IXPUG Annual Conference 2019

Performance and Scalability Analysis of CNN-based Deep Learning Inference in the Intel Distribution of OpenVINO Toolkit

Iosif Meyerov[1], Valentina Kustikova[1], Evgeny Vasilyev[1], Evgeny Kozinov[1], Valentin Volokitin[1], Nadezhda Ageeva[2], Yuri Gorbachev[2], Zakhar Matveev[2]

[1] Lobachevsky State University of Nizhni Novgorod, Russia
[2] Intel Corporation

CERN, Geneva, Switzerland

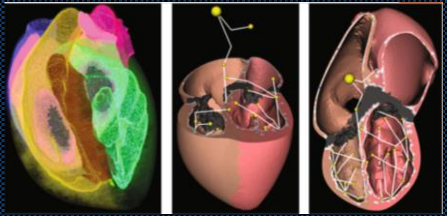
September 24, 2019

Contents

- Motivation
- Objective
- DL Frameworks
  - Intel Distribution of OpenVINO Toolkit
  - Intel Optimization for Caffe
  - OpenCV
- Models (ResNet-50, SSD-300)
- Computational infrastructure
- Numerical results
- Conclusions

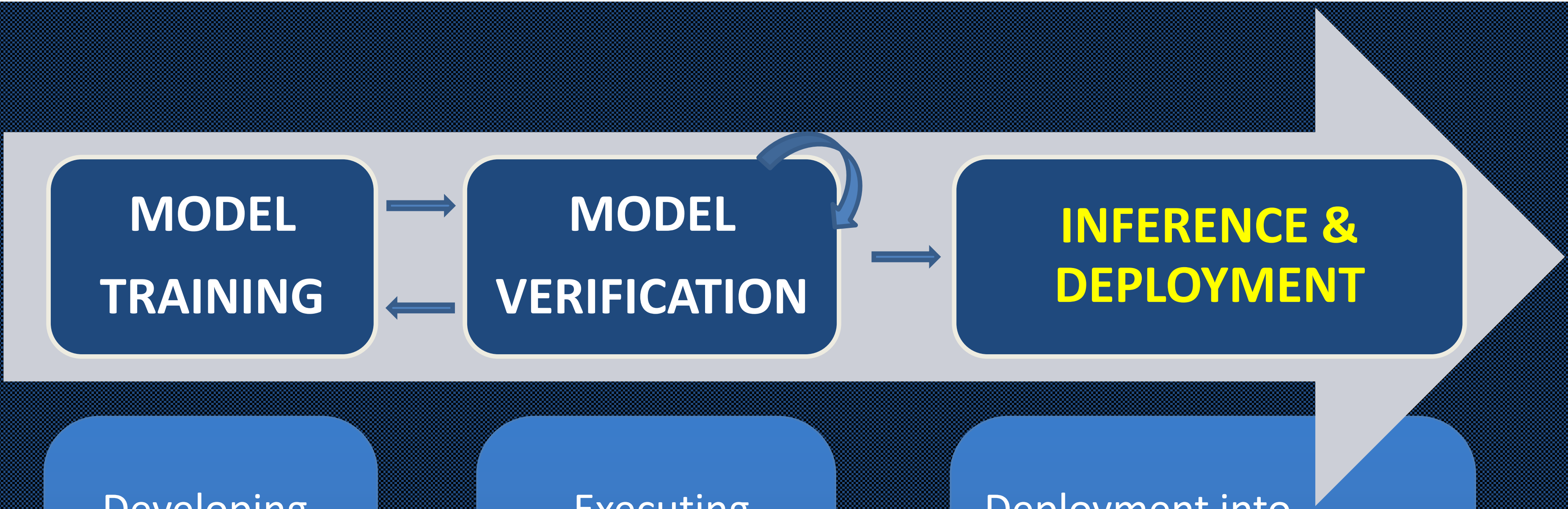
Motivation

- Deep learning is everywhere
  - Computer vision
  - Natural language processing
  - Bioinformatics
  - Biomedicine
  - …
- Deep learning is Supercomputing
  - Large-scale neural networks
  - Computationally intensive training
  - Need of real-time inference



[*] Source of the pictures: 30 amazing applications of deep learning http://www.yaronhadad.com/deep-learning-most-amazing-applications

Deep Learning Lifecycle



MODEL TRAINING

MODEL VERIFICATION

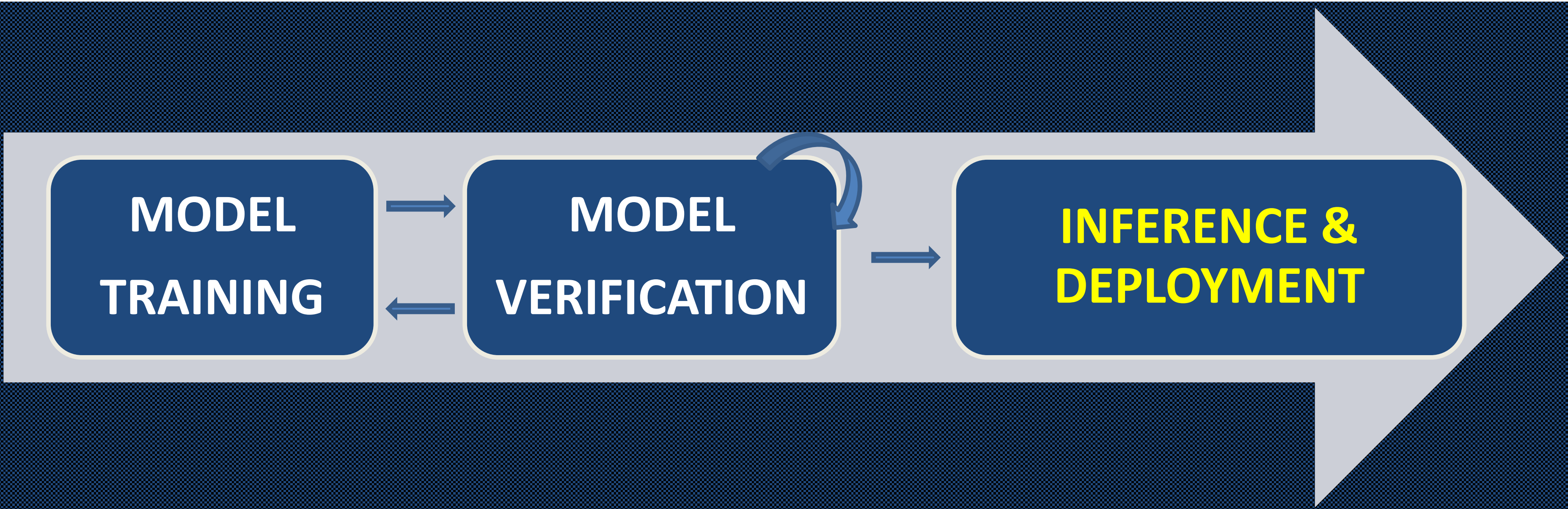
INFERENCE & DEPLOYMENT

Developing, training and fine-tuning of a deep model

Executing inference to verify a model **accuracy**

Deployment into production environment
Optimizing a model for inference & deployment

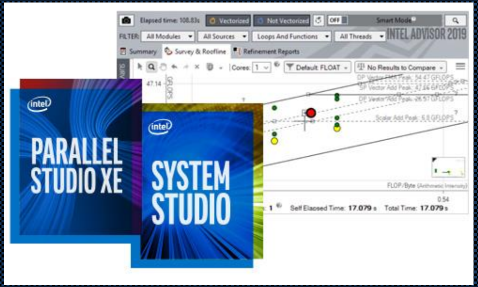
Deep Learning Lifecycle

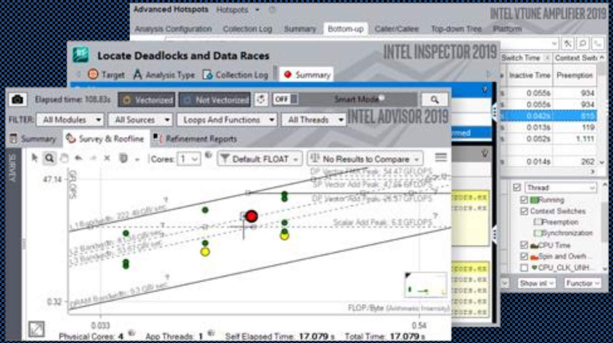


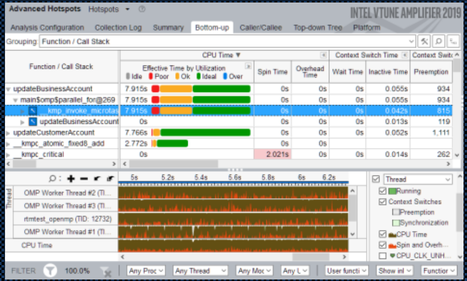
THIS TALK IS FOCUSED
ON DEEP LEARNING INFERENCE ON INTEL CPUs

Deep Learning Inference (1)

- Computationally intensive procedure
- Need of real-time inference in state-of-the-art applications
- Development problems:
  - Code optimization is not straightforward due to the variety of topologies of deep neural networks
  - Code parallelization is not trivial due to various possible usage scenarios (synchronous and asynchronous modes, load balancing)



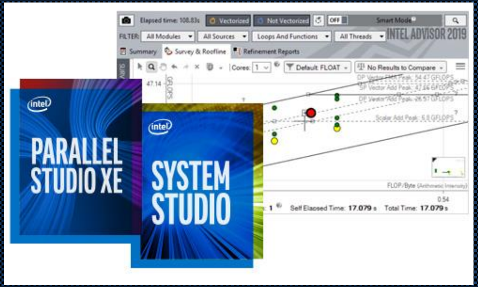


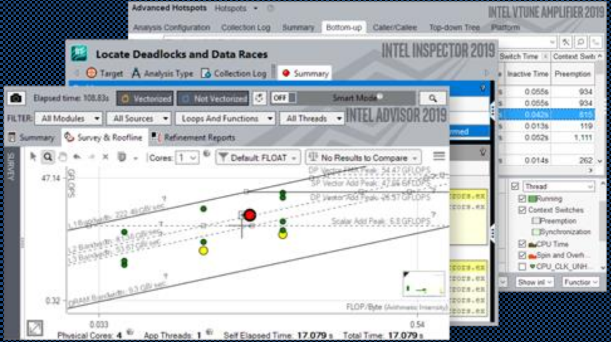


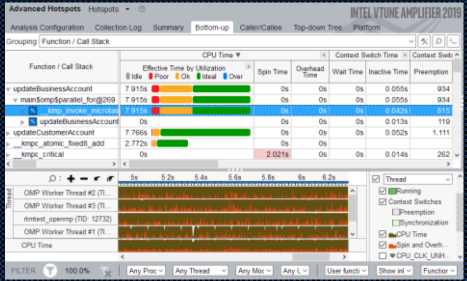
* Source of the pictures: www.intel.com

Deep Learning Inference (2)

- **Computationally intensive** procedure
- Need of **real-time** inference in state-of-the-art applications
- **User problems:**
  - **A lot** of parameters (mode, #threads, batch size, HT mode…)
  - How to find the best or at least relevant combination?







[*] Source of the pictures: www.intel.com

Deep Learning Frameworks

- **Three well established deep learning inference frameworks***
- **Intel Optimization for Caffe**
  - “This optimized fork is dedicated to improving Caffe performance when running on a CPU”
- **Intel Distribution of OpenVINO Toolkit**
  - “The toolkit extends workloads across Intel hardware (including accelerators) and maximizes performance”
- **OpenCV**
  - “OpenCV was built to provide a common infrastructure for CV applications and to accelerate the use of machine perception in the commercial products”

* *Ongoing work*: the number of frameworks will be extended

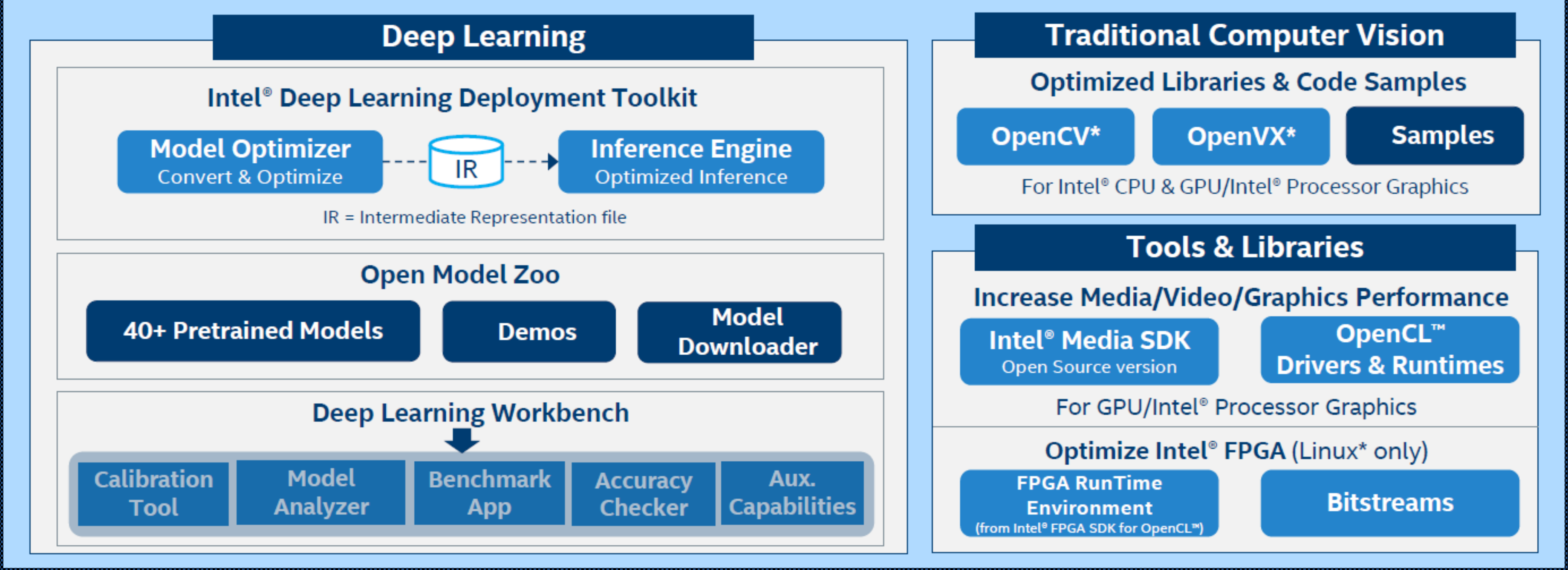
Objective: What Is This Talk About?

- Finding optimal run parameters for DL inference in OpenVINO
- Analysis of scaling efficiency of OpenVINO using dozens of CPU cores in different modes
- Comparison of performance of CNN-based DL inference frameworks on Intel CPUs
- Exploring performance improvement of int8 quantization for fast CPU inference using OpenVINO
- Exploring the results of Intel AVX512 VNNI performance acceleration in Intel CascadeLake CPUs by means of Intel Advisor

OpenVINO: Main Principles

- Focuses on the developing cross-platform solutions of CV problems based on image processing, ML, and DL
- Provides a wide range of algorithms optimized to achieve maximum performance on the Intel hardware (CPUs, IPG, Movidius, FPGAs…)
- Provides heterogeneous execution of algorithms on various Intel accelerators using the same API

OpenVINO: What Is Inside?



Deep Learning

Intel® Deep Learning Deployment Toolkit

Model Optimizer
Convert & Optimize

IR

Inference Engine
Optimized Inference

IR = Intermediate Representation file

Open Model Zoo

40+ Pretrained Models

Demos

Model Downloader

Deep Learning Workbench

Calibration Tool

Model Analyzer

Benchmark App

Accuracy Checker

Aux. Capabilities

Traditional Computer Vision

Optimized Libraries & Code Samples

OpenCV*

OpenVX*

Samples

For Intel® CPU & GPU/Intel® Processor Graphics

Tools & Libraries

Increase Media/Video/Graphics Performance

Intel® Media SDK
Open Source version

OpenCL™ Drivers & Runtimes

For GPU/Intel® Processor Graphics

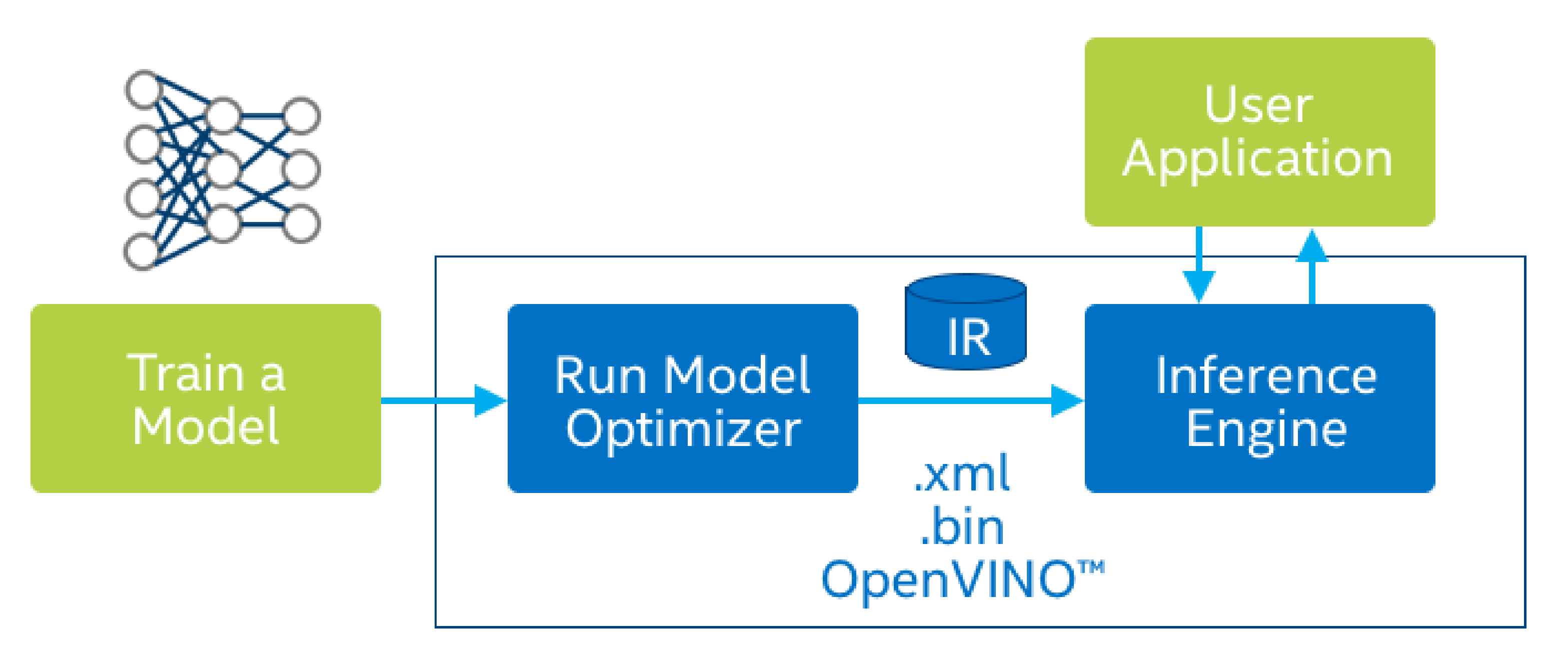
Optimize Intel® FPGA (Linux* only)

FPGA RunTime Environment
(from Intel® FPGA SDK for OpenCL™)

Bitstreams

OS Support: CentOS 7.4 (64 bit), Ubuntu 16.04.3 LTS (64 bit), Microsoft Windows 10 (64 bit), Yocto Project version Poky Jethro v2.0.3(64 bit), macOS 10.13 & 10.14 (64 bit)

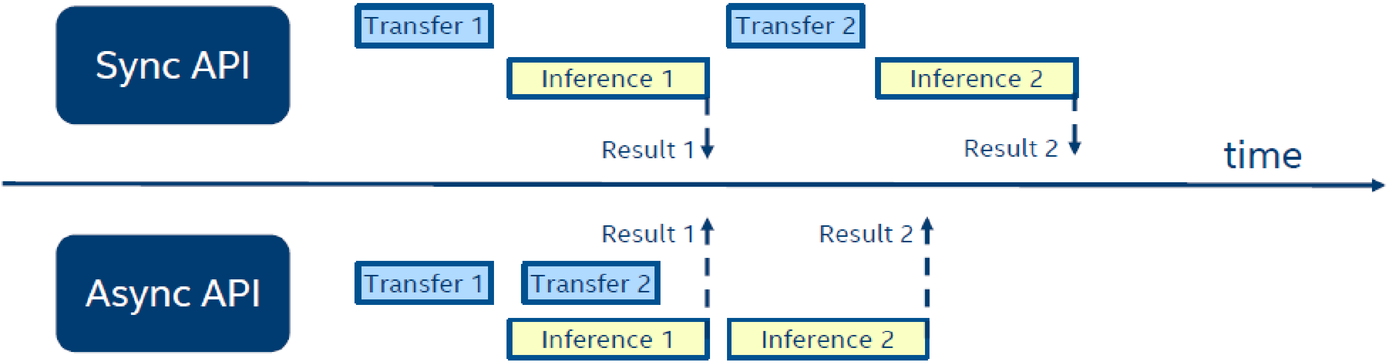
OpenVINO IE: How It Works?



* Source: Intel Distribution of OpenVINO Toolkit. Deep Learning For Computer Vision
https://software.intel.com/en-us/openvino-toolkit/deep-learning-cv

OpenVINO IE: Execution Modes

- **Inference Request** contains the batch of samples
- **Latency (or synchronous) mode,** provides the best **latency**
  - Supposes the next inference request is executed after the completion of the previous one
- **Throughput (or asynchronous) mode,** provides the best **throughput**
  - Assumes constructing a queue of inference requests, several requests can be executed in parallel



Models

- **ResNet-50**
  - He K., et al (2015) Deep Residual Learning for Image Recognition. [https://arxiv.org/pdf/1512.03385.pdf]
  - Image classification
  - ImageNet [http://www.image-net.org]
- **SSD300**
  - Liu W., et al (2015) SSD: Single Shot MultiBox Detector. [https://arxiv.org/pdf/1512.02325.pdf]
  - Object detection
  - Pascal Visual Object Challenge [http://host.robots.ox.ac.uk/pascal/VOC]

Computational Infrastructure

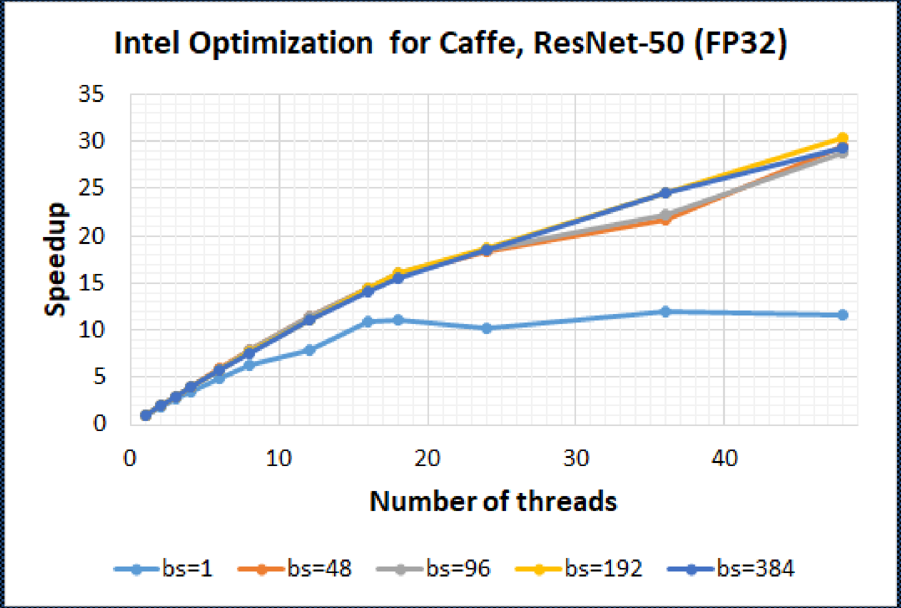
Intel Endeavour	
CPU	2x Intel Xeon Platinum 8260L 2.4GHz (2x24 cores), TurboBoost OFF *CascadeLake generation*
RAM	196 GB
OS	CentOS 7
Frameworks	Anaconda 4.5.12 Intel Optimization for Caffe 1.1.0 OpenCV 4.1.1 Intel Distribution of OpenVINO Toolkit 2019.2

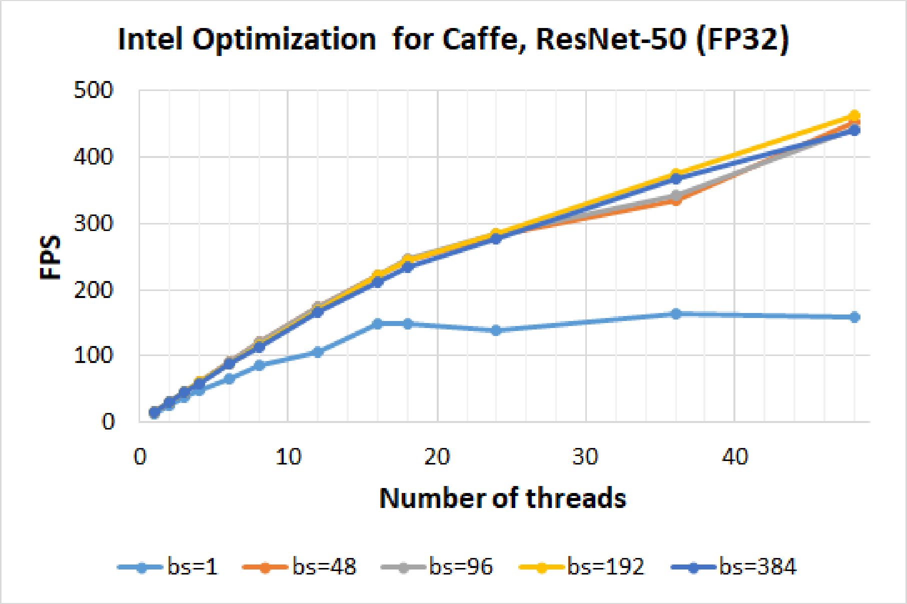
Experiment Setup

- Set of 1152 images from ImageNet/PASCAL VOC 2012 divided into batches
  (**Batch Size** is a parameter)
- **Caffe, OpenCV, OpenVINO (IE, sync. mode)**
  - For each request its run time is measured
  - The standard deviation is calculated on the set of obtained durations and the ones that goes beyond three standard deviations relative to the mean inference time are discarded
  - **Latency** is a median of execution times
  - **FPS** is the ratio of the batch size to the latency
- **OpenVINO (IE, async. mode)**
  - **FPS** is the ratio of the images number to the total execution time of all requests

Intel Optimization for Caffe

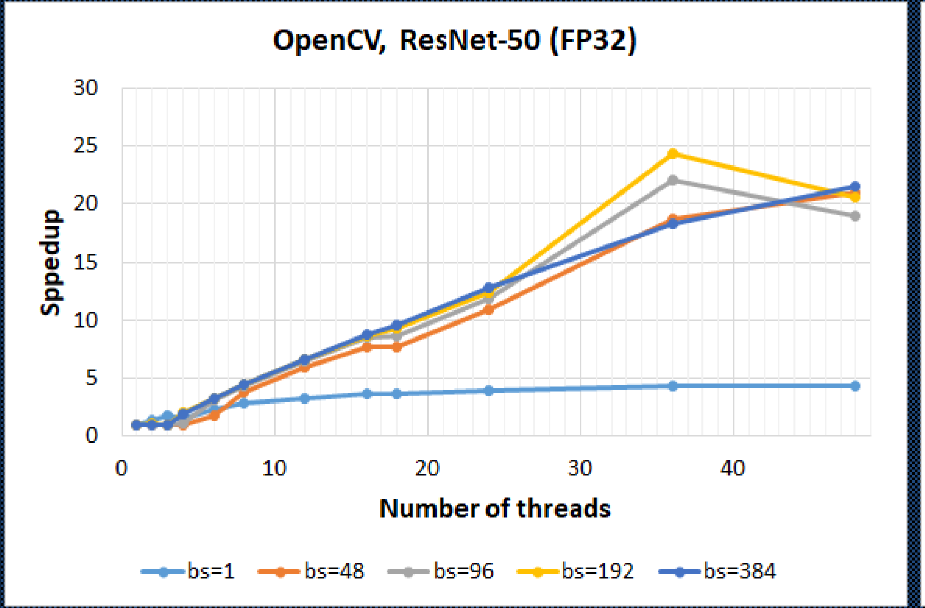
- Use OMP_NUM_TREADS to control #threads
- AFFINITY=compact,1,0
- 62.5% strong scaling efficiency on 48 cores, up to 450 FPS

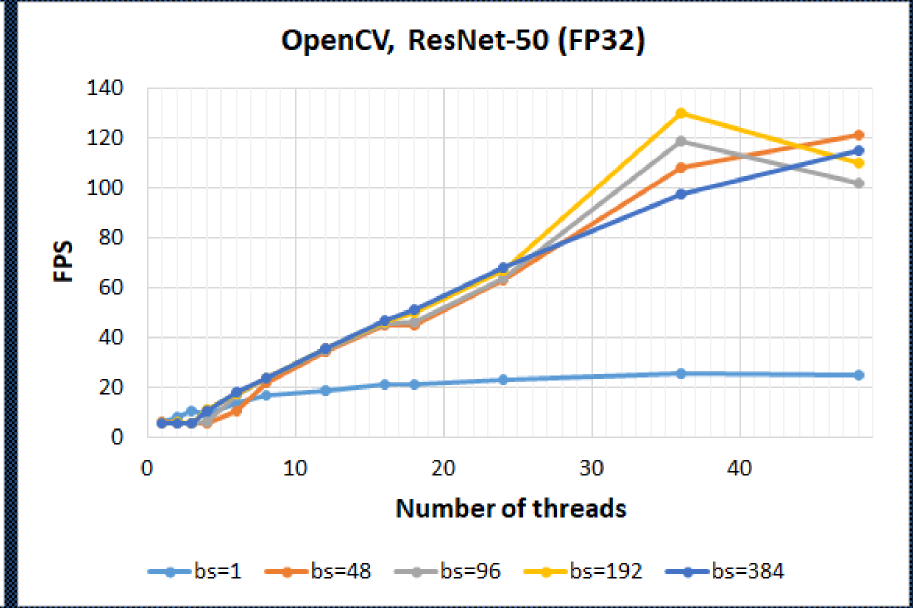




OpenCV

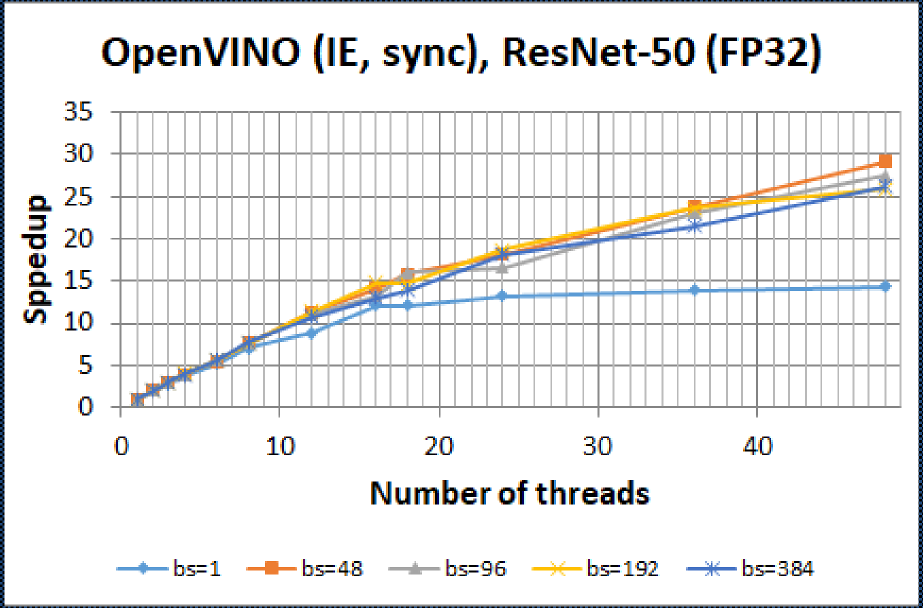
- Use TBB_NUM_TREADS to control #threads
- AFFINITY=compact,1,0
- 46% strong scaling efficiency on 48 cores, up to 130 FPS

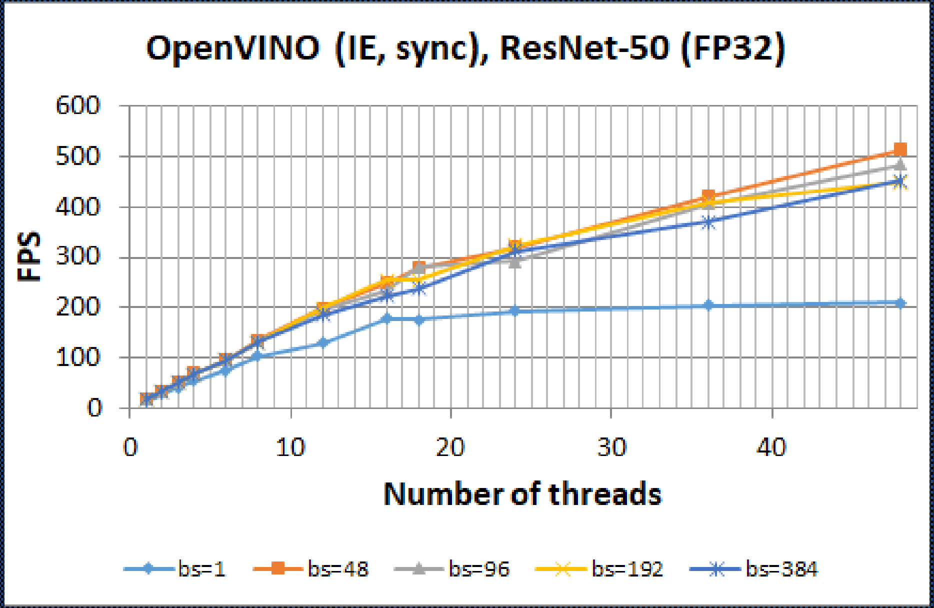




OpenVINO, IE, sync. mode

- Use plugin.set_config({'CPU_THREADS_NUM': str(thread_num)})
- AFFINITY=compact,1,0
- 62.5% strong scaling efficiency on 48 cores, up to 500 FPS!





Main Observations (1)

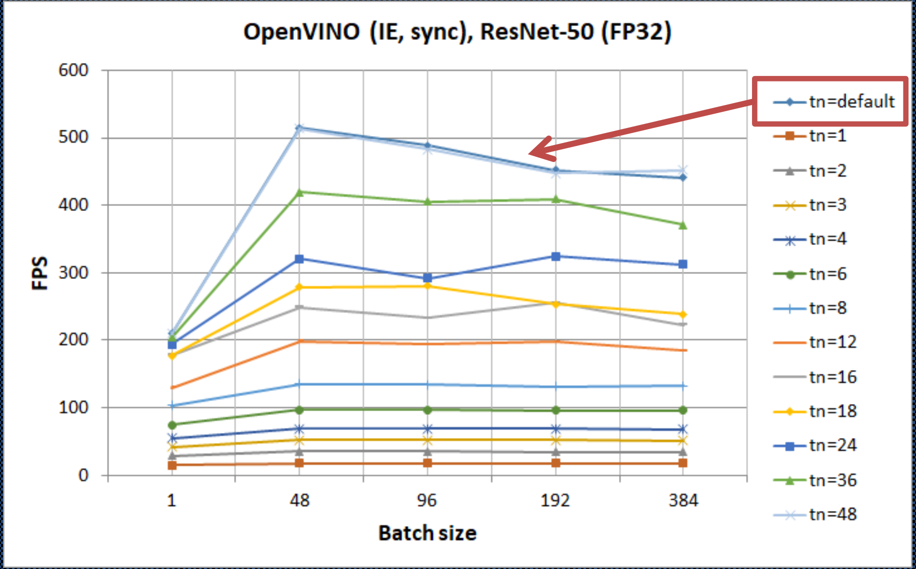
- All observations are the same **for both datasets**
- All frameworks scale quite well up to 48 cores (up to **62.5%** strong scaling efficiency for OpenVINO and Intel Caffe on ResNet-50)
- OpenVINO achieves the best performance in terms of FPS (up to **500 FPS** on ResNet-50)
- The choice of **batch size** highly affects performance and scaling efficiency, but **BS = 48+** is relevant

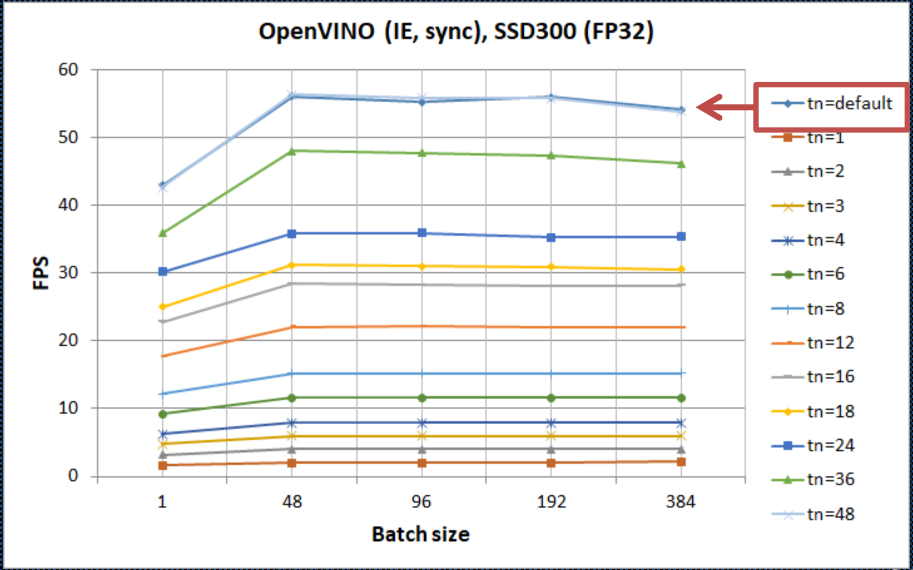
What about the number of cores?

Main Observations (2)

- What about the number of cores?
- OpenVINO IE can be executed with default settings.

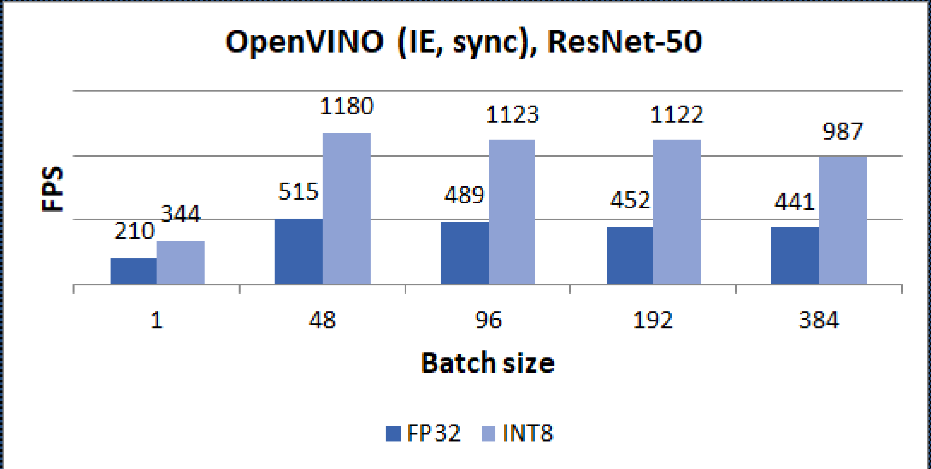
It is empirically the best choice!

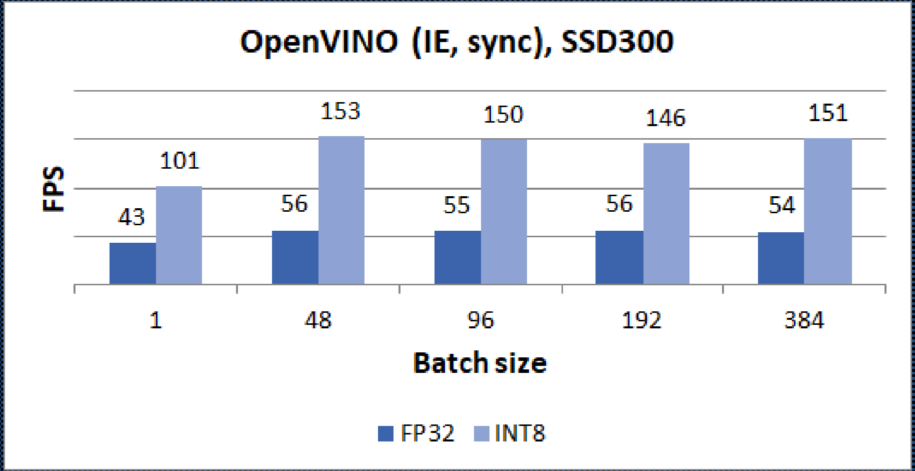




INT8 Quantization in OpenVINO

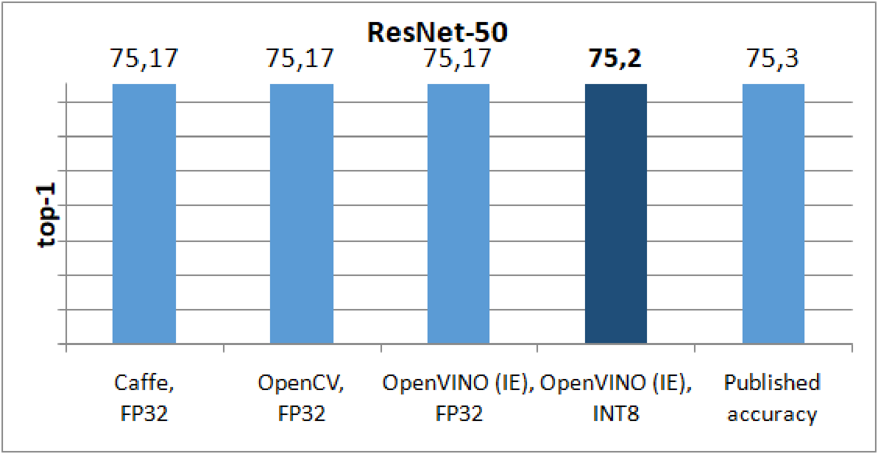
- The Calibration tool is used according to the documentation
- Default number of threads
- Sync. mode
- **Perf. improvement**: **2x** on ResNet-50 and **3x** on SSD300.

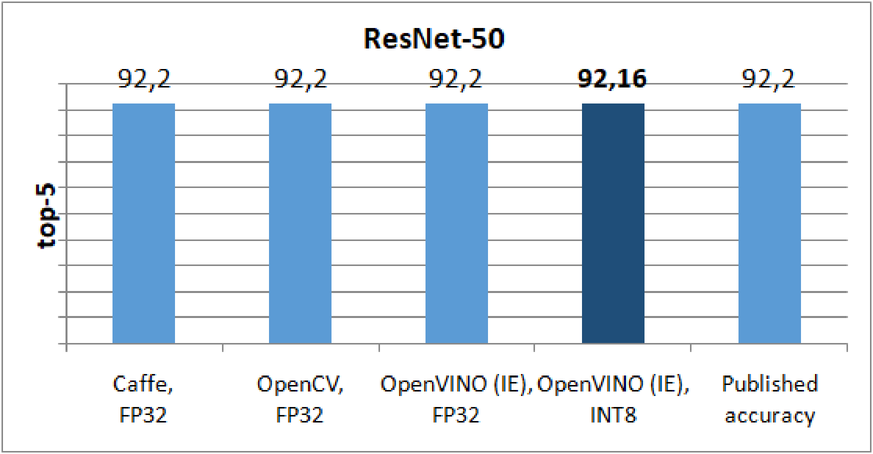




INT8 Quantization. What about accuracy?

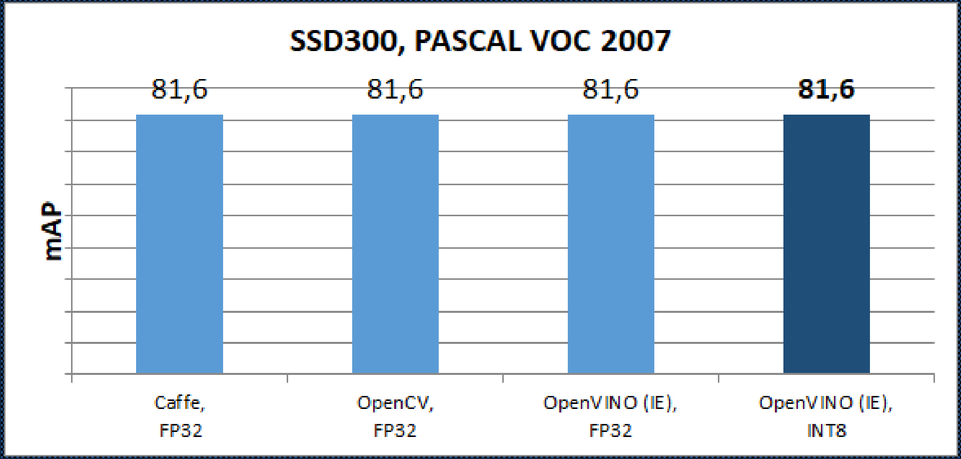
- ImageNet, validation dataset
- Classification error: top-1, top-5
- **Result**: accuracy is almost the same with good agreement with the current record

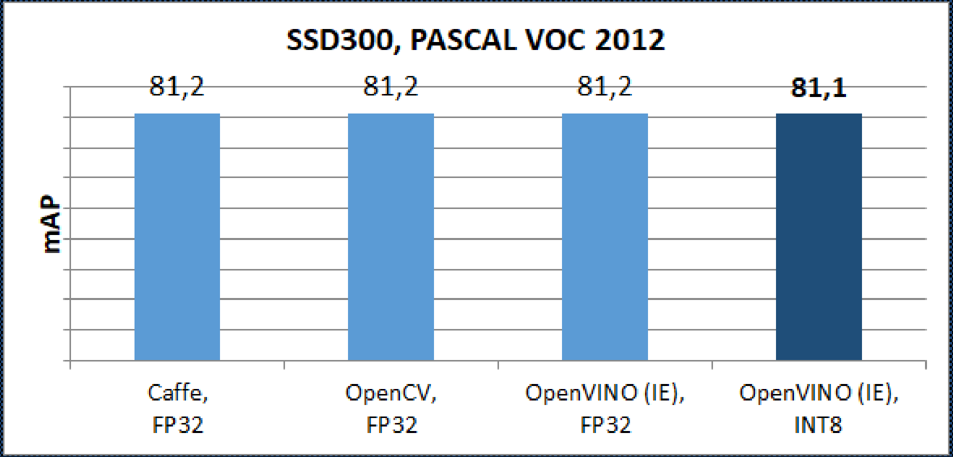




INT8 Quantization. What about accuracy?

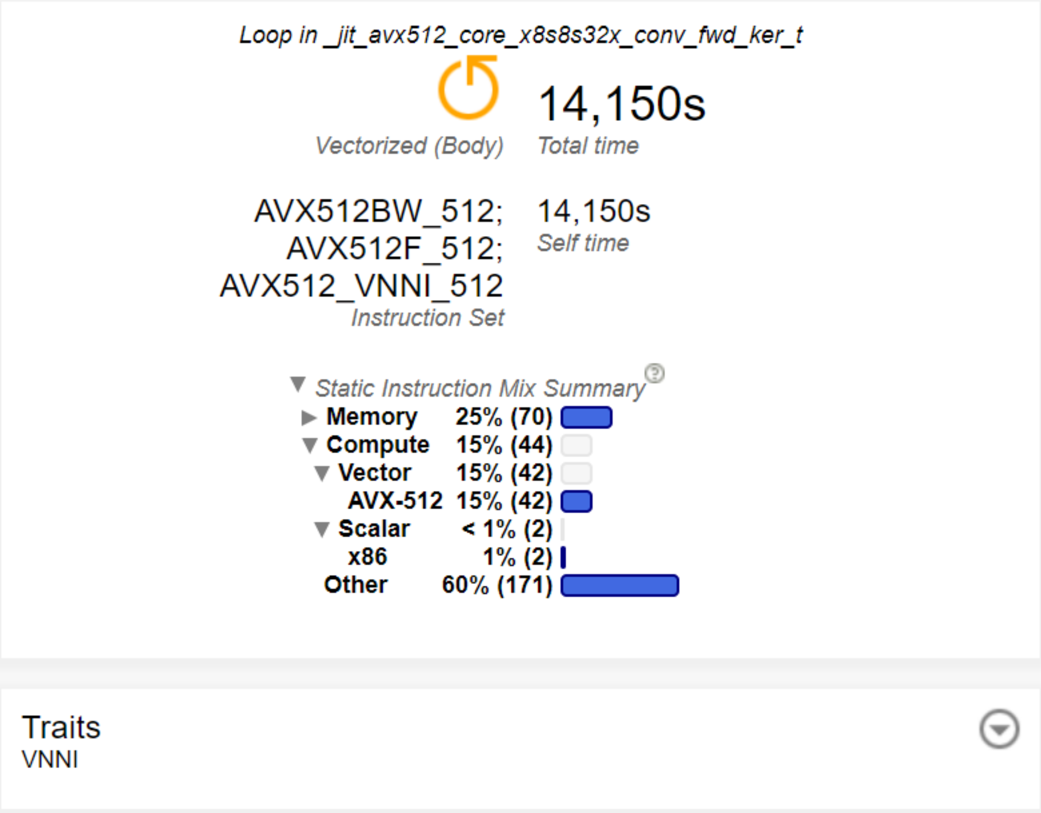
- PASCAL VOC 2007, test dataset (publicly available)
- PASCAL VOC 2012, validation dataset
- Object detection accuracy: mean average precision (mAP)
- **Result**: accuracy is almost the same





Use Intel Advisor

- **Intel Advisor**: Very useful and insightful tool to understand, discuss and overcome performance problems
- Intel Advisor tells about the reason of performance improvement



Loop in _jit_avx512_core_x8s8s32x_conv_fwd_ker_t

Vectorized (Body)

14,150s
Total time

AVX512BW_512;
AVX512F_512;
AVX512_VNNI_512
Instruction Set

14,150s
Self time

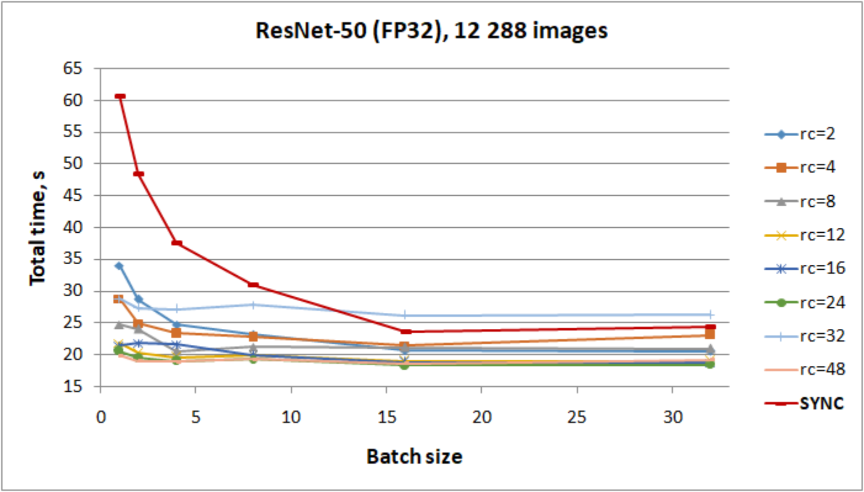
▼ *Static Instruction Mix Summary*

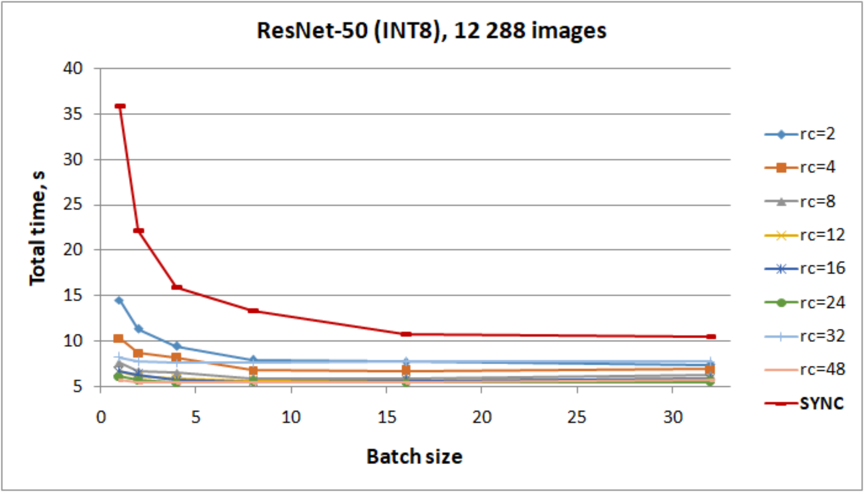
▶ **Memory**	**25% (70)**
▼ **Compute**	**15% (44)**
▼ **Vector**	**15% (42)**
AVX-512	**15% (42)**
▼ **Scalar**	**< 1% (2)**
x86	**1% (2)**
Other	**60% (171)**

Traits
VNNI

Asynchronous mode. ResNet-50

- Maximizes throughput (and as rule minimizes total time)
- **Parameter**: a queue size to collect batches (**Requests Count, rc**)*.
- 48 threads
- **Observation**: **~2x** speedup vs. Sync. mode in terms of total time

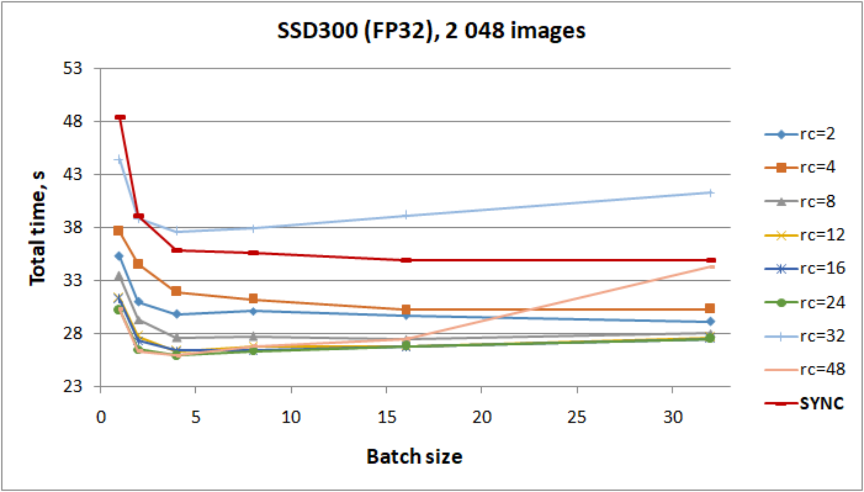


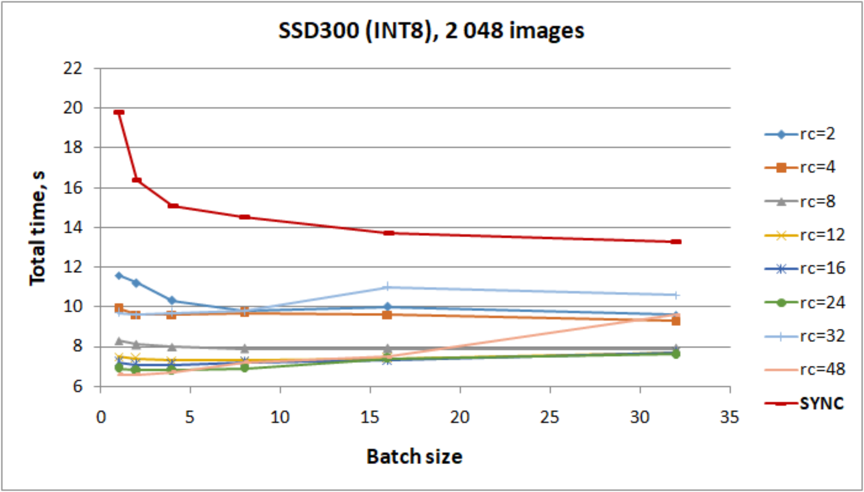


* Streams Count = Requests Count in all the experiments

Asynchronous mode. SSD300

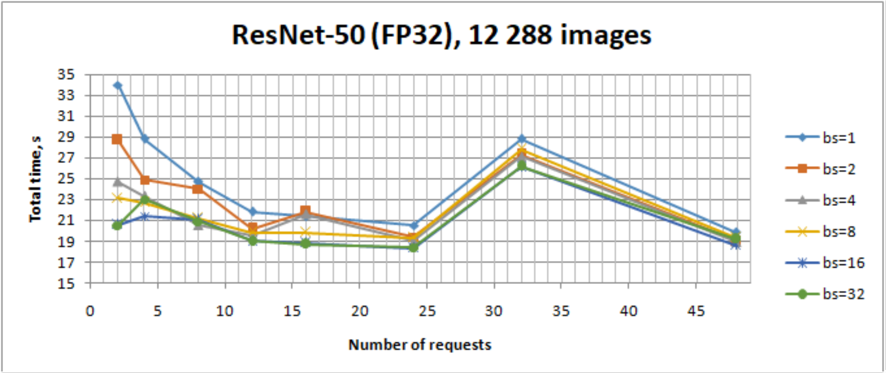
- Minimizes throughput (and as rule total time)
- **Parameter**: a queue size to collect batches (**Requests Count, rc**).
- 48 threads
- **Observation**: **~1.5x** speedup vs. Sync. mode in terms of total time



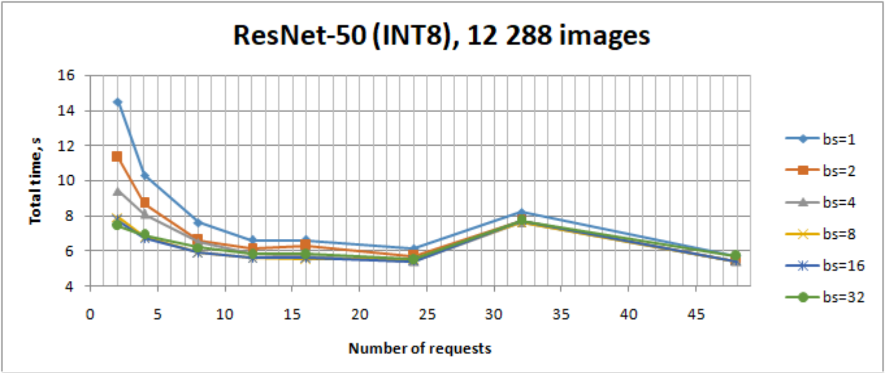


Asynchronous mode. Requests Count

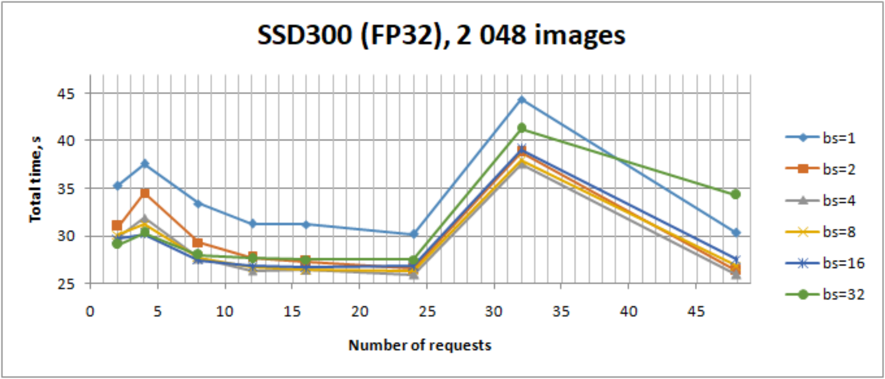
- How to choose Requests count?



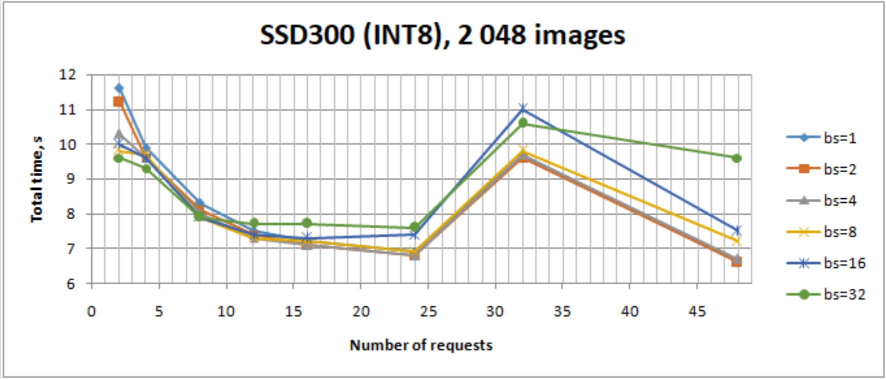
ResNet-50 (FP32), 12 288 images



ResNet-50 (INT8), 12 288 images



SSD300 (FP32), 2 048 images



SSD300 (INT8), 2 048 images

Asynchronous mode. Request Count

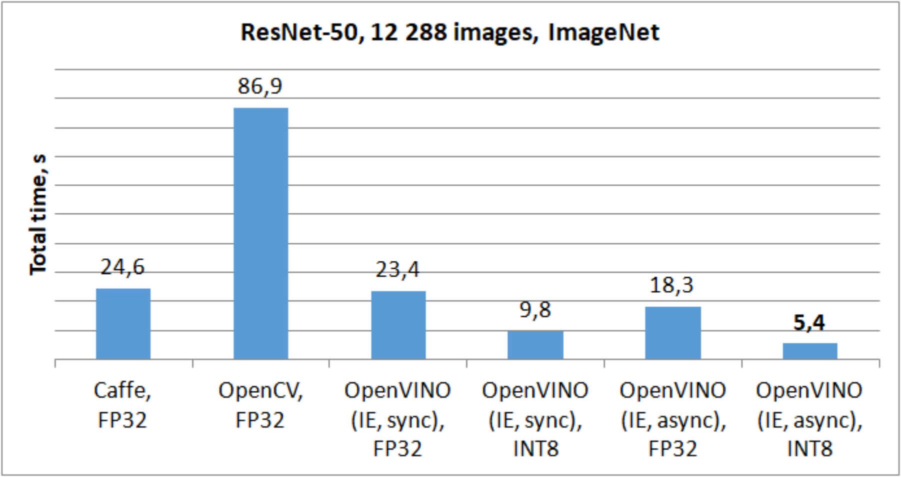
- How to choose Request count?

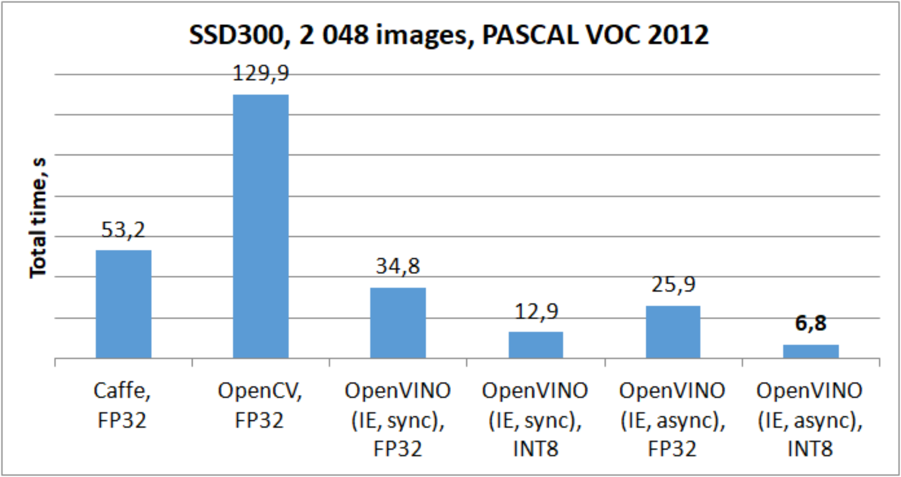
- **Reasonable choice**: RC is equal to the number of cores

Performance Comparison

- **The best values of all the parameters are used**

- **Observation**: OpenVINO IE in asynchronous mode outperforms other frameworks





Conclusions

- All frameworks perform reasonably in DL reference on two model on two 24-cores Cascade Lake CPUs but OpenVINO is better in terms of FPS and Total time
- OpenVINO scales well up to at least 48 cores
- The async. mode in OpenVINO results in ~2x perf. Improvement
- The choice of parameters values is crucial. We recommend to use default settings, find the batch size empirically and set RC (and SC) to the number of cores in asynchronous mode
- INT8 calibration greatly improves performance with almost the same accuracy
- Use Intel Advisor to understand performance of your Application

Contacts

Dr. Iosif Meyerov

Vice-head of the Mathematical Software and Supercomputing Technologies department,

Lobachevsky State University of Nizhni Novgorod

meerov@vmk.unn.ru