



ACCELERATE DEEP LEARNING INFERENCE USING INTEL TECHNOLOGIES

OPTIMIZATION: TOOLS AND TECHNIQUES

August 2019

Optimization Notice

Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness or any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice. Notice Revision #20110804.

Legal Notices and Disclaimers (1 of 2)

Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software, or service activation. Performance varies depending on system configuration. No computer system can be absolutely secure. Check with your system manufacturer or retailer or learn more at www.intel.com.

Performance estimates were obtained prior to implementation of recent software patches and firmware updates intended to address exploits referred to as "Spectre" and "Meltdown." Implementation of these updates may make these results inapplicable to your device or system.

Cost reduction scenarios described are intended as examples of how a given Intel-based product, in the specified circumstances and configurations, may affect future costs and provide cost savings. Circumstances will vary. Intel does not guarantee any costs or cost reduction.

This document contains information on products, services, and/or processes in development. All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest forecast, schedule, specifications, and roadmaps.

Any forecasts of goods and services needed for Intel's operations are provided for discussion purposes only. Intel will have no liability to make any purchase in connection with forecasts published in this document.

Arduino*101 and the Arduino* infinity logo are trademarks or registered trademarks of Arduino, LLC.

Altera, Arria, the Arria logo, Intel, the Intel logo, Intel Atom, Intel Core, Intel Nervana, Intel Xeon Phi, Movidius, Saffron, and Xeon are trademarks of Intel Corporation or its subsidiaries in the United States and other countries.

*Other names and brands may be claimed as the property of others.

Copyright © 2018, Intel Corporation. All rights reserved.

Legal Notices and Disclaimers (2 of 2)

This document contains information on products, services, and/or processes in development. All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest forecast, schedule, specifications, and roadmaps. Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software, or service activation. Learn more at [intel.com](https://www.intel.com) or from the OEM or retailer. No computer system can be absolutely secure.

Tests document performance of components on a particular test, in specific systems. Differences in hardware, software, or configuration will affect actual performance. Consult other sources of information to evaluate performance as you consider your purchase. For more complete information about performance and benchmark results, visit www.intel.com/performance.

Cost reduction scenarios described are intended as examples of how a given Intel-based product, in the specified circumstances and configurations, may affect future costs and provide cost savings. Circumstances will vary. Intel does not guarantee any costs or cost reduction.

Statements in this document that refer to Intel's plans and expectations for the quarter, the year, and the future, are forward-looking statements that involve a number of risks and uncertainties. A detailed discussion of the factors that could affect Intel's results and plans is included in Intel's SEC filings, including the annual report on Form 10-K.

The products described 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. Performance estimates were obtained prior to implementation of recent software patches and firmware updates intended to address exploits referred to as "Spectre" and "Meltdown." Implementation of these updates may make these results inapplicable to your device or system.

No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document. Intel does not control or audit third-party benchmark data or the web sites referenced in this document. You should visit the referenced web site and confirm whether referenced data are accurate.

Intel, the Intel logo, Pentium, Celeron, Atom, Core, Xeon, Movidius, Saffron, and others are trademarks of Intel Corporation in the United States and other countries.

*Other names and brands may be claimed as the property of others.

Copyright © 2018, Intel Corporation. All rights reserved.

Smart Video Workshop Overview

Introduction

1. Introduction to Intel technologies for deep learning inference
2. Hardware acceleration techniques

Each module contains a hands-on lab exercise that introduces various Intel technologies to accelerate computer vision application with hardware heterogeneity.

Intel® Distribution of OpenVINO™ 101

Hardware Acceleration

2. Basic End-to-End Object Detection Example

3./4./5. Hardware Acceleration with CPU, Integrated GPU, Intel® Movidius™ NCS, FPGA

Optimization

6. Optimization Tools and Techniques

Application

7. Advanced Video Analytics

Optimizing Computer Vision Applications

- Use OpenVINO:
 - Model Optimizer:
 - Use the right data type(FP16 or 32) for your target hardware and accuracy needs
 - Inference Engine:
 - Performance difference between using the inference engine (IE) and a non-optimized framework can be bigger than differences between accelerators
 - Set batch size
 - Use Async API while capturing video at the same time as doing the inference
- Other Recommendations:
 - Use/train a model with the right performance/accuracy tradeoffs. Performance differences between models can be bigger than any optimization you can do at the inference app level
 - Don't infer every frame if not needed
 - Use Intel® VTune™ Analyzer



Use the Correct Data Type

Use the Correct Data Type

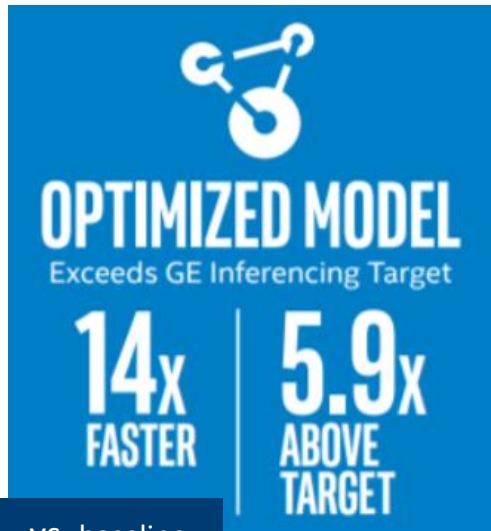
	FP32	FP16
CPU	yes	no
GPU	yes	recommended
Intel® Movidius™ Myriad™	no	yes
FPGA/DLA	no	yes

FPGA/DLA also supports FP11.



Use Inference Engine

Use an Optimized Inference Engine vs. a non-optimized framework on Intel® CPU



vs. baseline



The Deep Learning Deployment Toolkit from Intel helps deliver optimized inferencing on Intel® architecture, helping bring the power of AI to clinical diagnostic scanning and other healthcare workflows

https://ai.intel.com/wp-content/uploads/sites/53/2018/03/IntelSWDevTools_OptimizeDLforHealthcare.pdf

The performance difference between running inference engine vs. unoptimized can be bigger than the difference between accelerators.

Set Batch Size

Set Batch Size

- To increase throughput, a convolutional neural network (CNN) could process the input data in batches.
- What is batch size?
 - If we have a 3-channel(RGB) images that are 256x256 pixels. A single image can be represented as a 3 x 256 x 256 matrix. If we set the batch size to 10, then we are concatenating those 10 images together into a 10 x 3 x 256 x 256 matrix.
- What is the best Batch size? – It depends on the model and available compute power
 - Increase batch size could help to get a higher throughput which means better inference FPS, but if it goes too large, your device will run out of memory to hold it.



Use Async

Inference Engine async API

- Improves overall frame-rate of the application.
- Executes a request asynchronously (in the background) and waits until ready, when the result is actually needed.
- Continues doing things on the host, while the accelerator is busy.
- The demo keeps two parallel infer requests, and, while the current is processed, the input frame for the next is captured. This essentially hides the latency of capturing, so the overall framerate is determined by the `MAXIMUM(detection time, input capturing time)` and not the `SUM(detection time, input capturing time)`.

Running Object Detection Sample SSD async

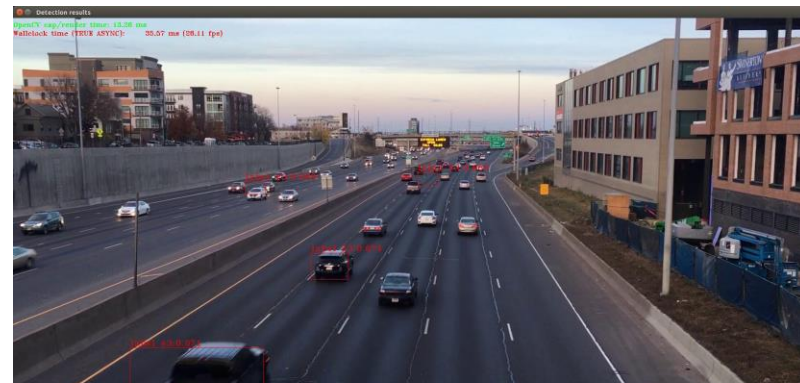
```
$ ./object_detection_demo_ssd_async -i /home/intel/workshopApr25/workshop-tutorials/test_content/video/cars_1920x1080.h264 -m /home/intel/workshopApr25/workshop-tutorials/test_content/IR/SSD/SSD_GoogleNet_v2_fp32.xml
```

Synchronous Mode



OpenCV cap/render time: 9.85 ms
Wallclock time (SYNC, press Tab): 42.96 ms (23.28 fps)
Detection time : 32.96 ms (30.34 fps)

Press **Tab** to Enable Asynchronous Mode



OpenCV cap/render time: 13.26 ms
Wallclock time (TRUE ASYNC): 35.57 ms (28.11 fps)

Pick the Right Model

Use/Train a Model with the Right Performance plus Accuracy Tradeoffs.

Performance is based on many factors:

- Topography complexity/layer implementation plus scheduling
- Number of color channels (that is, BGR vs. grayscale)
- Model resolution

Exercise: Range of Model Performance

Focus on the Inference Timing

Complete this table in the hands-on exercise:

	CPU ms/frame	GPU ms/frame	Intel® Movidius™ Myriad™ X ms/frame
ssd512			
ssd300			
Mobilnet-ssd*			



Don't Infer If Not Needed

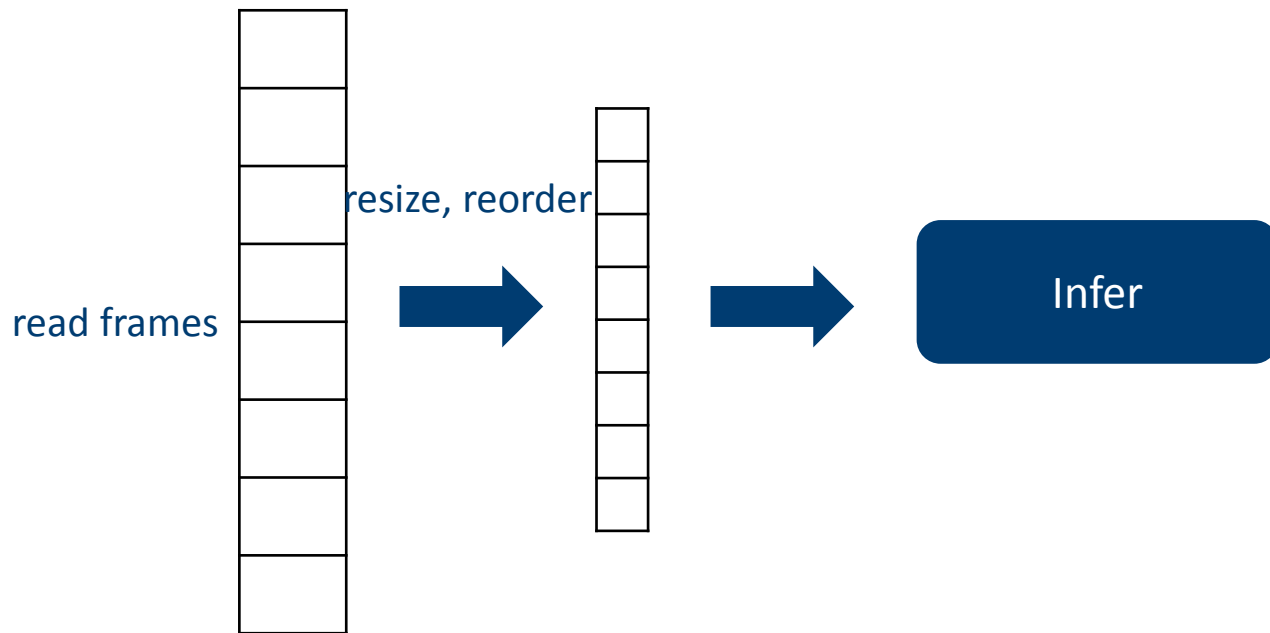
Tracking

Many ways to track:

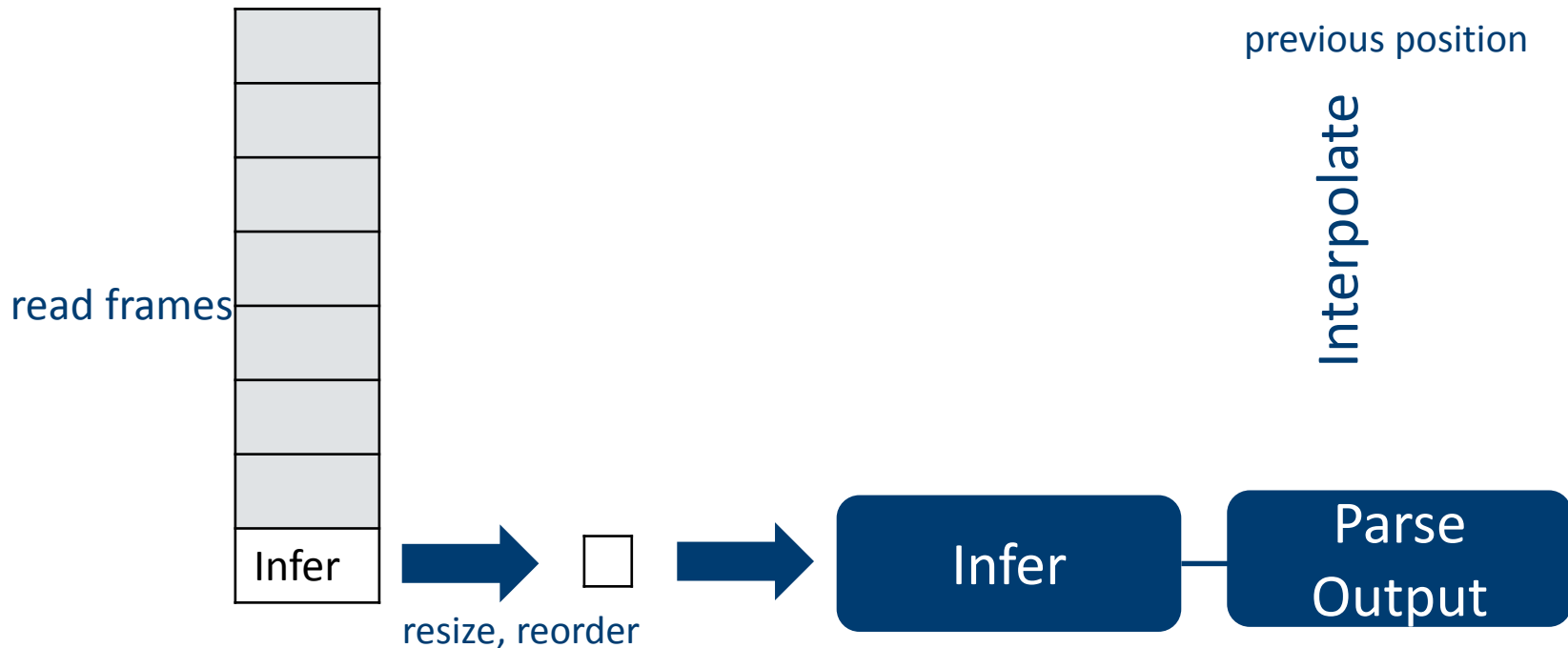
- Optical flow: the pattern of apparent motion of objects, surfaces, and edges in a visual scene caused by the relative motion between an observer and a scene.
- Kalman filters
- Single/simplistic tracking (what we show here)
- Multiple object tracking: like above but find ways to identify each region so rectangles have a persistent identity over time

A Tracking Thought Experiment

Processing in batches already introduces latency



What If We Did Less Inference?



Determine If There Is Nothing to See

Inference is expensive to run each frame. It can save time to not run when there is nothing to identify.

- Check motion vectors
- Frame sizes
- bgsbmg
- SAD

These methods can be several orders of magnitude less expensive than inference. Use techniques to increase the total # of streams a system can watch.



Lab5 - Optimizing Computer Vision Applications

URL: <https://github.com/intel-iot-devkit/smart-video-workshop/blob/master/optimization-tools-and-techniques/README.md>

Objective: This tutorial shows some techniques to get better performance for computer vision applications with the Intel® Distribution of OpenVINO™ toolkit.

Estimated Complete Time: 40min

Use Intel[®] VTune[™] Analyzer

Use Intel® VTune™ Analyzer



Lab6 - Intel® VTune™ Amplifier Tutorial

URL: https://github.com/intel-iot-devkit/smart-video-workshop/blob/master/optimization-tools-and-techniques/README_VTune.md

Objective: This tutorial will show how to run Intel® VTune™ Amplifier on an Intel® Distribution of OpenVINO™ toolkit Inference Engine application.

Estimated Complete Time: 30min

ADVANCED VIDEO ANALYTICS

SECURITY BARRIER DEMO

Video Analytics in Intel® Distribution of OpenVINO™ Toolkit

Topology	Type	Description
<u>license-plate-recognition-barrier-0001</u>	ocr	Chinese license plate recognition.
<u>vehicle-attributes-recognition-barrier-0010</u>	object_attributes	Vehicle attributes recognition with modified RESNET10* backbone.
<u>vehicle-license-plate-detection-barrier-0007</u>	detection	Multiclass (vehicle, license plates) detector based on RESNET10 plus SSD.

vehicle-attributes-recognition-barrier-0010

Use Case/High-Level Description

Vehicle attributes classification algorithm for a traffic analysis scenario.

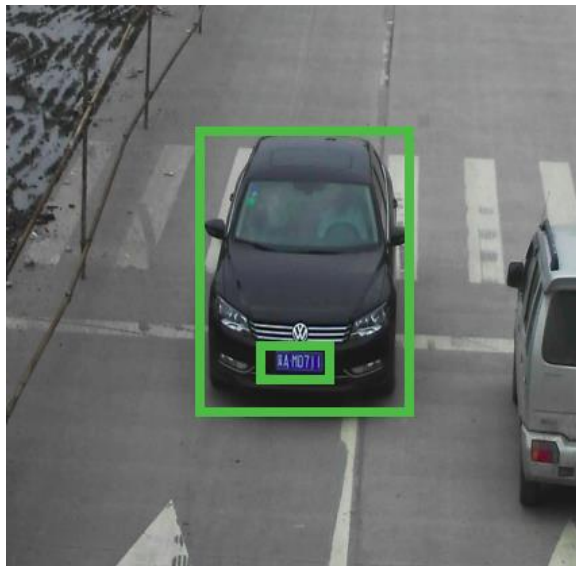


Type: regular
Color: black

vehicle-license-plate-detection-barrier-007

Use Case/High-Level Description

RESNET* 10 plus SSD-based vehicle and (Chinese) license plate detector for "Barrier" use case.



license-plate-recognition-barrier-0001

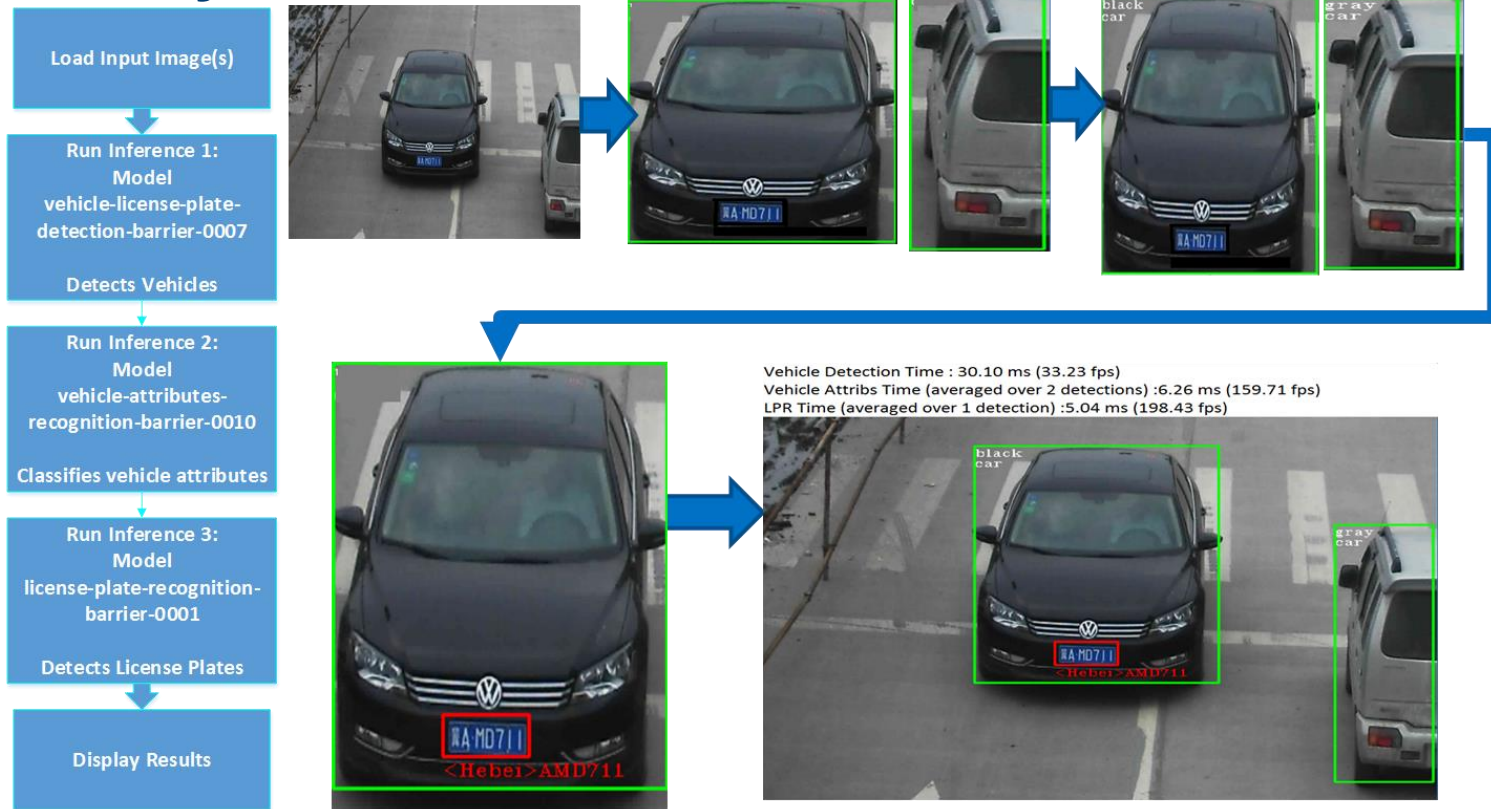
Use Case/High-Level Description

Small-footprint network trained E2E to recognize Chinese license plates in traffic scenarios.

Note: The license plates in the image are modified from the originals.



Security Barrier Demo



Lab7 - Advanced Video Analytics

URL: https://github.com/intel-iot-devkit/smart-video-workshop/blob/master/advanced-video-analytics/multiple_models.md

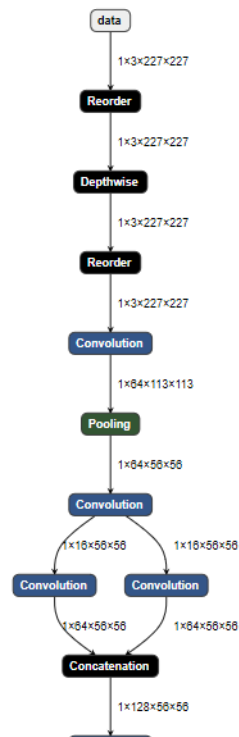
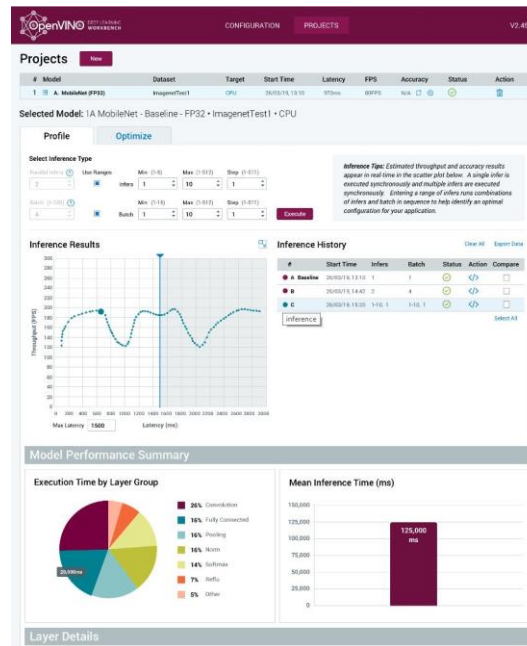
Objective: The tutorial shows some techniques for developing advanced video analytics applications.

Estimated Complete Time: 20min

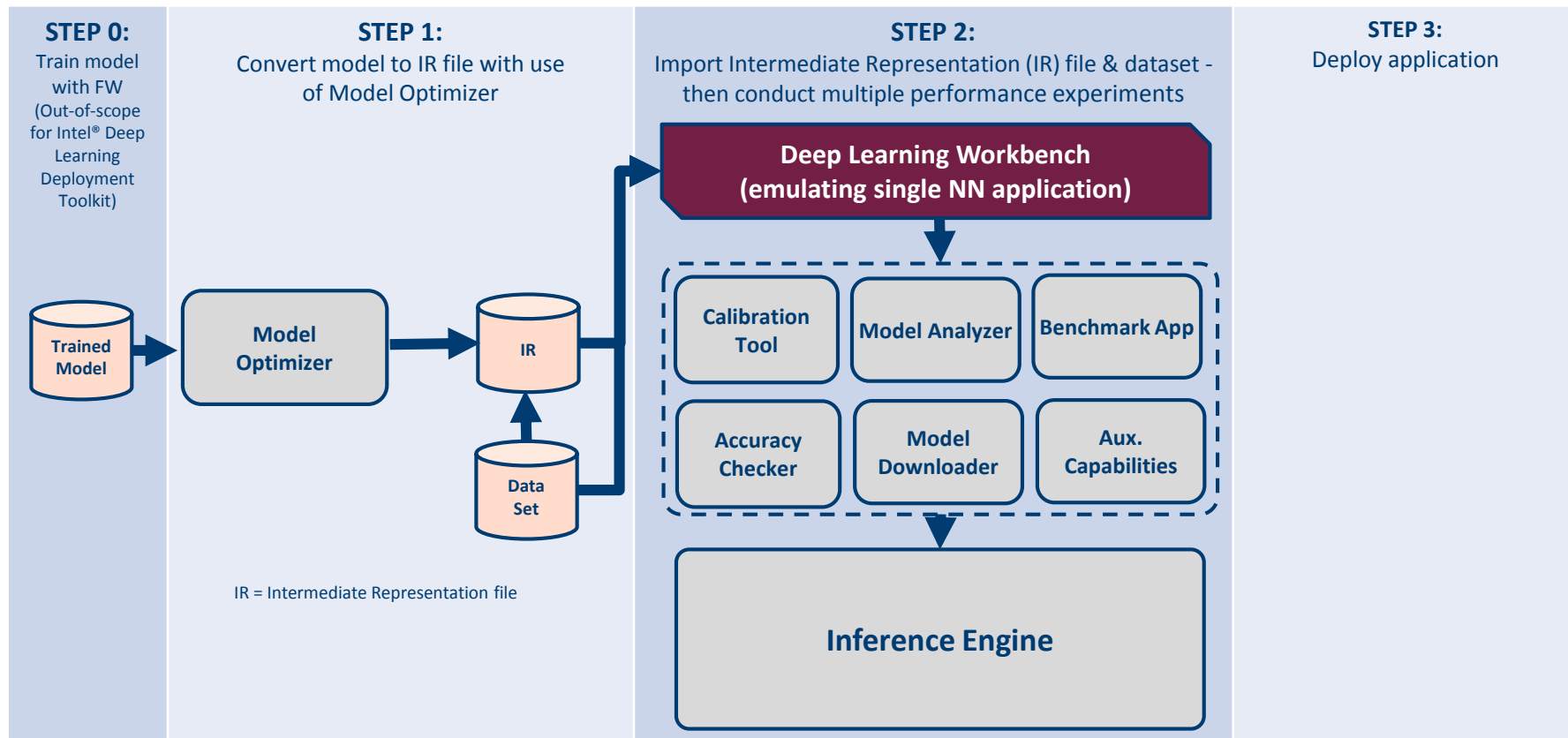
Deep Learning Workbench (Preview)

Deep Learning Workbench capabilities

- Web-based tool - UI extension of Intel® Distribution of OpenVINO™ toolkit functionality
- Visualizes performance data for topologies/ layers to aid in model analysis
- Automate analysis for optimal performance configuration (streams, batches, latency)
- Experiment with int8 calibration for optimal tuning
- Provide accuracy info through accuracy checker
- Direct access to Models from public set of Open Model Zoo



Deep Learning Workbench Data Flow



Workbench interfaces with key components

- **Calibration Tool** – calibrates model in original precision (FP32, FP16) to lower precision (Int8) to reduce memory footprint and increase neural model performance. Implies some model accuracy drop that can be configured before model calibration.
- **Model Analyzer** – provides theoretical data on the models like computational complexity (flops), number of neurons and memory consumption.
- **Benchmark App** – helps to measure performance (throughput and latency) of a model, get performance metrics on per layer and overall basis
- **Accuracy Checker Tool** – Check for accuracy of the model both original and after conversion to IR file using a know data set.
- **Model Downloader** – Provides an easy way of accessing a number of public neural network models as well as a set of pre-trained Intel models
- **Aux Capabilities** - Additional capabilities delivered by the toolset, e.g. ability to optimize model using Winograd algorithmic tuning.

Installation & Distribution

Distributed with Intel Distribution of OpenVINO toolkit

- Build your local docker image from package (build scripts)
- Build your local docker image by copying and running dockerfile from documentation

Download docker image from DockerHub

- Work in progress – will depend on decision of OpenSource PDT
- Can be used for internal experiments (at least)

