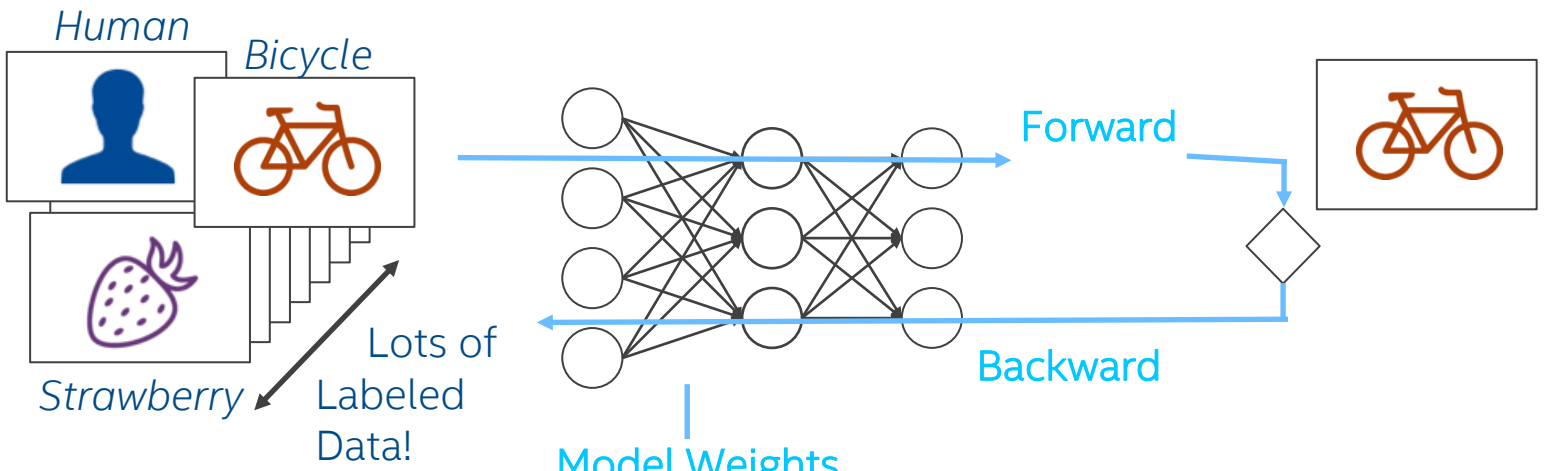
Intel® Distribution of OpenVINO™ toolkit

Dr. Séverine Habert, Deep Learning Software Engineer

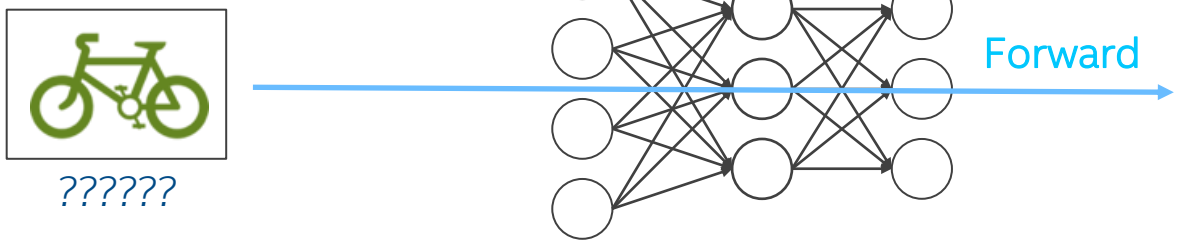
October 14th, 2021

Deep Learning: Training vs. Inference

Training

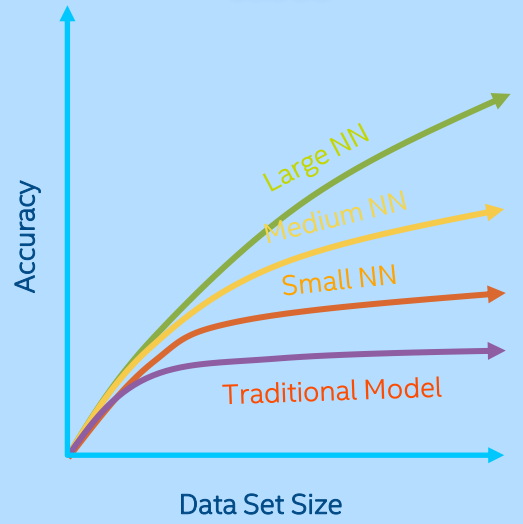


Inference



Did You Know?

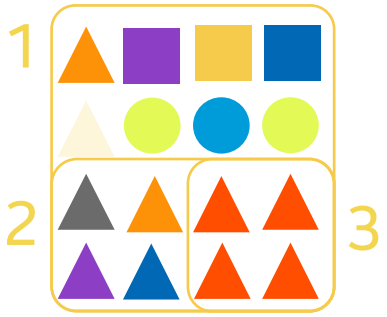
Training requires a very large data set and deep neural network (many layers) to achieve the highest accuracy in most cases



AI Compute Considerations

How do you determine the right computing for your AI needs?

WORKLOADS

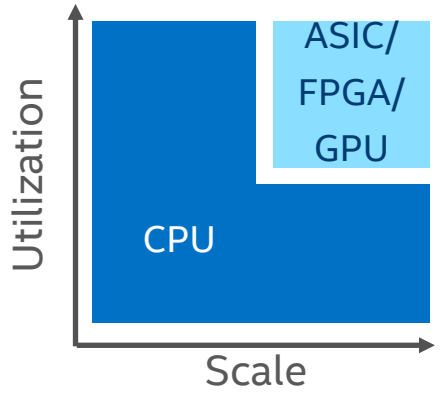


What is my workload profile?

REQUIREMENTS

What are my use case requirements?

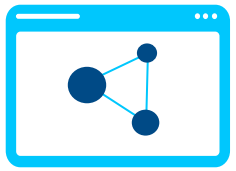
DEMAND



How prevalent is AI in my environment?

Challenges in Deep Learning

Development and deployment challenges in deep learning



Unique Inference Needs

Gap in performance and accuracy between trained and deployed models

Low performing, lower accuracy models deployed

Integration Challenges

No streamlined way for end-to-end development workflow

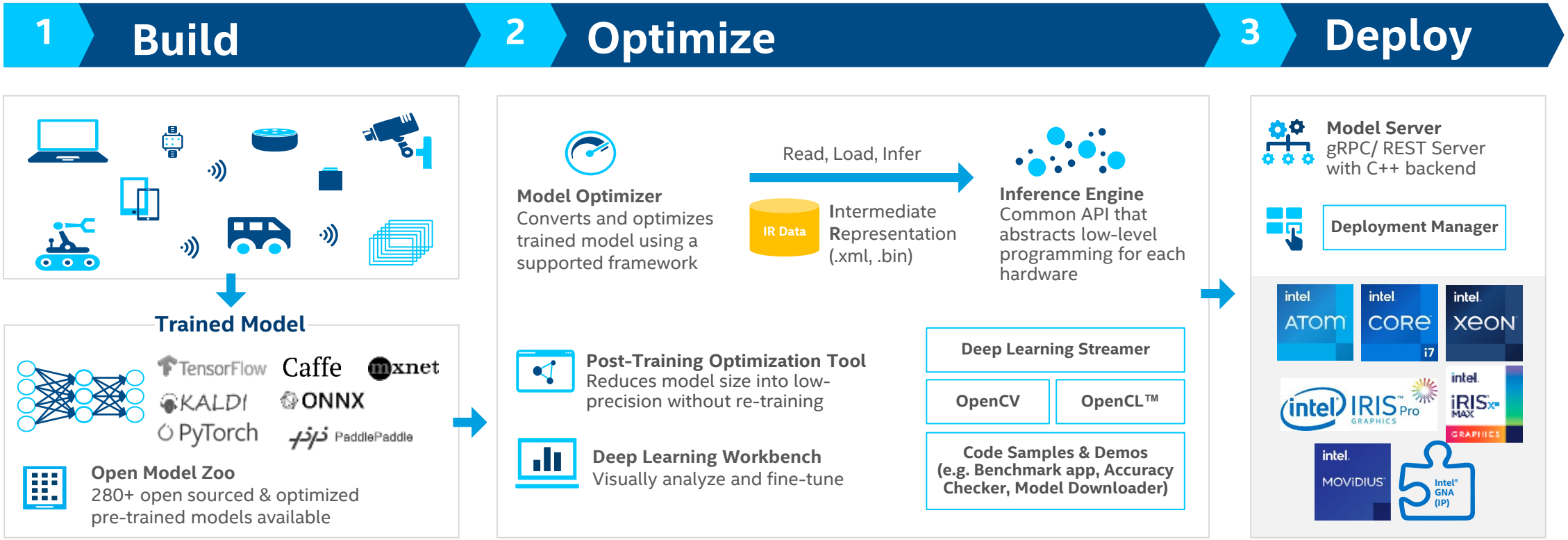
Slow time-to-solution and time–to-market

No One Size Fits All

Diverse requirements for myriad use cases require unique approaches

Inability to meet use-case specific requirements

Three steps for the Intel® Distribution of OpenVINO™ toolkit



Supported Frameworks

Breadth of supported frameworks to enable developers with flexibility

Supported Frameworks and Formats ▸ https://docs.openvinotoolkit.org/latest/_docs_IE_DG_Introduction.html#SupportedFW
Configure the Model Optimizer for your Framework ▸ https://docs.openvinotoolkit.org/latest/_docs_MO_DG_prepare_model_Config_Model_Optimizer.html

Model Optimization

Breadth of supported frameworks to enable developers with flexibility

Model Optimizer loads a model into memory, reads it, builds the internal representation of the model, optimizes it, and produces the **Intermediate Representation**.

Optimization techniques available are:

— Linear operation fusing

— Stride optimizations

— Group convolutions fusing

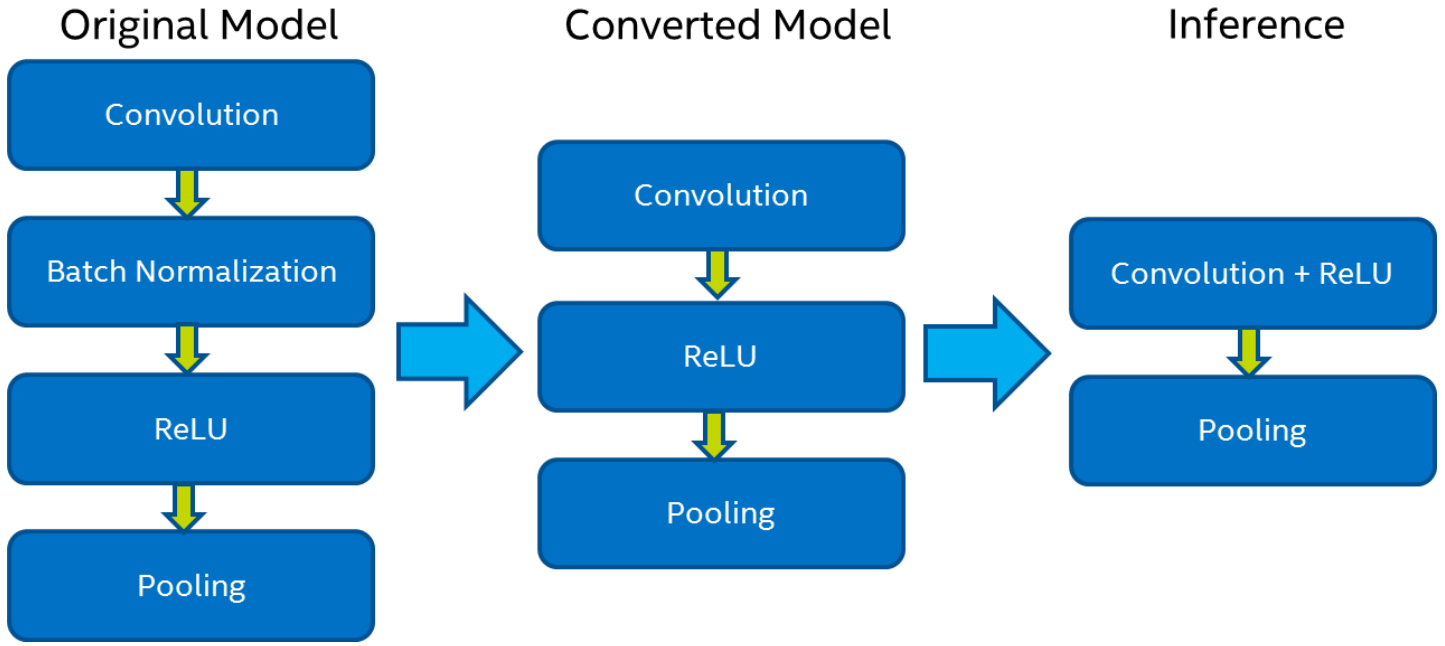
Note: Except for ONNX (.onnx model formats), all models have to be converted to an IR format to use as input to the Inference Engine

.xml – describes the network topology
.bin – describes the weights and biases binary data

Model Optimizer: Linear Operation Fusing

- Example

1. Remove Batch normalization stage.
2. Recalculate the weights to 'include' the operation.
3. Merge Convolution and ReLU into one optimized kernel.



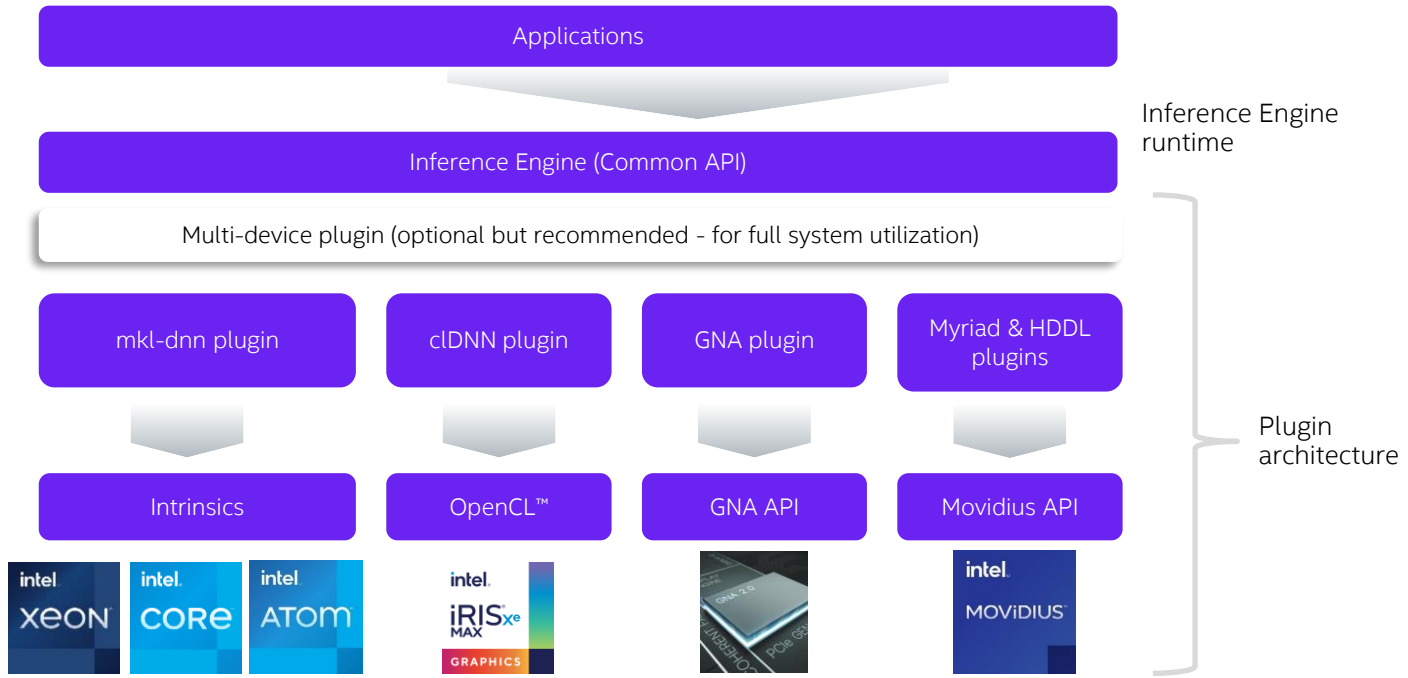
Optimal Model Performance Using the Inference Engine

Core Inference Engine Libraries

- Create Inference Engine Core object to work with devices
- Read the network
- Manipulate network information
- Execute and pass inputs and outputs

Device-specific Plugin Libraries

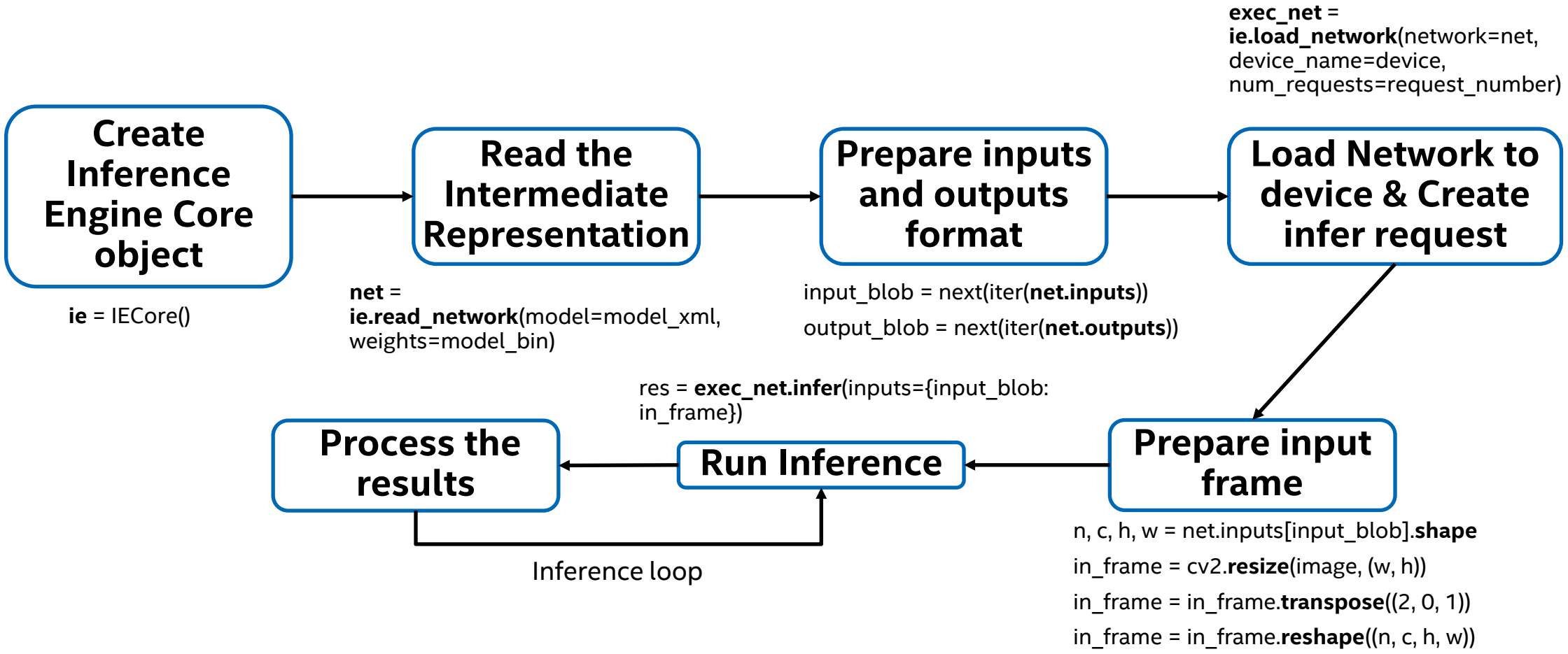
- For each supported target device, Inference Engine provides a plugin — a DLL/shared library that contains complete implementation for inference on this device.



GPU = Intel CPU with integrated graphics/Intel® Processor Graphics/GEN

GNA = Gaussian mixture model and Neural Network Accelerator

Common Workflow for Using the Inference Engine API



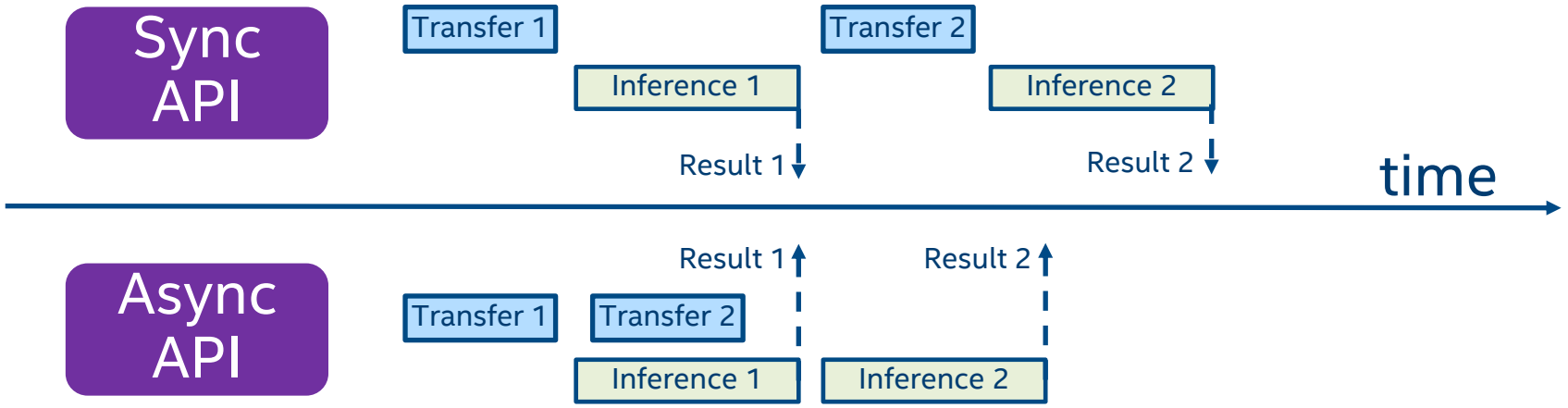
http://docs.openvinotoolkit.org/latest/_docs_IE_DG_Integrate_with_customer_application_new_API.html

Inference Engine

Synchronous vs Asynchronous Execution

- In IE API model can be executed by **Infer Request** which can be:
- **Synchronous** - blocks until inference is completed.
  - exec_net.infer(inputs = {input_blob: in_frame})
- **Asynchronous** – checks the execution status with the wait or specify a completion callback *(recommended way)*.
  - exec_net.start_async(request_id = id, inputs={input_blob: in_frame})
  - If exec_net.requests[id].wait() != 0

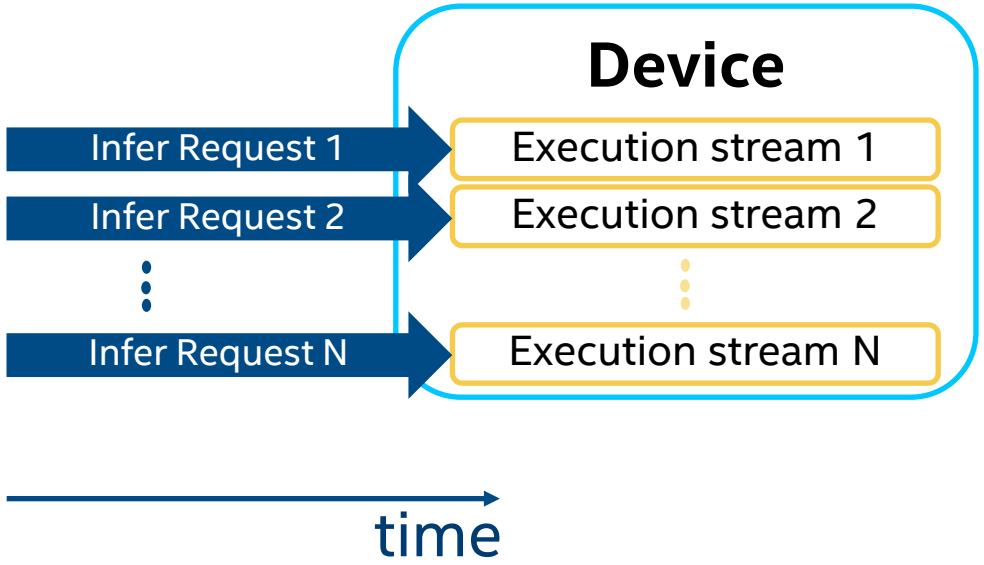
    do something



Inference Engine

Throughput Mode for CPU, iGPU and VPU

- **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



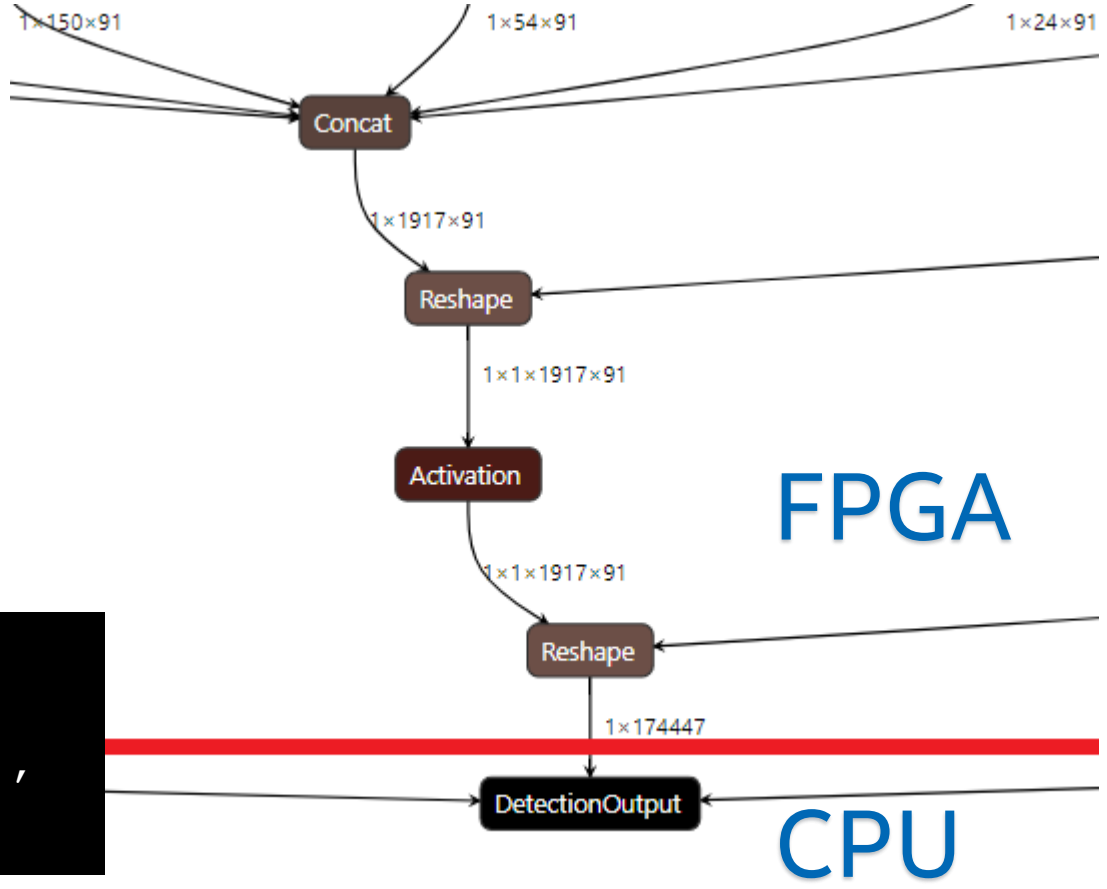
CPU Example:

```
ie = IECore()
ie.GetConfig(CPU, KEY_CPU_THROUGHPUT_STREAMS)
```

Inference Engine

Heterogeneous Support

- You can execute different layers on different HW units
- Offload unsupported layers on fallback devices:
  - Default affinity policy
  - Setting affinity manually (`CNNLayer::affinity`)
- All device combinations are supported (CPU, GPU, FPGA, MYRIAD, HDDL)
- Samples/demos usage "`-d HETERO:FPGA,CPU`"




```
InferenceEngine::Core core;
        auto executable_network =
        core.LoadNetwork(reader.getNetwork(),
        "HETERO:FPGA,CPU");
```

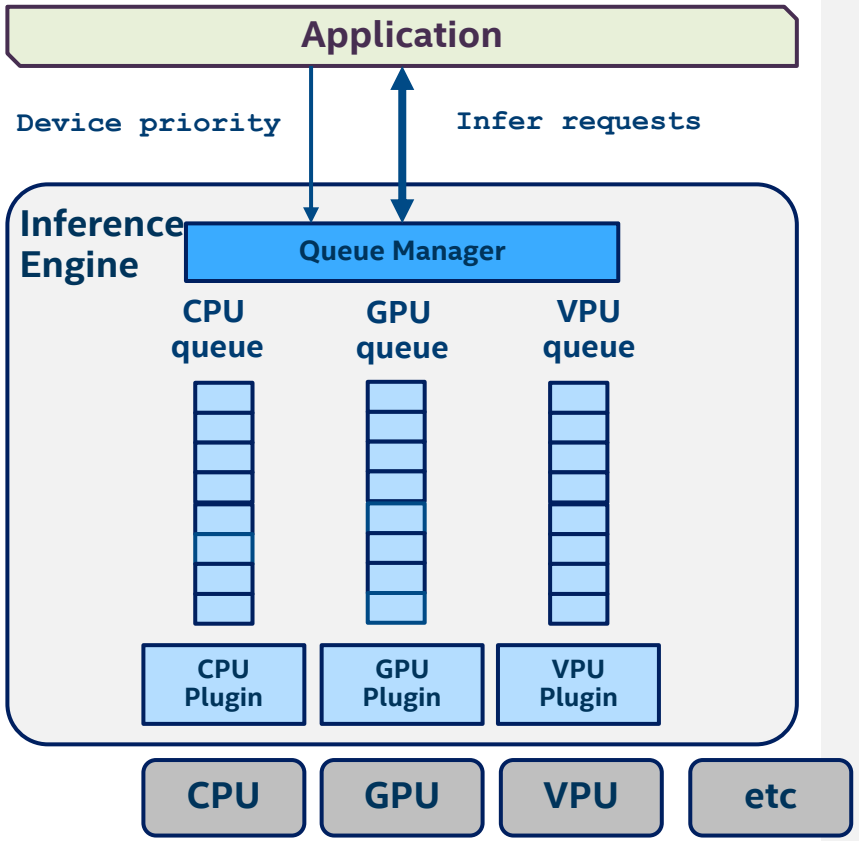
Inference Engine

Multi-device Support

Automatic load-balancing between devices (inference requests level) for full system utilization

- Any combinations of the following devices are supported (CPU, iGPU, VPU, HDDL)
- As easy as “-d MULTI:CPU,GPU” for cmd-line option of your favorite sample/demo
- C++ example (Python is similar)

```
Core ie;
ExecutableNetwork exec =
ie.LoadNetwork(network,{{“DEVICE_PRIORITIES”, “CPU,GPU”}},
“MULTI”)
```

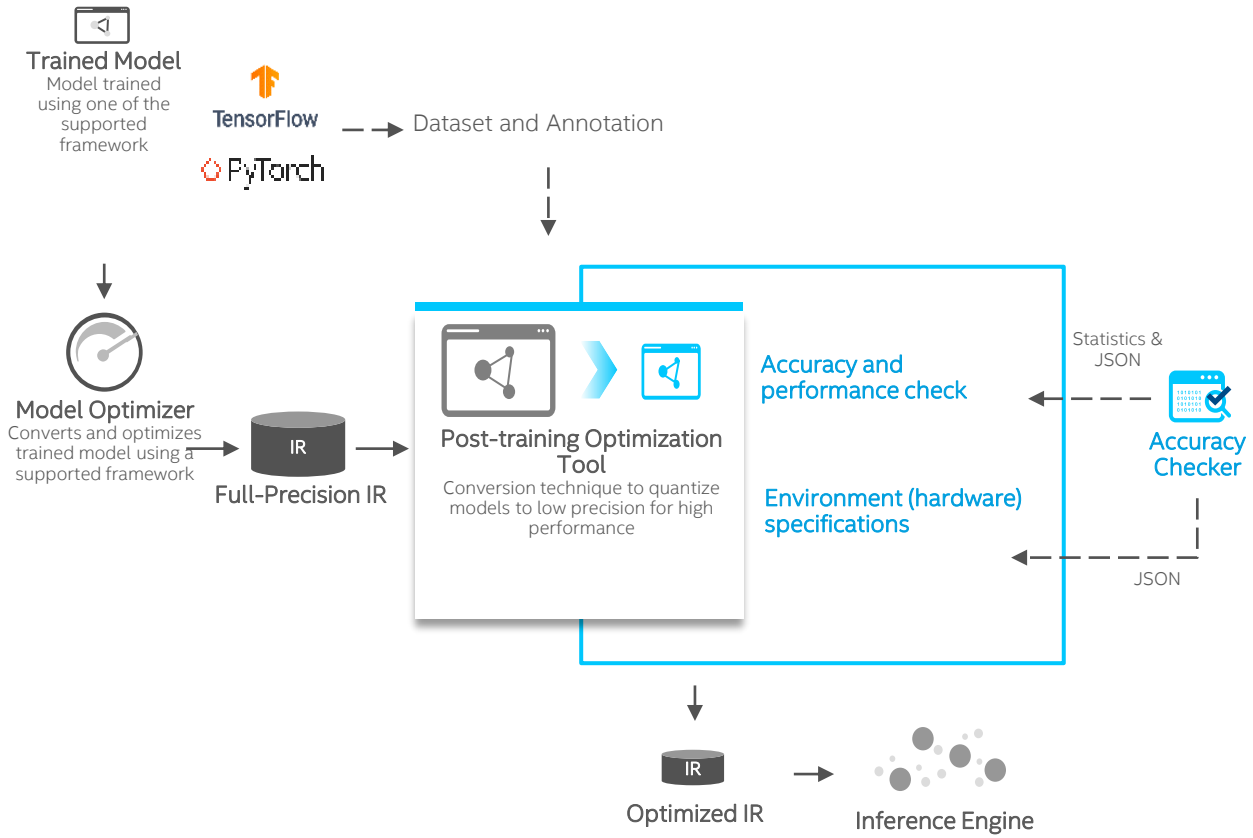


Post-Training Optimization Tool

Conversion technique that reduces model size into low-precision without re-training

Reduces model size **while also improving latency, with little degradation** in model accuracy and without model re-training.

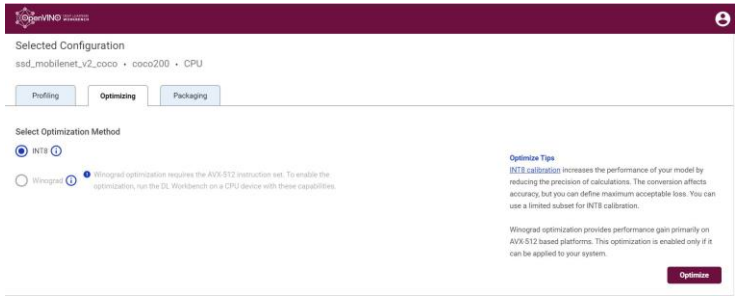
Different optimization approaches are supported: quantization algorithms, etc.



Deep Learning Workbench

Web-based UI extension tool for model analyses and graphical measurements

- **Visualizes performance data for** topologies and layers to aid in model analysis
- **Automates analysis** for optimal performance configuration (streams, batches, latency)
- **Experiment with INT8 or Winograd calibration** for optimal tuning using the Post Training Optimization Tool
- Provide **accuracy information** through accuracy checker
- **Direct access to models** from public set of Open Model Zoo
- Enables **remote profiling**, allowing the collection of performance data from multiple different machines without any additional set-up.



Additional Tools and Add-ons

Streamlined development experience and ease of use

Model Downloader

- Provides an easy way of accessing a number of public models as well as a set of pre-trained Intel models

Deployment Manager

- Generate an optimal, minimized runtime package for deployment
- Deploy with smaller footprint compared to development package

Benchmark App

- Measure performance (throughput, latency) of a model
- Get performance metrics per layer and overall basis

Accuracy Checker

- Check for accuracy of the model (original and after conversion) to IR file using a known data set

Computer Vision Annotation Tool
This web-based tool helps annotate videos and images before training a model

Deep Learning Streamer
Streaming analytics framework to create and deploy complex media analytics pipelines

OpenVINO™ Model Server
Scalable inference server for serving optimized models over gRPC or REST API endpoints

Dataset Management Framework
Use this add-on to build, transform and analyze datasets

Neural Network Compression Framework
Suite of compression algorithms for quantization-aware training with PyTorch* and TensorFlow frameworks

Training Extensions
Trainable deep learning models for action recognition, segmentation, image classification, object detection and text spotting

Pre-Trained Models and Public Models

Open-sourced repository of pre-trained models and support for public models

Use free **Pre-trained Models** to speed up development and deployment

Take advantage of the **Model Downloader** and other automation tools to quickly get started

Iterate with the **Accuracy Checker** to validate the accuracy of your models

100+ Pre-trained Models
Common AI tasks

Object Detection
Object Recognition
Reidentification
Semantic Segmentation
Instance Segmentation
Human Pose Estimation
Image Processing
Text Detection
Text Recognition
Text Spotting
Action Recognition
Image Retrieval
Compressed Models
Question Answering

100+ Public Models
Pre-optimized external models

Classification
Segmentation
Object Detection
Human Pose Estimation
Monocular Depth Estimation
Image Inpainting
Style Transfer
Action Recognition
Colorization

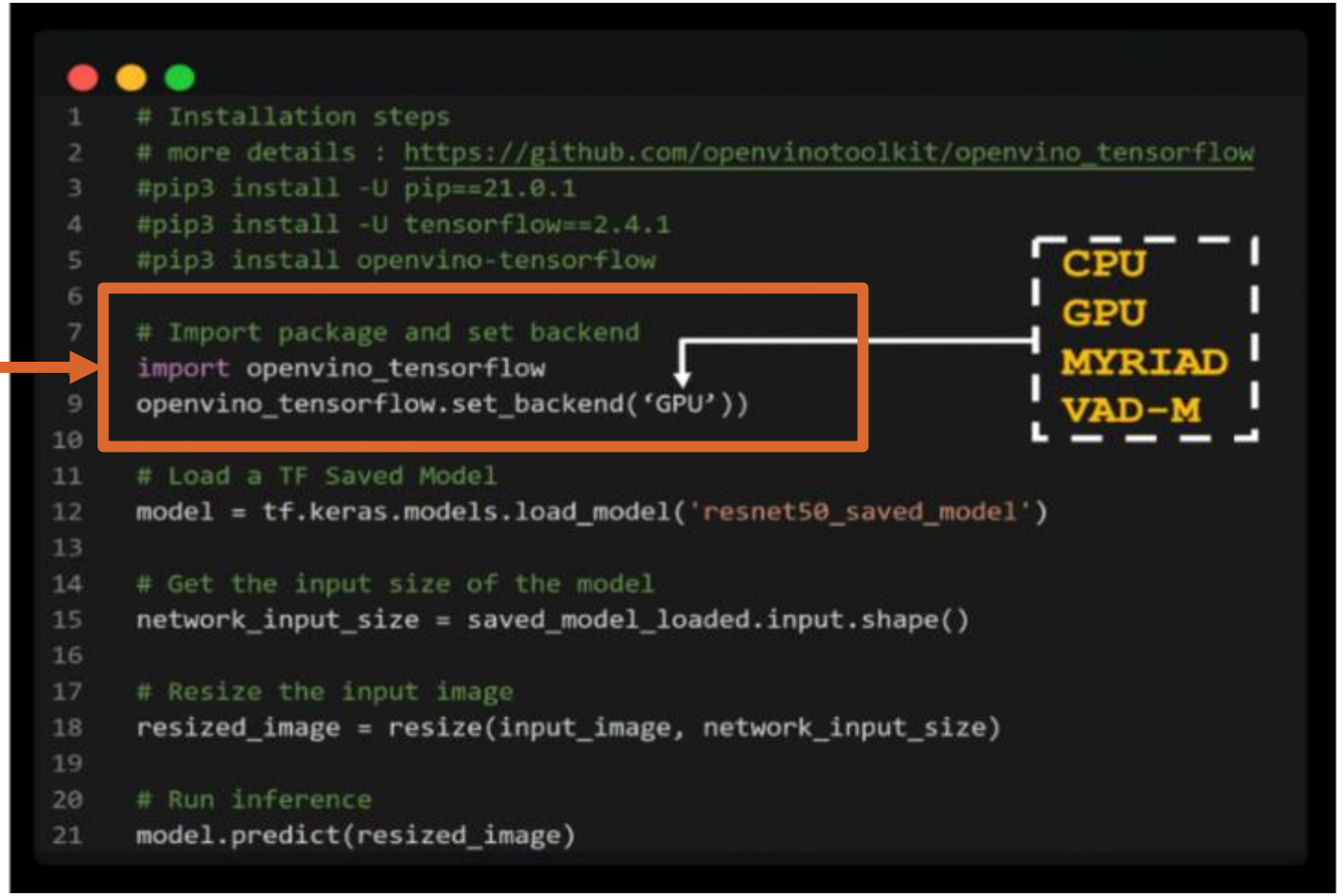
OpenVINO as execution provider

- You can use OpenVINO Inference Engine as backend of other DL Inference Frameworks such as Tensorflow and ONNX Runtime

- Benefit: the advantages of OpenVINO (multiple HW support and acceleration) in your favorite framework

OpenVINO™ Integration with TensorFlow*

Only 2 lines to be added to regular Tensorflow




```
# Installation steps
# more details : https://github.com/openvinotoolkit/openvino_tensorflow
#pip3 install -U pip==21.0.1
#pip3 install -U tensorflow==2.4.1
#pip3 install openvino-tensorflow

# Import package and set backend
import openvino_tensorflow
openvino_tensorflow.set_backend('GPU')

# Load a TF Saved Model
model = tf.keras.models.load_model('resnet50_saved_model')

# Get the input size of the model
network_input_size = saved_model_loaded.input.shape()

# Resize the input image
resized_image = resize(input_image, network_input_size)

# Run inference
model.predict(resized_image)
```

Intel® DevCloud for the Edge

Accelerate Test Cycles with the Intel® DevCloud for the Edge

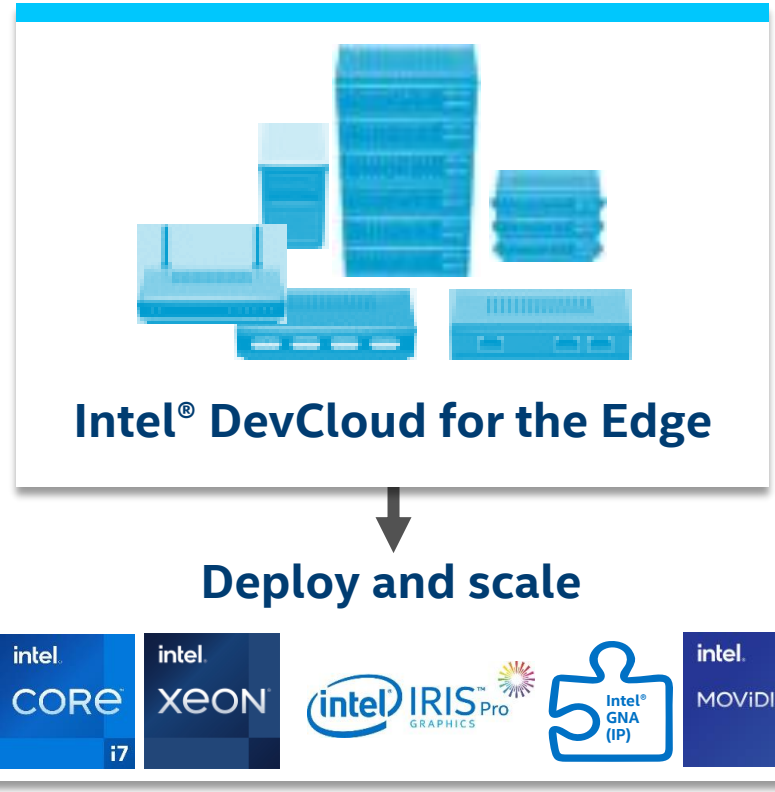
A Development Sandbox for Developers, Researchers, and Startups to Test AI and Vision Workloads Remotely before Deployment.

With the Intel® DevCloud for the Edge users can:

- **Prototype** on the latest hardware and software to future proof the solution
- **Benchmark** the customized AI application
- Run AI applications from **anywhere in the world**
- **Reduce** development time and cost

[New]DL Workbench + Intel® DevCloud for the Edge

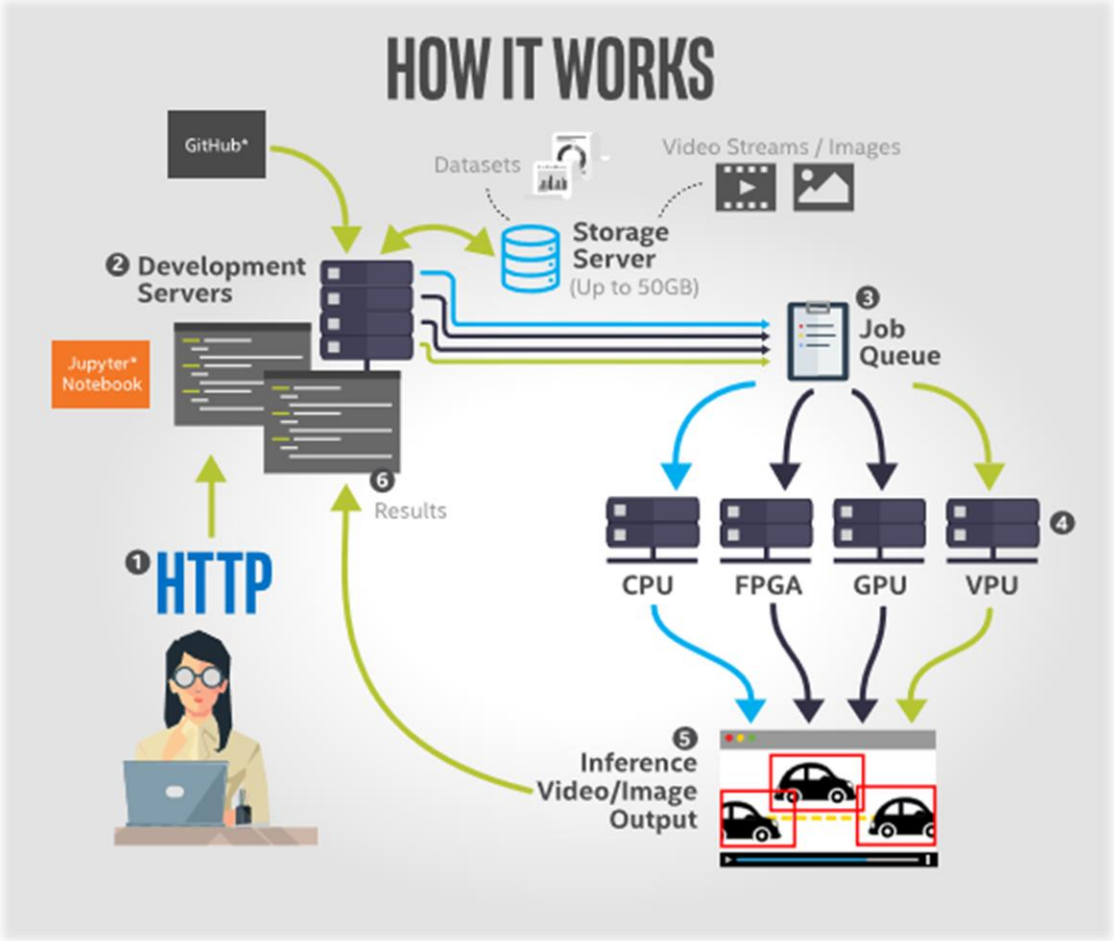
Developers can now graphically analyze models using the DL Workbench on Intel® DevCloud for the Edge (instead of local machine only) to compare, visualize and fine-tune a solution against multiple remote hardware configurations



For more information visit ▸ https://devcloud.intel.com/edge/

Accelerate Time to Production with Intel® DevCloud for the Edge

See immediate AI Model performance across Intel's vast array of Edge Solutions



- **Instant, Global Access**
  Run AI applications from anywhere in the world
- **Prototype on the Latest Hardware and Software**
  Develop knowing you're using the latest Intel technology
- **Benchmark your Customized AI Application**
  Immediate feedback - frames per second, performance
- **Reduce Development Time and Cost**
  Quickly find the right compute for your edge solution

Learn more

Sign up now for access

Demo

Pneumonia Classification with Class Activation Maps

https://notebooks.edge.devcloud.intel.com/user/u52111/notebooks/Reference-samples/iot-devcloud/openvino-lts/developer-samples/python/pneumonia-classification/classification_pneumonia.ipynb#

-->**Development Environment:** OpenVINO 2021.4 LTS Jupyter Notebook

Pneumonia Detection

Last Updated: 09/29/2021

Classify the probability of pneumonia in X-Ray images using a pretrained neural network and the Intel® Distribution of OpenVINO™ toolkit.

Ready to get started?

Download directly from Intel for free

Intel® Distribution of OpenVINO ™ toolkit
(Recommended)

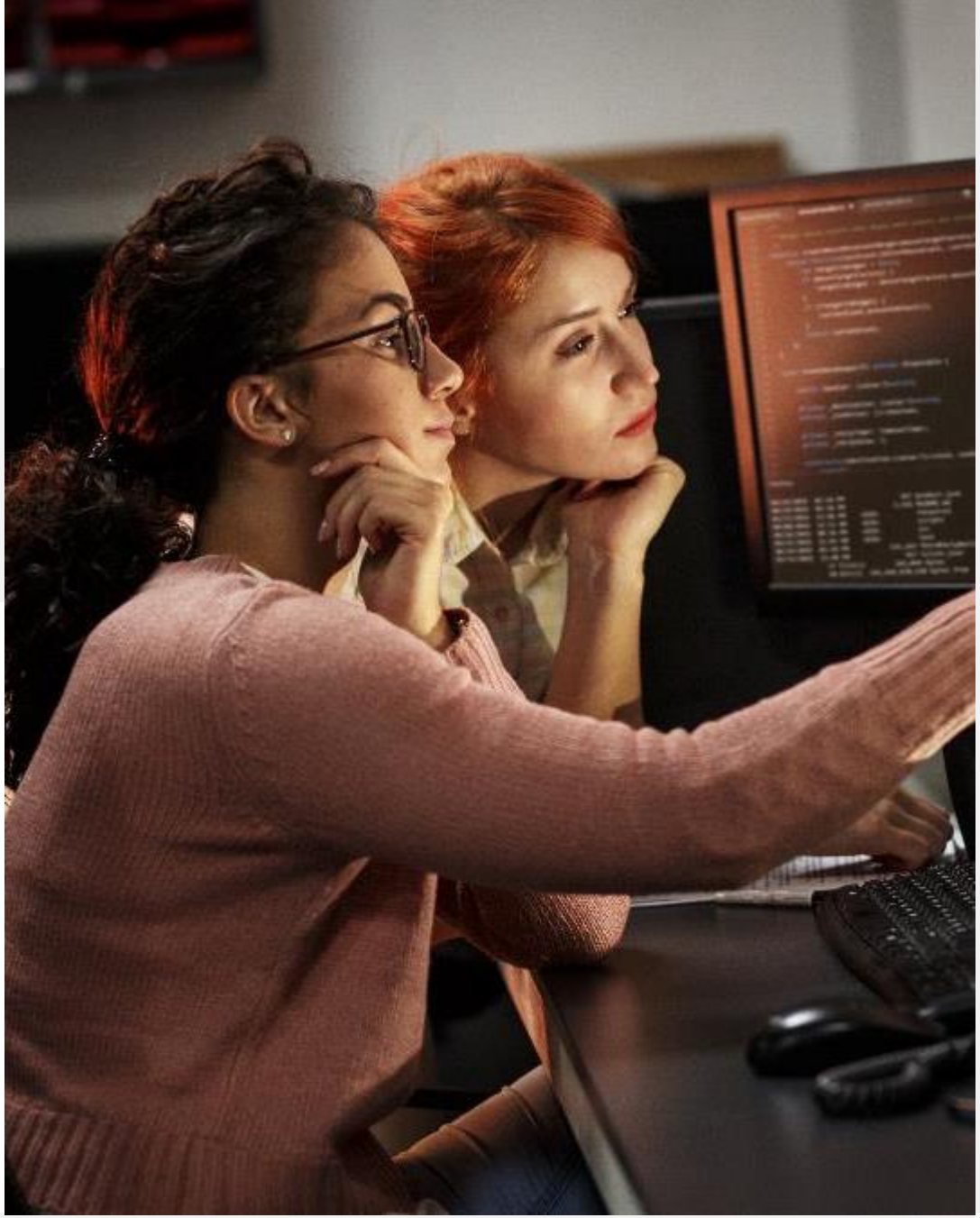
Also available from

Intel's Edge Software Hub | Intel® DevCloud for the Edge | PIP | DockerHub | Dockerfile | Anaconda Cloud | YUM | APT

Build from source

GitHub | Gitee (for China)

Choose & Download



Questions?


intel

Choose between Distributions

Tool/Component	Intel® Distribution of OpenVINO™ toolkit	OpenVINO™ toolkit (open source)	Open Source Directory
Installer (including necessary drivers)	✓		
Model Optimizer	✓	✓	https://github.com/openvinotoolkit/openvino/tree/master/model-optimizer
Inference Engine - Core	✓	✓	https://github.com/openvinotoolkit/openvino/tree/master/inference-engine
Intel CPU plug-in	✓ Intel® Math Kernel Library (Intel® MKL) only[1]	✓ BLAS, Intel® MKL[1], jit (Intel MKL)	https://github.com/openvinotoolkit/openvino/tree/master/inference-engine
Intel GPU (Intel® Processor Graphics) plug-in	✓	✓	https://github.com/openvinotoolkit/openvino/tree/master/inference-engine
Heterogeneous plug-in	✓	✓	https://github.com/openvinotoolkit/openvino/tree/master/inference-engine
Intel GNA plug-in	✓	✓	https://github.com/openvinotoolkit/openvino/tree/master/inference-engine
Intel® FPGA plug-in	✓		
Intel® Neural Compute Stick (1 & 2) VPU plug-in	✓	✓	https://github.com/openvinotoolkit/openvino/tree/master/inference-engine
Intel® Vision Accelerator based on Movidius plug-in	✓		
Multi-device & hetero plug-ins	✓	✓	
Public and Pretrained Models - incl. Open Model Zoo (IR models that run in IE + open sources models)	✓	✓	https://github.com/openvinotoolkit/open_model_zoo
Samples (APIs)	✓	✓	https://github.com/openvinotoolkit/openvino/tree/master/inference-engine
Demos	✓	✓	https://github.com/openvinotoolkit/open_model_zoo
Traditional Computer Vision			
OpenCV*	✓	✓	https://github.com/opencv/opencv
Intel® Media SDK	✓	✓[2]	https://github.com/Intel-Media-SDK/MediaSDK
OpenCL™ Drivers & Runtimes	✓	✓[2]	https://github.com/intel/compute-runtime
FPGA Runtime Environment, Deep Learning Acceleration & Bitstreams (Linux* only)	✓		

System Requirements

	Intel® Platforms	Compatible Operating Systems
Target Solution Platforms	**CPU** ▪ 6th-10th generation Intel® Core™ and Xeon® processors ▪ 1st and 2nd generation Intel® Xeon® Scalable processors	▪ Ubuntu* 18.04.3 LTS (64 bit) ▪ Microsoft Windows* 10 (64 bit) ▪ CentOS* 7.4 (64 bit) ▪ macOS* 10.13 & 10.14 (64 bit)
	▪ Intel® Pentium® processor N4200/5, N3350/5, N3450/5 with Intel® HD Graphics	▪ Yocto Project* Poky Jethro v2.0.3 (64 bit)
	Iris® Pro & Intel® HD Graphics ▪ 6th-10th generation Intel® Core™ processor with Intel® Iris™ Pro graphics & Intel® HD Graphics ▪ Intel® Xeon® processor with Intel® Iris™ Pro Graphics & Intel® HD Graphics (excluding E5 product family, which does not have graphics[1])	▪ Ubuntu 18.04.3 LTS (64 bit) ▪ Windows 10 (64 bit) ▪ CentOS 7.4 (64 bit)
	FPGA ▪ Intel® Arria® FPGA 10 GX development kit ▪ Intel® Programmable Acceleration Card with Intel® Arria® 10 GX FPGA operating systems ▪ OpenCV* & OpenVX* functions must be run against the CPU or Intel® Processor Graphics (GPU)	▪ Ubuntu 18.04.2 LTS (64 bit) ▪ CentOS 7.4 (64 bit)
	VPU: Intel Movidius™ Neural Compute Stick:, Intel® Neural Compute Stick2	▪ Ubuntu 18.04.3 LTS (64 bit) CentOS 7.4 (64 bit) ▪ Windows 10 (64 bit) macOS* (64 bit) Raspbian (target only)
	Intel® Vision Accelerator Design Products ▪ Intel® Vision Accelerator Design with Intel® Arria10 FPGA	▪ Ubuntu 18.04.2 LTS (64 bit)
	▪ Intel® Vision Accelerator Design with Intel® Movidius™ VPUs	▪ Ubuntu 8.04.3 LTS (64 bit) ▪ Windows 10 (64 bit)
Development Platforms	▪ 6th-10th generation Intel® Core™ and Intel® Xeon® processors ▪ 1st and 2nd generation Intel® Xeon® Scalable processors	▪ Ubuntu* 18.04.3 LTS (64 bit) ▪ Windows® 10 (64 bit) ▪ CentOS* 7.4 (64 bit) ▪ macOS* 10.13 & 10.14 (64 bit)
Additional Software Requirements	Linux* build environment required components ▪ OpenCV 3.4 or higher ▪ CMake* 2.8 or higher ▪ GNU Compiler Collection (GCC) 3.4 or higher ▪ Python* 3.4 or higher Microsoft Windows* build environment required components ▪ Intel® HD Graphics Driver (latest version)† ▪ Intel® C++ Compiler 2017 Update 4 ▪ Python 3.4 or higher ▪ OpenCV 3.4 or higher ▪ CMake 2.8 or higher ▪ Microsoft Visual Studio* 2015	
External Dependencies/Additional Software		View Product Site, detailed System Requirements

Commonly Asked Questions

Can I use the Intel® Distribution of OpenVINO™ toolkit for commercial usage? Yes, the Intel® Distribution of OpenVINO™ toolkit is licensed under Intel's End User License Agreements and the open-sourced OpenVINO™ toolkit is licensed under Apache License 2.0. For information, review the licensing directory inside the package.

Is the Intel® Distribution of OpenVINO™ toolkit subject to export control? Yes, the ECCN is EAR99.

How often does the software get updated? Standard releases are updated 3-4 times a year, while LTS releases are updated once a year.

What is the difference between Standard and LTS releases? Standard Releases are recommended for new users and users currently prototyping. It offers new features, tools and support to stay current with deep learning advancements. LTS Releases are recommended for experienced users that are ready to take their application into production and who do not require new features and capabilities for their application.

For technical questions, visit the Model Optimizer FAQ and Performance Benchmarks FAQ. If you don't find an answer, please visit the following community and support links.

Get Help

- Ask on the Community Forum
- Contact Intel Support
- File an Issue on GitHub*
- Get Answers on StackOverflow*

Get Involved

- Contribute to the Code Base
- Contribute to Documentation

Stay Informed

- Join the Mailing List
- Read the Documentation
- Read the Knowledge Base
- Read the Blog