embedded VISION SUMMIT 2018

Portability and Performance in Embedded Deep Neural Networks: Can We Have Both?

Cormac Brick, Director Embedded Machine Intelligence

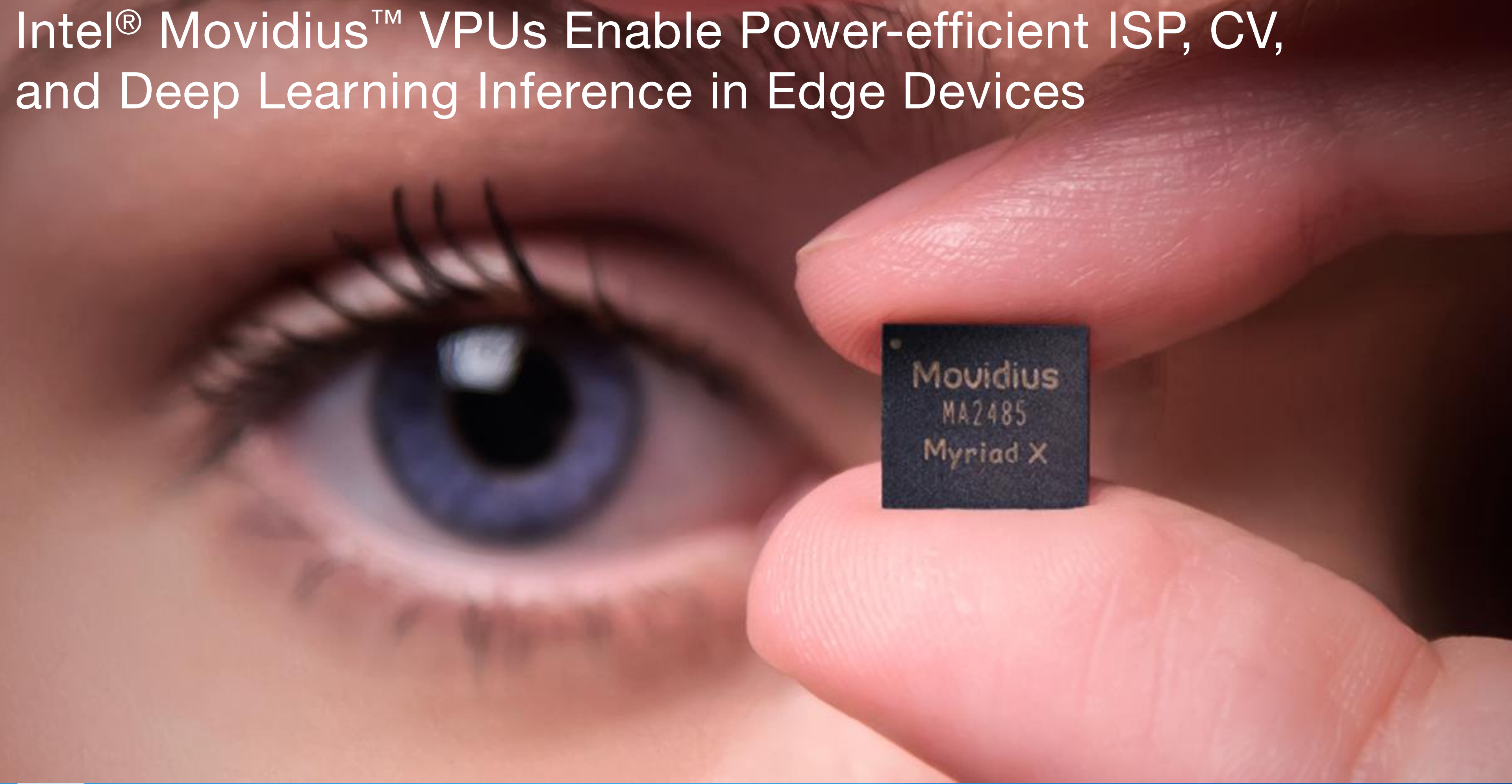
May 22, 2018

intel Movidius™

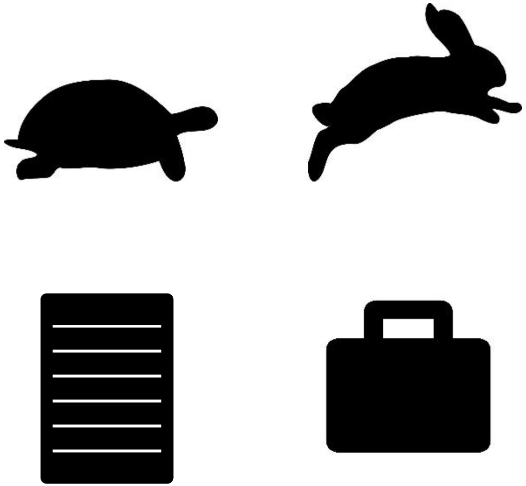
Legal Disclaimer

- INFORMATION IN THIS DOCUMENT IS PROVIDED IN CONNECTION WITH INTEL PRODUCTS. NO LICENSE, EXPRESS OR IMPLIED, BY ESTOPPEL OR OTHERWISE, TO ANY INTELLECTUAL PROPERTY RIGHTS IS GRANTED BY THIS DOCUMENT. EXCEPT AS PROVIDED IN INTEL'S TERMS AND CONDITIONS OF SALE FOR SUCH PRODUCTS, INTEL ASSUMES NO LIABILITY WHATSOEVER AND INTEL DISCLAIMS ANY EXPRESS OR IMPLIED WARRANTY, RELATING TO SALE AND/OR USE OF INTEL PRODUCTS INCLUDING LIABILITY OR WARRANTIES RELATING TO FITNESS FOR A PARTICULAR PURPOSE, MERCHANTABILITY, OR INFRINGEMENT OF ANY PATENT, COPYRIGHT OR OTHER INTELLECTUAL PROPERTY RIGHT.
- A "Mission Critical Application" is any application in which failure of the Intel Product could result, directly or indirectly, in personal injury or death. SHOULD YOU PURCHASE OR USE INTEL'S PRODUCTS FOR ANY SUCH MISSION CRITICAL APPLICATION, YOU SHALL INDEMNIFY AND HOLD INTEL AND ITS SUBSIDIARIES, SUBCONTRACTORS AND AFFILIATES, AND THE DIRECTORS, OFFICERS, AND EMPLOYEES OF EACH, HARMLESS AGAINST ALL CLAIMS COSTS, DAMAGES, AND EXPENSES AND REASONABLE ATTORNEYS' FEES ARISING OUT OF, DIRECTLY OR INDIRECTLY, ANY CLAIM OF PRODUCT LIABILITY, PERSONAL INJURY, OR DEATH ARISING IN ANY WAY OUT OF SUCH MISSION CRITICAL APPLICATION, WHETHER OR NOT INTEL OR ITS SUBCONTRACTOR WAS NEGLIGENT IN THE DESIGN, MANUFACTURE, OR WARNING OF THE INTEL PRODUCT OR ANY OF ITS PARTS.
- Intel may make changes to specifications and product descriptions at any time, without notice. Designers must not rely on the absence or characteristics of any features or instructions marked "reserved" or "undefined". Intel reserves these for future definition and shall have no responsibility whatsoever for conflicts or incompatibilities arising from future changes to them. The information here is subject to change without notice. Do not finalize a design with this information. The products described in this document may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.
- Contact your local Intel sales office or your distributor to obtain the latest specifications and before placing your product order.
- Copies of documents which have an order number and are referenced in this document, or other Intel literature, may be obtained by calling 1-800-548-4725, or go to: http://www.intel.com/design/literature.htm
- Software and workloads used in performance tests may have been optimized for performance only on Intel microprocessors. Performance tests, such as SYSmark and MobileMark, are measured using specific computer systems, components, software, operations and functions. Any change to any of those factors may cause the results to vary. You should consult other information and performance tests to assist you in fully evaluating your contemplated purchases, including the performance of that product when combined with other products.
- All products, computer systems, dates and figures specified are preliminary based on current expectations, and are subject to change without notice.
- All products, platforms, dates, and figures specified are preliminary based on current expectations, and are subject to change without notice. All dates specified are target dates, are provided for planning purposes only and are subject to change.
- This document contains information on products in the design phase of development. Do not finalize a design with this information. Revised information will be published when the product is available. Verify with your local sales office that you have the latest datasheet before finalizing a design.
- Code names featured are used internally within Intel to identify products that are in development and not yet publicly announced for release. Customers, licensees and other third parties are not authorized by Intel to use code names in advertising, promotion or marketing of any product or services and any such use of Intel's internal code names is at the sole risk of the user.
- Intel, Intel Inside, Intel Atom and Intel Core are trademarks of Intel Corporation in the U.S. and other countries.
- Other names and brands may be claimed as the property of others.
-

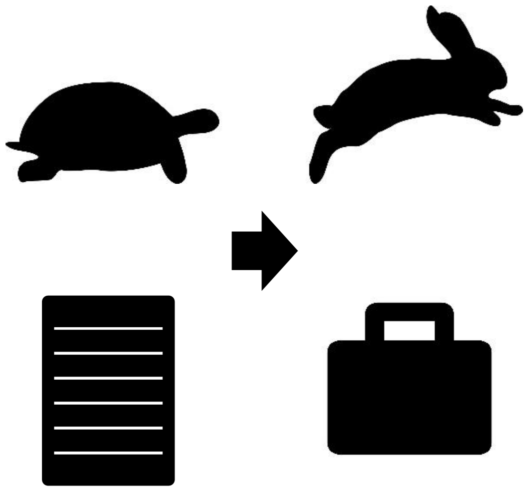
Intel® Movidius™ VPUs Enable Power-efficient ISP, CV, and Deep Learning Inference in Edge Devices



The Question:

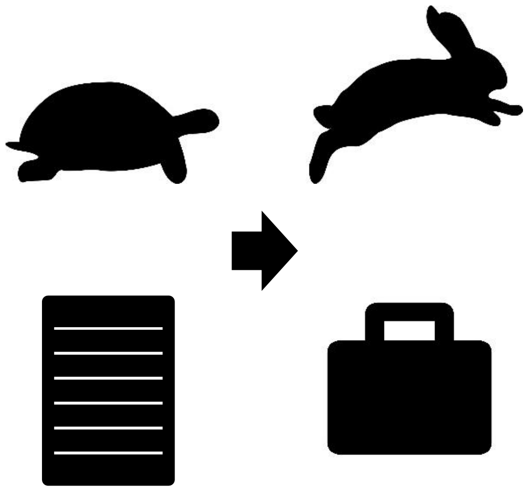


The Question:



Can we have neural networks that are fast and portable?

The Question (final version):



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
- Challenges & open items

What is a fast network?

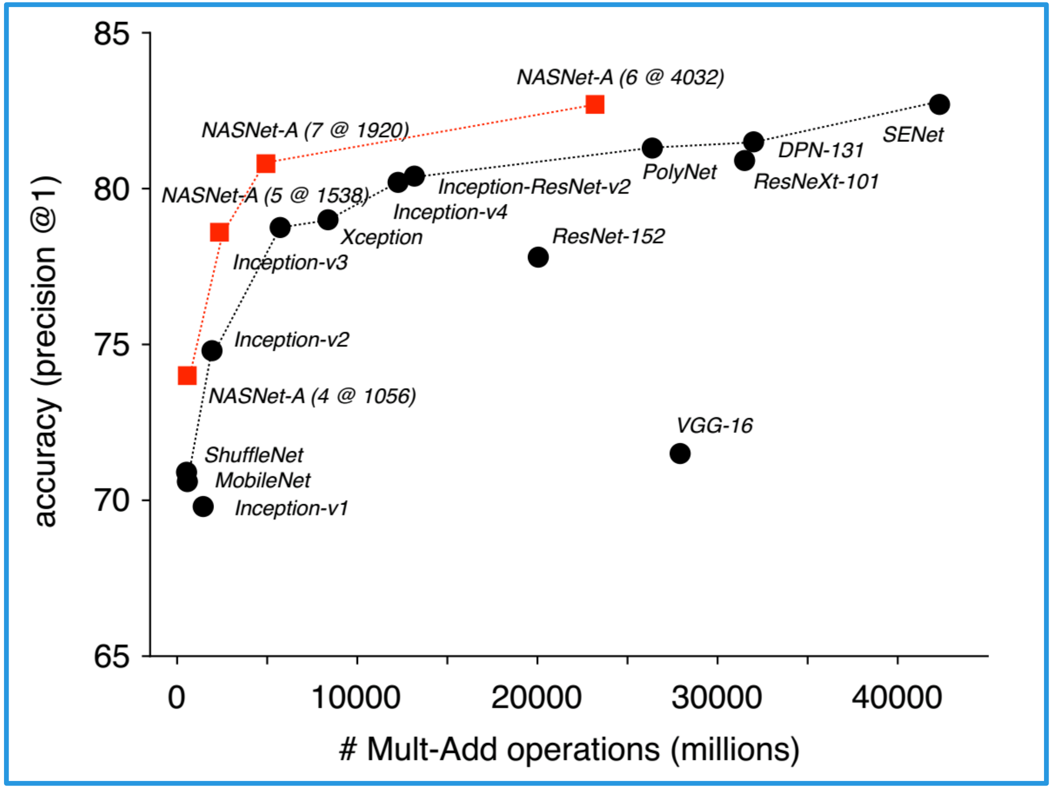
Classical approach, look at:

- Number of FLOPS
- Number of parameters

What is a fast network?

Classical approach, look at:

1. Number of FLOPS

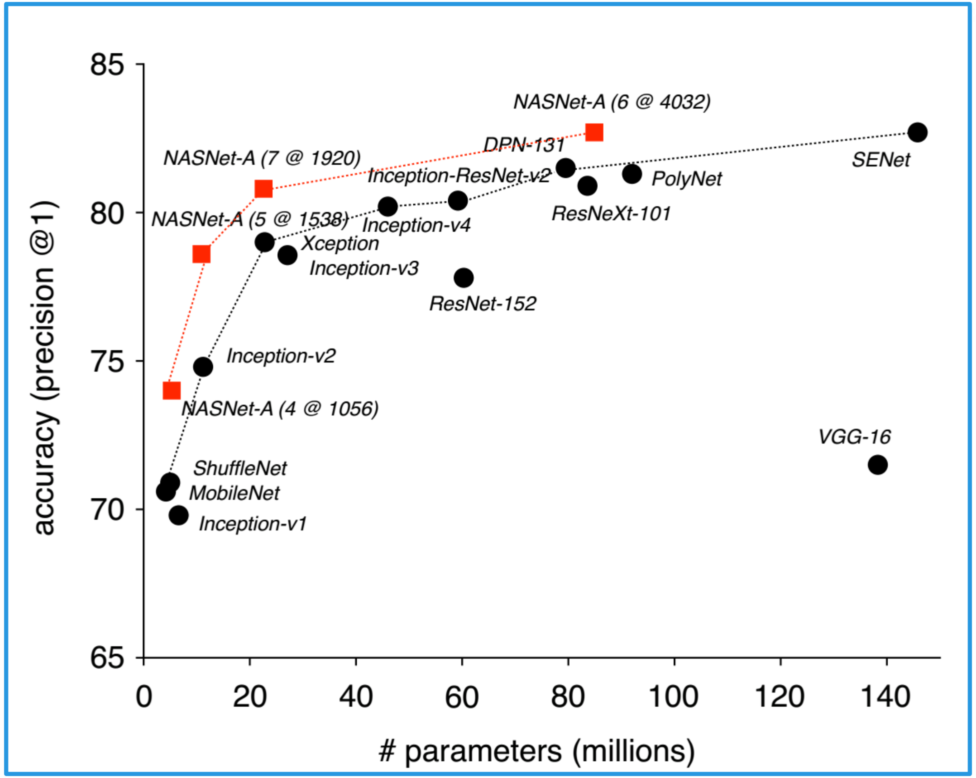


Ref[1]

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

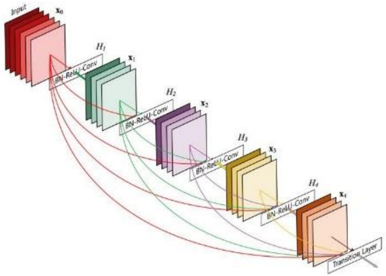
   → *Keep activations in local mem/cache*

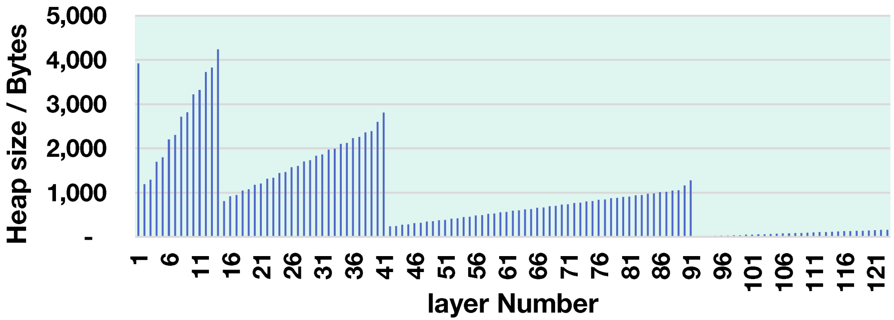
4. FLOPS/param/layer

   → *Avoid being DDR bound on weight fetch*

Activation heap size

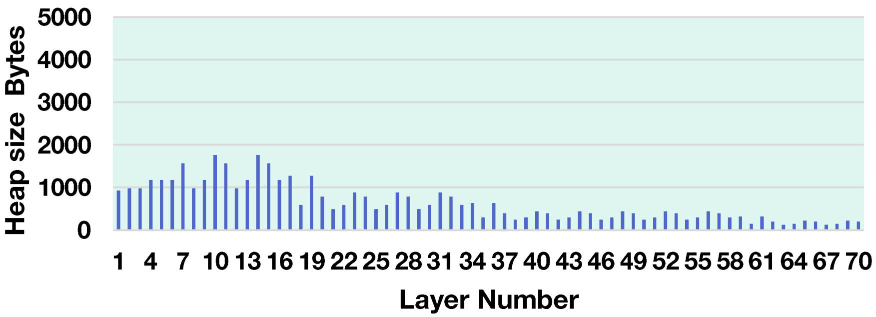
DenseNet





Long lifetime data – larger heap

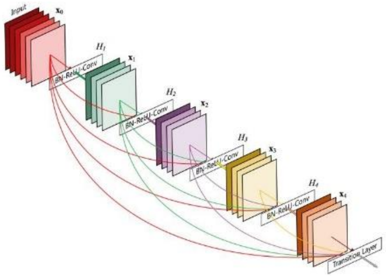
Resnet 50



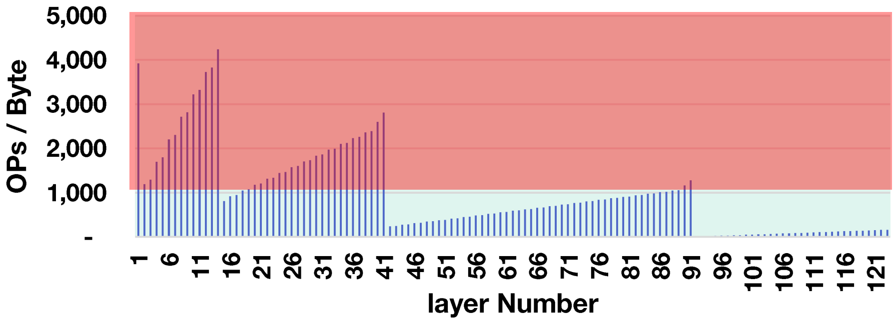
limited lifetime data – smaller heap

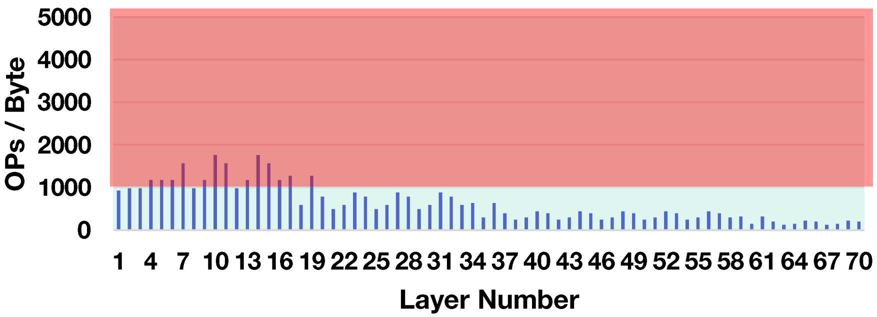
Activation heap size – what if only 1MB L0 mem?

DenseNet



Resnet 50

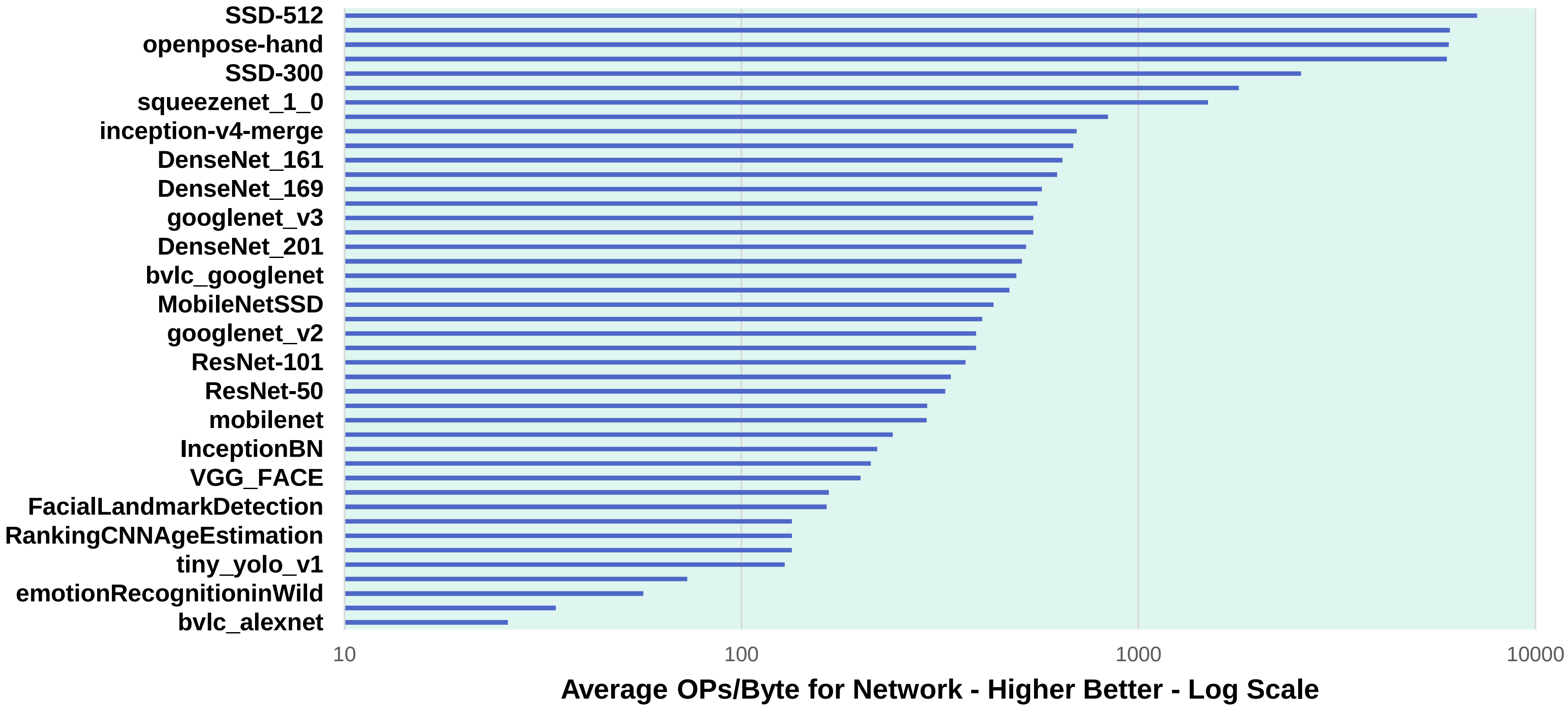




Long lifetime data – larger heap

limited lifetime data – smaller heap

Average OPs/Byte on Common Vision Networks



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

See resources slide for relevant references

Intersection of portability and Model refinement

- Model Pruning
  - Consider pruning to multiples of 8/16 channels. Many hardware implementations have this type of restriction
- Reducing Precision of Weights (4b / 2b / 1b)
  - Using discrete values for weights will
    - 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 risks for:
    - Accumulator overflows
    - Edge cases when determining scale factor
  - Solutions:
    - Train with RELU6: $y = \min(\max(x, 0), 6)$
    - Train with Batch Norm, by default keeps $\sigma=1$

Intersection of portability and Model refinement

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

Results

(PyTorch) ResNet50	#Param bytes (Non Zero)	TOPs	Accuracy @Top1	Ops/Parameter 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	6.9M	3.82	73.81%	1107

⇒ 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

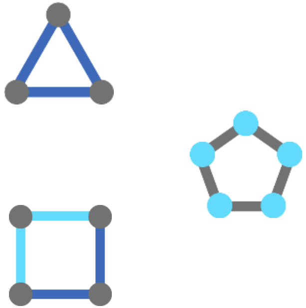
Technique	Portability
Prune Networks	Good, benefit varies
Use 8 bits for activation and weights	Good, when used with care
Use <8 bits for weights / codebook	Good, benefit varies
Sparsify	Good, benefit varies
Split 3x3 conv in to depth-wise separable conv	Varies
Use <8 bits for activations	Poor

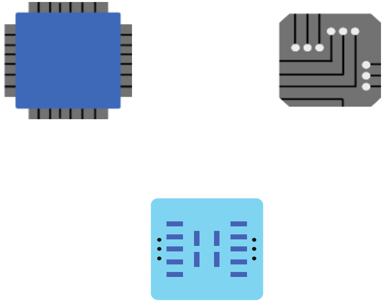
Portable Network Ecosystems

- **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:

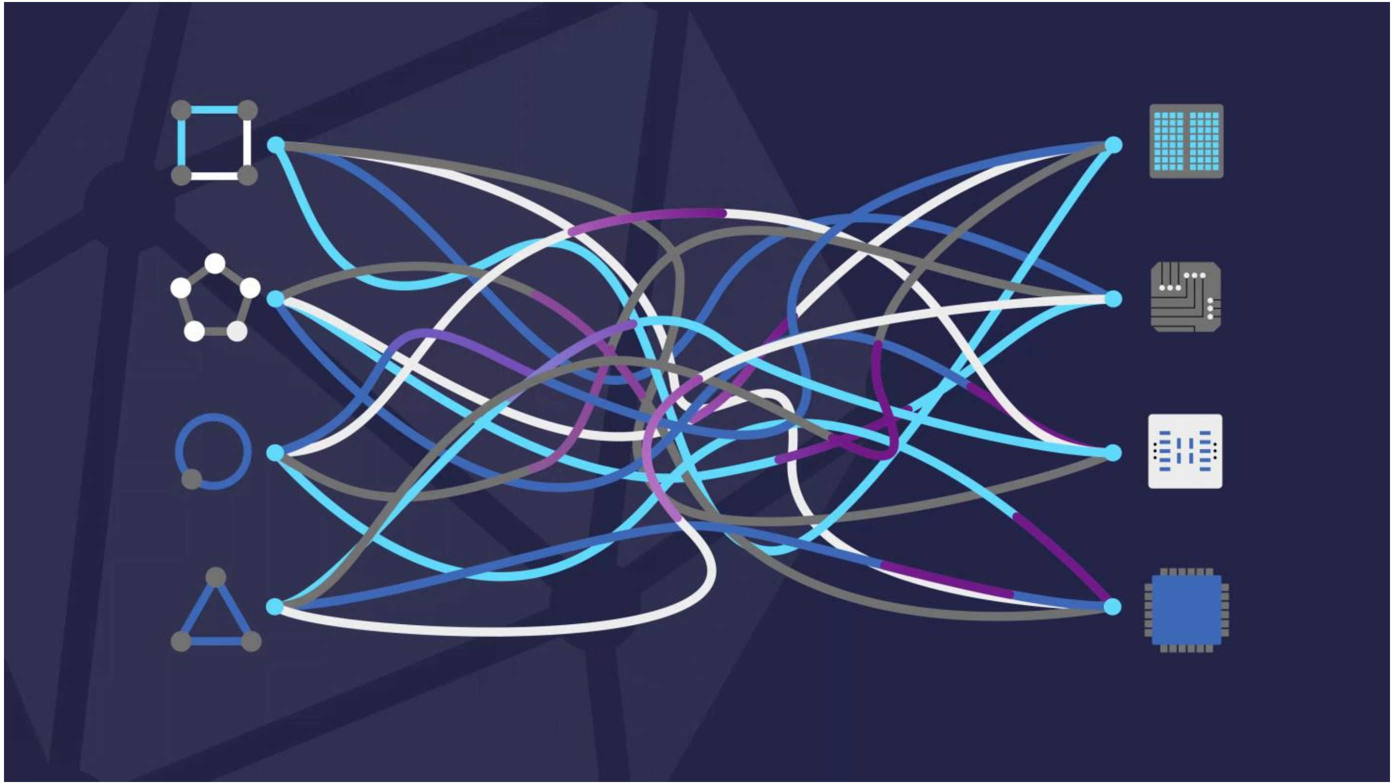




Serialized models are interoperable between frameworks

Have a common target for optimization for different backends

ONNX - Overview



ONNX: Open Ecosystem for AI Models

High level API & Framework Frontends

- Caffe2
- Cognitive Toolkit
- mxnet
- PYTORCH
- Chainer
- PaddlePaddle

ONNX

Hardware Vendor Libraries & Devices

- ML HW
- GPU
- CPU
- FPGA
- DSP

Challenges

- **8-bit Support still emerging**
  - ONNX today does not support this (being fixed quickly)
  - Poor support for 8 bits in training frameworks:
    - Tensorflow notable exception
  - More than 1 flavor of 8 bits (symmetric/asymmetric)

- **Custom layers still a hard problem**
  - Multiple candidate solutions: ANSI-C, directML HLSL, OpenCL, Halide, …

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/
- 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