INTEL OPENVINO & AZURE IOT EDGE

Roy Allela

Software Technical Consulting Engineer

Workshop Agenda

- Intel OpenVINO Overview:
    - Model Optimizer
    - Inference Engine
    - Pretrained Models & Demos
    - INT8 Calibration

- OpenVINO Release 2020.1 New Features

- OpenVINO & Azure IoTEdge Workflow

- OpenVINO & Azure IoTEdge Demo

INTEL® DISTRIBUTION OF OPENVINO™ TOOLKIT

Take your computer vision solutions to a new level with deep learning inference intelligence.

What it is

A toolkit to accelerate **high performance computer vision** & **deep learning inference into vision/AI applications** used from edge to cloud. It enables deep learning on hardware accelerators and easy deployment across multiple types of Intel® platforms.

Who needs this product?

- Computer vision, deep learning software developers
- Data scientists
- OEMs, ISVs, System Integrators

Usages

Security surveillance, robotics, retail, healthcare, AI, office automation, transportation, non-vision use cases (speech, text) & more.

HIGH PERFORMANCE, PERFORM AI AT THE EDGE

STREAMLINED & OPTIMIZED DEEP LEARNING INFERENCE

HETEROGENEOUS, CROSS-PLATFORM FLEXIBILITY

Free Download ▸ software.intel.com/openvino-toolkit

Open Source version ▸ 01.org/openvinotoolkit

Optimization Notice

Intel Computer Vision/AI Portfolio

EXPERIENCES

TOOLS

Intel® Parallel Studio XE
Intel® System Studio
Intel® Media SDK

Intel® Distribution of OpenVINO™ toolkit
Intel® SDK for OpenCL™ Applications
Nauta

FRAMEWORKS

Apache Spark MLlib BigDL

TensorFlow

mxnet

Caffe

torch

KALDI

ONNX

LIBRARIES

Intel® Data Analytics Acceleration Library

Intel® Distribution for Python*

Intel® Math Kernel Library

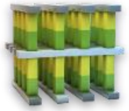
Intel® nGraph™ Compiler

Movidius Stack

HARDWARE

intel CELERON inside
intel ATOM inside
intel CORE i7 inside
intel XEON inside
intel ARRIA 10 inside
intel MOVIDIUS inside

Compute



Memory & Storage

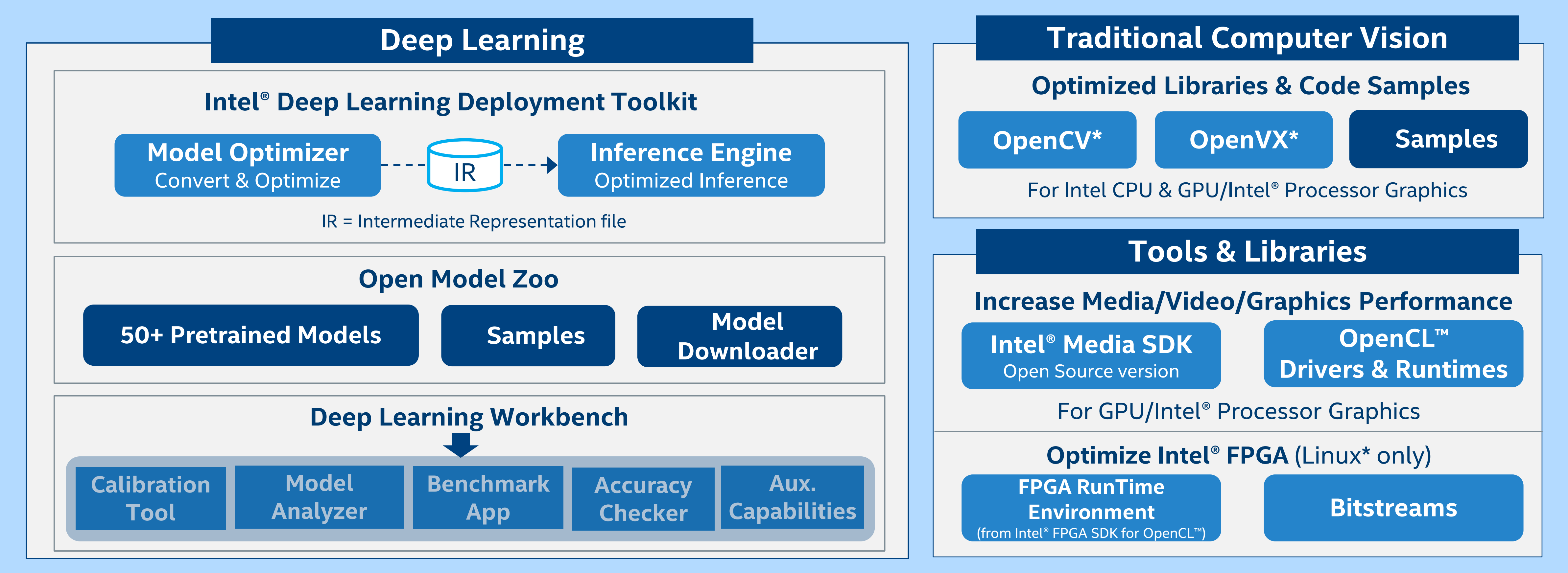
Networking

intel REALSENSE TECHNOLOGY
intel Movidius™

Visual Intelligence

UNLEASH FULL POTENTIAL

What's Inside Intel® Distribution of OpenVINO™ toolkit



Deep Learning

Intel® Deep Learning Deployment Toolkit

Model Optimizer Convert & Optimize → IR → **Inference Engine** Optimized Inference

IR = Intermediate Representation file

Open Model Zoo

- 50+ Pretrained Models
- Samples
- 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)

Intel® Architecture-Based Platforms Support

intel CELERON inside · intel ATOM inside · intel CORE inside · intel XEON inside · intel ARRIA 10 inside · intel MOVIDIUS inside · intel IRIS Pro GRAPHICS

Intel® Vision Accelerator Design Products & AI in Production/ Developer Kits

An open source version is available at 01.org/openvinotoolkit (deep learning functions support for Intel CPU/GPU/NCS/GNA).

Optimization Notice

Intel® Deep Learning Deployment Toolkit

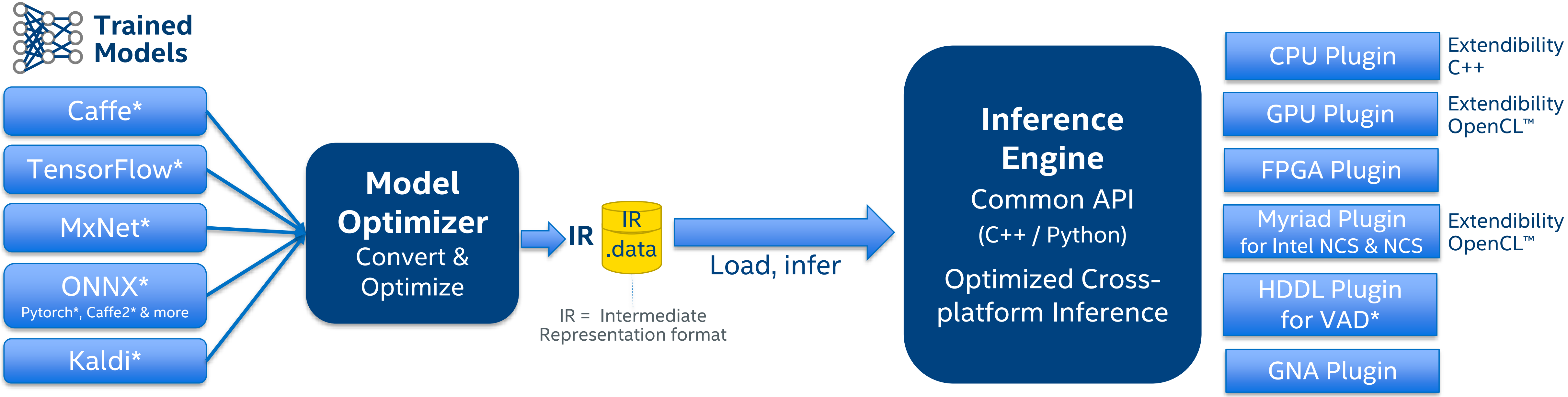
For Deep Learning Inference

Model Optimizer

- **What it is:** A Python*-based tool to import trained models and convert them to Intermediate representation.
- **Why important:** Optimizes for performance/space with conservative topology transformations; biggest boost is from conversion to data types matching hardware.

Inference Engine

- **What it is:** High-level inference API
- **Why important:** Interface is implemented as dynamically loaded plugins for each hardware type. Delivers best performance for each type without requiring users to implement and maintain multiple code pathways.



Trained Models

Caffe*

TensorFlow*

MxNet*

ONNX*
Pytorch*, Caffe2* & more

Kaldi*

Model Optimizer
Convert & Optimize

IR

IR .data

IR = Intermediate Representation format

Load, infer

Inference Engine
Common API
(C++ / Python)
Optimized Cross-platform Inference

CPU Plugin — Extendibility C++

GPU Plugin — Extendibility OpenCL™

FPGA Plugin

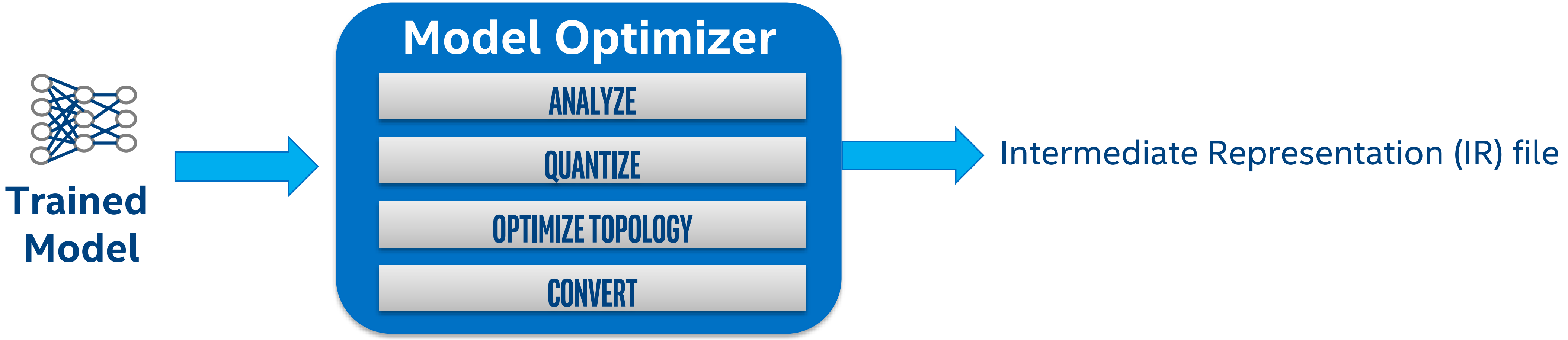
Myriad Plugin for Intel NCS & NCS — Extendibility OpenCL™

HDDL Plugin for VAD*

GNA Plugin

GPU = Intel CPU with integrated GPU/Intel® Processor Graphics, Intel® NCS = Intel® Neural Compute Stick (VPU)
*VAD = Intel® Vision Accelerator Design Products (HDDL-R)

Improve Performance with Model Optimizer

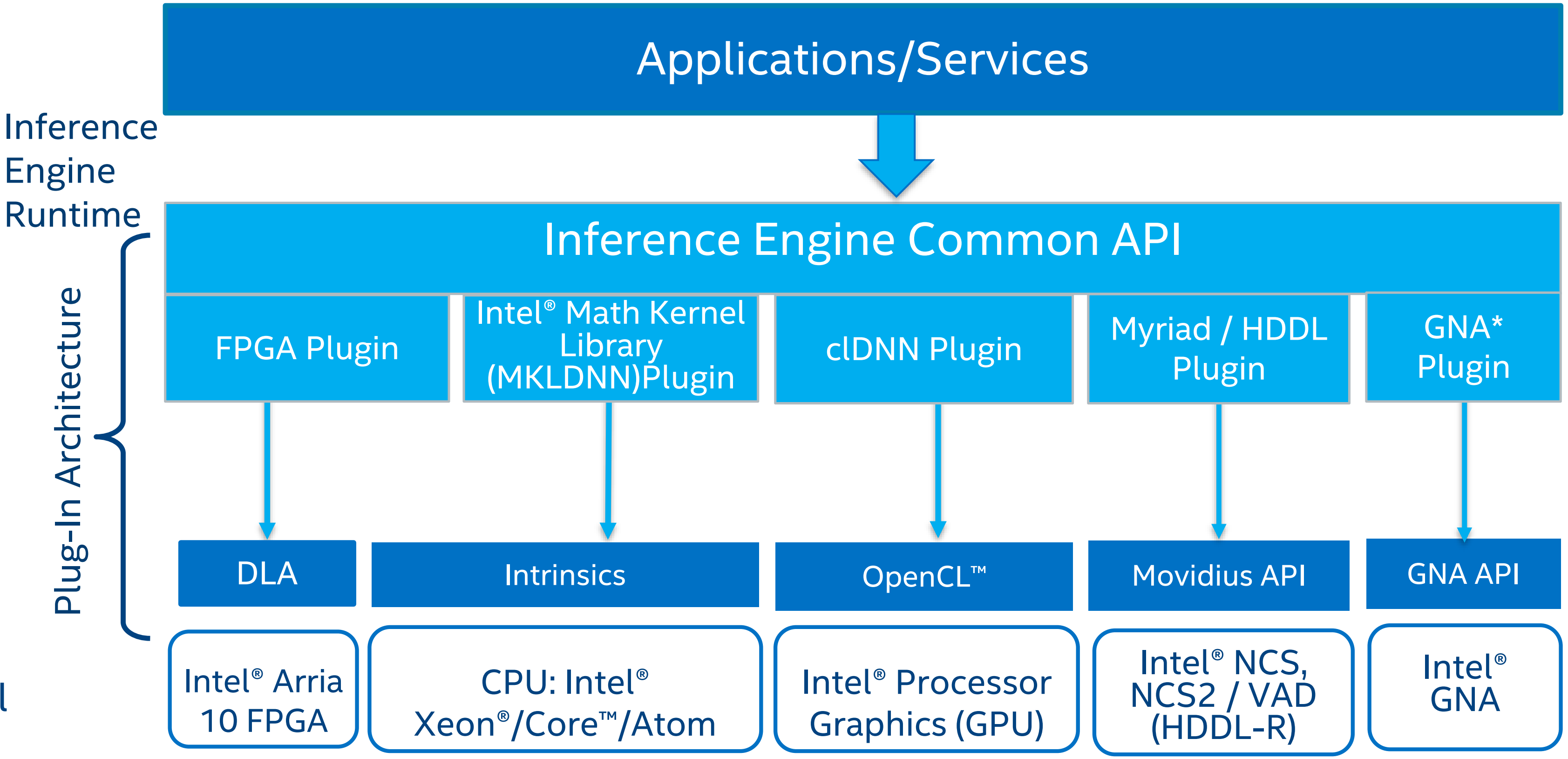


- Easy to use, Python*-based workflow does not require rebuilding frameworks.
- Import Models from many supported frameworks: Caffe*, TensorFlow*, MXNet*, Kaldi*, exchange formats like ONNX* (Pytorch*, Caffe2* and others through ONNX).
- 100+ models for Caffe, MXNet, TensorFlow validated. Supports all ONNX* model zoo public models.
- Extends inferencing for non-vision networks with support of LSTM, Bert, GNMT, TDNN-LSTM, ESPNet and more.
- IR files for models using standard layers or user-provided custom layers do not require Caffe.
- Fallback to original framework is possible in cases of unsupported layers, but requires original framework.

Optimization Notice

Optimal Model Performance Using the Inference Engine

- Simple & unified API for inference across all Intel® architecture
- Optimized inference on large IA hardware targets (CPU/GEN/FPGA)
- Heterogeneity support allows execution of layers across hardware types
- Asynchronous execution improves performance
- Futureproof/scale your development for future Intel® processors
- Supports serialized FP16 IR across all plugins / platforms (CPU inference remains at FP32)



Applications/Services

Inference Engine Runtime

Inference Engine Common API

Plug-In Architecture

FPGA Plugin	Intel® Math Kernel Library (MKLDNN)Plugin	clDNN Plugin	Myriad / HDDL Plugin	GNA* Plugin
DLA	Intrinsics	OpenCL™	Movidius API	GNA API
Intel® Arria 10 FPGA	CPU: Intel® Xeon®/Core™/Atom	Intel® Processor Graphics (GPU)	Intel® NCS, NCS2 / VAD (HDDL-R)	Intel® GNA

Transform Models & Data into Results & Intelligence

GPU = Intel CPU with integrated graphics/Intel® Processor Graphics/GEN
GNA = Gaussian mixture model and Neural Network Accelerator

Optimization Notice

Speed Deployment with Pretrained Models & Samples

Expedite development, accelerate deep learning inference performance, speed production deployment

Pretrained Models in Intel® Distribution of OpenVINO™ toolkit

- Age & Gender
- Face Detection–standard & enhanced
- Head Position
- Human Detection–eye-level & high-angle detection
- Detect People, Vehicles & Bikes
- License Plate Detection: small & front facing
- Vehicle Metadata
- Human Pose Estimation
- Action recognition–encoder & decoder
- Text Detection & Recognition
- Vehicle Detection
- Retail Environment
- Pedestrian Detection
- Pedestrian & Vehicle Detection
- Person Attributes Recognition Crossroad
- Emotion Recognition
- Identify Someone from Different Videos–standard & enhanced
- Facial Landmarks
- Gaze estimation
- Identify Roadside objects
- Advanced Roadside Identification
- Person Detection & Action Recognition
- Person Re-identification–ultra small/ultra fast
- Face Re-identification
- Landmarks Regression
- Smart Classroom Use Cases
- Super Resolution
- Instance segmentation
- Image retrieval
- & more…

Binary Models

- Face Detection Binary
- Pedestrian Detection Binary
- Vehicle Detection Binary
- ResNet50 Binary

Optimization Notice

Speed Deployment with Demos

Expedite development, accelerate deep learning inference performance, speed production deployment

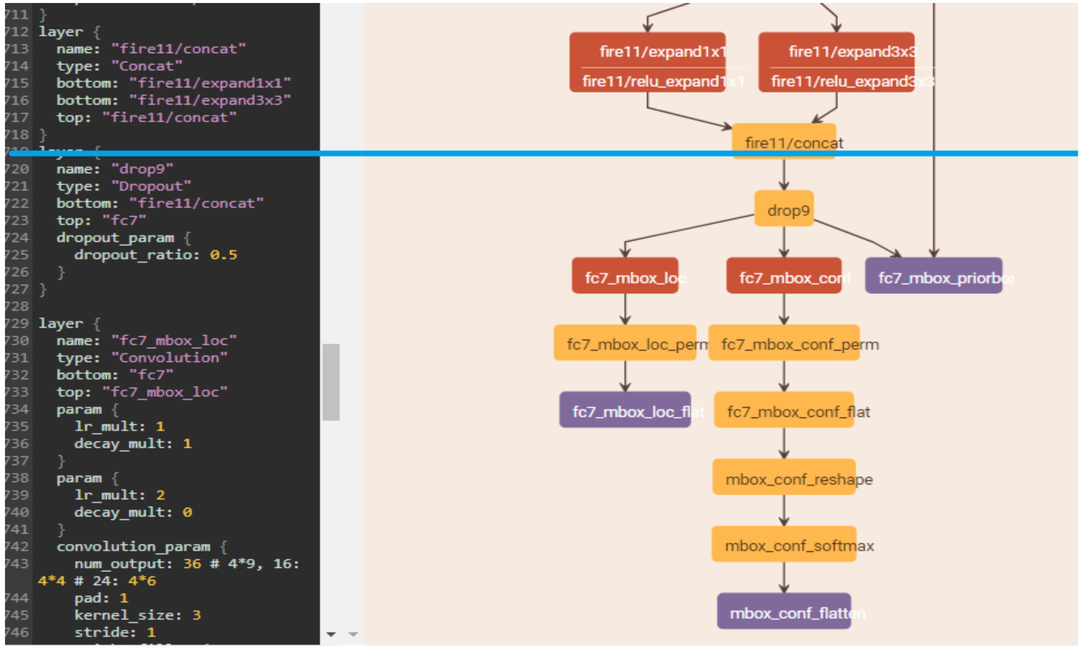
Some Available Demos in Intel® Distribution of OpenVINO™ toolkit

- Gaze/Pose Estimation
- Action Recognition
- Crossroad Camera
- Gaze Estimation
- Image Segmentation
- 3D Segmentation
- Instance Segmentation
- Interactive Face Recognition

- Text Detection & Recognition
- Mask R-CNN Object Detection
- Object Detection for Faster R-CNN
- Object Detection for SSD
- Object Detection for YOLO
- Super Resolution
- Text Detection
- Image Retrieval

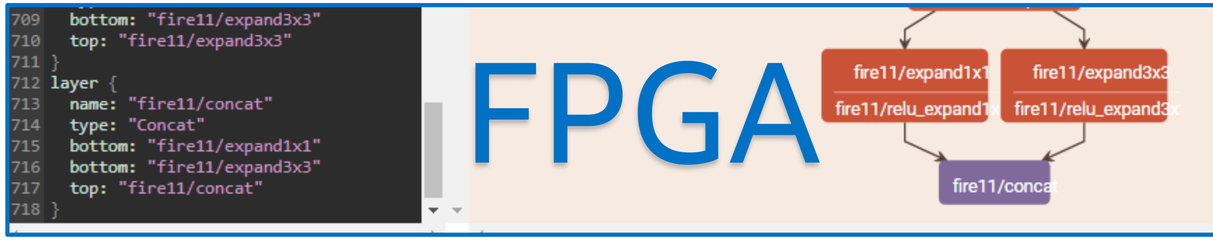
Heterogeneous support

- Possibility to execute different layers on different HW units

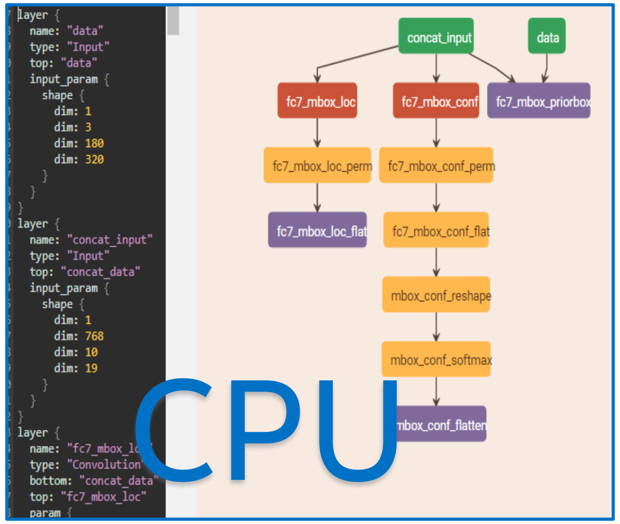


```
}
layer {
  name: "fire11/concat"
  type: "Concat"
  bottom: "fire11/expand1x1"
  bottom: "fire11/expand3x3"
  top: "fire11/concat"
}
layer {
  name: "drop9"
  type: "Dropout"
  bottom: "fire11/concat"
  top: "fc7"
  dropout_param {
    dropout_ratio: 0.5
  }
}

layer {
  name: "fc7_mbox_loc"
  type: "Convolution"
  bottom: "fc7"
  top: "fc7_mbox_loc"
  param {
    lr_mult: 1
    decay_mult: 1
  }
  param {
    lr_mult: 2
    decay_mult: 0
  }
  convolution_param {
    num_output: 36 # 4*9, 16:
4*4 # 24: 4*6
    pad: 1
    kernel_size: 3
    stride: 1
```



```
  bottom: "fire11/expand3x3"
  top: "fire11/expand3x3"
}
layer {
  name: "fire11/concat"
  type: "Concat"
  bottom: "fire11/expand1x1"
  bottom: "fire11/expand3x3"
  top: "fire11/concat"
}
```



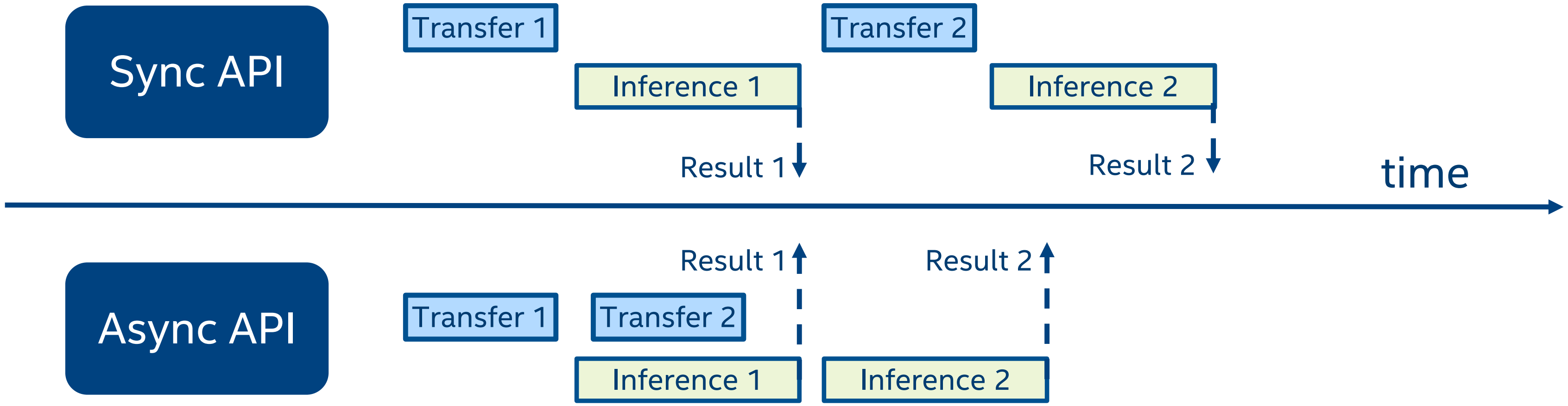
```
layer {
  name: "data"
  type: "Input"
  top: "data"
  input_param {
    shape {
      dim: 1
      dim: 3
      dim: 180
      dim: 320
    }
  }
}
layer {
  name: "concat_input"
  type: "Input"
  top: "concat_data"
  input_param {
    shape {
      dim: 1
      dim: 768
      dim: 10
      dim: 19
    }
  }
}
layer {
  name: "fc7_mbox_loc"
  type: "Convolution"
  bottom: "concat_data"
  top: "fc7_mbox_loc"
  param {
```

Optimization Notice

Synchronous vs Asynchronous execution

In IE API model executes by **Infer request** which can be:

- **Synchronous -** blocks until inference is completed.
- **Asynchronous –** checks the execution status with the wait, or specify a completion callback *(recommended way)*.



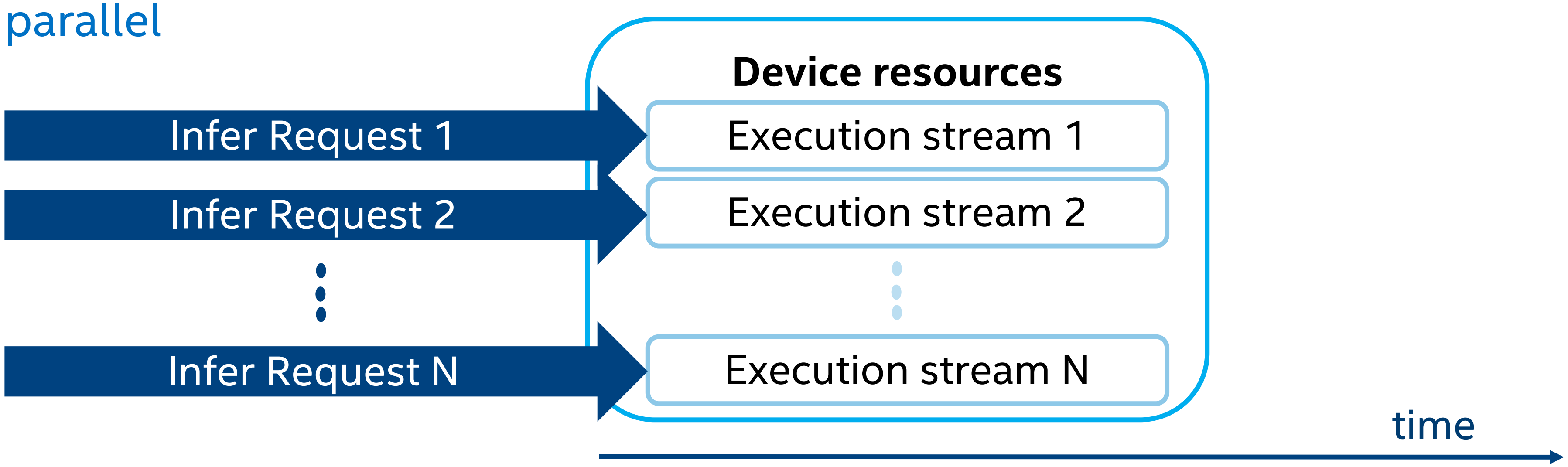
Inference Engine “Throughput” mode for CPU and iGPU

Latency – inference time of 1 frame (ms).

Throughput – overall amount of frames inferred per 1 second (FPS)

“Throughput" mode allows the Inference Engine to efficiently run multiple infer requests simultaneously, greatly improving the overall throughput.

Device resources are divided into execution “**streams**” – parts which runs infer requests in parallel



Inference Engine Multi-Device Support

Automatic load-balancing between devices (inference requests level)

Fully general machinery: any combinations of devices

- CPU+iGPU
- Multiple NCS2, etc

As easy as "`-d MULTI:HDDL,GPU`" for cmd-line option of your favorite sample

C++ example (Python is similar)

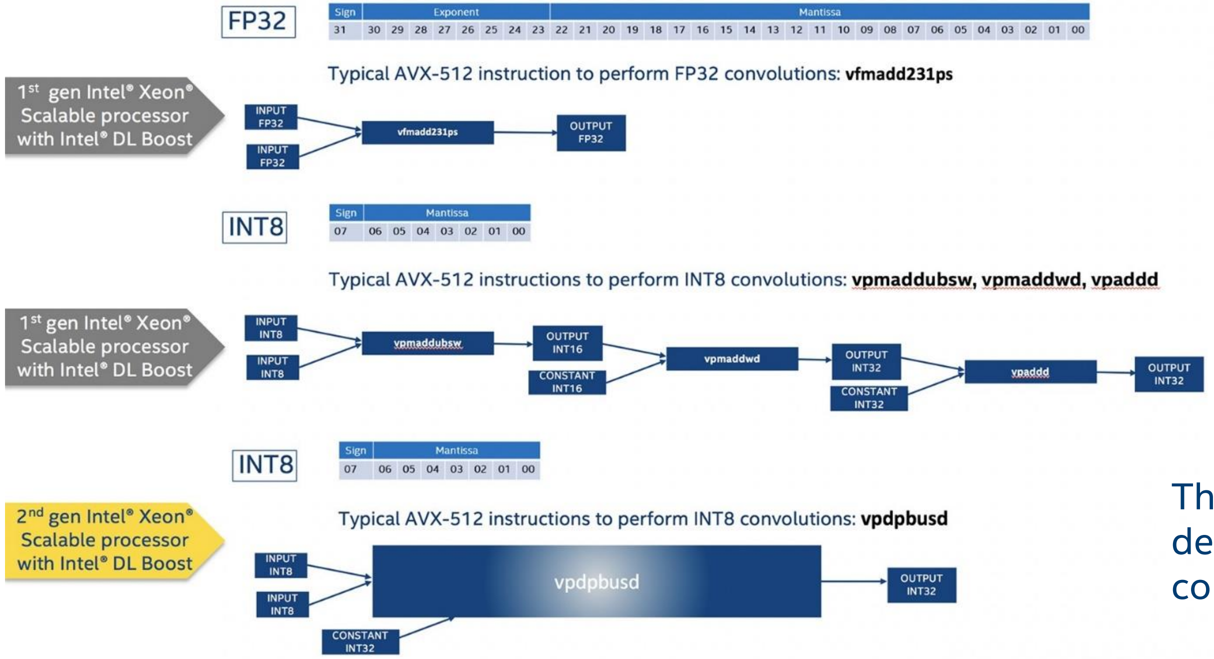
```
// New IE-centric API
    Core ie;
    ExecutableNetwork exec = ie.LoadNetwork(network,{{"DEVICE_PRIORITIES", "HDDL,GPU"}}, "MULTI");
    // Old plugin-centric API
    auto plugin =  PluginDispatcher().getPluginByDevice("MULTI:CPU,GPU");
    ExecutableNetwork executable_network = plugin.LoadNetwork(network, config);
```

Optimization Notice

Int8 support on CPU – why important?

Significant performance boost and little loss of accuracy because

- Benefit from less data size on Intel® platforms with Intel® AVX-512, Intel® AVX2, Intel® SSE4.2
- Take advantage from VNNI (Vector Neural Network Instructions) on 2nd Generation Intel® Xeon® Scalable



FP32

Sign	Exponent								Mantissa																						
31	30	29	28	27	26	25	24	23	22	21	20	19	18	17	16	15	14	13	12	11	10	09	08	07	06	05	04	03	02	01	00

Typical AVX-512 instruction to perform FP32 convolutions: **vfmadd231ps**

1st gen Intel® Xeon® Scalable processor with Intel® DL Boost
INPUT FP32
INPUT FP32
vfmadd231ps
OUTPUT FP32

INT8

Sign	Mantissa						
07	06	05	04	03	02	01	00

Typical AVX-512 instructions to perform INT8 convolutions: **vpmaddubsw, vpmaddwd, vpaddd**

INT8

Sign	Mantissa						
07	06	05	04	03	02	01	00

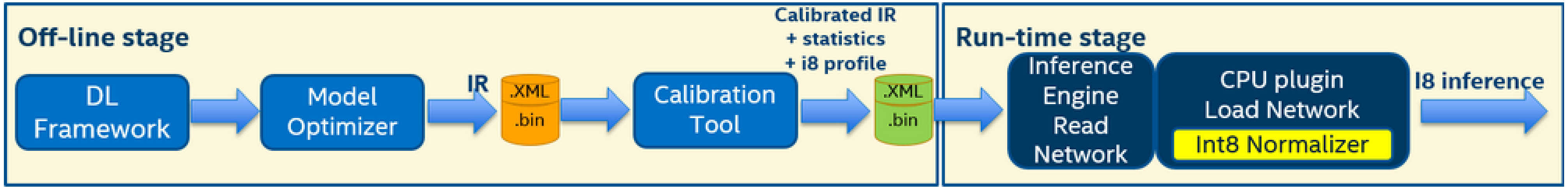
Typical AVX-512 instructions to perform INT8 convolutions: **vpdpbusd**

2nd gen Intel® Xeon® Scalable processor with Intel® DL Boost
INPUT INT8
INPUT INT8
CONSTANT INT32
vpdpbusd
OUTPUT INT32

The size of performance gain is dependent on the topology (how big is convolutional part) and system

Int8 Calibration – Calibration Tool

Calibration tool - command line app which collects statistics from FP32 or FP16 IR (intermediate representations)



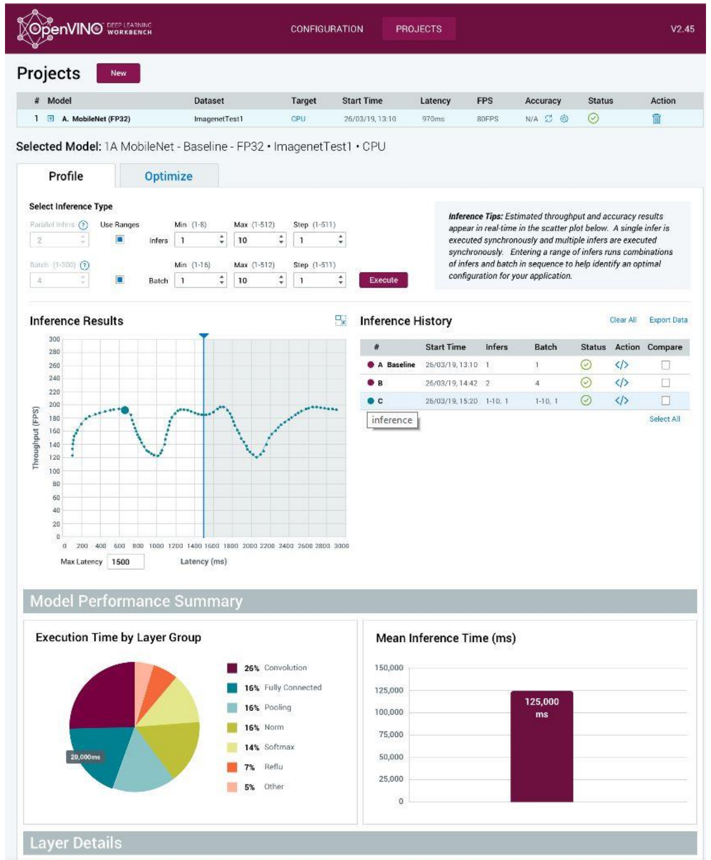
Calibration in **“simplified” mode** – see the maximum of potential performance gain from Int8 without accuracy calculation

Pass full calibration process to get working Int8 model with accuracy statistics

Deep Learning Workbench

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



Projects New

#	Model	Dataset	Target	Start Time	Latency	FPS	Accuracy	Status	Action
1	A. MobileNet (FP32)	ImagenetTest1	CPU	26/03/19, 13:10	970ms	80FPS	N/A	✓	

Selected Model: 1A MobileNet - Baseline - FP32 • ImagenetTest1 • CPU

Select Inference Type

Parallel Infers 2 · Use Ranges ■ · Infers Min (1-8) 1 · Max (1-512) 10 · Step (1-511) 1

Batch (1-300) 4 · ■ · Batch Min (1-16) 1 · Max (1-512) 10 · Step (1-511) 1

Inference Tips: Estimated throughput and accuracy results appear in real-time in the scatter plot below. A single infer is executed synchronously and multiple infers are executed synchronously. Entering a range of infers runs combinations of infers and batch in sequence to help identify an optimal configuration for your application.

Max Latency 1500

Inference History

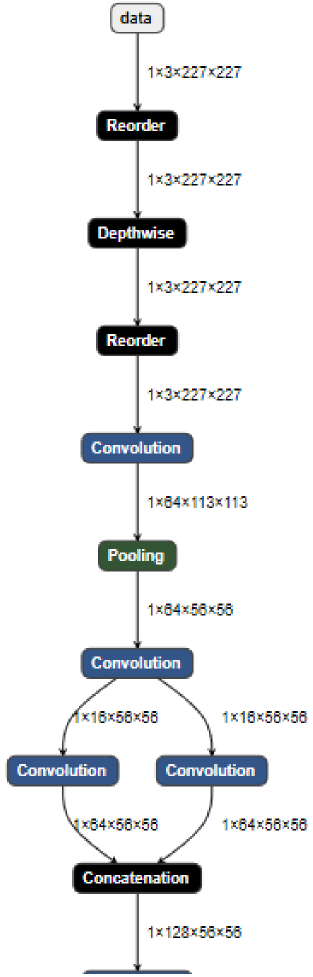
#	Start Time	Infers	Batch	Status	Action	Compare
● A Baseline	26/03/19, 13:10	1	1	✓	</>	□
● B	26/03/19, 14:42	2	4	✓	</>	□
● C	26/03/19, 15:20	1-10, 1	1-10, 1	✓	</>	□

Model Performance Summary

Execution Time by Layer Group

■	26%	Convolution
■	16%	Fully Connected
■	16%	Pooling
■	16%	Norm
■	14%	Softmax
■	7%	Reflu
■	5%	Other

Mean Inference Time (ms)



OpenVINO™ Toolkit

Open Source Version

- Provides flexibility and availability to the developer community to extend OpenVINO™ toolkit for custom needs
- Components that are open sourced
  - **Deep Learning Deployment Toolkit** with **CPU, GPU, Heterogeneous, Myriad** (for Intel® Neural Compute Stick (Intel® NCS) & Intel® NCS2), and **GNA** plugins github.com/opencv/dldt
  - **Open Model Zoo** - Includes pretrained models, model downloader, demos and samples: github.com/opencv/open_model_zoo
- See FAQ and next slides for key differences between the open source and Intel distribution

DESCRIPTION

OpenVINO™ toolkit, short for Open Visual Inference and Neural network Optimization toolkit, provides developers with improved neural network performance on a variety of Intel® processors and helps them further unlock cost-effective, real-time vision applications.The toolkit enables deep learning inference and easy heterogeneous execution across multiple Intel® platforms (CPU, Intel® Processor Graphics)—providing implementations across cloud architectures to edge devices. This open source distribution provides flexibility and availability to the developer community to innovate deep learning and AI solutions.

OpenVINO™ toolkit contains:

- Deep Learning Deployment Toolkit
- Open Model Zoo

The **Intel® Distribution of OpenVINO™ toolkit** is also available with additional, proprietary support for Intel® FPGAs, Intel®

Learn More ▶ 01.org/openvinotoolkit

Optimization Notice

OpenVINO™ Toolkit

Open Source Version

- Provides flexibility and availability to the developer community to extend OpenVINO™ toolkit for custom needs
- Components that are open sourced
  - **Deep Learning Deployment Toolkit** with **CPU, GPU, Heterogeneous, Myriad** (for Intel® Neural Compute Stick (Intel® NCS) & Intel® NCS2), and **GNA** plugins github.com/opencv/dldt
  - **Open Model Zoo** - Includes pretrained models, model downloader, demos and samples: github.com/opencv/open_model_zoo
- See FAQ and next slides for key differences between the open source and Intel distribution

DESCRIPTION

OpenVINO™ toolkit, short for Open Visual Inference and Neural network Optimization toolkit, provides developers with improved neural network performance on a variety of Intel® processors and helps them further unlock cost-effective, real-time vision applications.The toolkit enables deep learning inference and easy heterogeneous execution across multiple Intel® platforms (CPU, Intel® Processor Graphics)—providing implementations across cloud architectures to edge devices. This open source distribution provides flexibility and availability to the developer community to innovate deep learning and AI solutions.

OpenVINO™ toolkit contains:

- Deep Learning Deployment Toolkit
- Open Model Zoo

The **Intel® Distribution of OpenVINO™ toolkit** is also available with additional, proprietary support for Intel® FPGAs, Intel®

Learn More ▸ 01.org/openvinotoolkit

Optimization Notice

OpenVINO™ R2020.1 New Features

- Ngraph integration into OpenVINO
- Microsoft Visual Studio 2017/2019 compiler support
- Deployment Package Manager Windows
- New Int8 runtime –Post Training Optimization Tool (POT)
- New NUMA support (Multi-socket support for Windows
- Inference Engine C API
- Speech Libraries and End-to-End Speech Demos
- Inference & Streaming in OpenCV G-API
- 3D convolution with int8

DESCRIPTION

OpenVINO™ toolkit, short for Open Visual Inference and Neural network Optimization toolkit, provides developers with improved neural network performance on a variety of Intel® processors and helps them further unlock cost-effective, real-time vision applications.The toolkit enables deep learning inference and easy heterogeneous execution across multiple Intel® platforms (CPU, Intel® Processor Graphics)—providing implementations across cloud architectures to edge devices. This open source distribution provides flexibility and availability to the developer community to innovate deep learning and AI solutions.

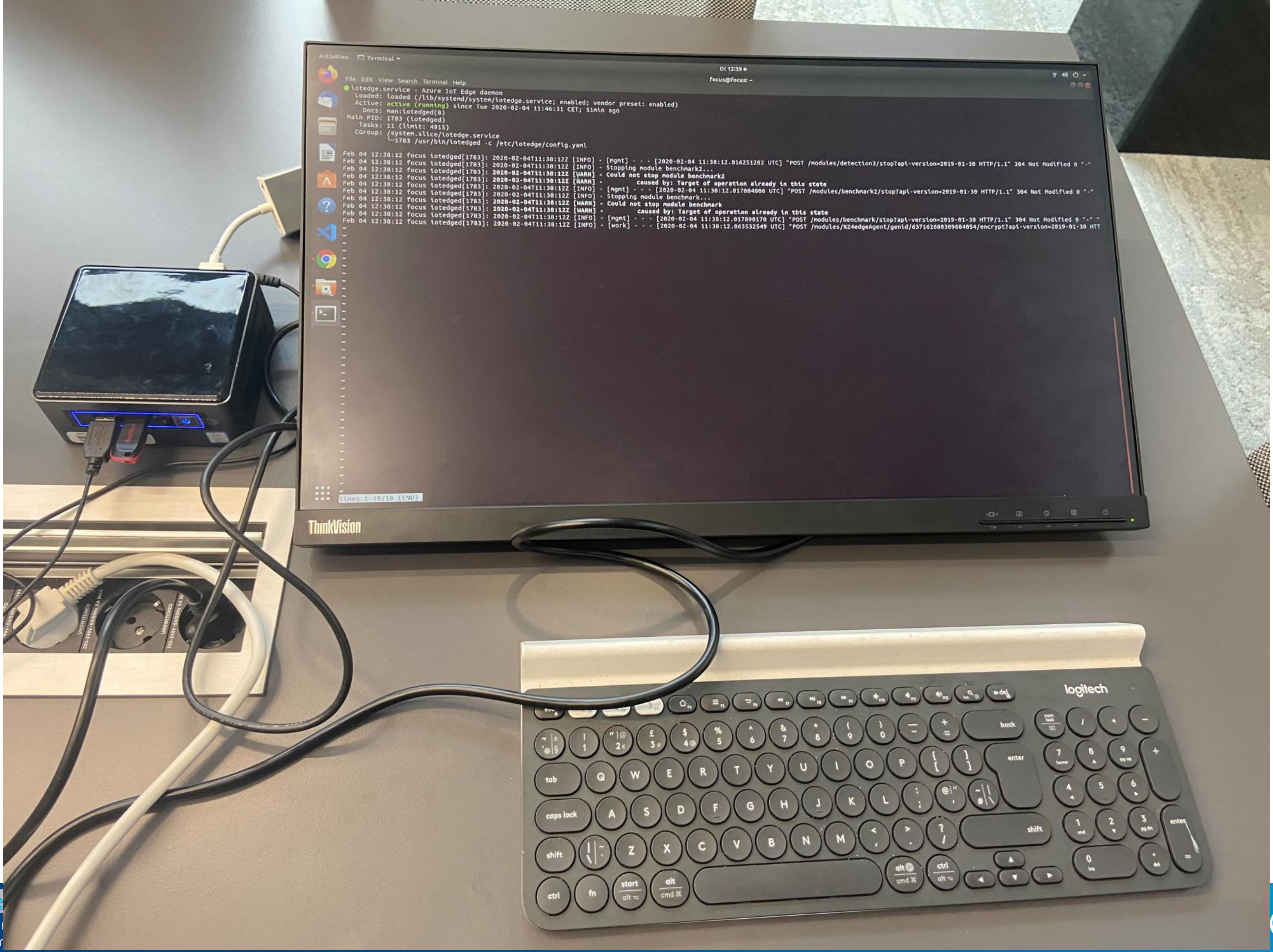
OpenVINO™ toolkit contains:

- Deep Learning Deployment Toolkit
- Open Model Zoo

The **Intel® Distribution of OpenVINO™ toolkit** is also available with additional, proprietary support for Intel® FPGAs, Intel®

Optimization Notice

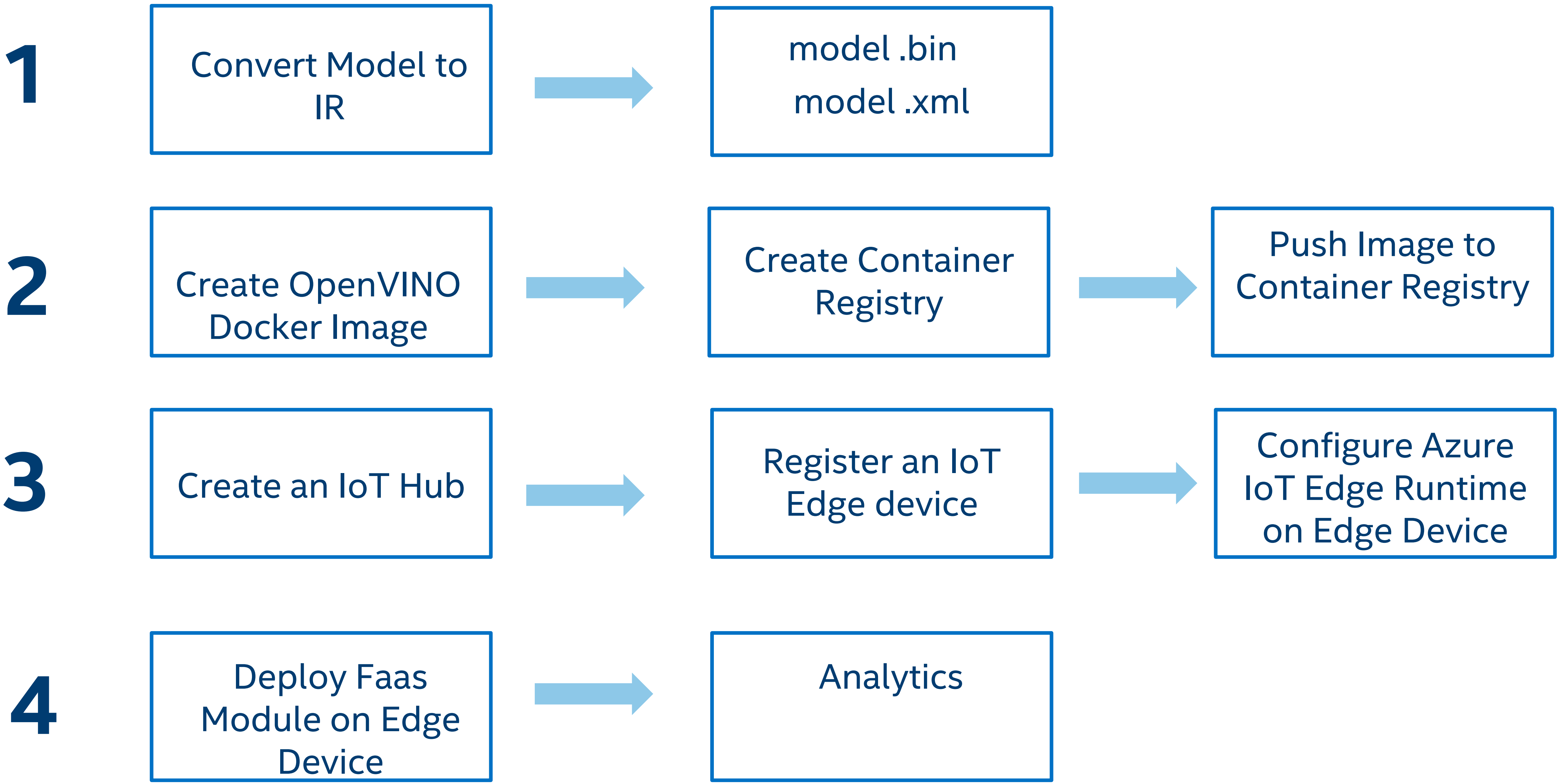
OPENVINO & AZURE IOT EDGE DEMO



```
● iotedge.service - Azure IoT Edge daemon
   Loaded: loaded (/lib/systemd/system/iotedge.service; enabled; vendor preset: enabled)
   Active: active (running) since Tue 2020-02-04 11:46:31 CET; 51min ago
     Docs: man:iotedged(8)
 Main PID: 1783 (iotedged)
    Tasks: 11 (limit: 4915)
   CGroup: /system.slice/iotedge.service
           └─1783 /usr/bin/iotedged -c /etc/iotedge/config.yaml

Feb 04 12:38:12 focus iotedged[1783]: 2020-02-04T11:38:12Z [INFO] - [mgmt] - - - [2020-02-04 11:38:12.016251282 UTC] "POST /modules/detection3/stop?api-version=2019-01-30 HTTP/1.1" 304 Not Modified 0 "-"
Feb 04 12:38:12 focus iotedged[1783]: 2020-02-04T11:38:12Z [INFO] - Stopping module benchmark2...
Feb 04 12:38:12 focus iotedged[1783]: 2020-02-04T11:38:12Z [WARN] - Could not stop module benchmark2
Feb 04 12:38:12 focus iotedged[1783]: 2020-02-04T11:38:12Z [WARN] -         caused by: Target of operation already in this state
Feb 04 12:38:12 focus iotedged[1783]: 2020-02-04T11:38:12Z [INFO] - [mgmt] - - - [2020-02-04 11:38:12.017084806 UTC] "POST /modules/benchmark2/stop?api-version=2019-01-30 HTTP/1.1" 304 Not Modified 0 "-"
Feb 04 12:38:12 focus iotedged[1783]: 2020-02-04T11:38:12Z [INFO] - Stopping module benchmark...
Feb 04 12:38:12 focus iotedged[1783]: 2020-02-04T11:38:12Z [WARN] - Could not stop module benchmark
Feb 04 12:38:12 focus iotedged[1783]: 2020-02-04T11:38:12Z [WARN] -         caused by: Target of operation already in this state
Feb 04 12:38:12 focus iotedged[1783]: 2020-02-04T11:38:12Z [INFO] - [mgmt] - - - [2020-02-04 11:38:12.017890170 UTC] "POST /modules/benchmark/stop?api-version=2019-01-30 HTTP/1.1" 304 Not Modified 0 "-" "
Feb 04 12:38:12 focus iotedged[1783]: 2020-02-04T11:38:12Z [INFO] - [work] - - - [2020-02-04 11:38:12.063532549 UTC] "POST /modules/%24edgeAgent/genid/637162680309684054/encrypt?api-version=2019-01-30 HTT
lines 1-19/19 (END)
```

Workflow



1

Convert Model to IR

model .bin

model .xml

2

Create OpenVINO Docker Image

Create Container Registry

Push Image to Container Registry

3

Create an IoT Hub

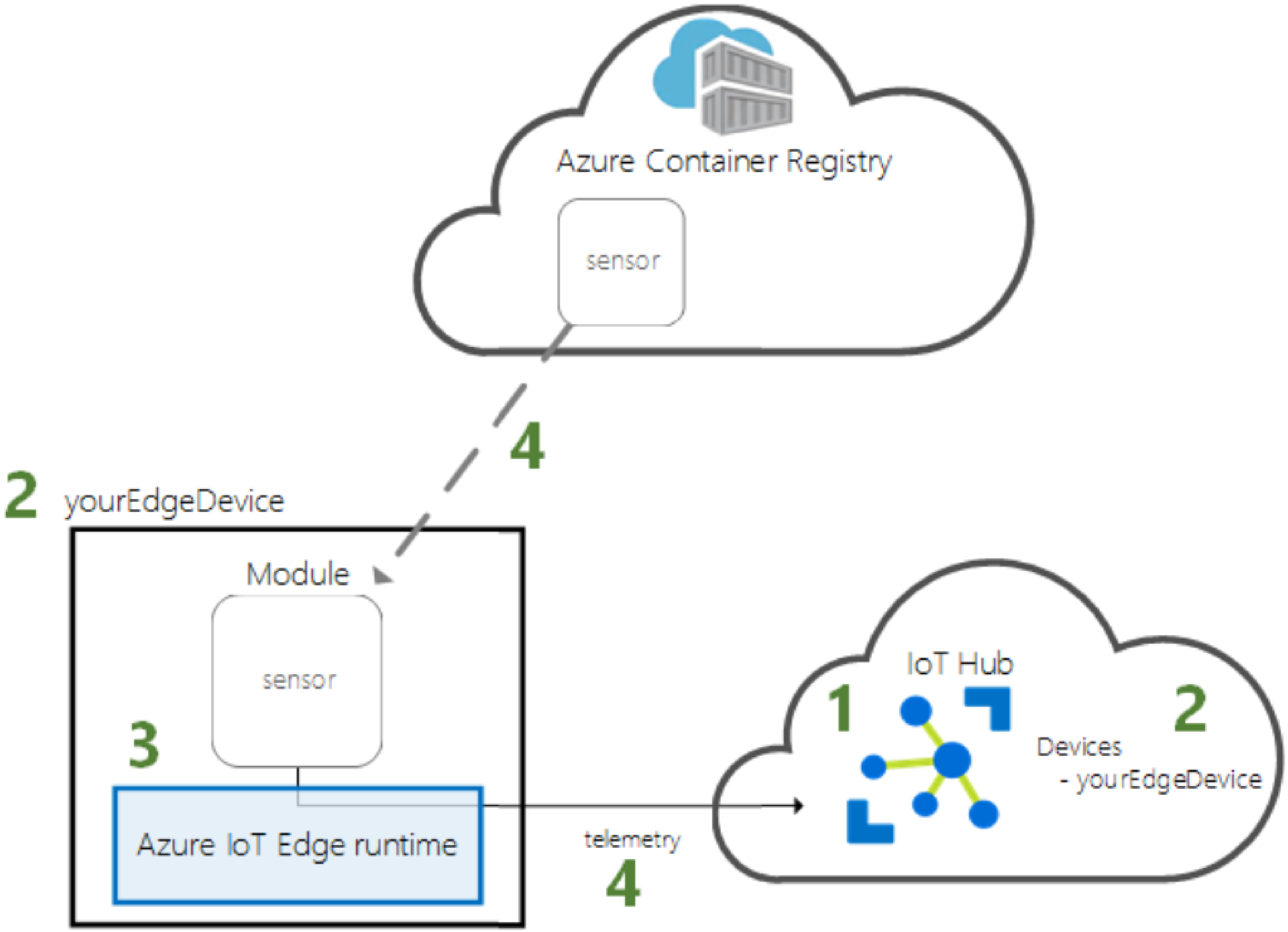
Register an IoT Edge device

Configure Azure IoT Edge Runtime on Edge Device

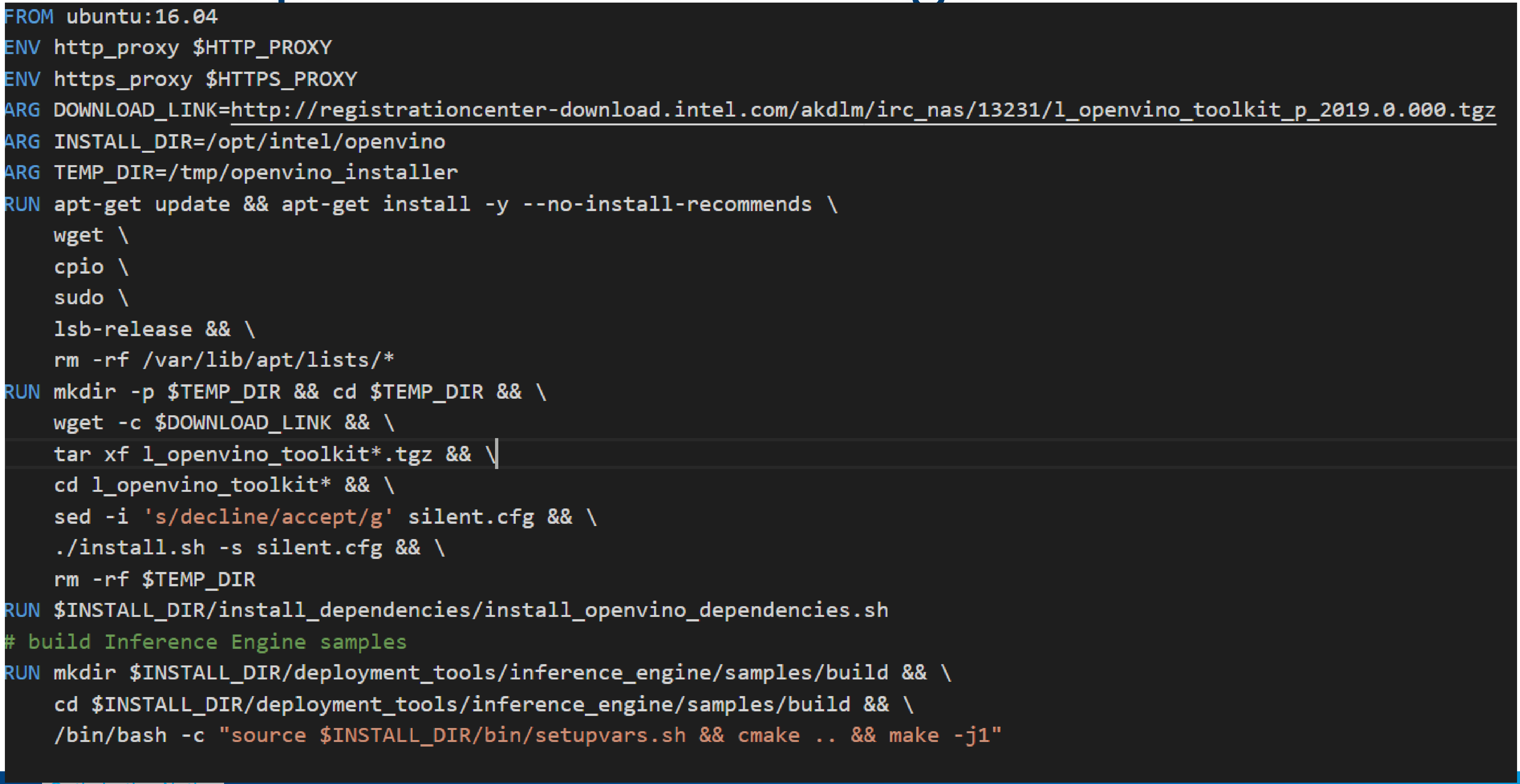
4

Deploy Faas Module on Edge Device

Analytics


Azure Container Registry
sensor
4
2
yourEdgeDevice
Module
sensor
3
Azure IoT Edge runtime
telemetry
4
IoT Hub
1
2
Devices
- yourEdgeDevice

Create OpenVINO Docker Image

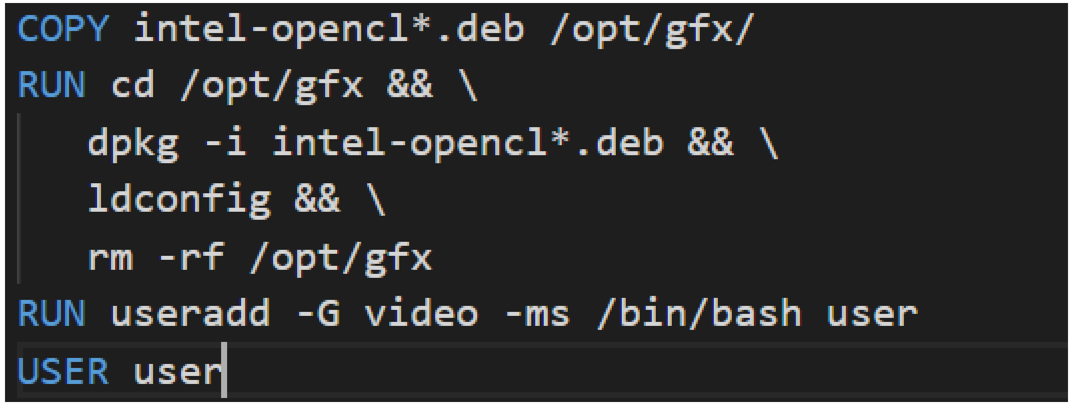


```
FROM ubuntu:16.04
ENV http_proxy $HTTP_PROXY
ENV https_proxy $HTTPS_PROXY
ARG DOWNLOAD_LINK=http://registrationcenter-download.intel.com/akdlm/irc_nas/13231/l_openvino_toolkit_p_2019.0.000.tgz
ARG INSTALL_DIR=/opt/intel/openvino
ARG TEMP_DIR=/tmp/openvino_installer
RUN apt-get update && apt-get install -y --no-install-recommends \
    wget \
    cpio \
    sudo \
    lsb-release && \
    rm -rf /var/lib/apt/lists/*
RUN mkdir -p $TEMP_DIR && cd $TEMP_DIR && \
    wget -c $DOWNLOAD_LINK && \
    tar xf l_openvino_toolkit*.tgz && \
    cd l_openvino_toolkit* && \
    sed -i 's/decline/accept/g' silent.cfg && \
    ./install.sh -s silent.cfg && \
    rm -rf $TEMP_DIR
RUN $INSTALL_DIR/install_dependencies/install_openvino_dependencies.sh
# build Inference Engine samples
RUN mkdir $INSTALL_DIR/deployment_tools/inference_engine/samples/build && \
    cd $INSTALL_DIR/deployment_tools/inference_engine/samples/build && \
    /bin/bash -c "source $INSTALL_DIR/bin/setupvars.sh && cmake .. && make -j1"
```

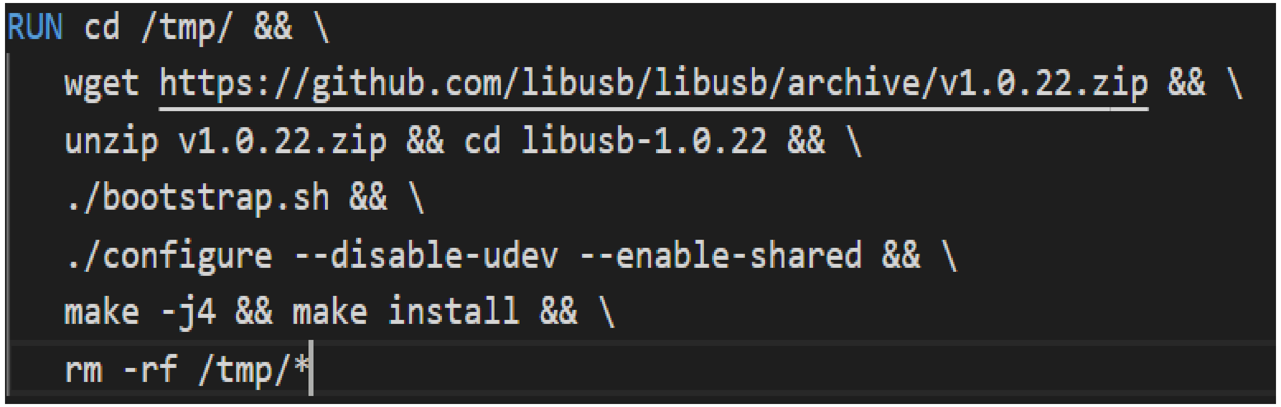
Create OpenVINO Docker Image

- https://docs.openvinotoolkit.org/latest/_docs_install_guides_installing_openvino_docker_linux.html

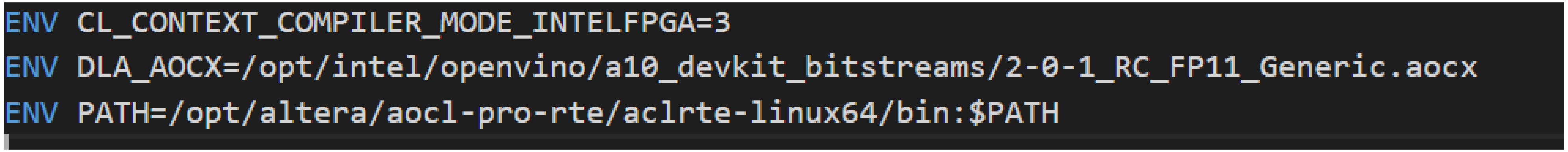
- Implement additional dependencies for building GPU, Movidius NCS and FPGA images



```
COPY intel-opencl*.deb /opt/gfx/
RUN cd /opt/gfx && \
   dpkg -i intel-opencl*.deb && \
   ldconfig && \
   rm -rf /opt/gfx
RUN useradd -G video -ms /bin/bash user
USER user
```



```
RUN cd /tmp/ && \
   wget https://github.com/libusb/libusb/archive/v1.0.22.zip && \
   unzip v1.0.22.zip && cd libusb-1.0.22 && \
   ./bootstrap.sh && \
   ./configure --disable-udev --enable-shared && \
   make -j4 && make install && \
   rm -rf /tmp/*
```



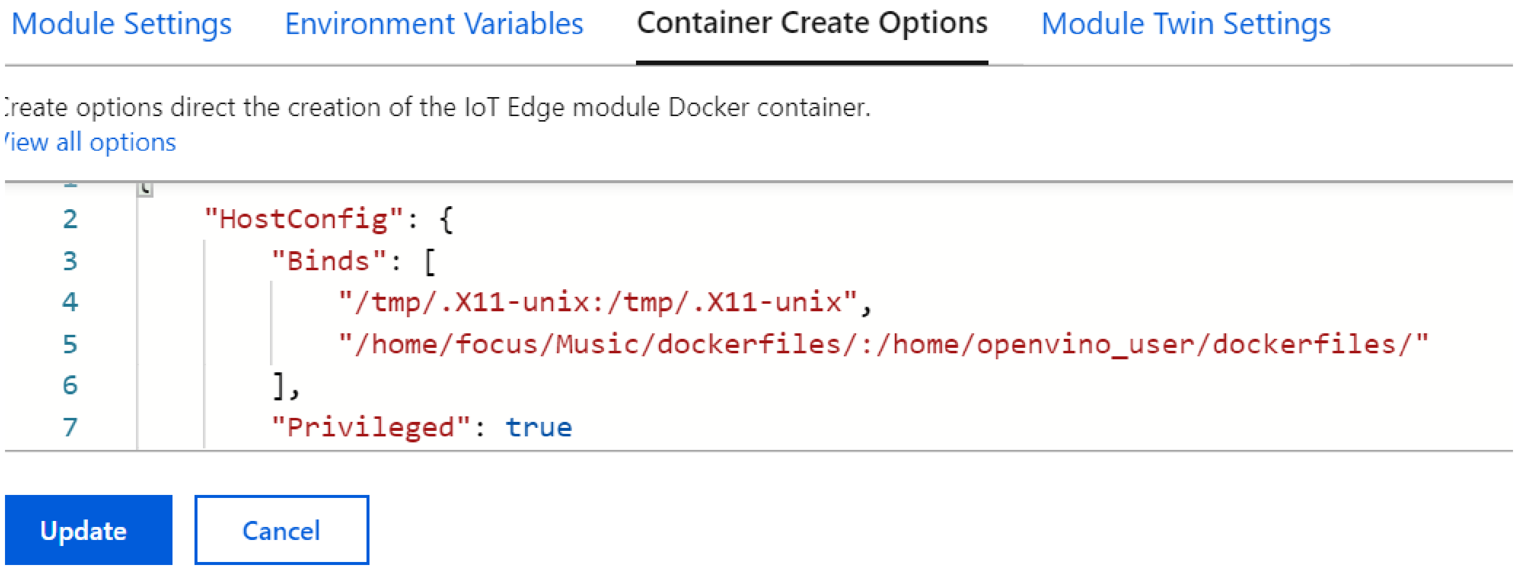
```
ENV CL_CONTEXT_COMPILER_MODE_INTELFPGA=3
ENV DLA_AOCX=/opt/intel/openvino/a10_devkit_bitstreams/2-0-1_RC_FP11_Generic.aocx
ENV PATH=/opt/altera/aocl-pro-rte/aclrte-linux64/bin:$PATH
```

Create and Push OpenVINO Docker Image

- docker build -t dockerimage .
- docker login openvinoregistry.azurecr.io -u XXXXX -p XXXXX
- docker tag dockerimage openvinoregistry.azurecr.io.azurecr.io/dockerimage:v1
- docker push openvinoregistry.azurecr.io/dockerimage:v1

Configure Module on IoT Hub

- Create IoT Hub. Add Device. Set Connection String on Edge Device(/etc/iotedge/config.yaml)
- Set Device Module Image URI to Repo in Container Registry
- Set Device Module Container Create Options



reate options direct the creation of the IoT Edge module Docker container.

iew all options

```
    "HostConfig": {
        "Binds": [
            "/tmp/.X11-unix:/tmp/.X11-unix",
            "/home/focus/Music/dockerfiles/:/home/openvino_user/dockerfiles/"
        ],
        "Privileged": true
```

OPENVINO & AZURE IOT EDGE DEMO

QNA

intel®

experience
what's inside™

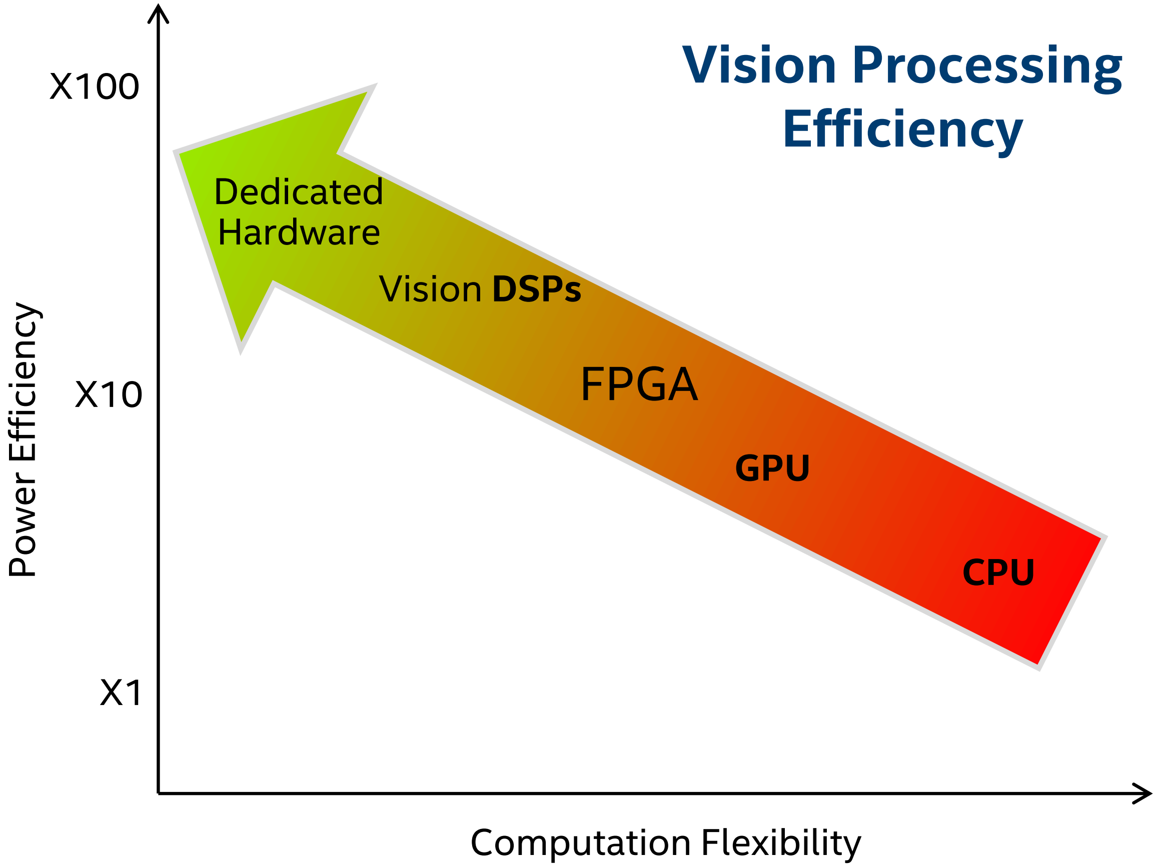
Choosing the “Right” Hardware

Power/Performance Efficiency Varies

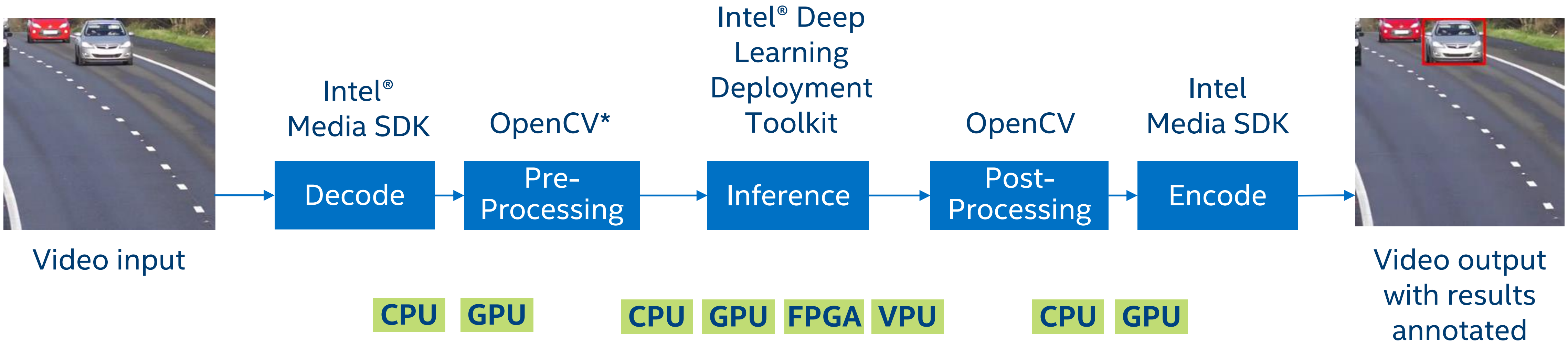
- Running the right workload on the right piece of hardware → higher efficiency
- Hardware acceleration is a must
- Heterogeneous computing?

Tradeoffs

- Power/performance
- Price
- Software flexibility, portability



End-to-End Vision Workflow



Video input

Intel® Media SDK

Decode

CPU GPU

OpenCV*

Pre-Processing

Intel® Deep Learning Deployment Toolkit

Inference

CPU GPU FPGA VPU

OpenCV

Post-Processing

CPU GPU

Intel Media SDK

Encode

Video output with results annotated

Model Optimizer

Model optimizer performs generic optimization:

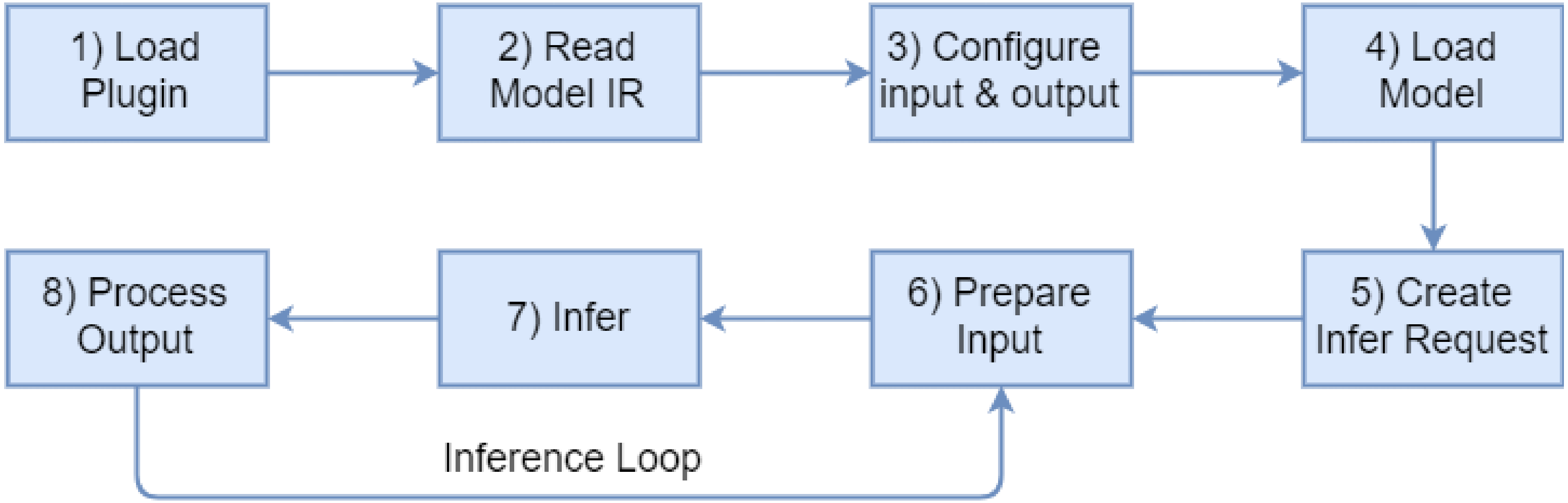
- Node merging
- Horizontal fusion
- Batch normalization to scale shift
- Fold scale shift with convolution
- Drop unused layers (dropout)
- FP16/Int8 quantization
- Model optimizer can add normalization and mean operations, so some preprocessing is ‘added’ to the IR

  --mean_values (104.006, 116.66, 122.67)

  --scale_values (0.07, 0.075, 0.084)

Hardware	FP32	FP16	FP11	INT8
CPU	yes	yes	no	yes
GPU	yes	yes	no	no
MYRIAD	no	yes	no	no
FPGA/DLA	no	yes	yes	no

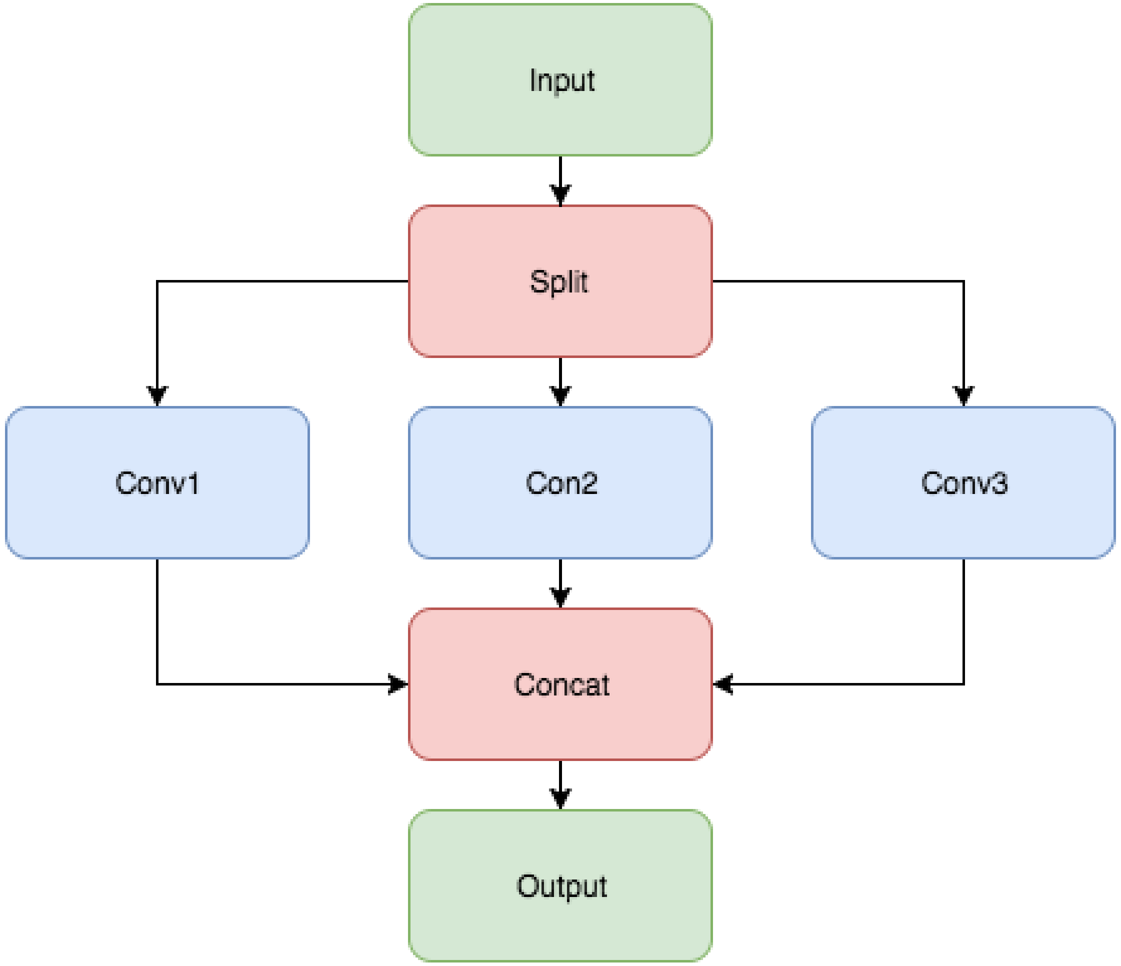
Application Workflow for Inference Engine



Internal CPU Plugin Optimizations

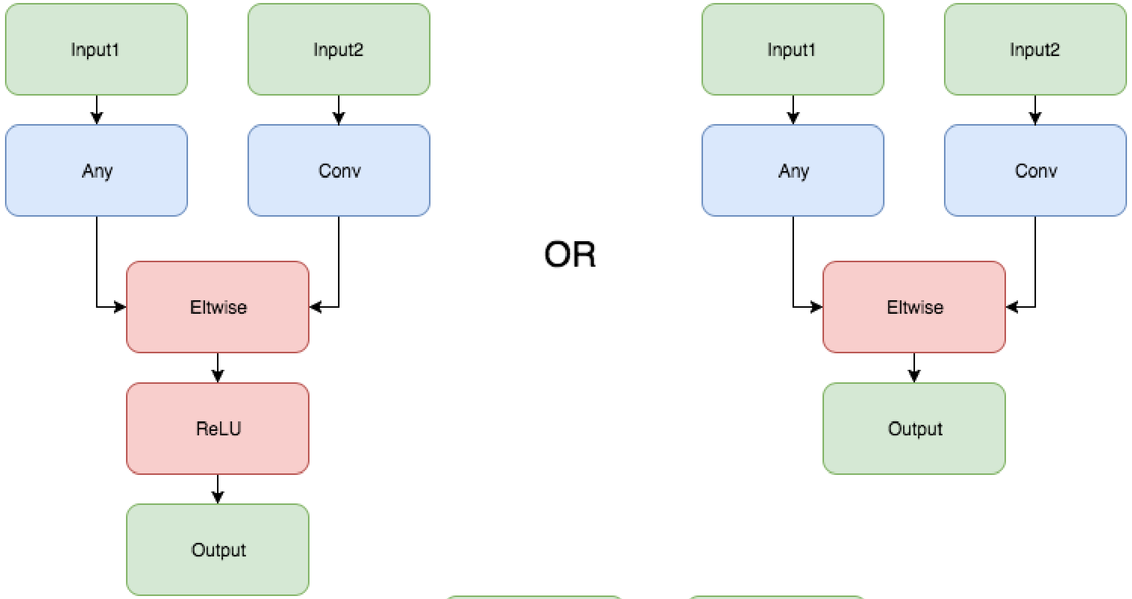
Merging of group convolutions.

- It means that if a topology contains the following pipeline →
- CPU plugin will merge it into one Convolution with the group parameter (Convolutions should have the same parameters).

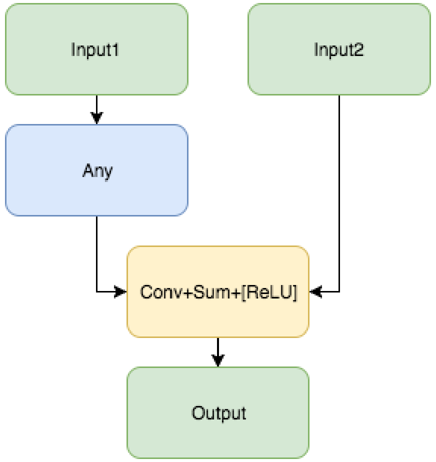


Internal CPU Plugin Optimizations

- **Fusing Convolution with ReLU or ELU.** CPU plugin is fusing all Convolution with ReLU or ELU layers if these layers are located after the Convolution layer.



- **Fusing Convolution + Sum or Convolution + Sum + ReLu.** To improve performance, the CPU plugin fuses the following structure:


Input1
Input2
Any
Conv+Sum+[ReLU]
Output