD, YoloV2

Object Detection

Mask-R-CNN, SSD

Semantic Segmentation

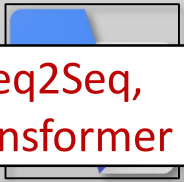
eech Recognition
RNNs, LSTMs

Speaker Diarization

DeepSpeech2

Speech Recognition

Natural Language Processing
Sequence to sequence



Seq2Seq, Transformer

Translation

Seq-CNN

Sentiment Analysis

Others

NCF

Recommender

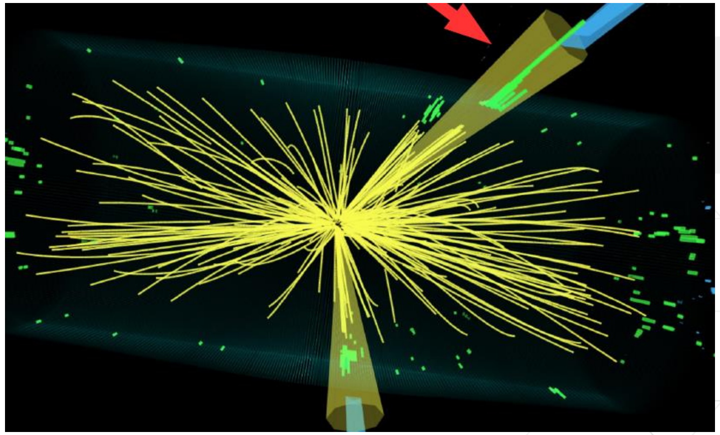
MiniGo, DeepQ, A3C

GamePlay

Adopted from MLPerf, Fathom, TDP

XILINX

New uses – High Energy Physics



Collision rate is 40 MHz
A new collision every 25ns

Detector → ASIC → FPGA → CPU Local → CPU Grid

Detector / ASIC

Preprocessing of detectors
High radiation environment
High Magnetic field
Specialized detectors

Fixed

FPGA

Programmable Combination of detectors

Changes with LHC beam

CPU Local

Detailed Programmable Combination of detectors

Access to all detectors

CPU Grid

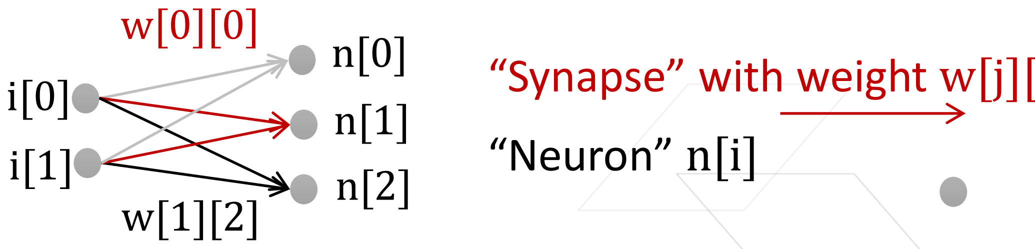
Final processing of detectors

This is data
We use to make discoveries

Latency : 10μs → 100ms → 10s

Convolutional Neural Networks (CNNs)
from a computational point of view

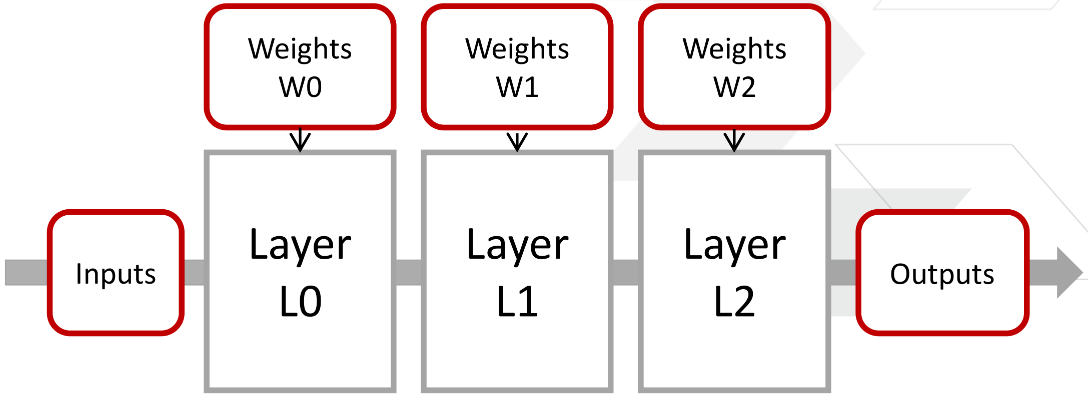
- **CNNs are usually feed forward* computational graphs constructed from one or more layers**
  - Up to 1000s of layers



- **Each layer consists of “neurons” n[i] which are interconnected with synapses, associated with weights w[i][j]**

$$n[0] = \text{Activation}(w[0][0]*i[0] + w[1][0]*i[1])$$

- **Each neuron computes:**
  - Typically a dot-product (multiply-accumulate)
  - Followed by a non-linear “activation” function
  - Without non-linear function, L0, L1, L2 could be collapsed into a single layer

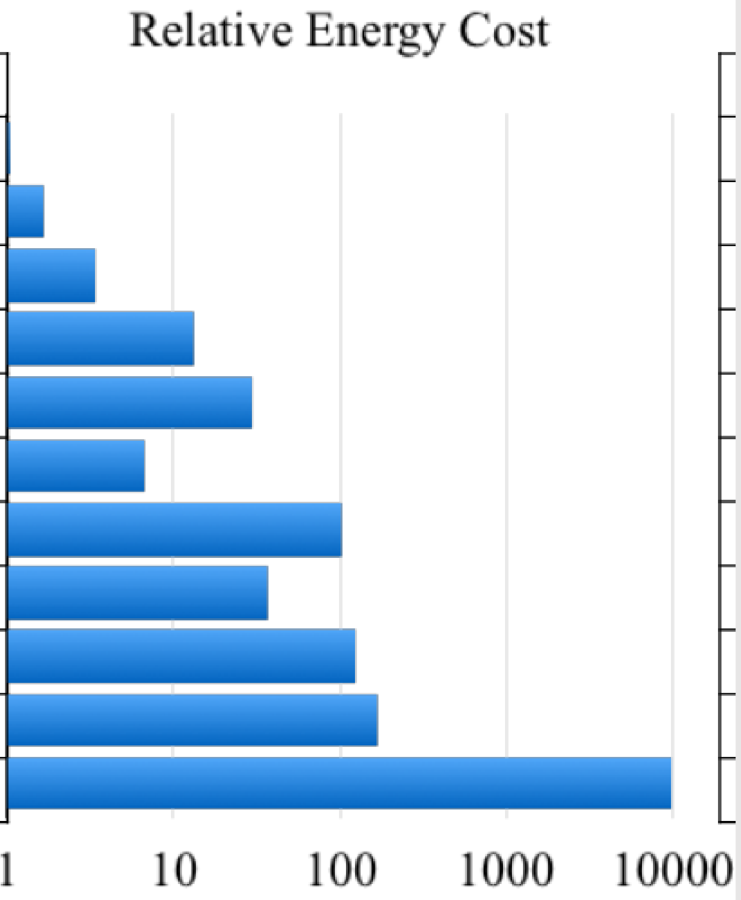


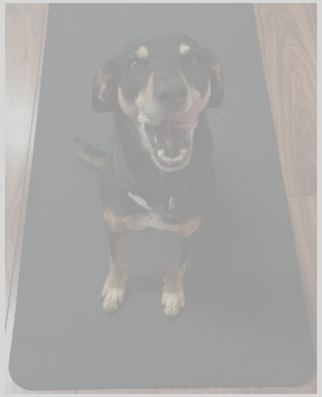
* With exception of RNNs

From Training to Inference

Training dataset

Operation:	Energy (pJ)
8b Add	0.03
16b Add	0.05
32b Add	0.1
16b FP Add	0.4
32b FP Add	0.9
8b Mult	0.2
32b Mult	3.1
16b FP Mult	1.1
32b FP Mult	3.7
32b SRAM Read (8KB)	5
32b DRAM Read	640

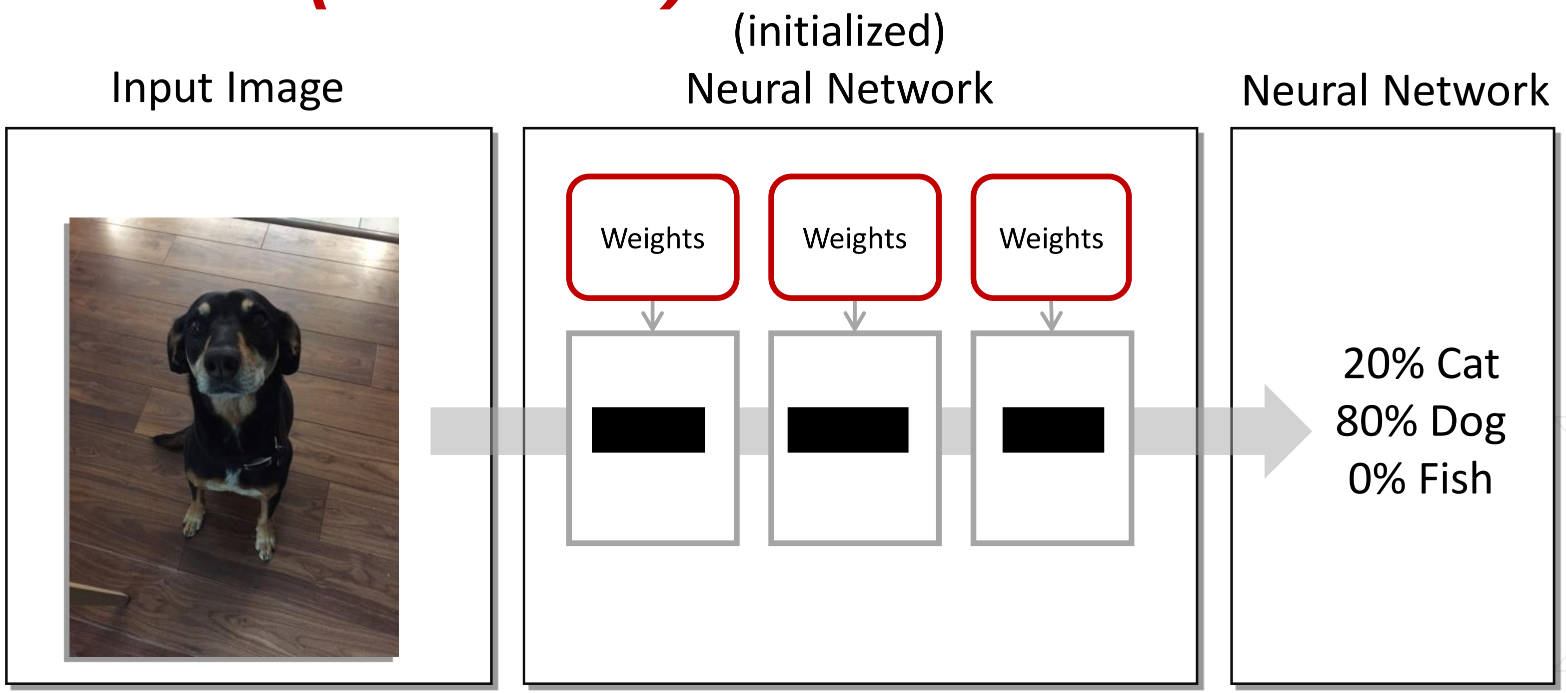




Deployment at the edge

Example: ResNet50

Forward Pass (Inference)



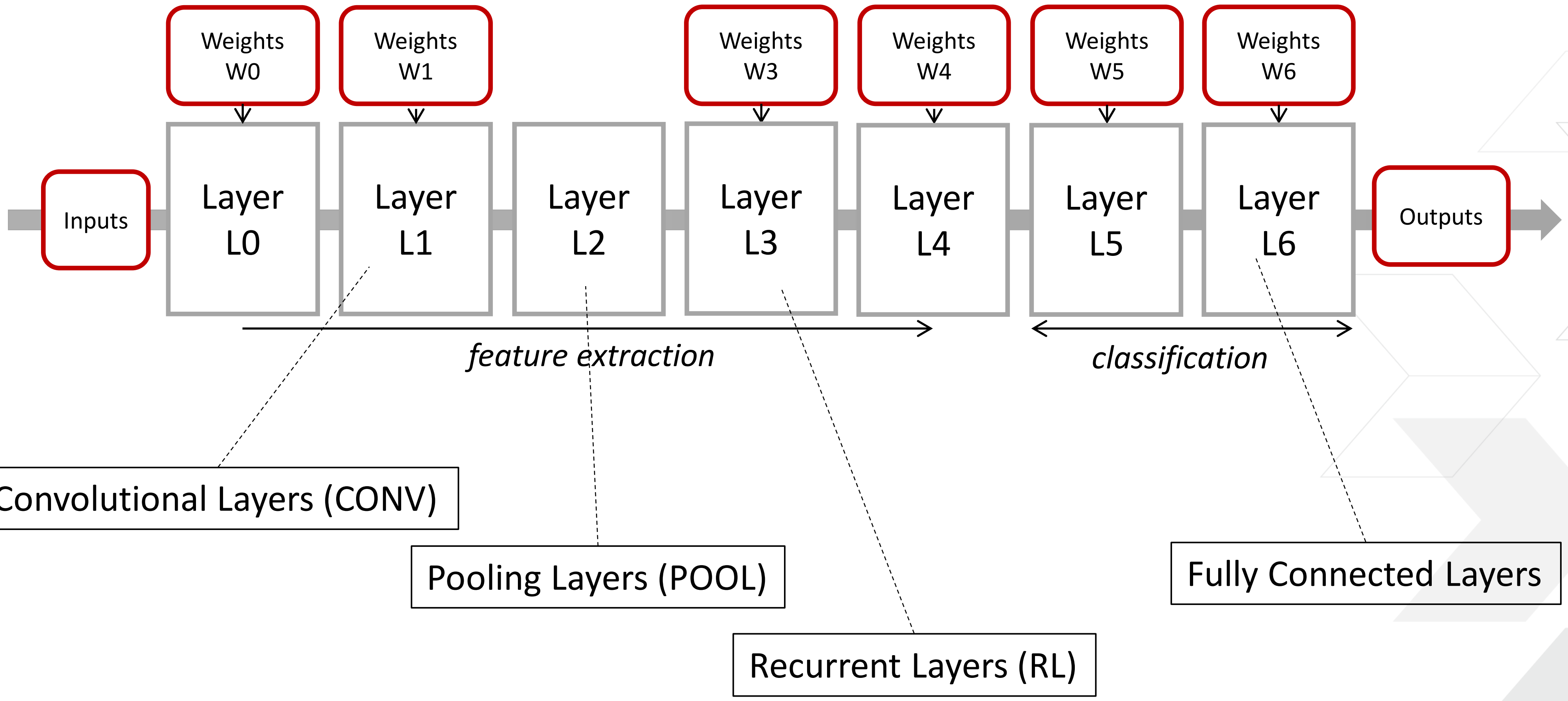
For ResNet50:

- 70 Layers
- 7.7 Billion operations
- 25.5 MBytes of weight storage*
- 10.1 MBytes for activations*

**Assuming int8*

XILINX

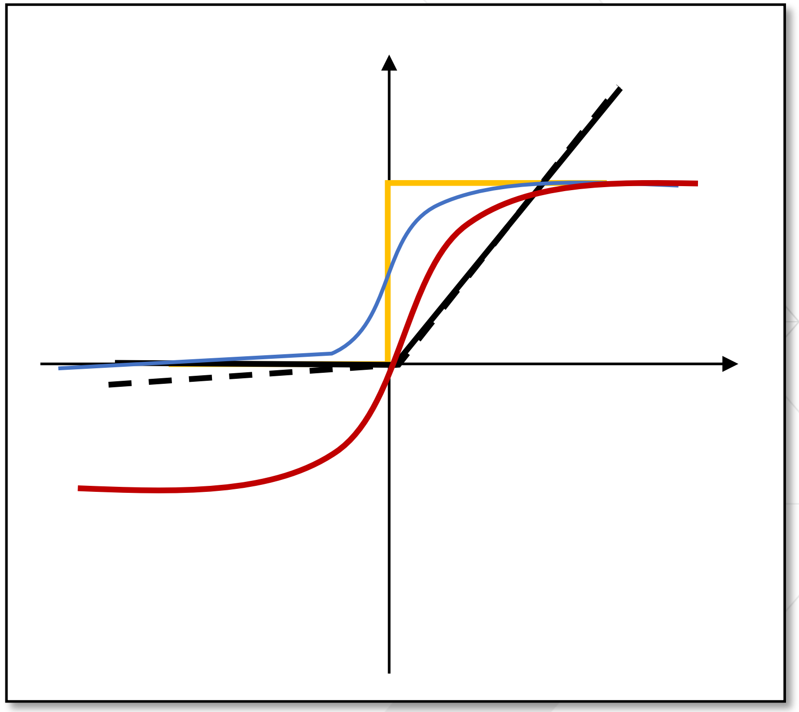
NNs in More Detail



Activation & Batch Normalization (RELU, PRELU, BN, SCALE, etc.)

Activation Functions

- > **Implements the concept of "Firing"**
  - >> Non-linear so we can approximate more complex functions
- > **Most popular for CNN: Rectified Linear Unit (ReLU)***
  - >> Popular as it propagates gradients better than bounded and easy to compute, x = max(0,x)
- > **Other common ones include: tanh, leaky ReLU, sigmoid, threshold functions for quantized neural networks**
- > **Implementation:**
  - >> Support for special functions as well as some level of flexibility



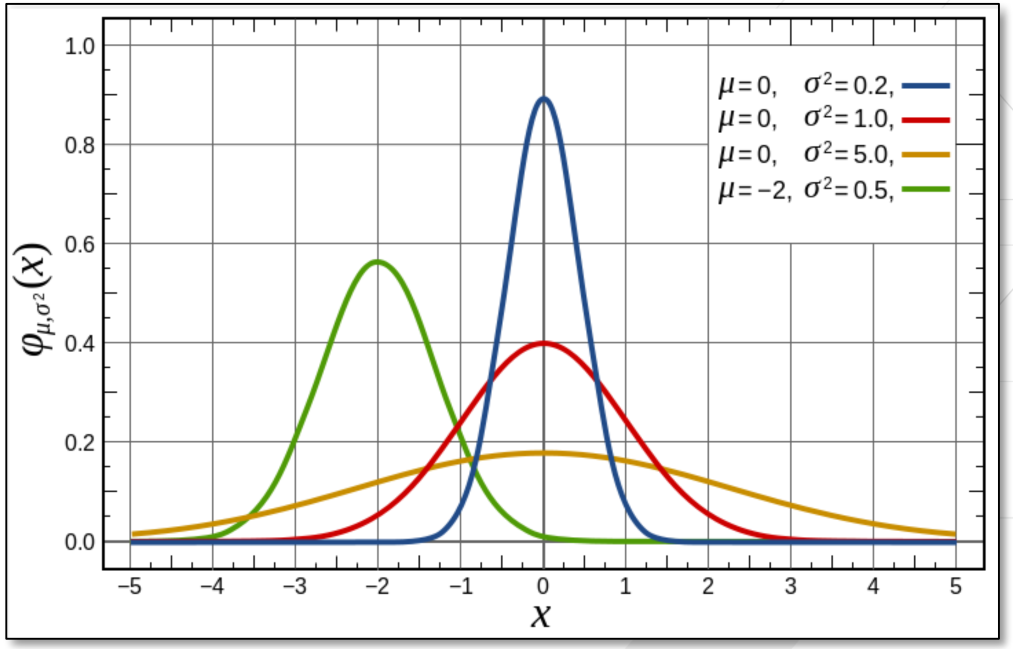
**Nair, V. and Hinton, G.E., 2010. Rectified linear units improve restricted boltzmann machines. In Proceedings of the 27th international conference on machine learning (ICML-10) (pp. 807-814).*

Batch Normalization

- **Normalizes the statistics of activation values across layers**

- **Significantly reduces the training time of networks, can improve accuracy and makes it less sensitive to initialization**

- **Compute:**
  - Lightweight at inference
  - Heavy duty during training
    - Subtract mean, divide by standard deviation to achieve zero-centered distribution with unit variance



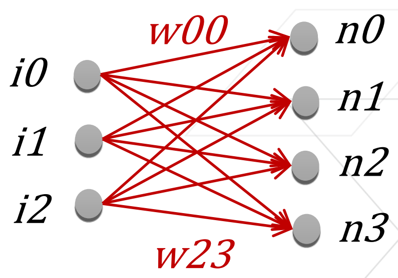
https://en.wikipedia.org/wiki/Normal_distribution

Ioffe, S. and Szegedy, C., 2015. Batch normalization: Accelerating deep network training by reducing internal covariate shift. arXiv preprint arXiv:1502.03167.

Fully Connected Layers

(aka inner product or dense layers)

- **Each input activation is connected to every output activation**
  - Why it's termed "Fully Connected"



- **Can be written as a matrix-vector product with an element-wise non-linearity applied afterwards.**

$$\begin{pmatrix} i0 & i1 & i2 \end{pmatrix} \times \begin{pmatrix} W00 & W01 & W02 & W03 \\ W10 & W11 & W12 & W13 \\ W20 & W21 & W22 & W23 \end{pmatrix} = \begin{pmatrix} n0' & n1' & n2' & n3' \end{pmatrix}$$

$$(n0\ n1\ n2\ n3) = Act(\,n0'\ n1'\ n2'\ n3')$$

- **Implementation Challenges**
  - Connectivity
  - High weight memory requirement: #IN * #OUT * BITS
  - Low arithmetic intensity assuming weights off-chip

    2 * #IN* #OUT / #IN * #OUT * BITS/8

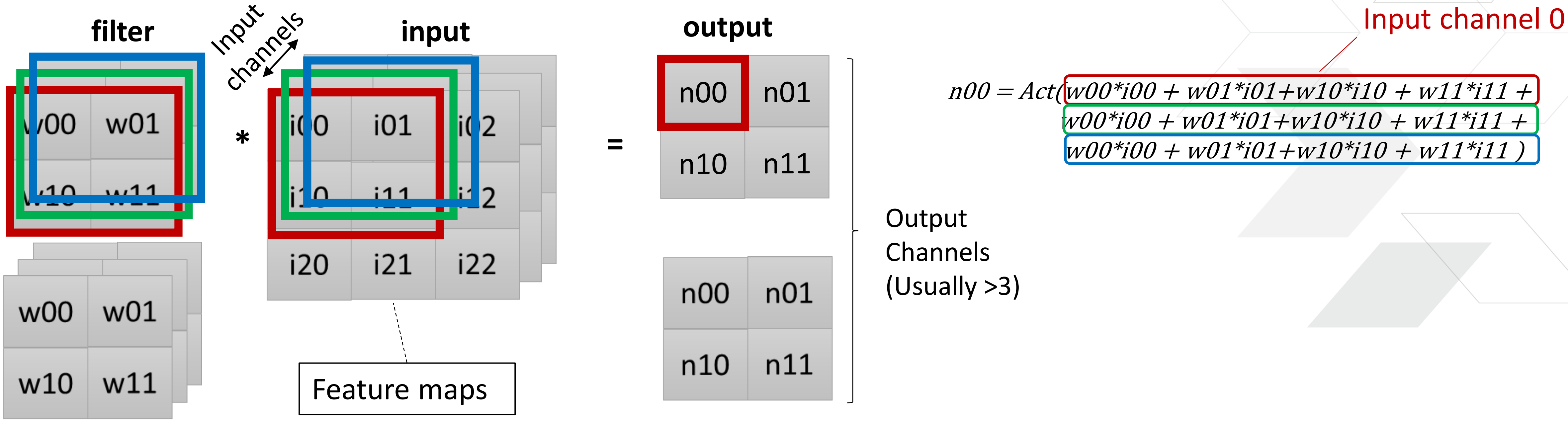
MODEL	CONV WEIGHTS (M)	FC WEIGHTS (M)
ResNet50	23.454912	2.048
AlexNet	2.332704	58.621952
VGG16	14.710464	123.633664

IN: *number of input channels*
OUT: *number of output channels*
BITS: *bit precision in data types*

Convolutional Layers

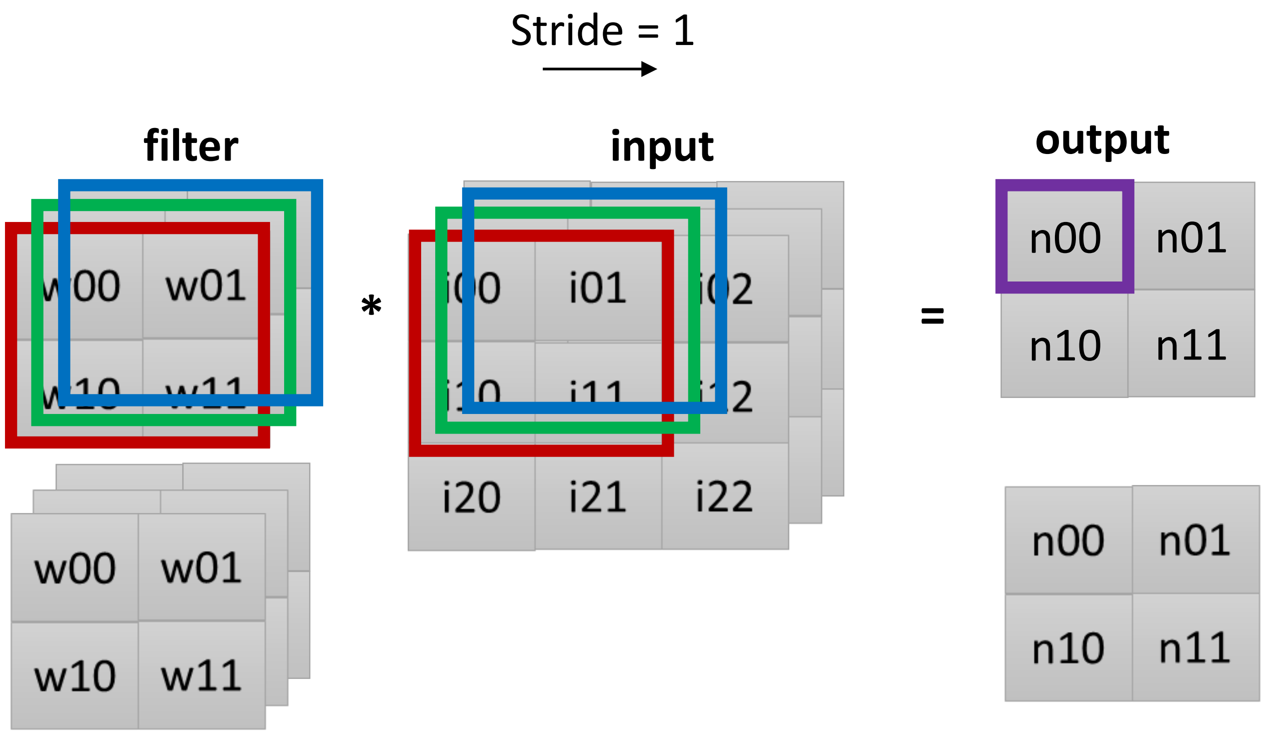
Example 2D Convolution

- Convolutions capture some kind of locality, spatial or temporal, that we know exists in the domain
- Input of each “neuron” reduced – Be
  - Applying convolution to all images in the previous layer
- Weights represent the filters used for convolutions



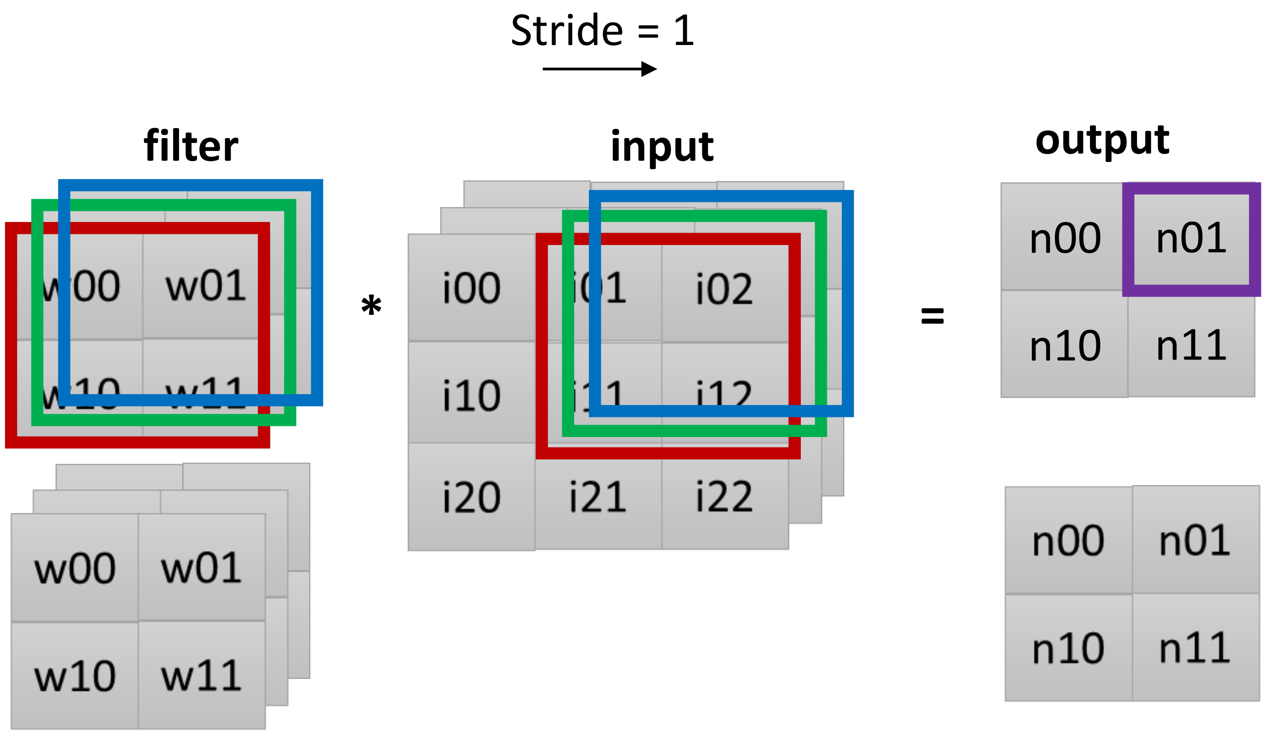
2D Convolutional Layers

- Slide the window till one feature map is complete
  - With a given stride size



2D Convolutional Layers

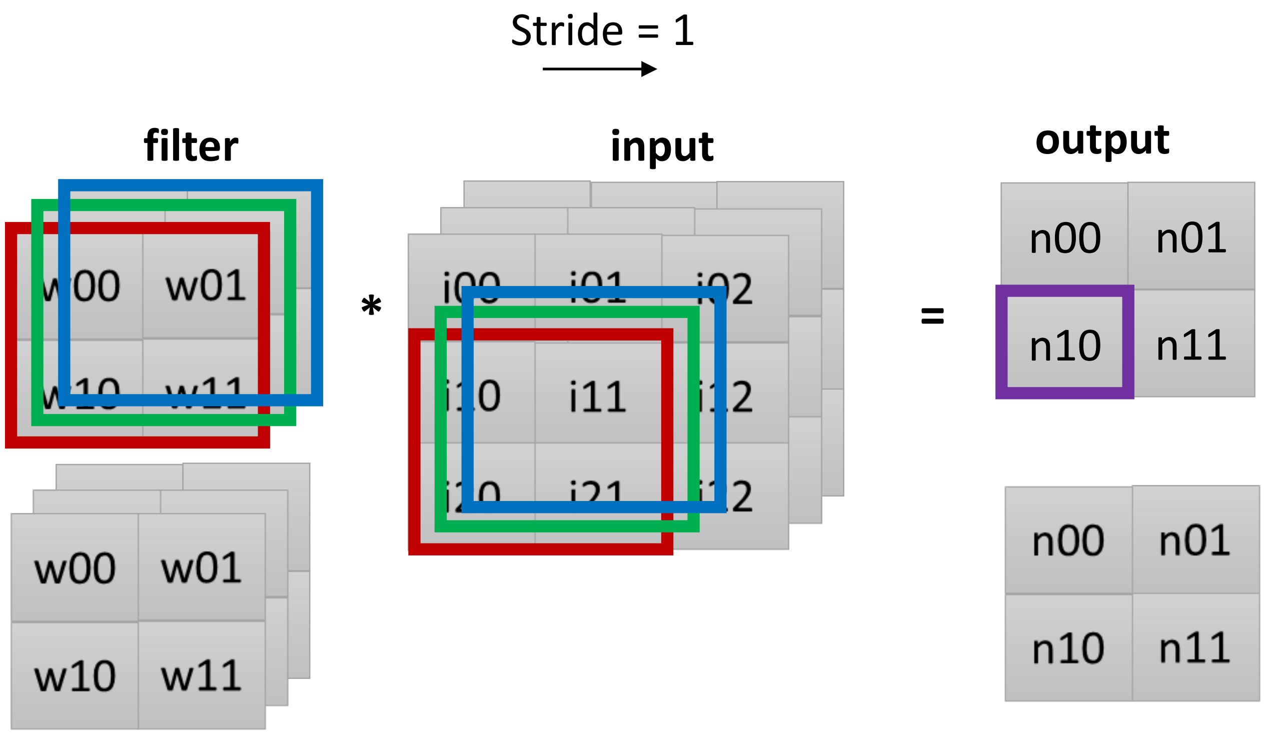
- Slide the window till one feature map is complete
  - With a given stride size



XILINX

2D Convolutional Layers

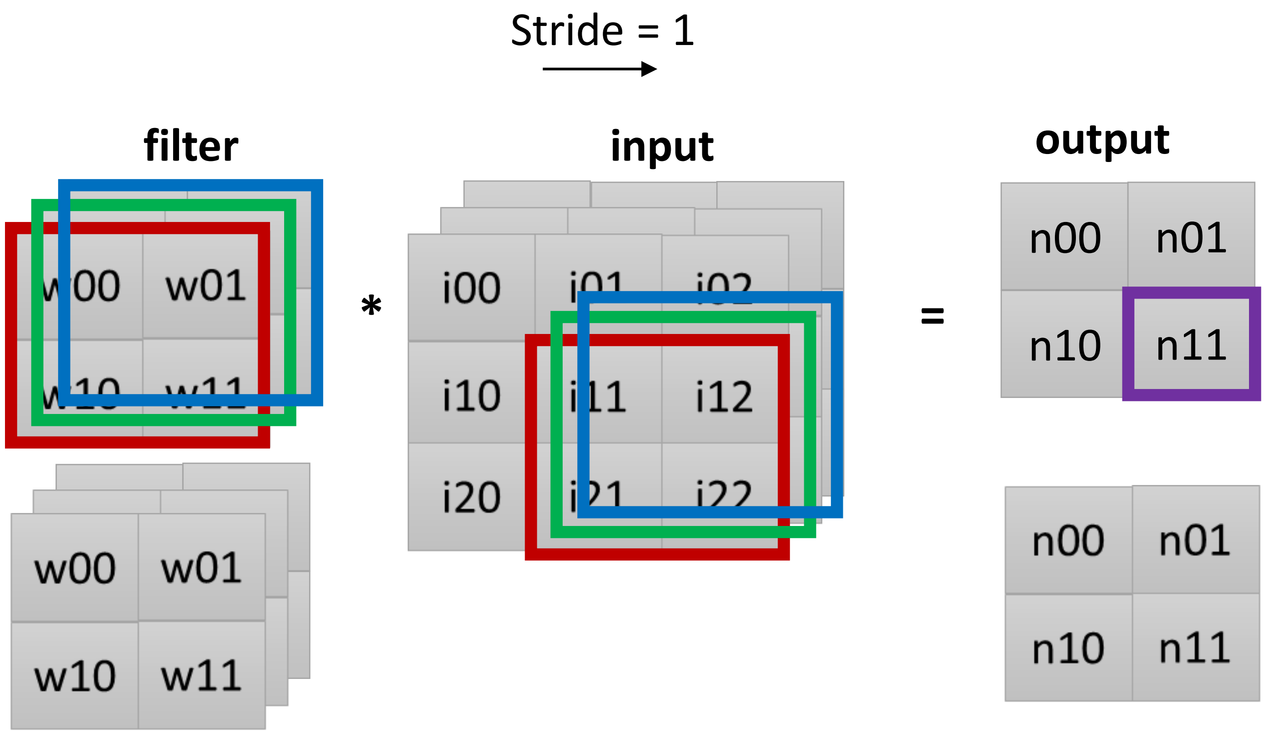
- **Slide the window till one feature map is complete**
  - With a given stride size



XILINX

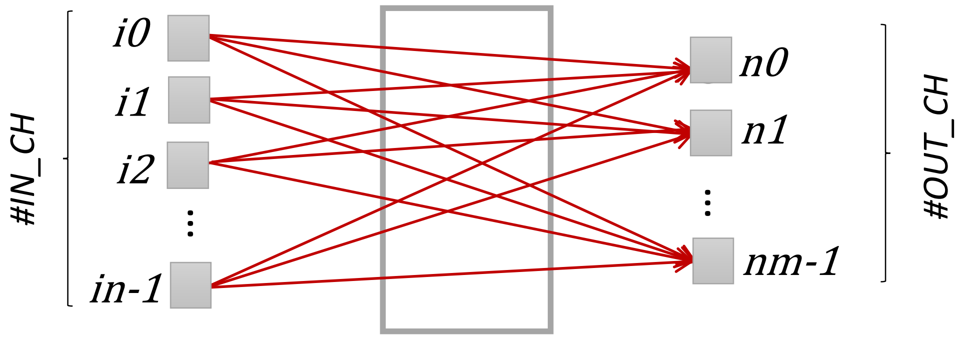
2D Convolutional Layers

- **Slide the window till one feature map is complete**
  - With a given stride size



Convolutions
Challenges

> **Channel connectivity issue**
  >> Every input channel information broadcasts to every output channel



100s to 1000 channels

MODEL	CONV [GOPS]	FC [GOPS]
ResNet50	7.712	0.004
AlexNet	1.332	0.044
VGG16	30.693	0.247

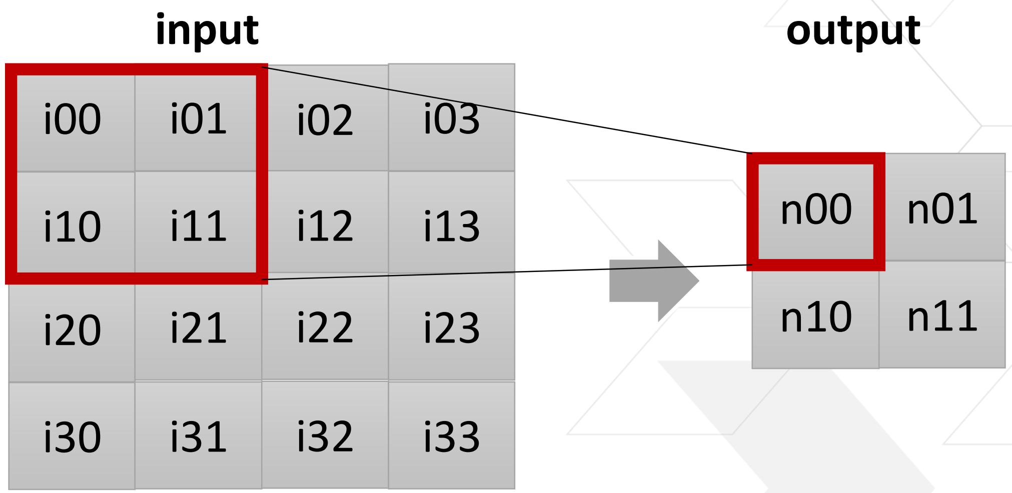
> **Huge amounts of compute**
  >> Dense convolutions account for the majority of the compute
  >> Image based so inputs could be HD video, 1920x1080

> **Lots of memory used**
  >> 1920x1080x3 = 6.2MB
  >> YOLOv2 First Output, 1920x1080x32 = 66.3MB

Pooling Layer

- Down-samplers of images
- Reduces compute in subsequent layers
- May use MAX or AVERAGE
- Compute:
  - Low amount of compute
  - Potentially replaceable with larger strides in previous convolution

Max pool with 2x2 filters and stride of 2:



$$n00 = \mathrm{MAX}(i00, i01, i10, i11)$$

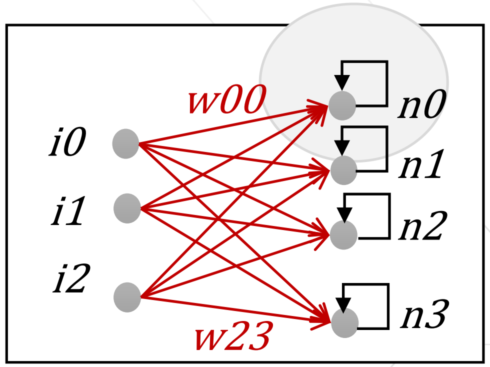
Or

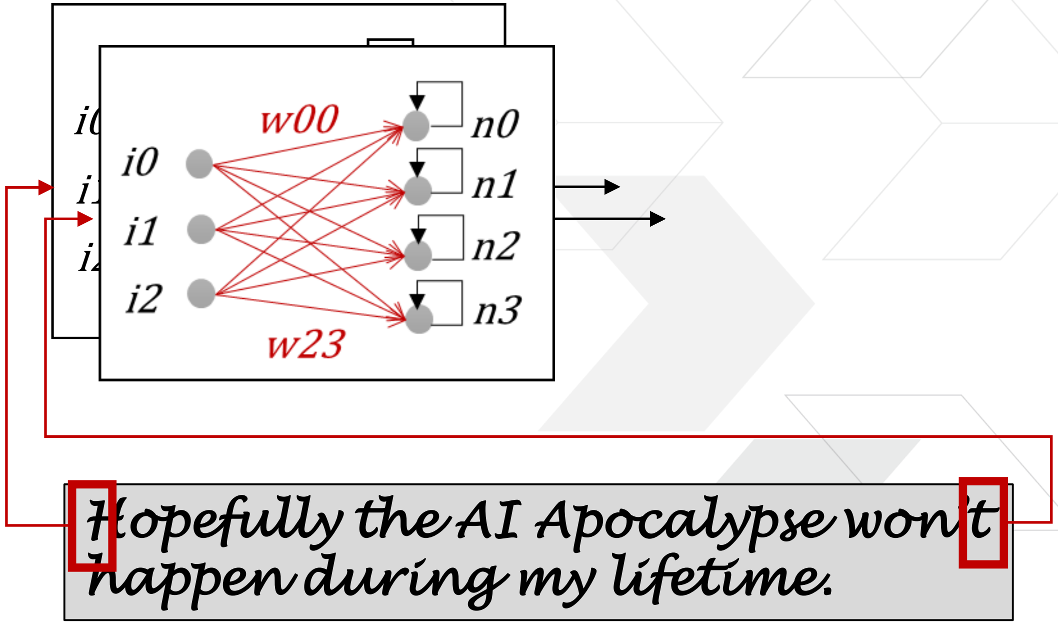
$$n00 = \mathrm{AVG}(i00, i01, i10, i11)$$

*Springenberg, J. T., Dosovitskiy, A., Brox, T., & Riedmiller, M. (2014). Striving for Simplicity: The all convolutional net.

Recurrent Layer Types

- **Contain state for processing sequences**
  - For example needed in speech or optical character recognition
  - “Apocal???”
- **Uni-directional or bi-directional**
  - “I ???? You”
- **More sophisticated types to address the vanishing gradients problem for learning more than 5-10 timesteps**
  - GRU (gated recurrent unit)
  - LSTM (long short term memory)

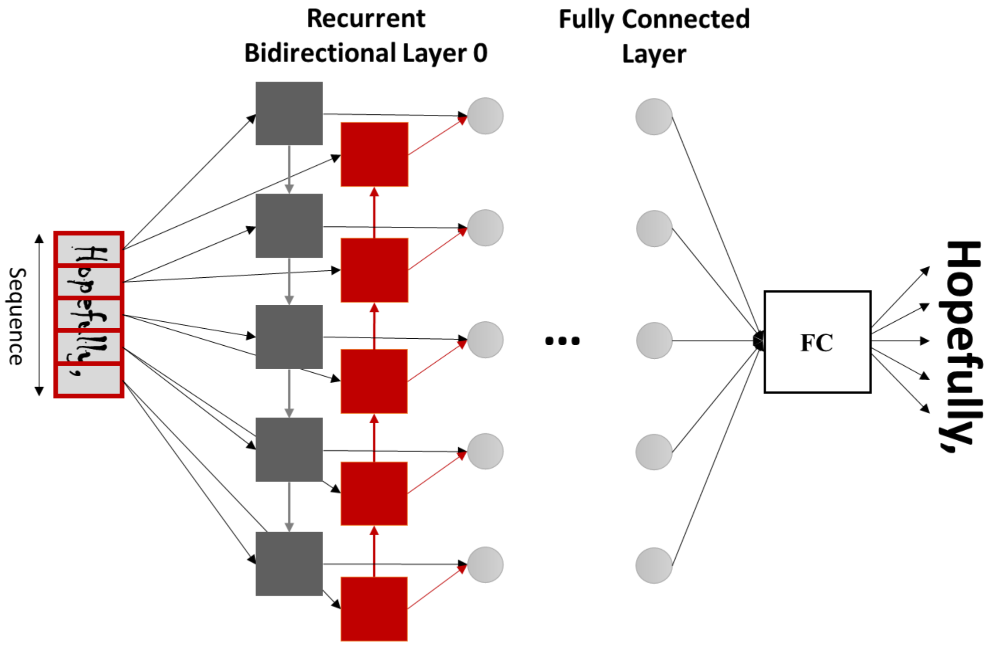




Recurrent Layers

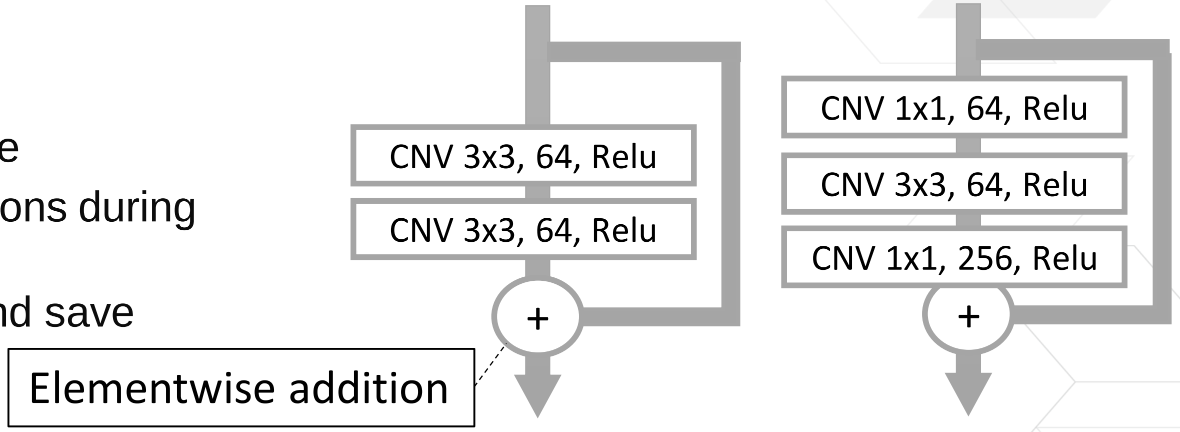
Challenges in Additional Data Dependencies

- **Input sequence**
  - Unlike batch, additional data dependencies between inputs of the same sequence and state
- **Bi-directional NNs**
  - Full sequence needs to be completed before the next layer.
  - Fewer opportunities for parallelism
  - Operations generally stress memory bandwidth (compared to CONV)

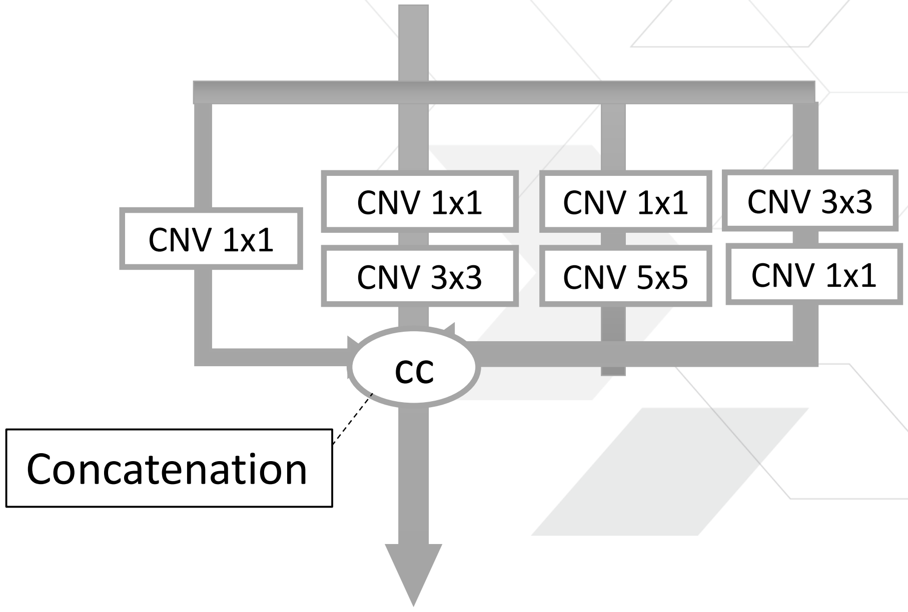


Meta-Layers

> **Residual layers (ResNets *)**
>> Introduced to make larger networks more trainable
>> Better gradient propagation through skip connections during training
>> Plus 1x1 convolutions to reduce dimensionality and save compute



> **Inception layers (GoogleNet**)**
>> Huge variation in spatial features => combining different size convolutions in one layer
>> Plus 1x1 convolutions to reduce dimensionality and save compute
>> Later on additional factorization to reduce compute
– 3x3 = 1x3 and 3x1



> **Many more…**

> **Implementation: support for non-linear topologies!**

*He, K., Zhang, X., Ren, S. and Sun, J. "Deep residual learning for image recognition." CVPR' 2016.
** Szegedy, C., Vanhoucke, V., Ioffe, S., Shlens, J. and Wojna, Z. "Rethinking the inception architecture for computer vision." CVPR' 2016.

Computation & Memory Requirements

XILINX

Compute and Memory Requirements
Architecture Neutral, Per Layer

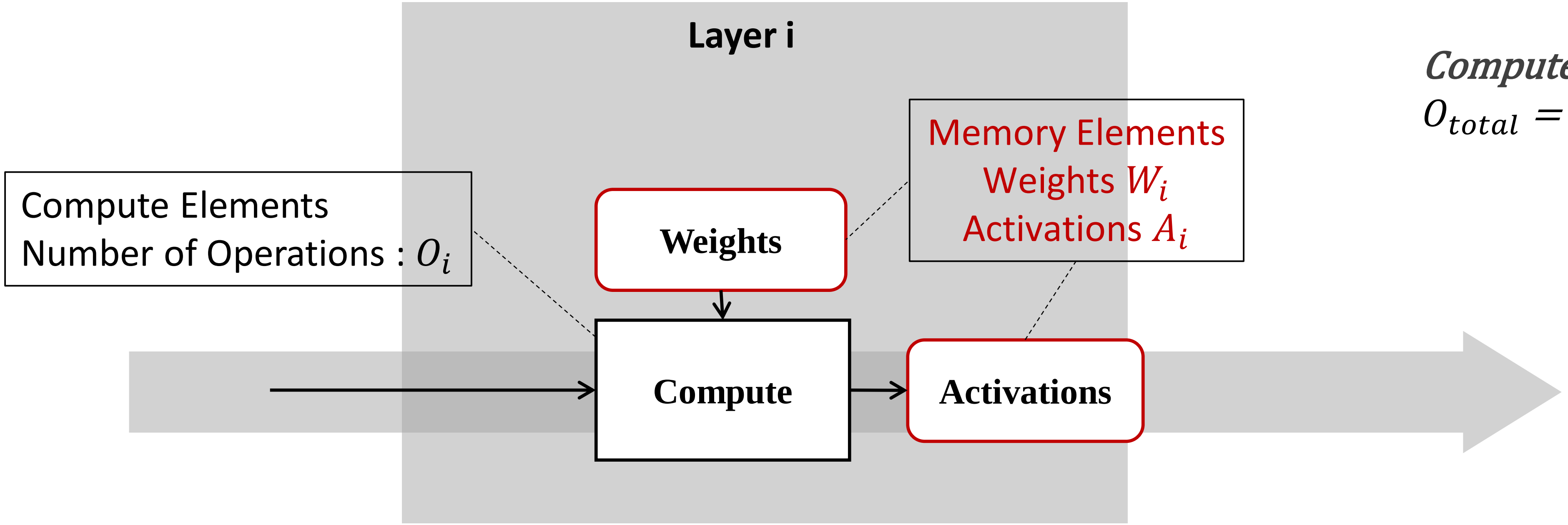
Memory Requirements:

$A_{total} = \sum A_i$

$W_{total} = \sum W_i$

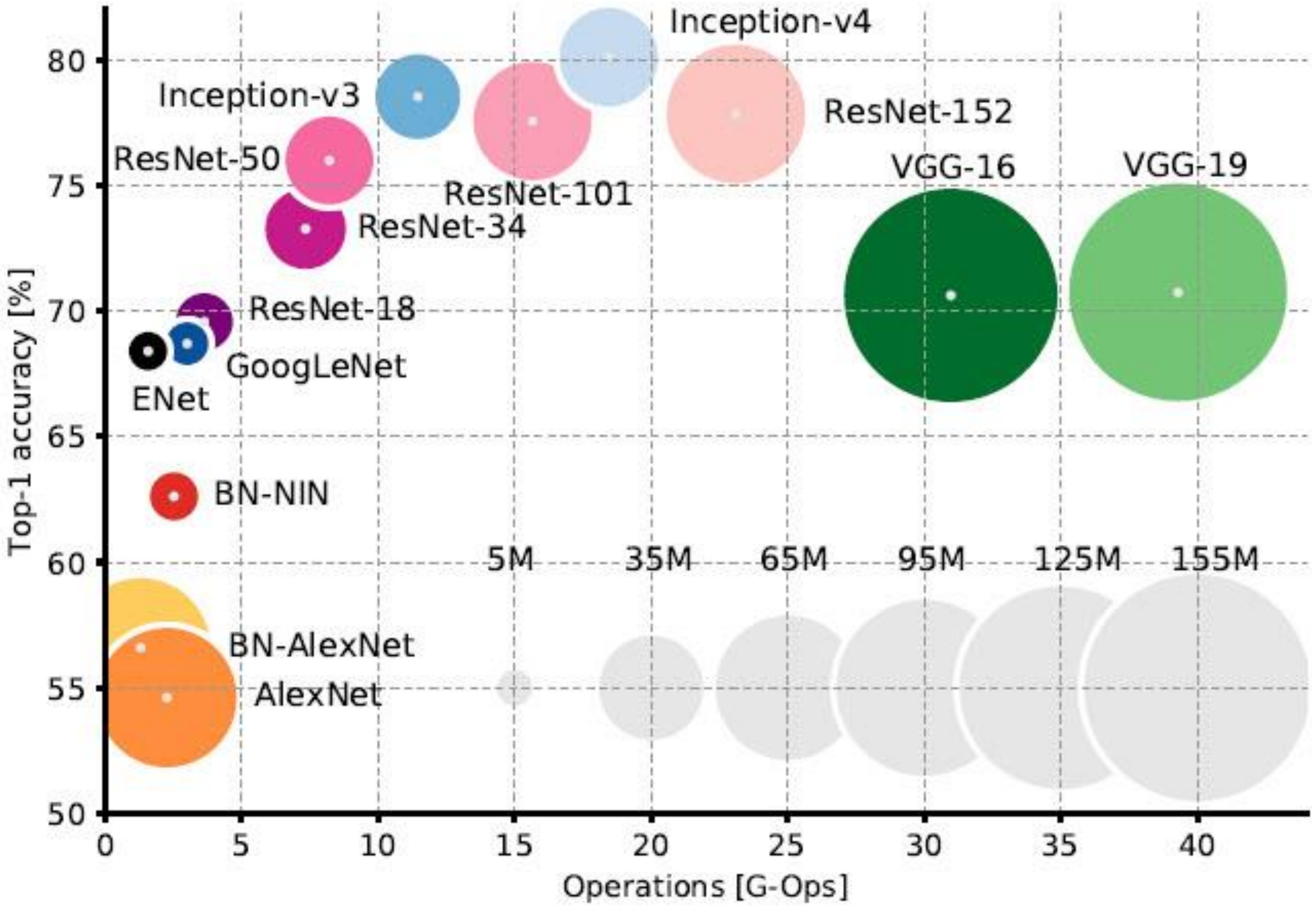
Compute Requirements:

$O_{total} = \sum O_i$



IN, IN_CH:	*number of inputs and input channels*
OUT, OUT_CH:	*number of outputs and output channels*
F_DIM, FM_DIM:	*filter and feature map dimensions (assumed square)*
BATCH:	*batch size*
BITS:	*bit precision in data types*
GATES:	*number of gates in RNNs:*
STATES:	*worst case*
SEQ:	*sequence length*
HID:	*hidden size (state + output from each state)*
DIRS:	*1 for unidirectional and 2 for bidirectional RNN*

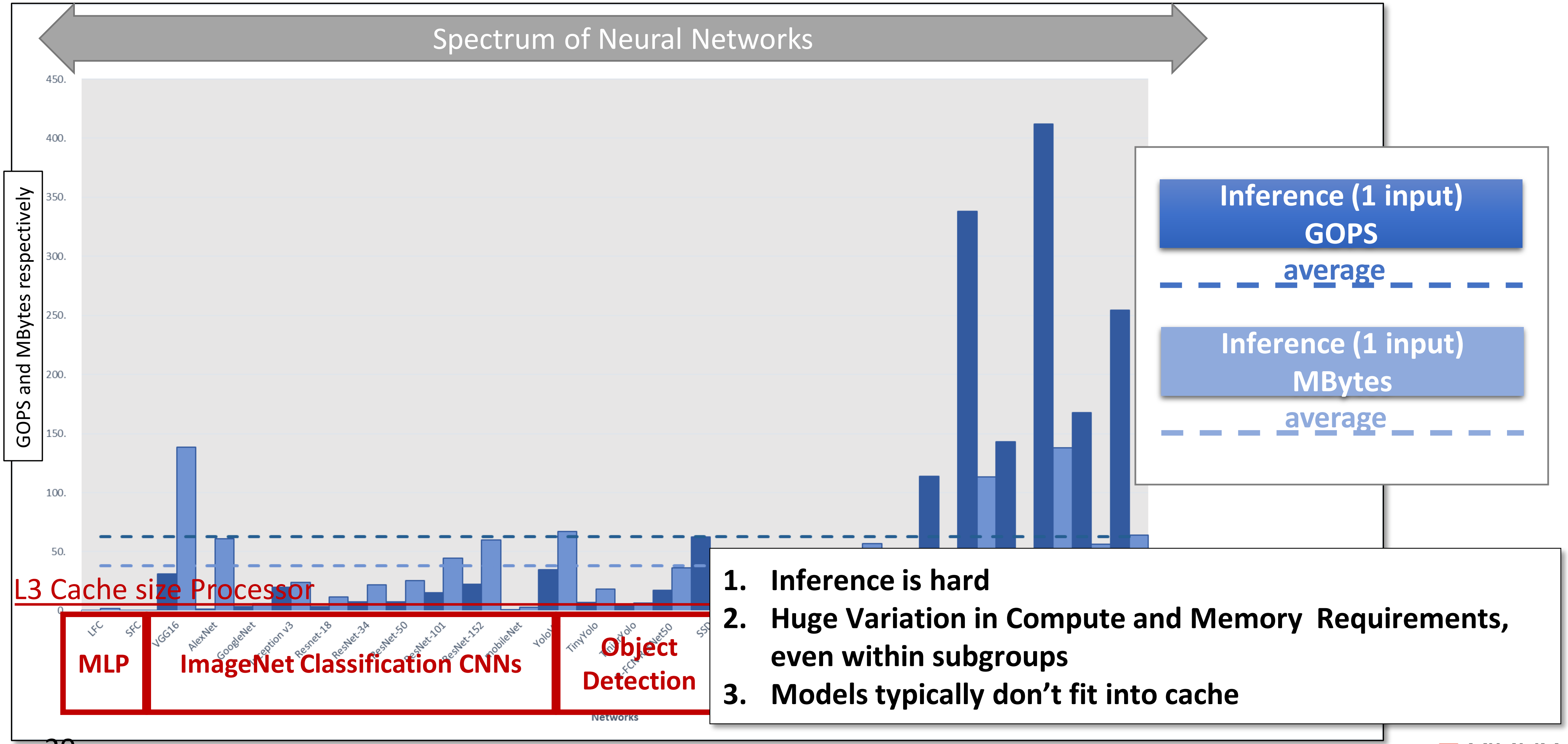
Network Complexity by Operations & Weights



An Analysis of Deep Neural Network Models for Practical Applications, Canziani et al.

Inference Compute and Memory
Across a Spectrum of Neural Networks

*architecture independent
**1 image forward
*** batch = 1
**** int8

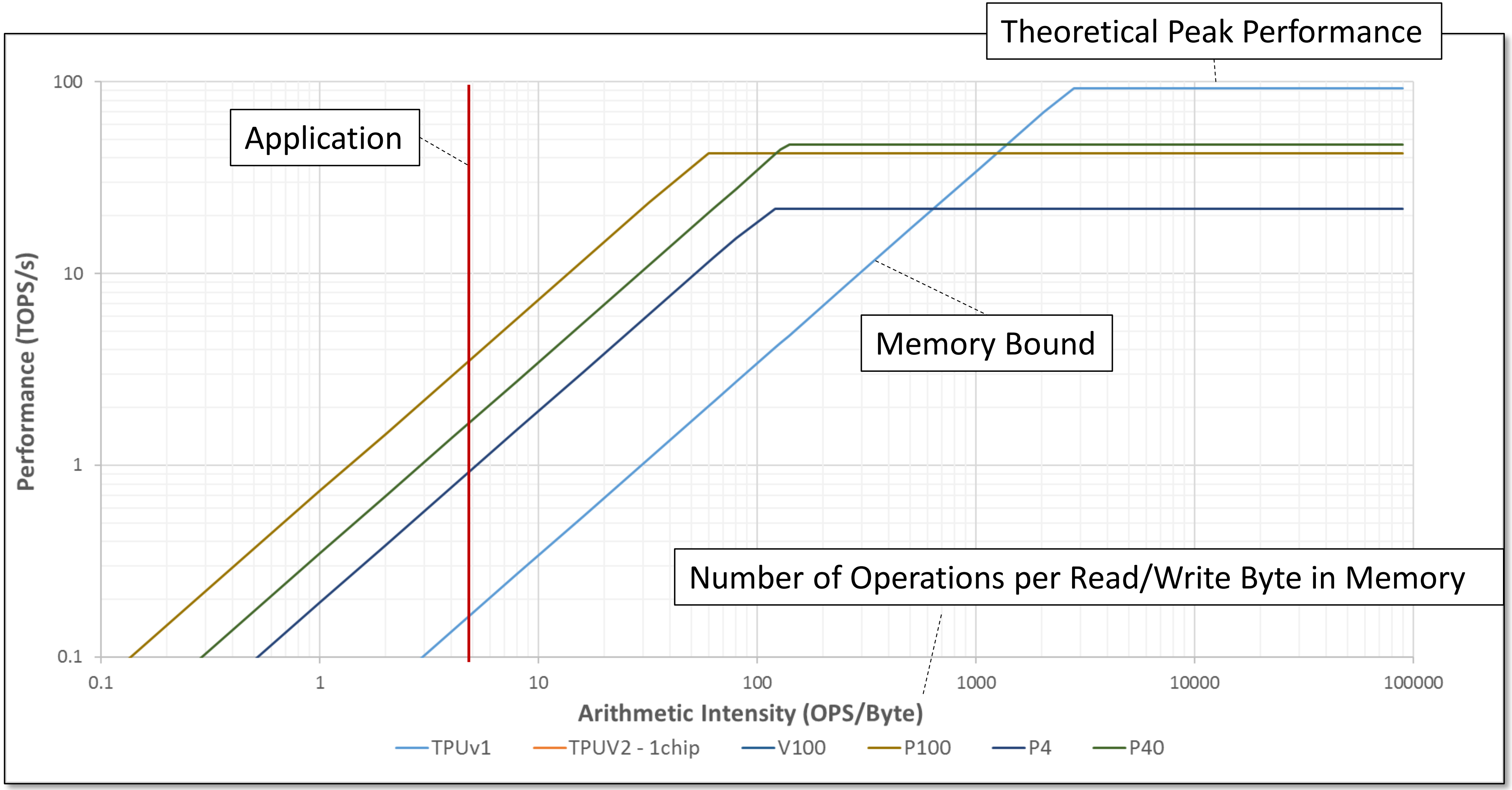


1. **Inference is hard**
2. **Huge Variation in Compute and Memory Requirements, even within subgroups**
3. **Models typically don't fit into cache**

XILINX

Rooflines*

log axes



*Williams, S., Waterman, A. and Patterson, D., 2009.
Roofline: an insightful visual performance model for multicore architectures. Communications of the ACM

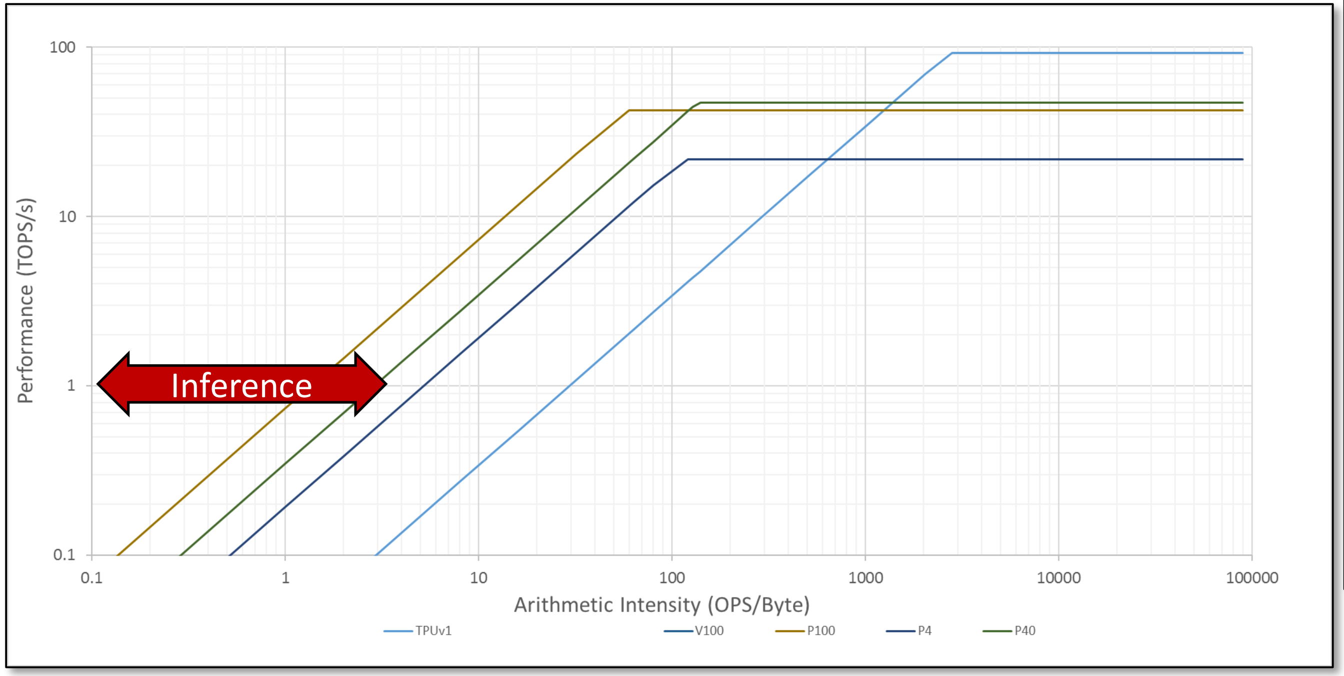
Arithmetic Intensity
Across a Spectrum of Neural Networks

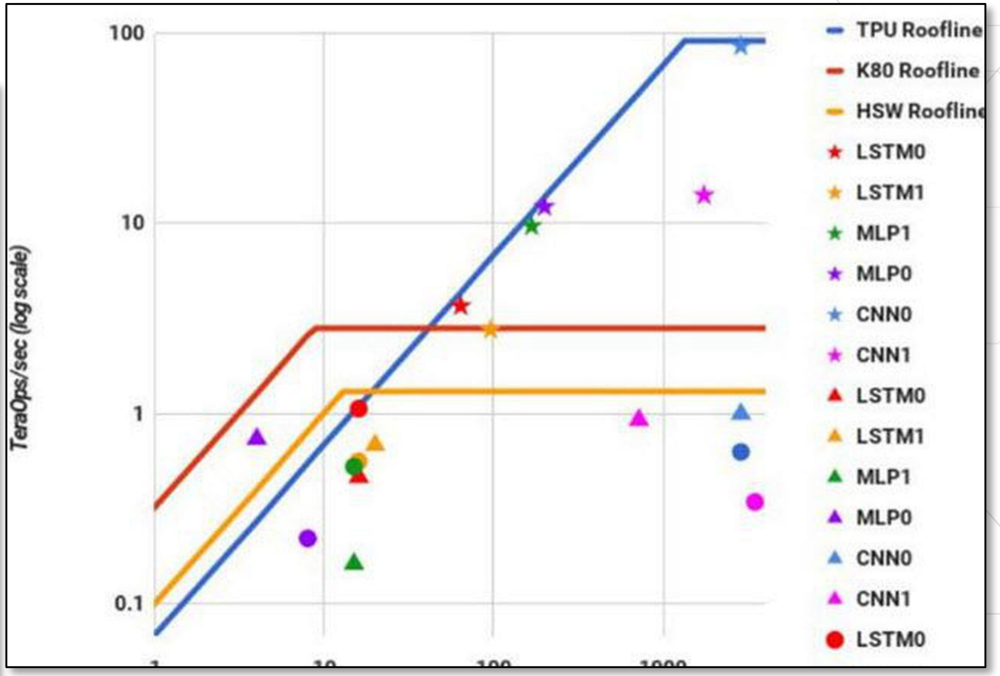
* batch = 1
** with respect to weights assuming weights are off-chip

> **Memory requirement for weights, activations are beyond typically available on-chip memory**

> **This yields low arithmetic intensity**

>> For example for inference, assuming weights off-chip and naïve implementation, majority of networks is below 6OPS:Byte





Jouppi, N.P., Young, C., Patil, N., Patterson, D., Agrawal, G., Bajwa, R., Bates, S., Bhatia, S., Boden, N., Borchers, A. and Boyle, R., 2017, June. In-datacenter performance analysis of a tensor processing unit. ISCA'2017

XILINX

In Summary: CNNs are associated with…

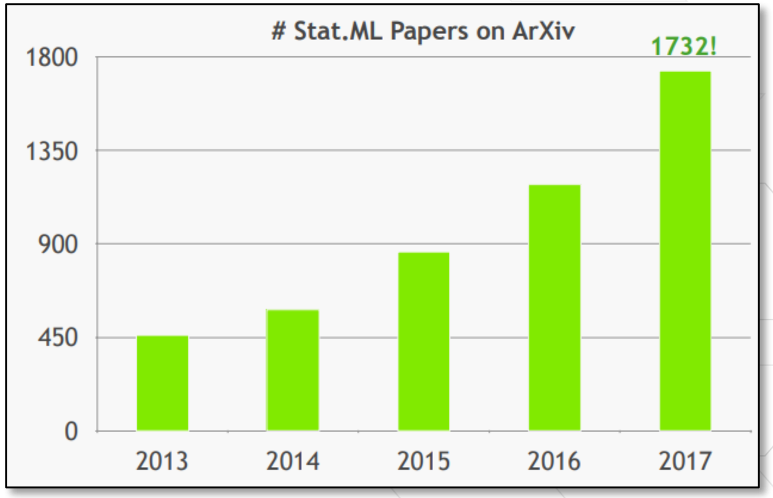
> Significant amounts of memory and computation

> Huge variation between topologies and within them

> Fast changing algorithms

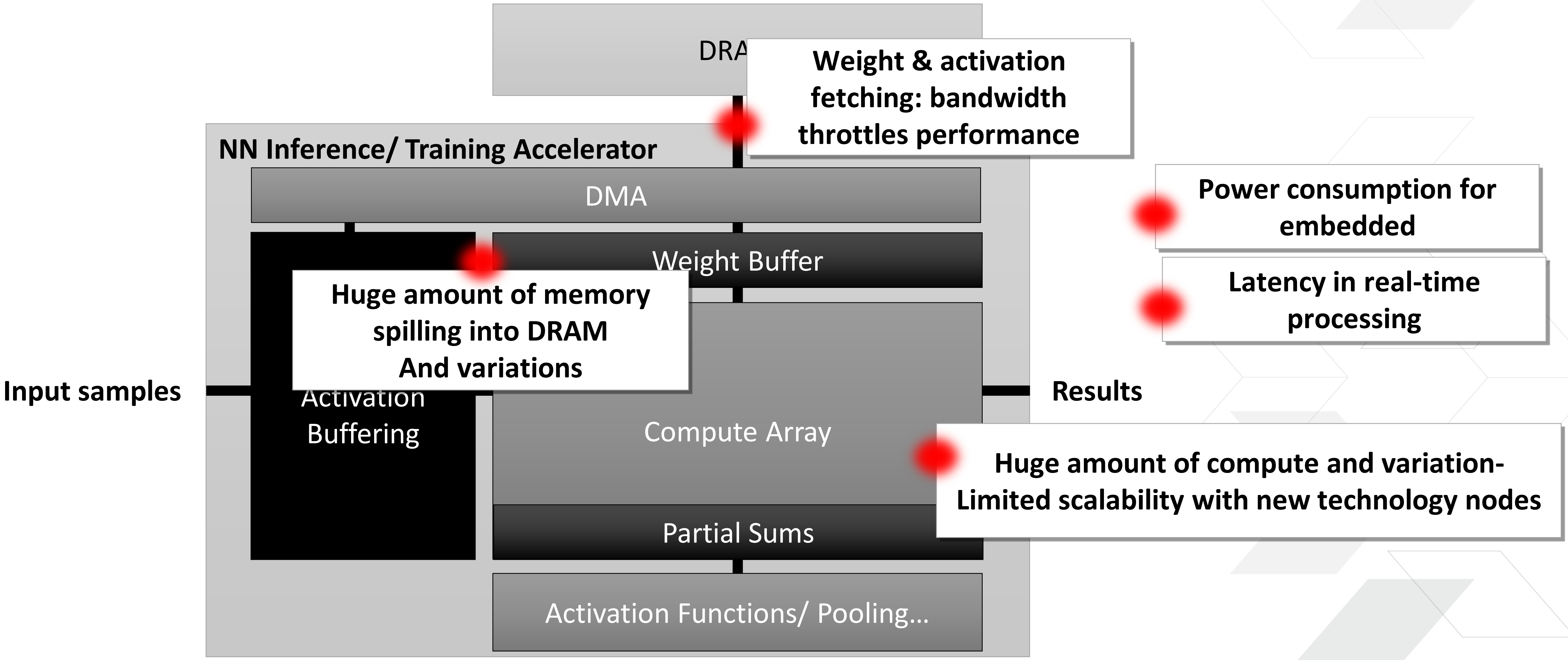
> Special functions, non-linear topologies

> However, incredibly parallel!
>> For convolutions: filter dimensions, feature map dimensions, input & output channels, batches, layers, and even precisions (discussed later)



Adopted from Ce Zhang, ETH Zurich, Systems Group Retreat

Architectural Challenges/ Pain Points



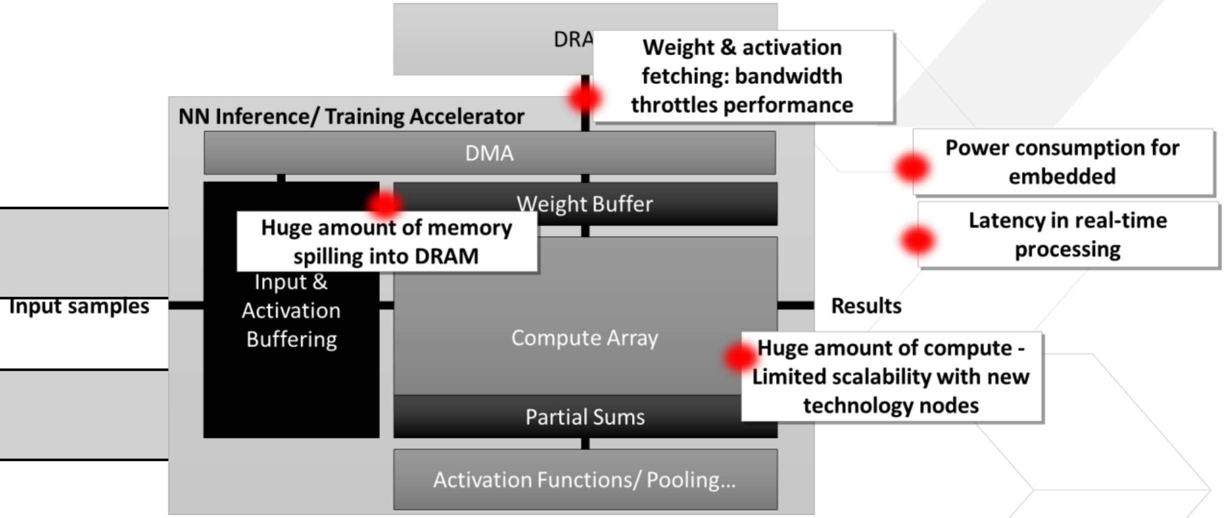
Requires algorithmic & architectural innovation

Algorithmic Optimization Techniques

XILINX.

Optimization Techniques

- Loop transformations to minimize memory access*
- Pruning
- Compression
- Winograd, Strassen and FFT
- Novel layer types (squeeze, shuffle, shift)
- Numerical Representations & Reducing Precision

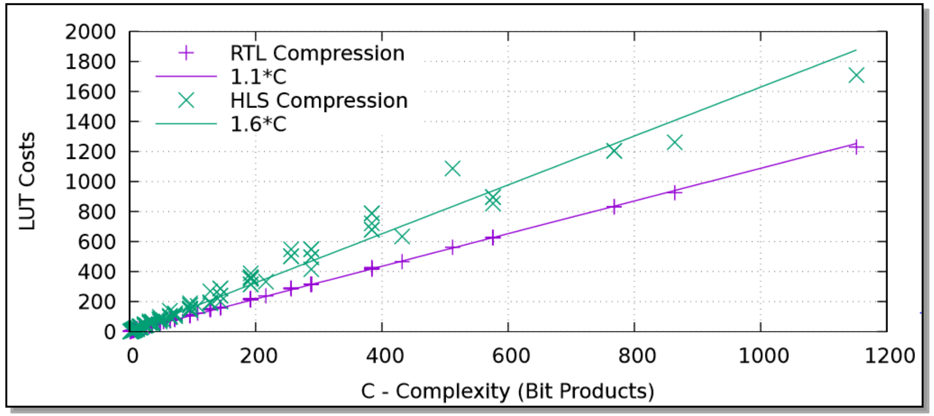


*Chen, Y.H., Krishna, T., Emer, J.S. and Sze, V., 2017. Eyeriss: An energy-efficient reconfigurable accelerator for deep convolutional neural networks. IEEE Journal of Solid-State Circuits, 52(1), pp.127-13

Example: Reducing Bit-Precision

- **Linear reduction in memory footprint**
  - Reduces weight fetching memory bandwidth
  - NN model may even stay on-chip
- **Reducing precision shrinks inherent arithmetic cost in both ASICs and FPGAs**
  - Instantiate **100x** more compute within the same fabric and thereby scale performance

Precision	Modelsize [MB] (ResNet50)
1b	3.2
8b	25.5
32b	102.5

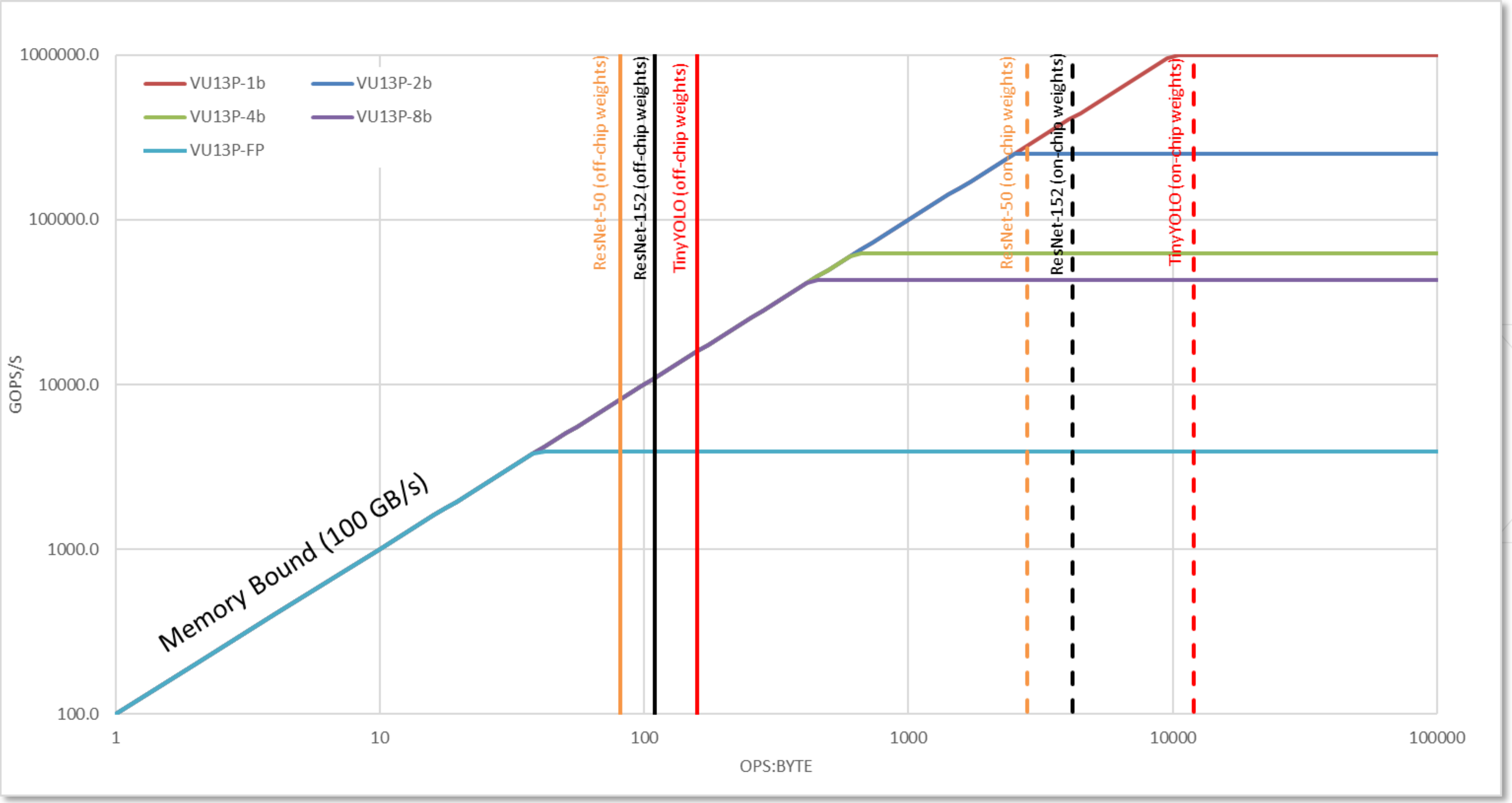


C= size of accumulator * size of weight * size of activation

(to appear in ACM TRETS SE on DL, FINN-R)

Reducing Precision provides Performance Scalability

Example: ResNet50, ResNet152 and TinyYolo



Theoretical Peak Performance for a VU13P with different Precision Operations
Assumptions: Application can fill device to 90% (fully parallelizable) 710MHz

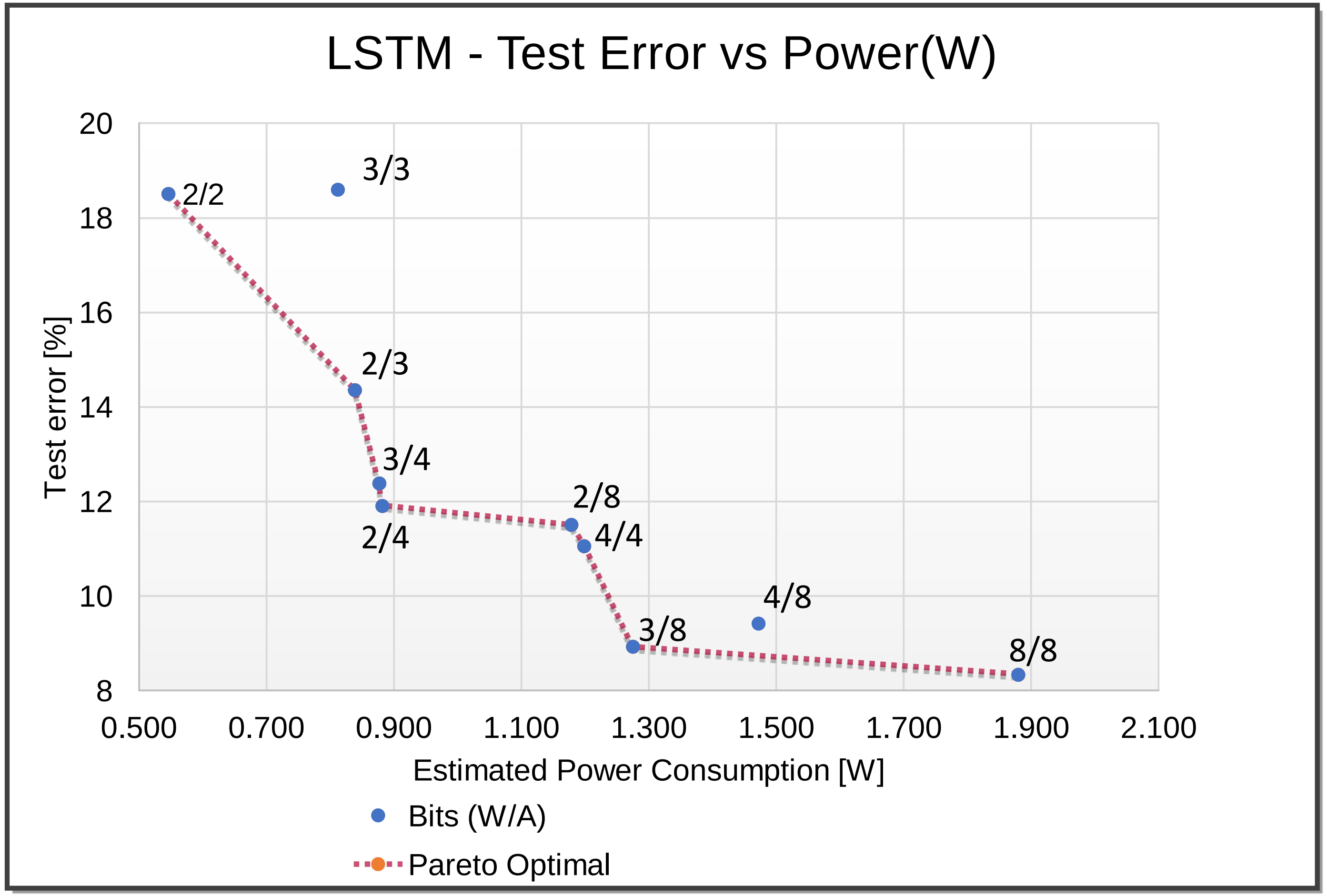
RP scales compute performance

RP reduces model size=> to stay on-chip

XILINX

Reducing Precision Inherently Saves Power

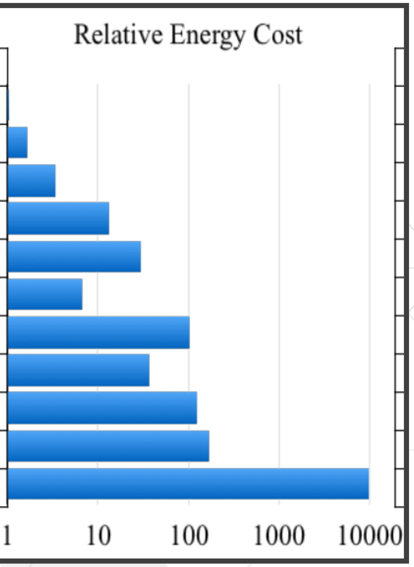
FPGA:



Target Device ZU7EV ● Ambient temperature: 25 °C ● 12.5% of toggle rate ● 0.5 of Static Probability ● Power reported for PL accelerated block only

ASIC:

Operation:	Energy (pJ)
8b Add	0.03
16b Add	0.05
32b Add	0.1
16b FP Add	0.4
32b FP Add	0.9
8b Mult	0.2
32b Mult	3.1
16b FP Mult	1.1
32b FP Mult	3.7
32b SRAM Read (8KB)	5
32b DRAM Read	640



Source: Bill Dally (Stanford), Cadence Embedded Neural Network Summit, February 1, 2017

Rybalkin, V., Pappalardo, A., Ghaffar, M.M., Gambardella, G., Wehn, N. and Blott, M. "FINN-L: Library Extensions and Design Trade-off Analysis for Variable Precision LSTM Networks on FPGAs."

Hardware Architectures and their Specialization Towards CNN Workloads

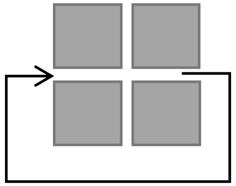
Exciting Times in Computer Architecture Research!

XILINX

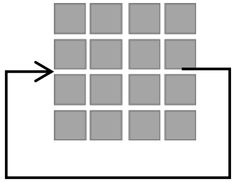
Spectrum of New Architectures for Deep Learning

DPU: Deep Learning Processing Unit

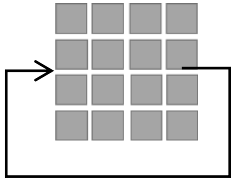
CPUs	GPUs	Soft DPUs (FPGA)	Hard DPUs (ASIC)	*In-Memory Compute*



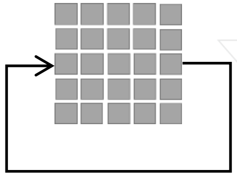
Intel
AMD
ARM



AMD
NVIDIA



DeePhi
Teradeep
XDNN



TPU, Cerebras, Graphcore, Groq, Nervana, Wave Computing, Eyeriss, Movidius, Kalray

Using non-volatile resistive memories or stacked DRAM*

ISAAC, Tetris, Neurcube

Vector-based SIMD processors
becoming increasingly customized for Deep Learning
(Tensor Cores, Reduced Precision,…)

**Shafiee, A., Nag, A., Muralimanohar, N., Balasubramonian, R., Strachan, J.P., Hu, M., Williams, R.S. and Srikumar, V., 2016. ISAAC: A convolutional neural network accelerator with in-situ analog arithmetic in crossbars. ACM SIGARCH*

Chi, P., Li, S., Xu, C., Zhang, T., Zhao, J., Liu, Y., Wang, Y. and Xie, Y., 2016, June. Prime: A novel processing-in-memory architecture for neural network computation in reram-based main memory. In ACM SIGARCH

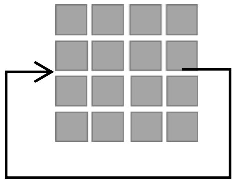
Chen, Y., Luo, T., Liu, S., Zhang, S., He, L., Wang, J., Li, L., Chen, T., Xu, Z., Sun, N. and Temam, O., 2014, December. Dadiannao: A machine-learning supercomputer. In Proceedings of the 47th Annual IEEE/ACM International Symposium on Microarchitecture (pp. 609-622). IEEE Computer Society.

Architectural Choices – Macro-Architecture

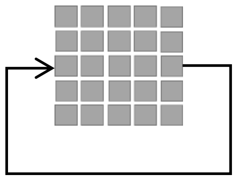
Soft DPUs (FPGA)

Hard DPUs (ASIC)

Matrix of PE

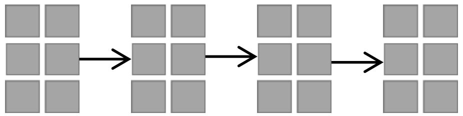


DeePhi
Teradeep
XDNN

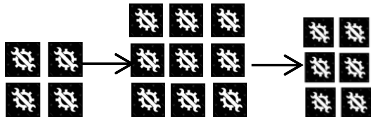


TPU, Cerebras, Graphcore, Groq, Nervana, Wave Computing, Eyeriss, Movidius, Kalray

Customized macro-architecture (Synchronous Dataflow)



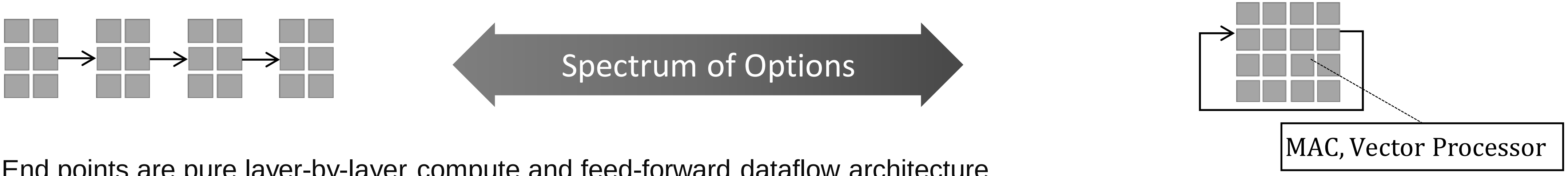
MSR Brainwave*



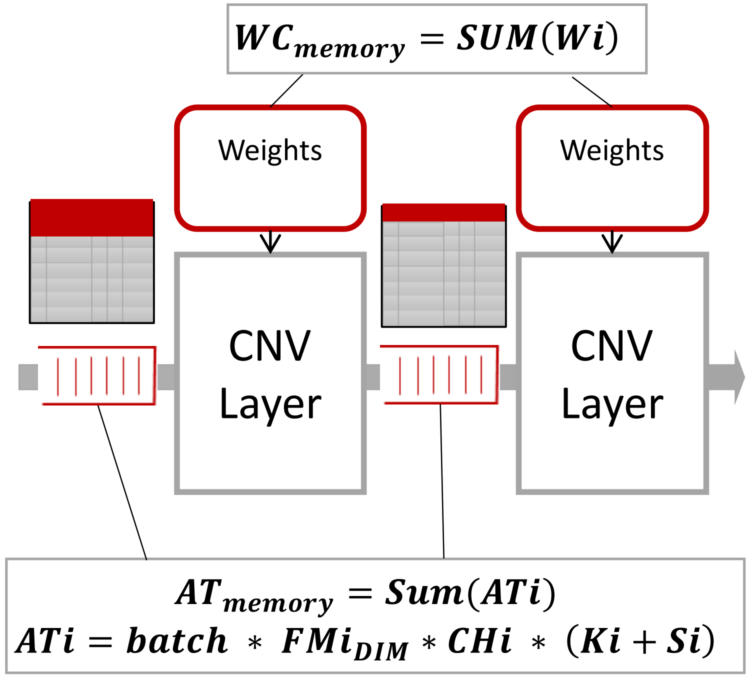
FINN**

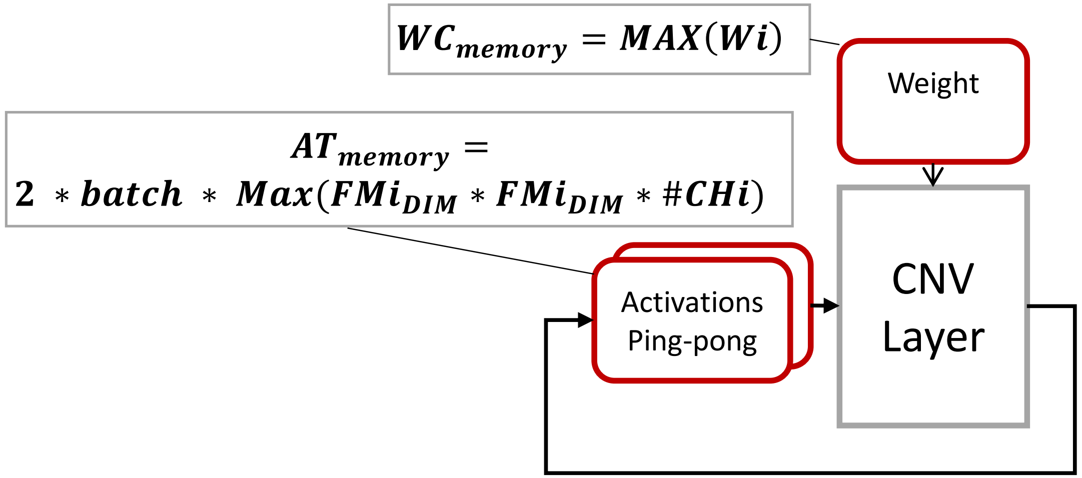
**Chung, E., Fowers, J., Ovtcharov, K., Papamichael, M., Caulfield, A., Massengill, T., Liu, M., Lo, D., Alkalay, S., Haselman, M. and Abeydeera, M.Serving DNNs in Real Time at Datacenter Scale with Project Brainwave. IEEE Micro, 38(2)*
https://www.microsoft.com/en-us/research/uploads/prod/2018/06/ISCA18-Brainwave-CameraReady.pdf
***Umuroglu, Yaman, Umuroglu, Y., Fraser, N.J., Gambardella, G., Blott, M., Leong, P., Jahre, M. and Vissers, K. "FINN: A framework for fast, scalable binarized neural network inference." ISFPGA'2017*

Synchronous Dataflow (SDF) vs Matrix of Processing Elements (MPE)



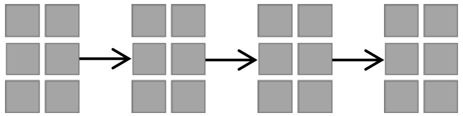
- End points are pure layer-by-layer compute and feed-forward dataflow architecture



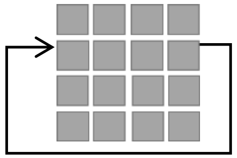


Lin, X., Yin, S., Tu, F., Liu, L., Li, X. and Wei, S. LCP: a layer clusters paralleling mapping method for accelerating inception and residual networks on FPGA. DAC'2016
Alwani, M., Chen, H., Ferdman, M. and Milder, P. Fused-layer CNN accelerators. MICRO 2016.

Synchronous Dataflow (SDF) vs Matrix of Processing Elements (MPE)



Degree of parallelization across layers



- Requires less activation buffering
- Higher compute and memory efficiency due to custom-tailored hardware design
- Less flexibility
- Less latency (reduced buffering)
- No control flow (static schedule)

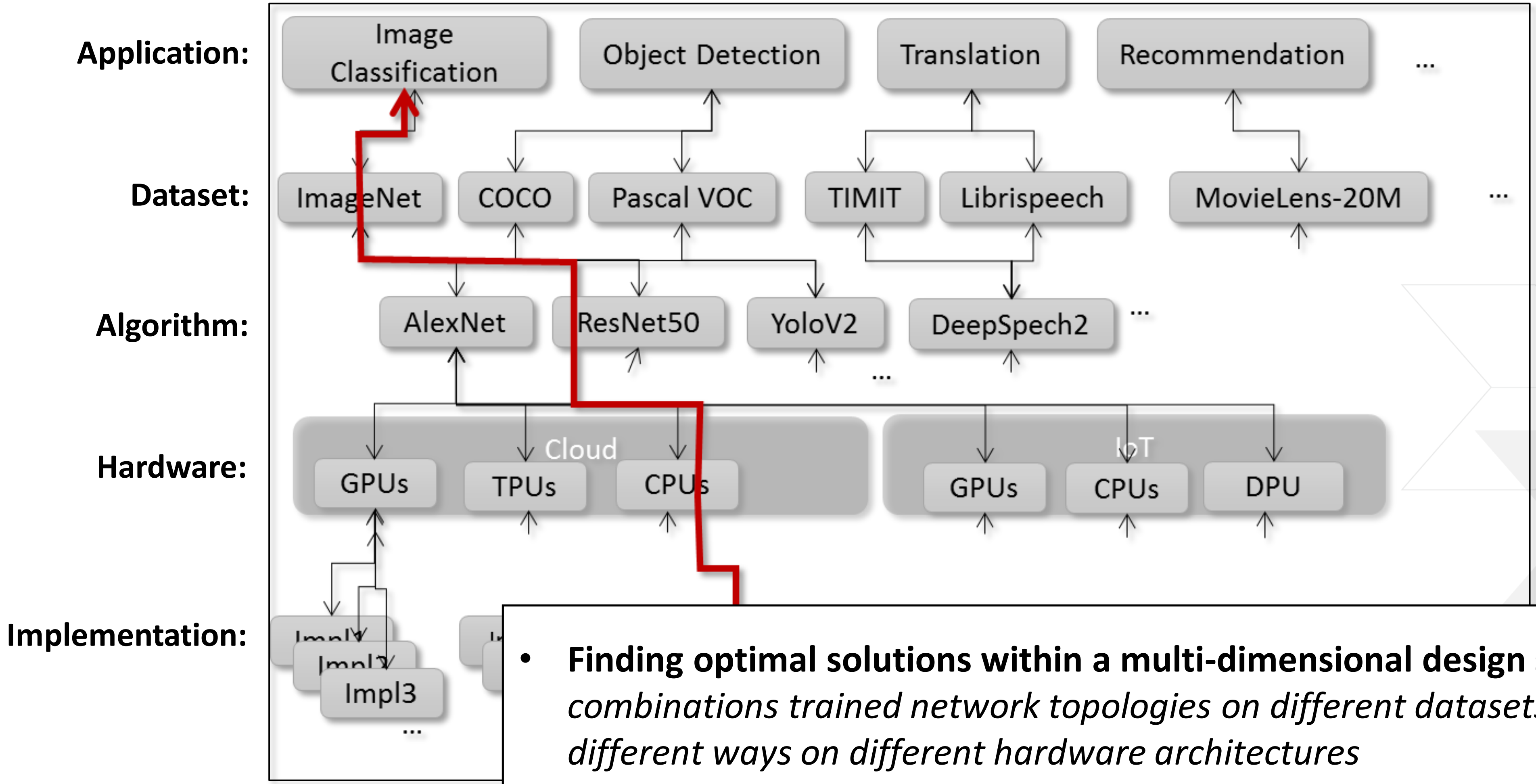
- Requires less on-chip weight memory, but more activation buffers
- Efficiency of memory for weights and activations depends on how well balanced the topology is
- Flexible hardware, which can scale to arbitrary large networks
- Compute efficiency is a scheduling problem
  => generating sophisticated scheduling algorithms

Summary

Summary

- Deep NNs are increasingly being adopted for new workloads and key to the current industrial revolution and perhaps the next
- Associated with significant challenges to increase performance
- Requires algorithmic and architectural innovation (co-designed)
- Emerging: Huge spectrum of algorithms and increasingly diverse & heterogenous hardware architectures
- Clear metrics for comparison needed
  - Hardware performance always tying back to application performance (accuracy) to allow for algorithmic optimizations
  - Always trading off accuracy, performance (op/sec), latency (ms), power consumption (W)

Exciting Times for our Community:
Many New Architectures Evolving - Programmable and Hardened



- **Finding optimal solutions within a multi-dimensional design space**
  combinations trained network topologies on different datasets implemented in different ways on different hardware architectures