AIDC

INTEL AI DEVCON 2018

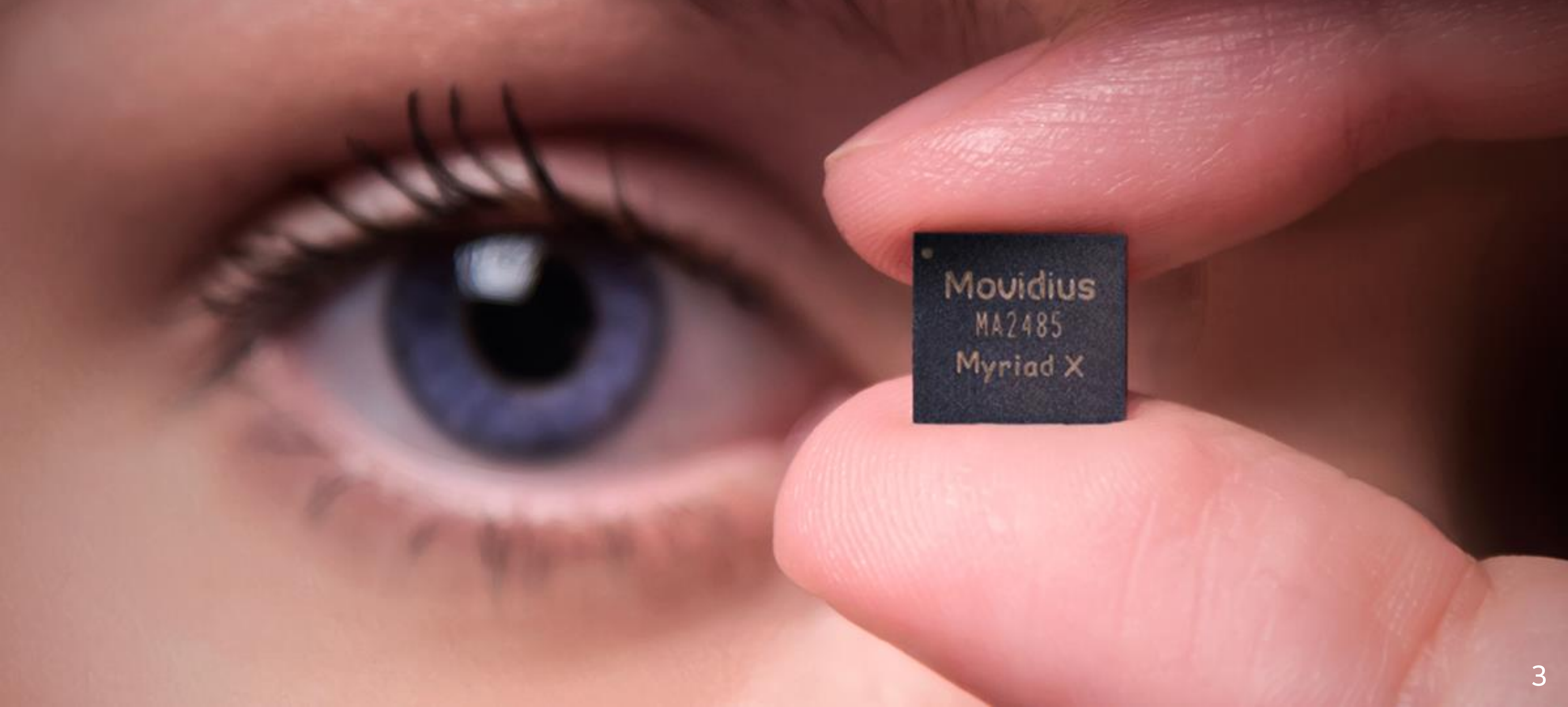
INTEL AI DEVCON 2018

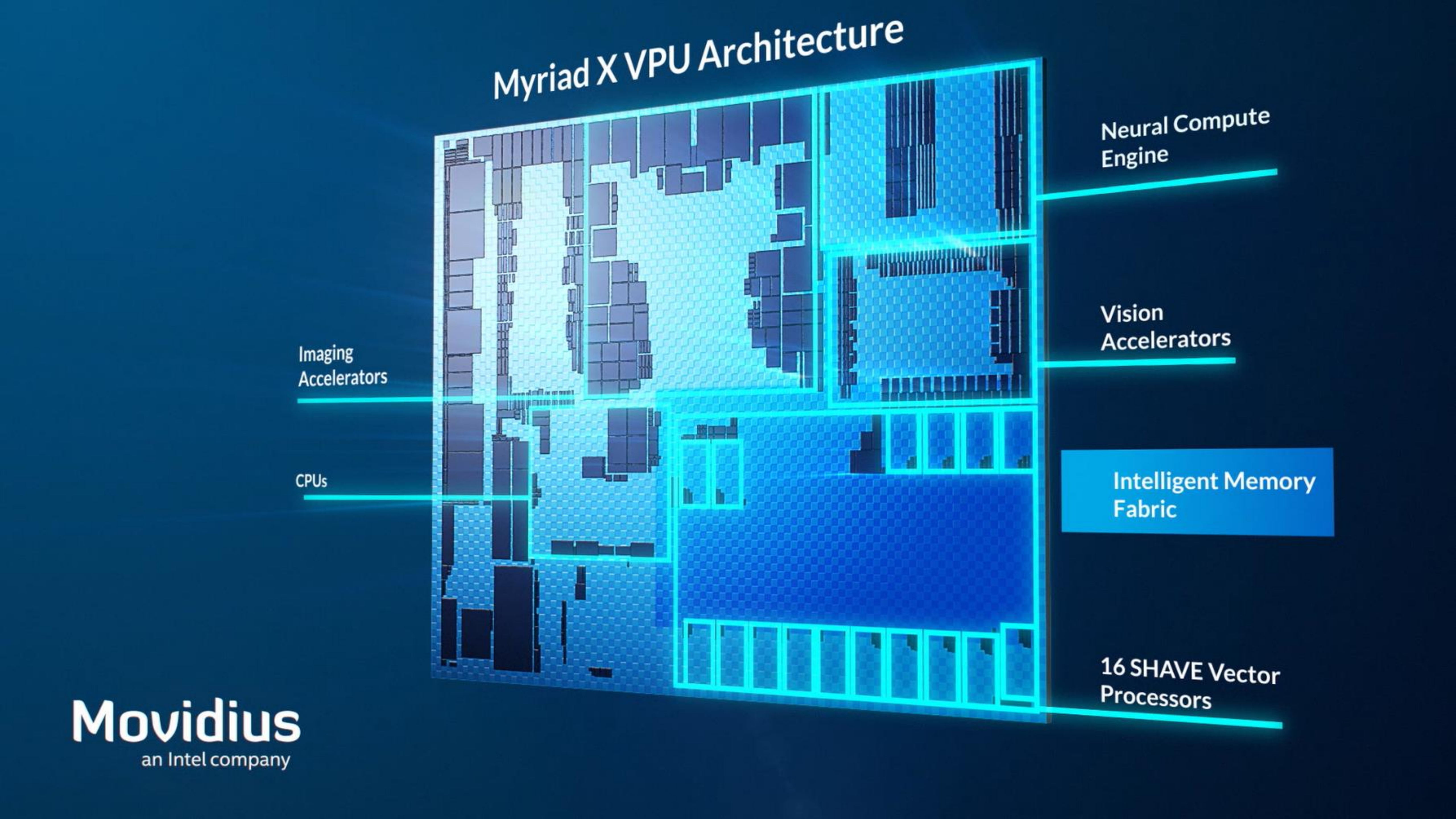
PORTABILITY AND PERFORMANCE IN EMBEDDED DEEP NEURAL NETWORKS: CAN WE HAVE BOTH?

Cormac Brick

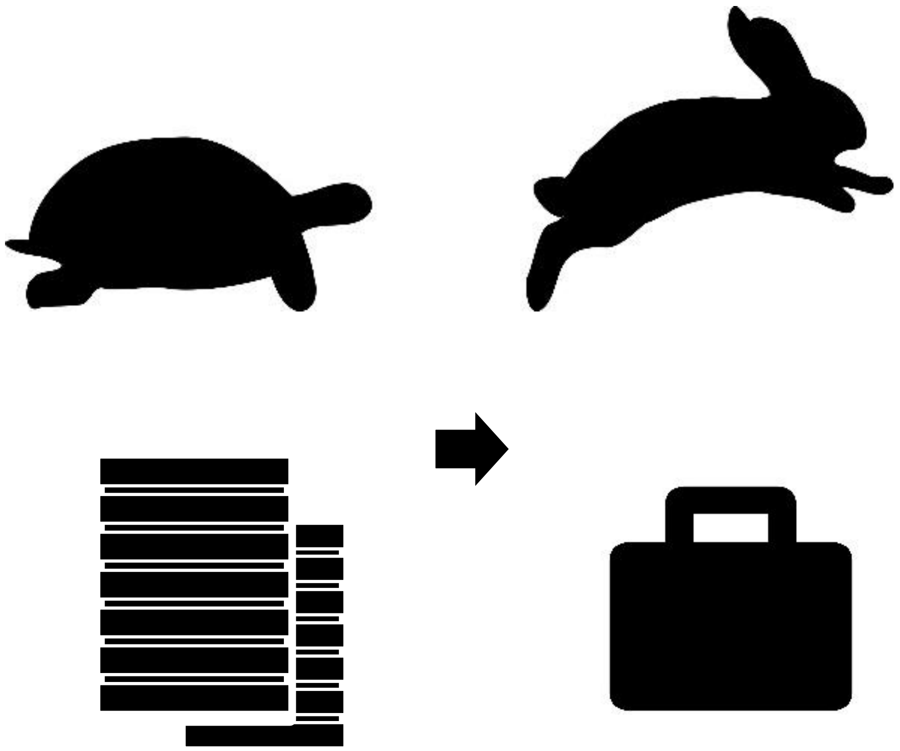
May 24th, 2018

INTEL® MOVIDIUS™ VPU TECHNOLOGY ENABLES POWER-EFFICIENT IMAGE PROCESSING, CV, AND DEEP LEARNING INFERENCE IN EDGE DEVICES



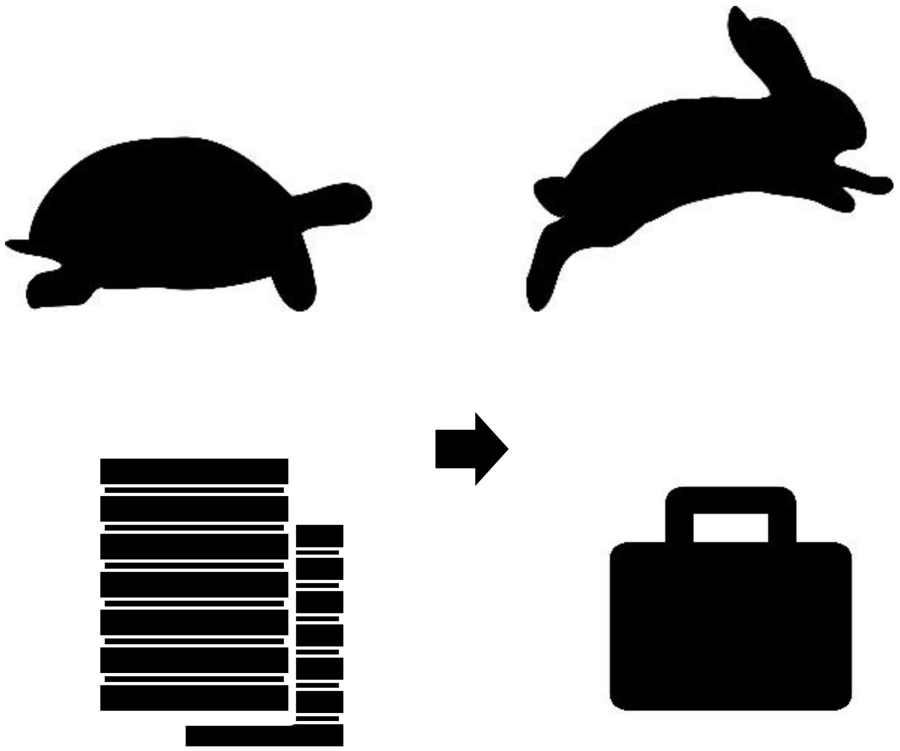

Myriad X VPU Architecture
Neural Compute Engine
Vision Accelerators
Intelligent Memory Fabric
16 SHAVE Vector Processors
Imaging Accelerators
CPUs
Movidius
an Intel company

THE QUESTION:



Can we have neural networks that are fast and portable?

THE QUESTION:



Can we have neural networks that are fast and portable without losing any accuracy?

OVERVIEW

- Selecting a fast network
- Making a fast network faster in a portable way
- Network portability: Ecosystems including ONNX

WHAT IS A FAST NETWORK?

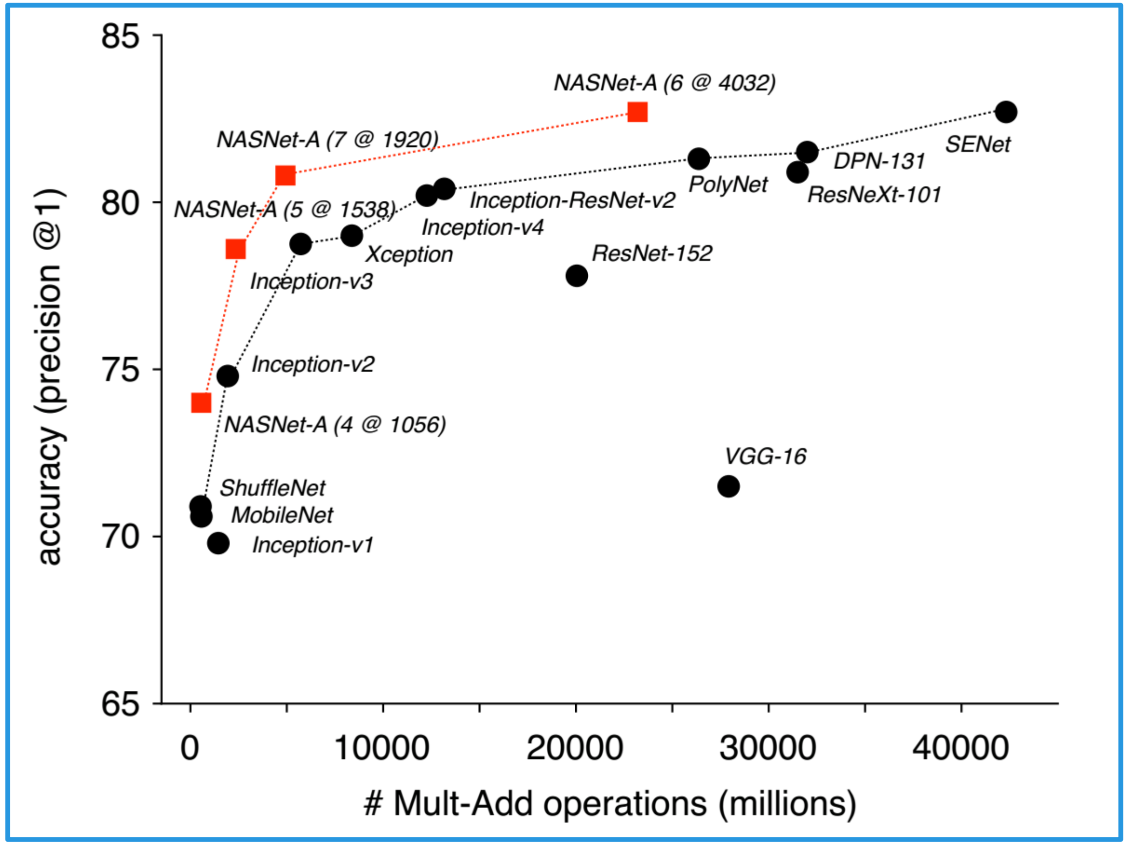
Classical approach, look at:

- Number of FLOPS
- Number of parameters

WHAT IS A FAST NETWORK?

Classical approach, look at:

1. Number of flops

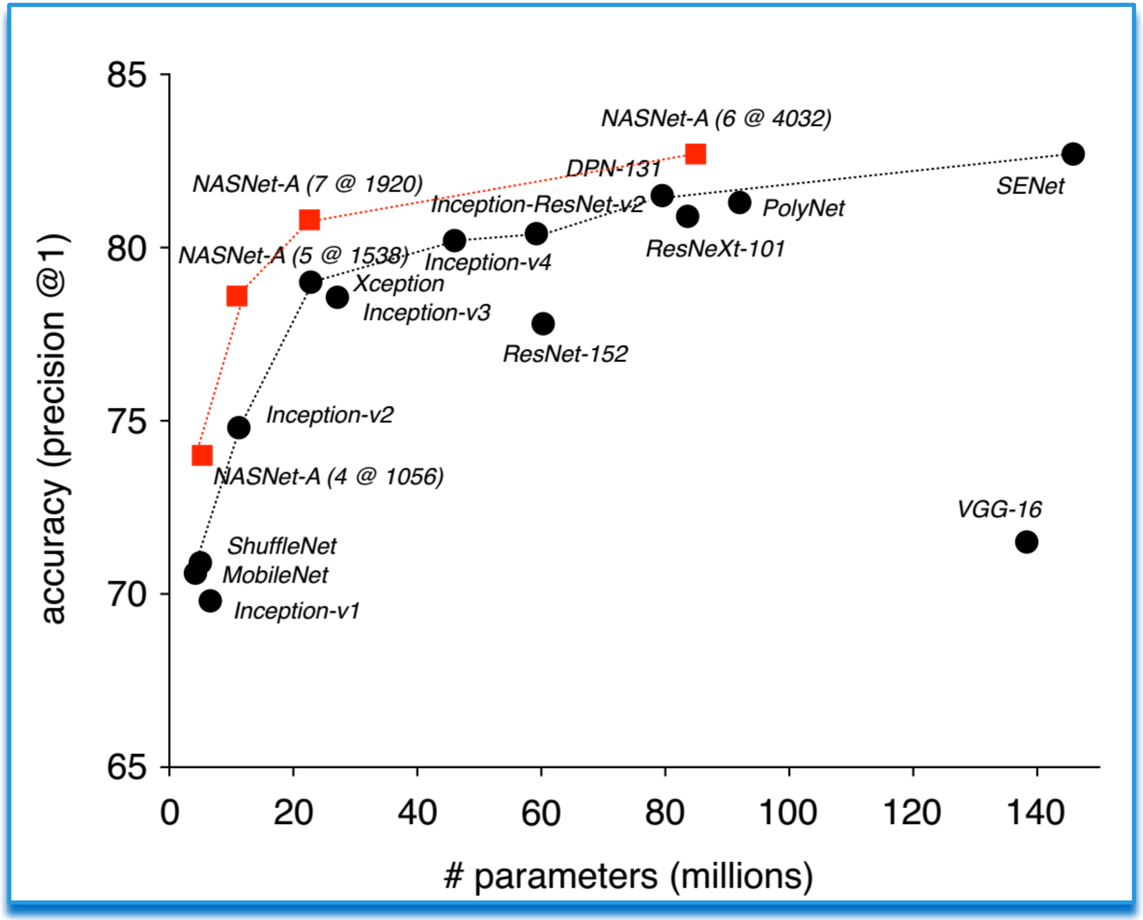


Ref[1]

WHAT IS A FAST NETWORK?

Classical approach, look at:

1. Number of FLOPS
2. Number of parameters



Ref[1]

WHAT IS A FAST NETWORK?

Classical approach, look at:

1. Number of FLOPS
2. Number of parameters

On Embedded platforms, dataflow is a key determinant of performance, so we should also consider:

3. Activation heap size

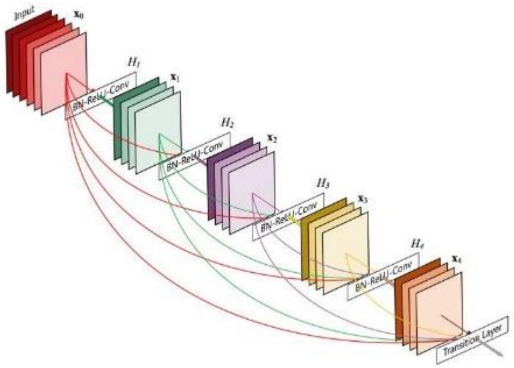
   → *Keep activations in local mem/cache*

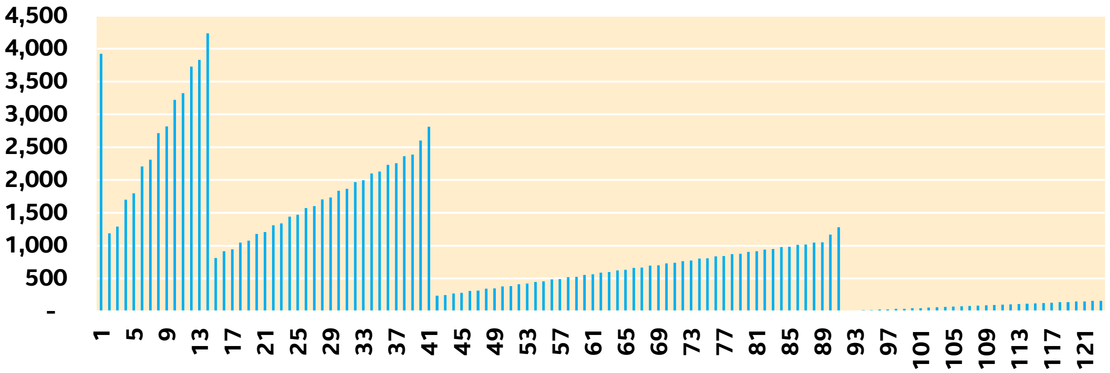
4. FLOPs/param/layer

   → *Avoid being DDR bound on weight fetch*

ACTIVATION HEAP SIZE

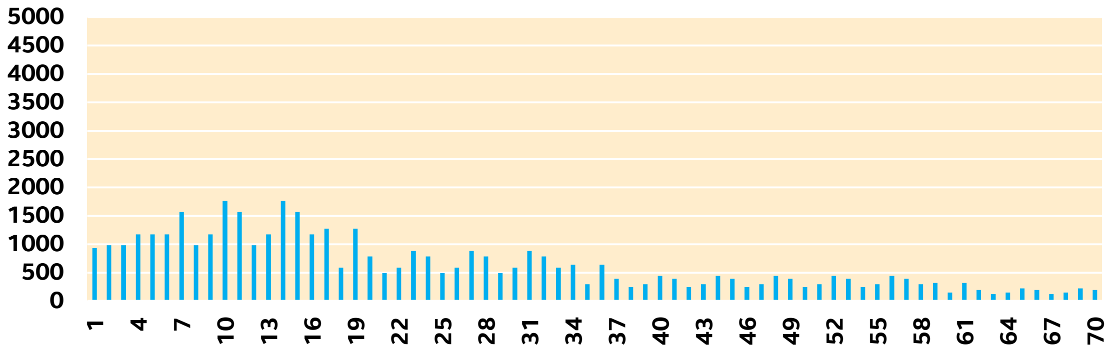
DenseNet





Long lifetime data – larger heap

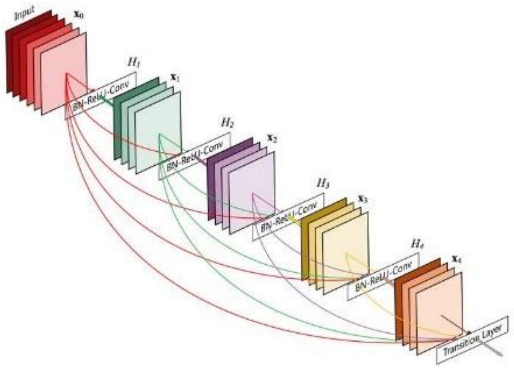
Resnet 50

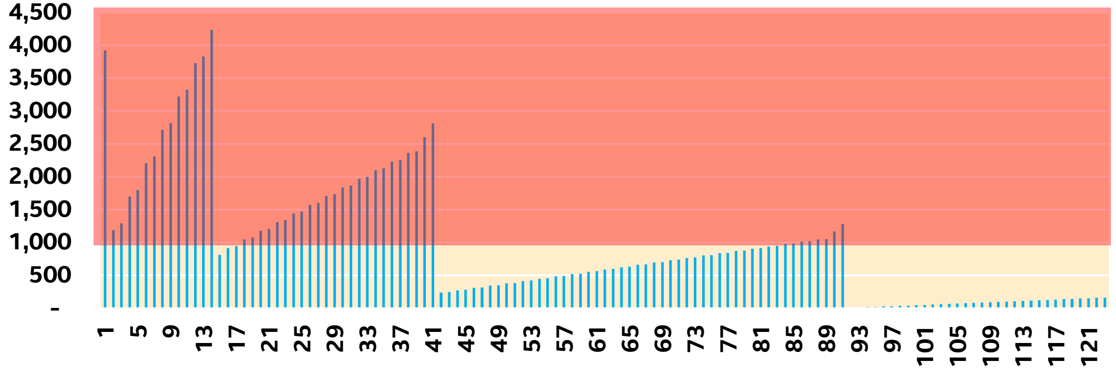


Limited lifetime data – smaller heap

ACTIVATION HEAP SIZE – WHAT IF ONLY 1MB L1 MEM?

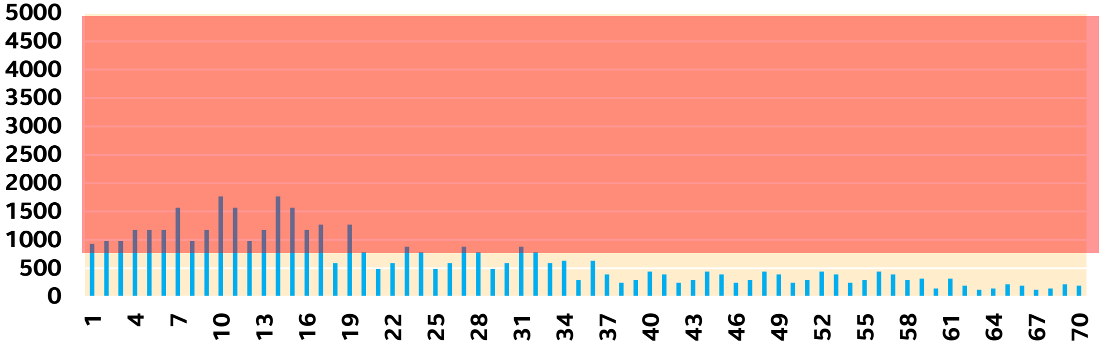
DenseNet





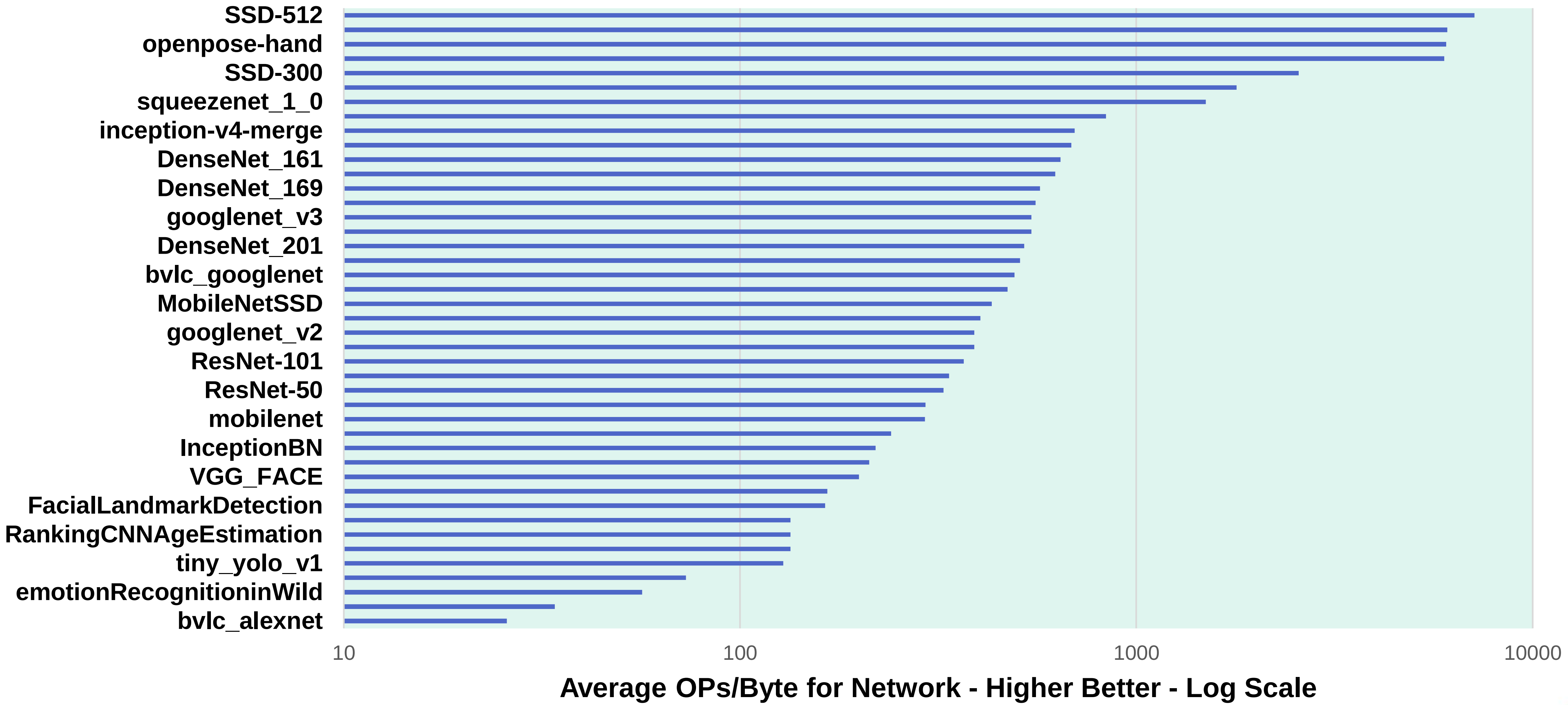
Long lifetime data – larger heap

Resnet 50



Limited lifetime data – smaller heap

AVERAGE OPS/BYTE ON COMMON VISION NETWORKS



AIDC
INTEL AI DEVCON 2018

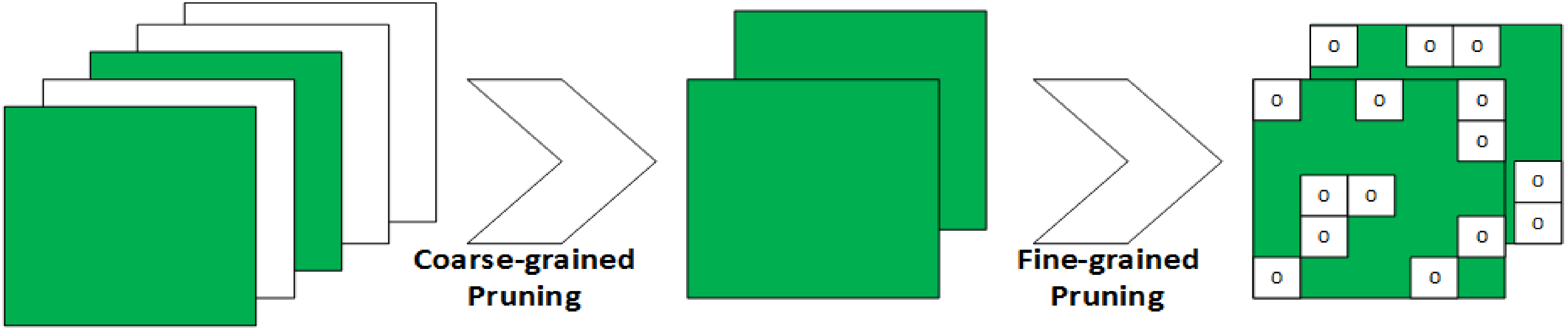
OVERVIEW

- Selecting a fast network
- Making a fast network faster in a portable way
- Network portability: Ecosystems including ONNX

REFINING A FAST MODEL TO MAKE IT FASTER

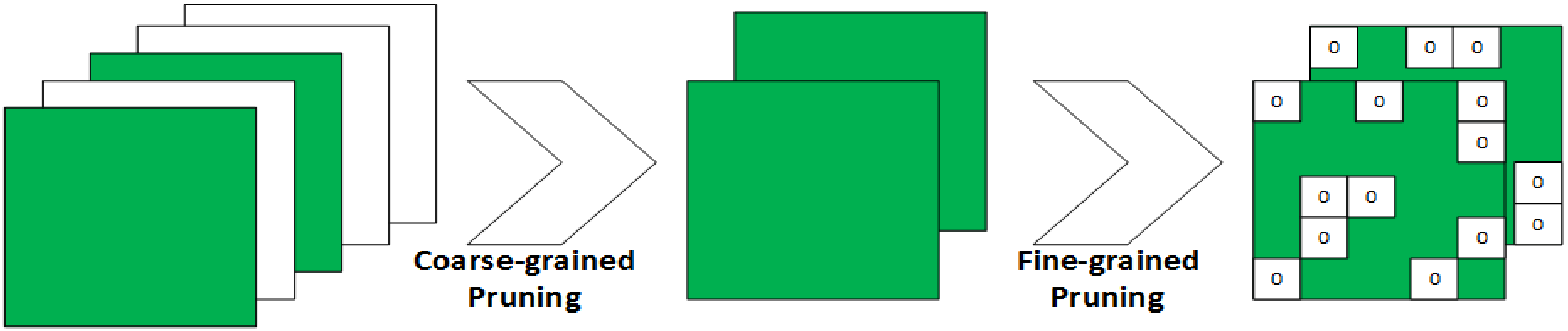
Technique	What are we reducing?
Prune Networks	OPs, heapSize, #params
Use 8 bits for activation and weights	OPs
Use <8 bits for weights / codebook	Parameter bytes
Sparsify	ModelSize, OPs
Split 3x3 Conv in to DW separable Conv	OPs
Use <8 bits for activations	heapSize, OPS

PORTABILITY: COARSE GRAINED PRUNING



- Model Pruning
  - Very effective when transfer learning to simpler domains
  - Consider pruning to multiples of 8/16 channels. Many hardware implementations have this type of restriction

PORTABILITY: FINE GRAINED PRUNING



- Fine Grained Pruning for Sparsity
  - Good benefit by reducing deployment model size
  - Less Weight bandwidth on platforms supporting compression

INTERSECTION OF PORTABILITY AND MODEL REFINEMENT

- Reducing Precision of Weights (4b / 2b / 1b)
  - Reduce precision
  - Benefits over a range of platforms:
    - Save bandwidth on platforms that directly support low precision weights
    - Save a little less bandwidth on platforms that just support compression
    - Can still work on all platforms

INTERSECTION OF PORTABILITY AND MODEL REFINEMENT

- Enhancing portability of 8 bits
  - Dynamic range of activations introduces some risks when determining scale factor
  - Some layers can require higher precision

  - Solutions:
    - Train with RELU6: $y = \min(\max(x, 0), 6)$
    - Train with Batch Norm, by default keeps $\sigma=1$

RESULTS

(PyTorch) ResNet50	#Param bytes (Non Zero)	TOPs	Accuracy @Top1	Ops/Paramter Byte (higher better)
Baseline	25.5M	7.66	76.01%	300
Fine-grained (80% sparse)	5.1M (5x)	7.66	75.68%	1502
Coarse-grained Pruning	17.2M	3.82 (2x)	74.87%	222
Hybrid: Coarse then Fine (73% sparse thin)	6.9M	3.82	74.32%	554
Hybrid + 4b weights	3.5M	3.82	73.81%	1107

⇒ Pruning, sparsity and low precision are compatible and portable
⇒ 0.3%-2.2% accuracy loss, gap reducing over time

RESULTS

(PyTorch) ResNet50	#Param bytes (Non Zero)	TOPs		
Baseline	25.5M	7.66		
Fine-grained (80% sparse)	5.1M (5x)	7.66		
Coarse-grained Pruning	17.2M	3.82 (2x)	74.87%	222
Hybrid: Coarse then Fine (73% sparse thin)	6.9M	3.82	74.32%	554
Hybrid + 4b weights	6.9M	3.82	73.81%	1107

Learn more by visiting Intel Movidius Team Members in Poster Session

Starting at 12pm Today

"Low-precision Sparse Thin Network for Fast Inference on Edge Devices"

⇒ Pruning, sparsity and low precision are compatible and portable

⇒ 0.3%-2.2% accuracy loss, gap reducing over time

INTERSECTION OF PORTABILITY AND MODEL REFINEMENT
SUMMARY

Technique	Portability
Prune Networks	Good, benefit varies
Use 8 bits for activation and weights	Good, when used with care
Use <8 bits for weights / codebook	Good, benefit varies
Sparsify	Good, benefit varies
Split 3x3 Conv in to DW separable Conv	Varies
Use <8 bits for activations	Poor

OVERVIEW

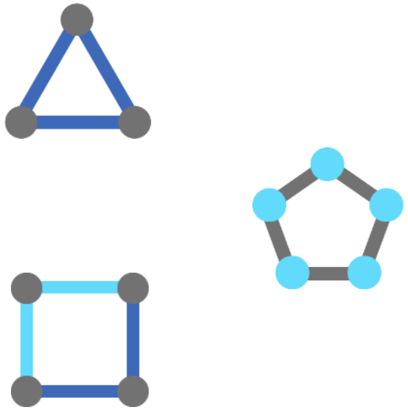
- Selecting a fast network
- Making a fast network faster in a portable way
- Network portability: Ecosystems including ONNX

PORTABLE NETWORK ECOSYSTEMS

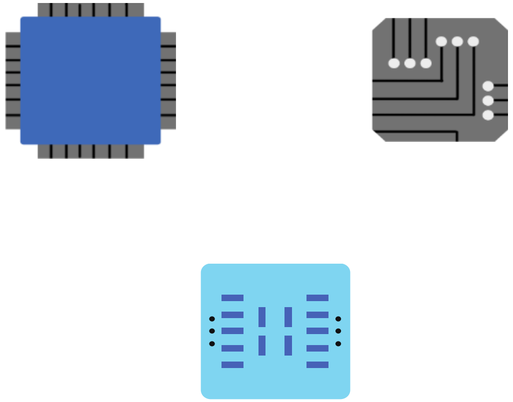
- **Deploying Model on Multiple Targets**
    - OS Specific frameworks
        - DirectML
        - AndroidNN API
        - CoreML
    - Network interchange:
        - ONNX

PORTABILITY: ONNX GOALS

Provide a standard way to represent models so that:

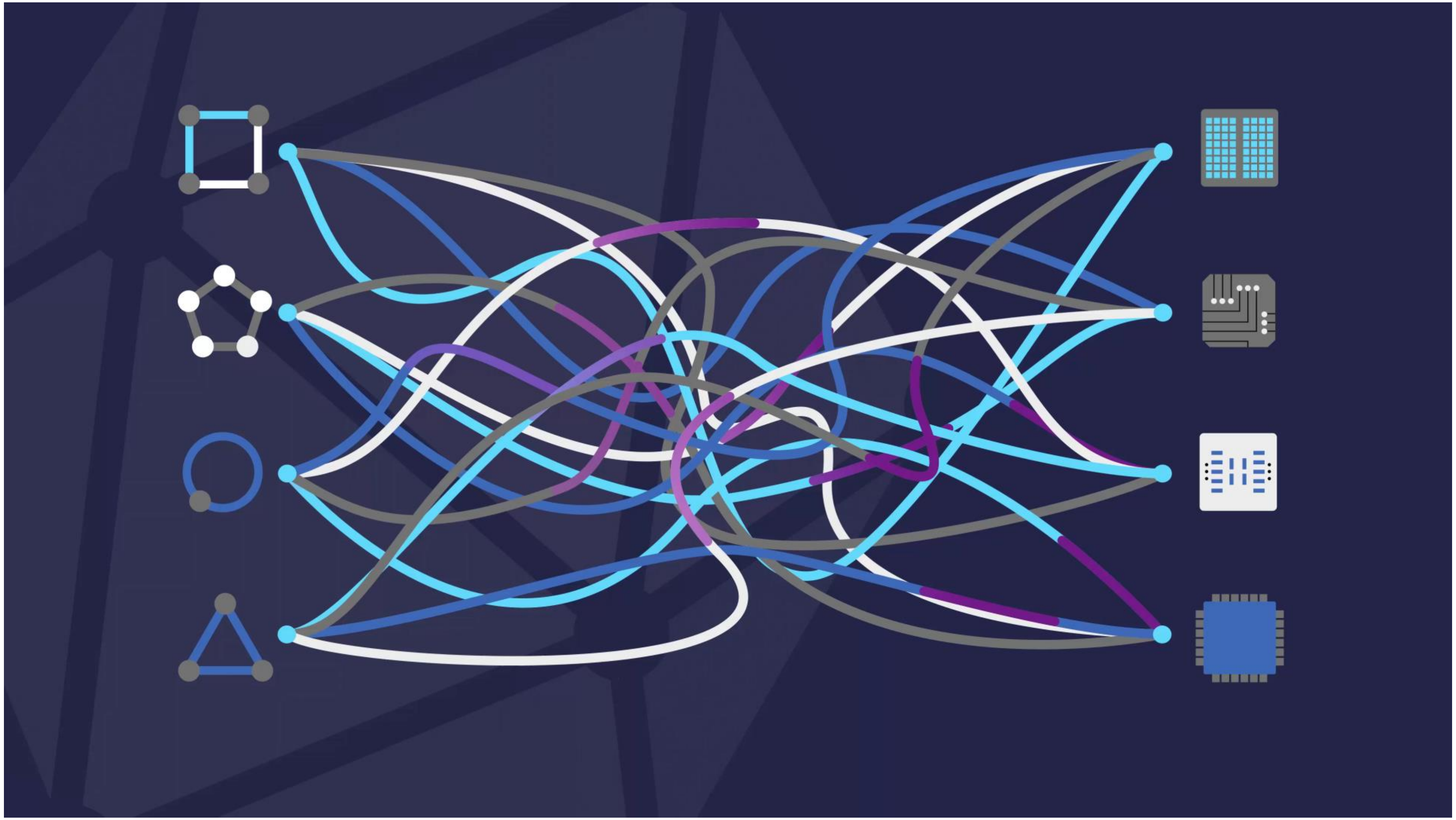


Serialized models are interoperable between frameworks



Have a common target for optimization for different backends

ONNX - OVERVIEW



ONNX: OPEN ECOSYSTEM FOR AI MODELS

INTEL OPENVINO

DELIVERS COMPUTER VISION AND DEEP LEARNING CAPABILITIES FROM EDGE TO CLOUD

- AGNOSTIC, COMPLEMENTARY TO MAJOR FRAMEWORKS
- HIGH PERFORMANCE, HIGH EFFICIENCY FOR THE EDGE
- CROSS-PLATFORM FLEXIBILITY
- OPEN SOURCE: COMING SOON

KEY TAKEAWAYS

- Select network carefully considering dataflow implications
- Optimize networks using portable techniques, specifically:
  - Pruning, 8 bit activations, low precision weights, sparsity
- ONNX has strong momentum as ecosystem for portable models

RESOURCES

- Useful Resources:
  - Intel Nervana AI Academy
  - http://www.arxiv-sanity.com/
  - https://github.com/NervanaSystems/distiller
- References:

  - [1] Learning Transferable Architectures for Scalable Image Recognition, https://arxiv.org/abs/1707.07012
  - [2] MobileNets: Efficient Convolutional Neural Networks for Mobile Vision Applications, https://arxiv.org/abs/1704.04861
  - [3] To prune, or not to prune: exploring the efficacy of pruning for model compression, https://arxiv.org/abs/1710.01878
  - [4] Learning both weights and connections for efficient neural networks, https://arxiv.org/abs/1506.02626
  - [5] Quantization and Training of Neural Networks for Efficient Integer-Arithmetic-Only Inference, https://arxiv.org/abs/1712.05877

LEGAL DISCLAIMER

intel® AI