Choose the Best Accelerated Technology

Intel® Distribution of OpenVINO™ toolkit

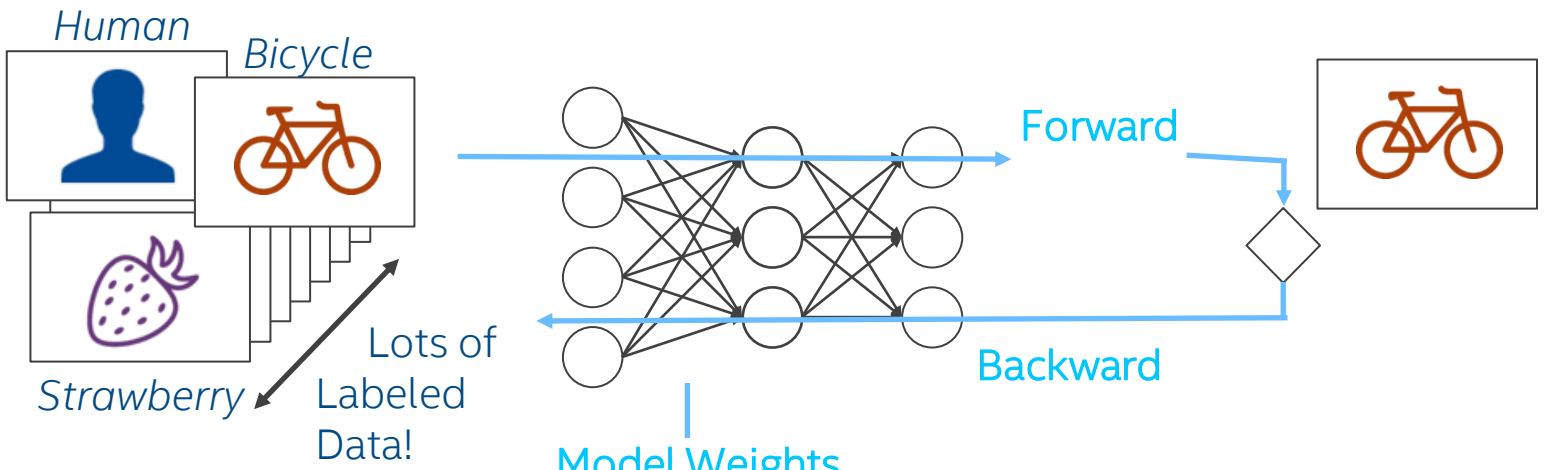
LRZ AI Workshop

Roy Allela – AI Software Solutions Engineer

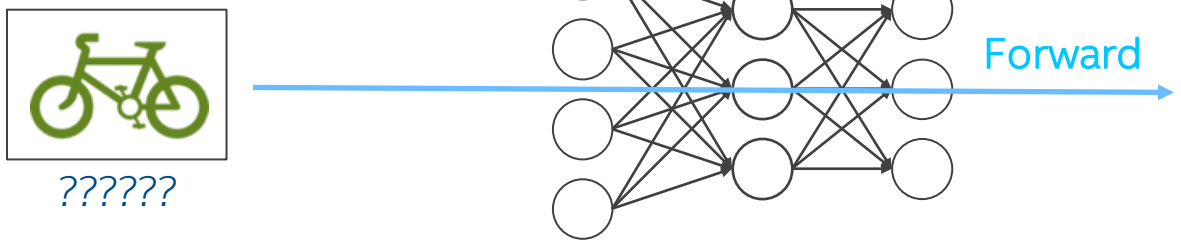
21.04.2022

Deep Learning: Training vs. Inference

Training

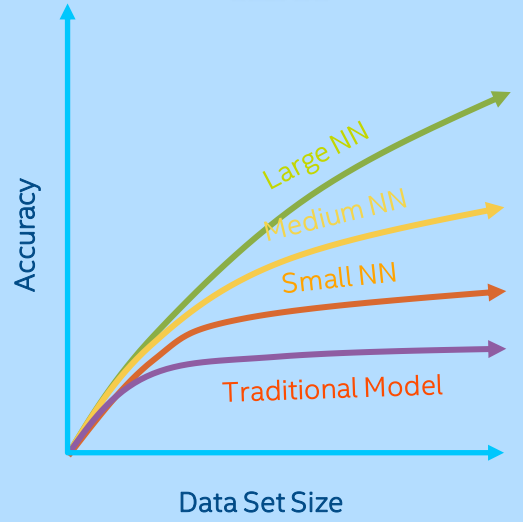


Inference



Did You Know?

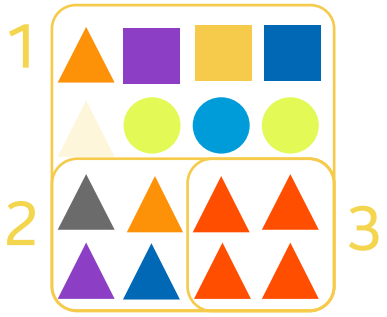
Training requires a very large data set and deep neural network (many layers) to achieve the highest accuracy in most cases



AI Compute Considerations

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

WORKLOADS

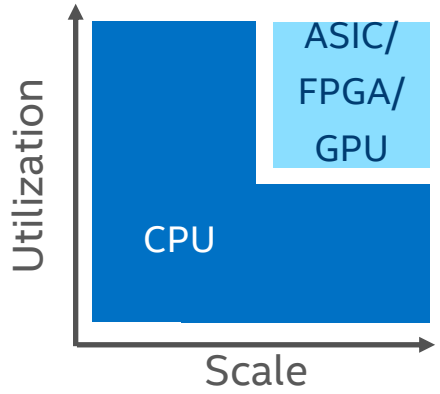


What is my workload profile?

REQUIREMENTS

What are my use case requirements?

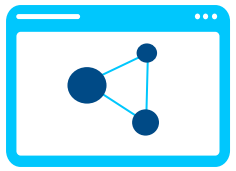
DEMAND



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

1 Build

Trained Model

TensorFlow Caffe mxnet KALDI ONNX PyTorch PaddlePaddle

Open Model Zoo
280+ open sourced & optimized pre-trained models available

2 Optimize

Model Optimizer
Converts and optimizes trained model using a supported framework

Read, Load, Infer

IR Data

Intermediate Representation (.xml, .bin)

Inference Engine
Common API that abstracts low-level programming for each hardware

Post-Training Optimization Tool
Reduces model size into low-precision without re-training

Deep Learning Workbench
Visually analyze and fine-tune

Deep Learning Streamer

OpenCV | **OpenCL™**

Code Samples & Demos (e.g. Benchmark app, Accuracy Checker, Model Downloader)

3 Deploy

Model Server
gRPC/ REST Server with C++ backend

Deployment Manager

intel ATOM | intel CORE i7 | intel XEON

intel IRIS Pro GRAPHICS | intel IRIS MAX GRAPHICS

intel MOVIDIUS | Intel® GNA (IP)

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

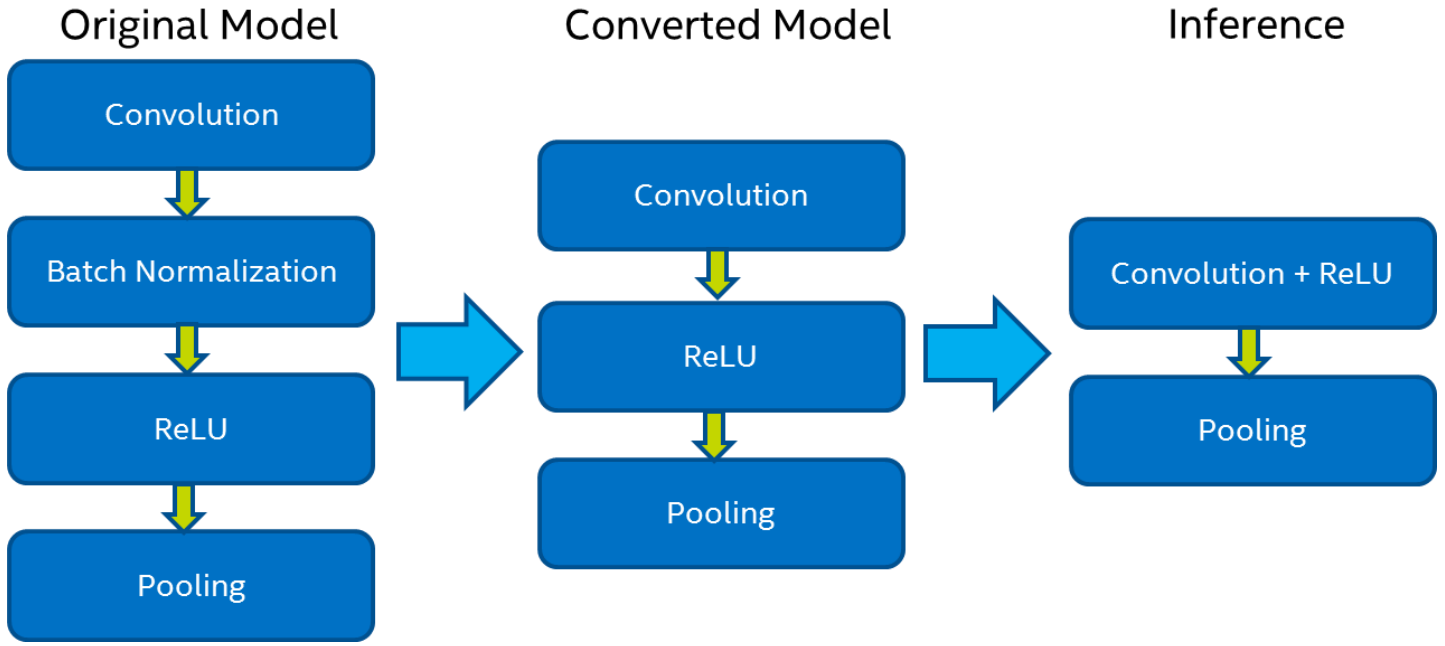
Note: Except for ONNX (.onnx model formats), all models have to be converted to an IR format to use as input to the Inference Engine

.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.



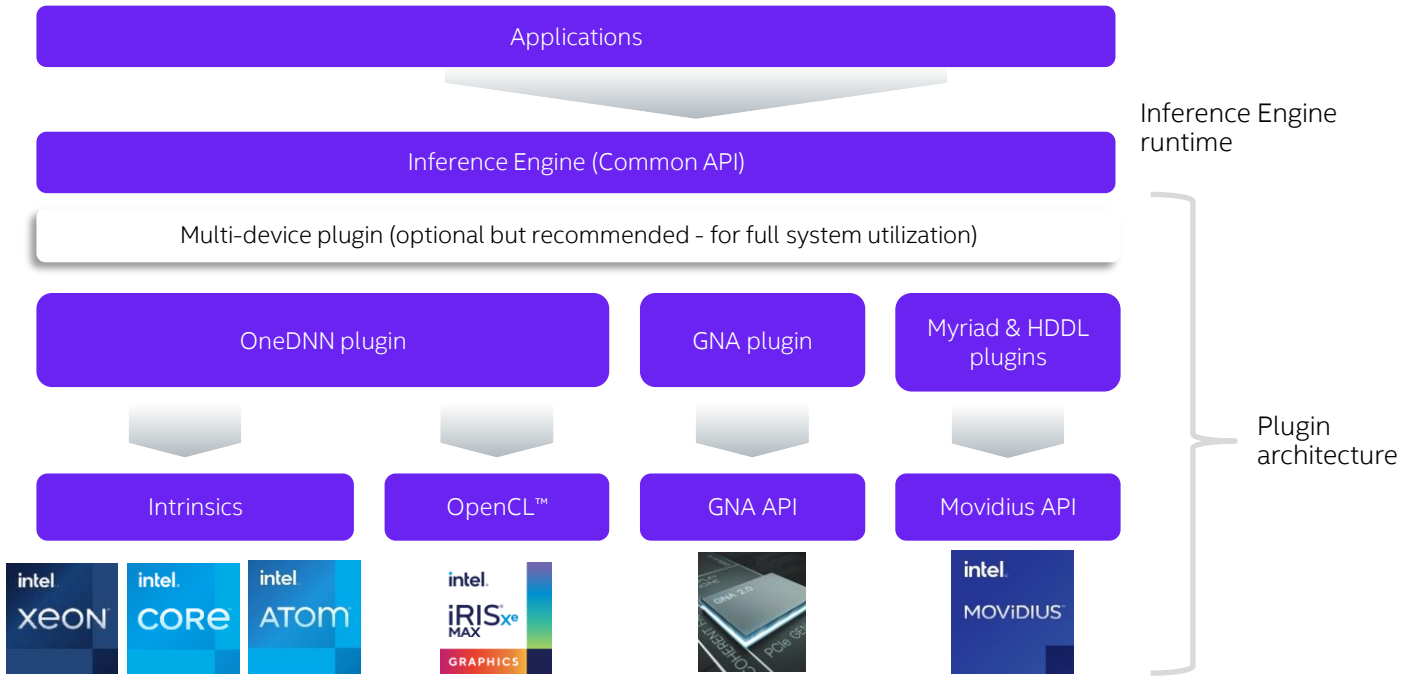
Optimal Model Performance Using the Inference Engine

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

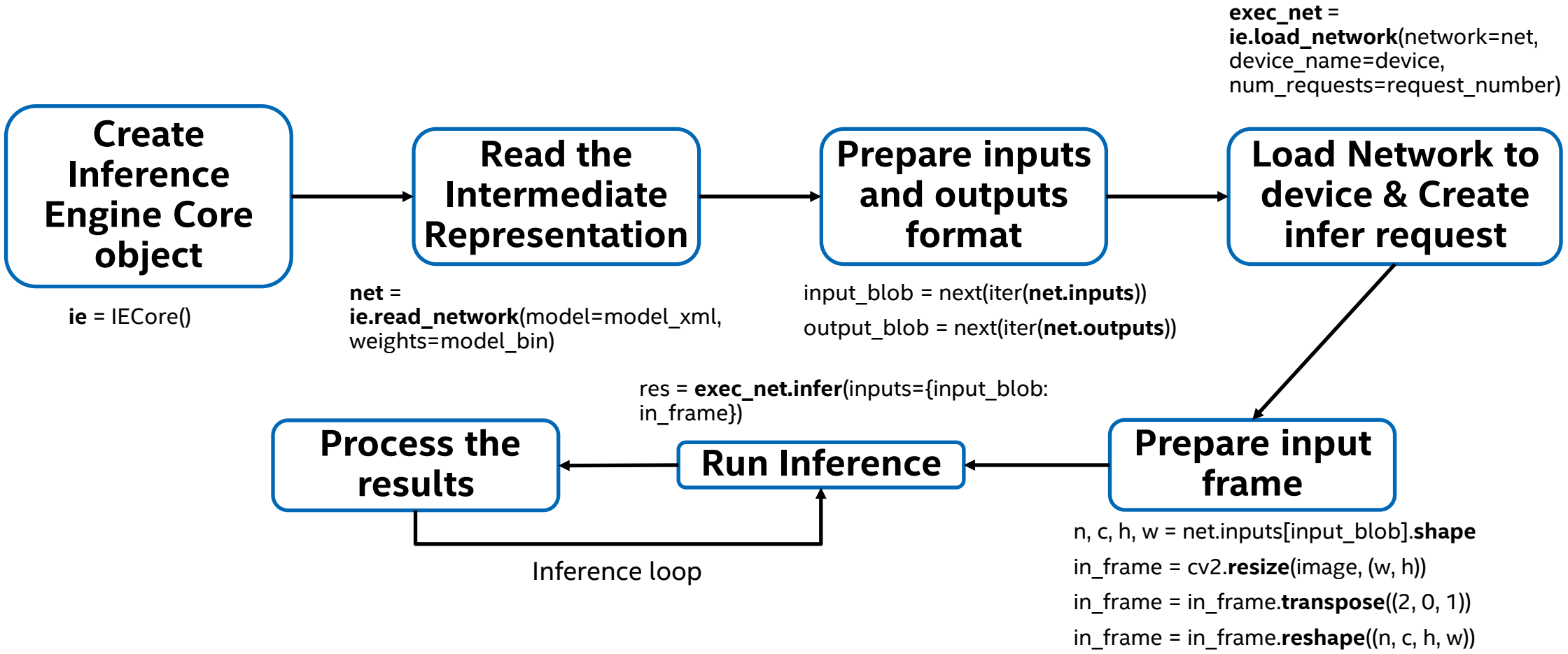
- For each supported target device, Inference Engine provides a plugin — a DLL/shared library that contains complete implementation for inference on this device.



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

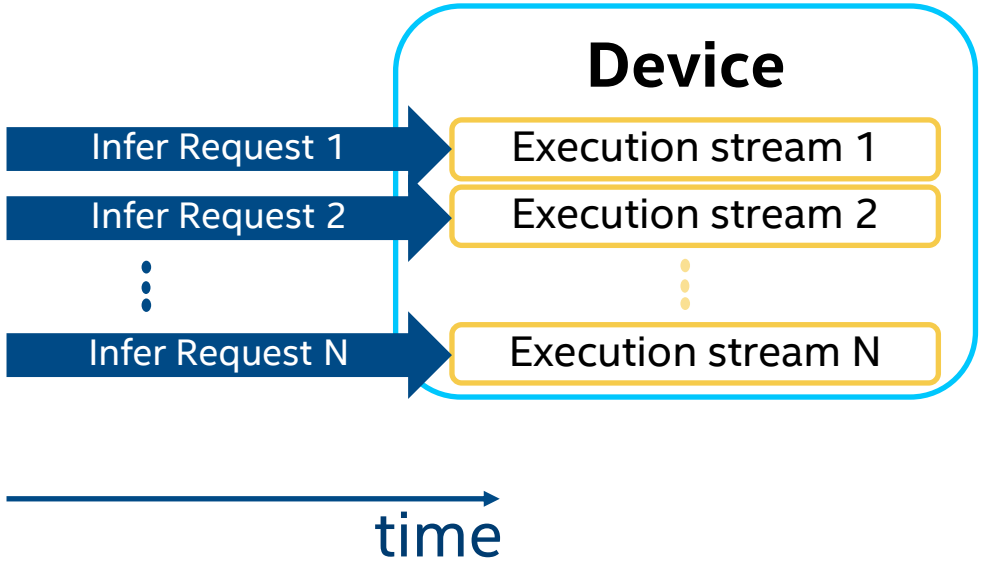


http://docs.openvinotoolkit.org/latest/_docs_IE_DG_Integrate_with_customer_application_new_API.html

Inference Engine

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



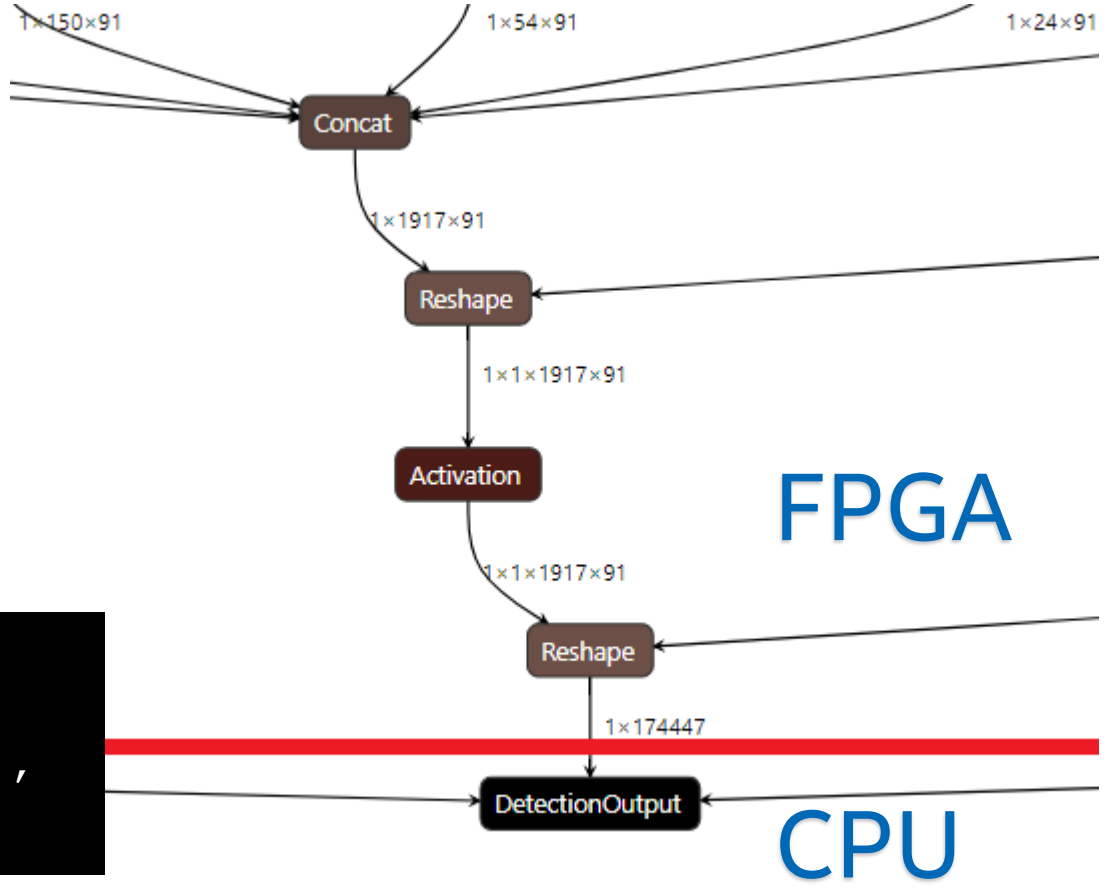
CPU Example:

```
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");
```

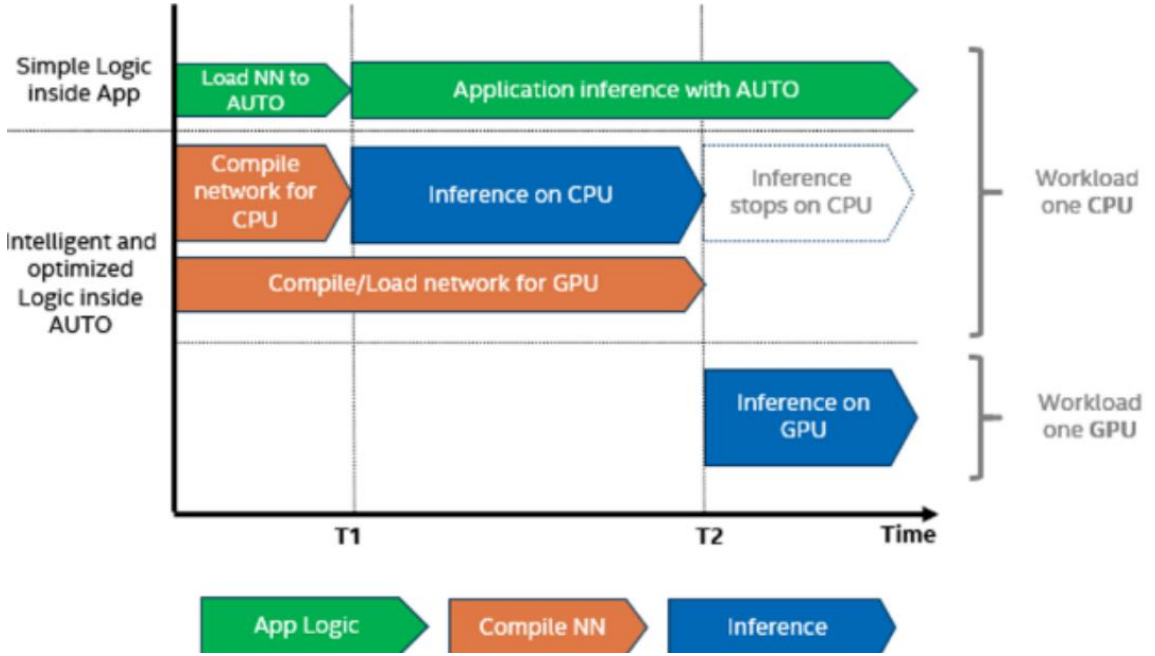
Inference Engine

Automatic Device Selection: AUTO

- AUTO discovers the accelerators & capabilities in the system, chooses and configures the device matching to the NN's precision, topology, user's performance requirements.

- Application portability, HW agnostic, dispatching workload to best fit device (CPU, GPU, VPU).

- AUTO always chooses the best device, if compiling model fails on this device, AUTO will try to compile it on next best device until one of them succeeds. If priority list is set, AUTO only select devices according to the list.




```
ov::CompiledModel model0 = core.compile_model(model);
ov::CompiledModel model1 = core.compile_model(model, "AUTO");
ov::CompiledModel model2 = core.compile_model(model, "AUTO", {});
```

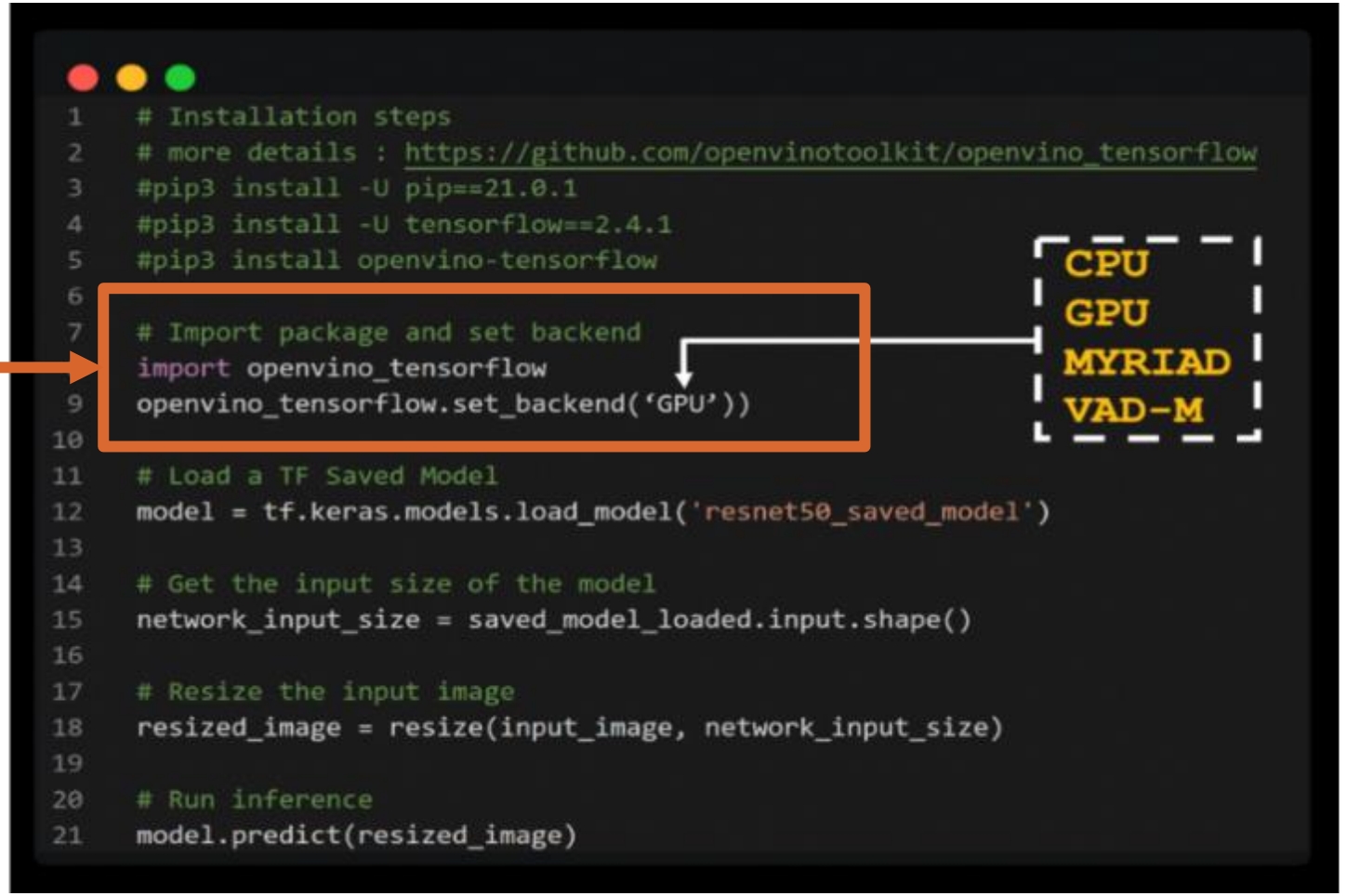
OpenVINO as execution provider

- You can use OpenVINO Inference Engine as backend of other DL Inference Frameworks such as Tensorflow or 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)
```

Pre-trained Models

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

Intel Pre-trained Models

Action Recognition Models
Classification Models
Head Pose Estimation Models
Human Pose Estimation Models
Image Processing Models
Instance Segmentation Models
Machine Translation Models
Object Attribute Estimation Models
Object Detection Models
Optical Character Recognition Models
Question Answering Models
Semantic Segmentation Models
Text-to-speech Models
Token Recognition Models

Public Pre-trained Models

Action Recognition Models
Classification Models
Colorization Models
Face Recognition Models
Human Pose Estimation Models
Image Inpainting Models
Image Processing Models
Image Translation Models
Instance Segmentation Models
Monocular Depth Estimation Models
Object Attribute Estimation Models
Object Detection Models
Optical Character Recognition Models
Place Recognition Models
Semantic Segmentation Models
Sound Classification Models
Speech Recognition Models
Style Transfer Models
Text-to-speech Models

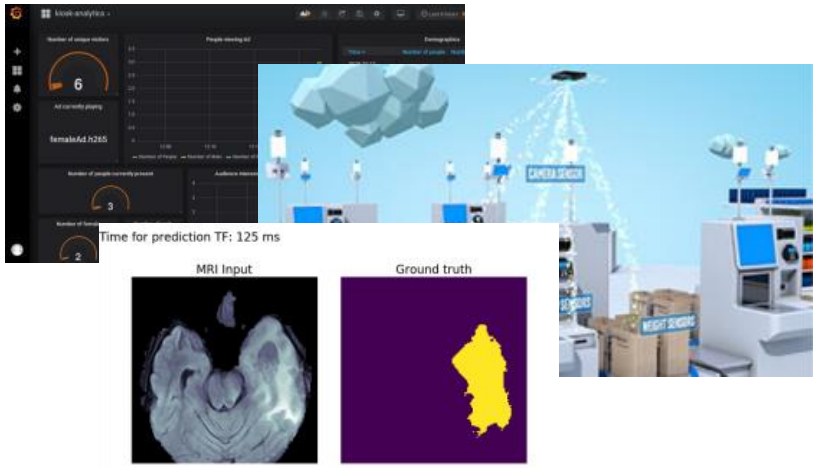
PRE-TRAINED MODELS

https://github.com/openvinotoolkit/open_model_zoo/tree/master/models

Demos Applications

Quickly get started with example demo applications

Take advantage of **pre-built, open-sourced** example implementations with step-by-step guidance and required components list



[3D Human Pose Estimation Python* Demo]
[3D Segmentation Python* Demo]
[Action Recognition Python* Demo]
[BERT Question Answering Embedding Python* Demo]
[BERT Question Answering Python* Demo]
[Classification C++ Demo]
[Colorization Demo]
[Crossroad Camera C++ Demo]
[Face Detection MTCNN Python* Demo]
[Formula Recognition Python* Demo]
[G-API Interactive Face Detection Demo]
[Gaze Estimation Demo]
[Gesture Recognition Python* Demo]
[Handwritten Text Recognition Demo]
[Human Pose Estimation C++ Demo]
[Human Pose Estimation Python* Demo]
[Image Deblurring Python* Demo]
[Image Inpainting Python Demo]
[Image Retrieval Python* Demo]
[Image Segmentation C++ Demo]
[Image Segmentation Python* Demo]
[Image Translation Demo]
[Instance Segmentation Python* Demo]
[Interactive Face Detection C++ Demo]
[Machine Translation Python* Demo]
[MonoDepth Python Demo]
[Multi Camera Multi Target Python* Demo]
[Multi-Channel Face Detection C++ Demo]
[Multi-Channel Human Pose Estimation C++ Demo]
[Multi-Channel Object Detection Yolov3 C++ Demo]
[Object Detection C++ Demo]
[Object Detection Python* Demo]
[Pedestrian Tracker C++ Demo]
[Place Recognition Python* Demo]
[Security Barrier Camera C++ Demo]
[Single Human Pose Estimation Demo (top-down pipeline)]
[Smart Classroom C++ Demo]
[Sound Classification Python* Demo]
[Speech Recognition Demo]
[Super Resolution C++ Demo]
[TensorFlow* Object Detection Mask R-CNNs Segmentation C++ Demo]
[Text Detection C++ Demo]
[Text Spotting Python* Demo]
[Text-to-speech Python* Demo]
[Whiteboard Inpainting Demo]

DEMO APPLICATIONS

https://github.com/openvinotoolkit/open_model_zoo/tree/master/demos

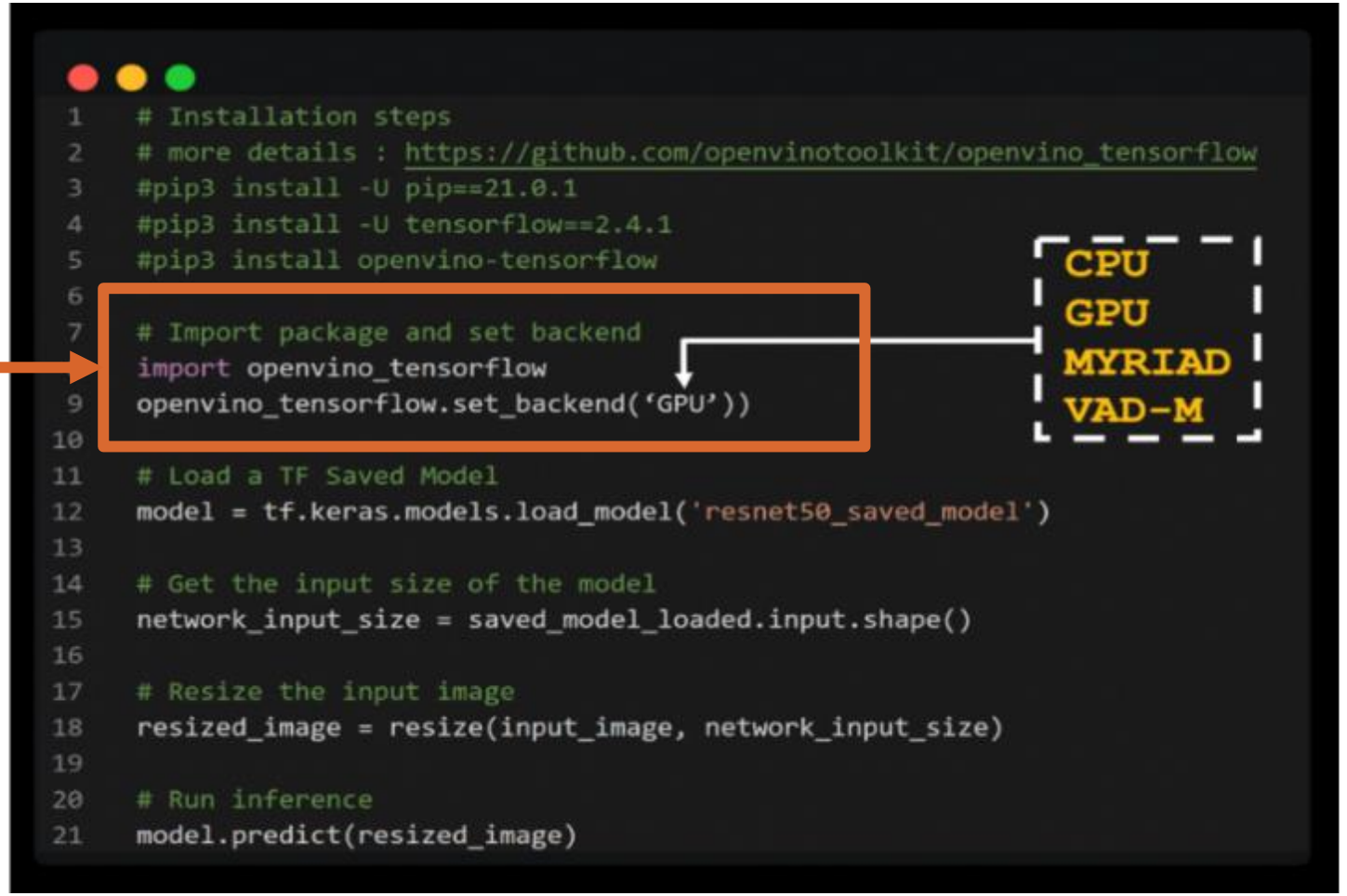
OpenVINO as execution provider

- You can use OpenVINO Inference Engine as backend of other DL Inference Frameworks such as Tensorflow or 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



Pre-trained Models

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

Intel Pre-trained Models

[Action Recognition Models](#)
[Classification Models](#)
[Head Pose Estimation Models](#)
[Human Pose Estimation Models](#)
[Image Processing Models](#)
[Instance Segmentation Models](#)
[Machine Translation Models](#)
[Object Attribute Estimation Models](#)
[Object Detection Models](#)
[Optical Character Recognition Models](#)
[Question Answering Models](#)
[Semantic Segmentation Models](#)
[Text-to-speech Models](#)
[Token Recognition Models](#)

Public Pre-trained Models

[Action Recognition Models](#)
[Classification Models](#)
[Colorization Models](#)
[Face Recognition Models](#)
[Human Pose Estimation Models](#)
[Image Inpainting Models](#)
[Image Processing Models](#)
[Image Translation Models](#)
[Instance Segmentation Models](#)
[Monocular Depth Estimation Models](#)
[Object Attribute Estimation Models](#)
[Object Detection Models](#)
[Optical Character Recognition Models](#)
[Place Recognition Models](#)
[Semantic Segmentation Models](#)
[Sound Classification Models](#)
[Speech Recognition Models](#)
[Style Transfer Models](#)
[Text-to-speech Models](#)

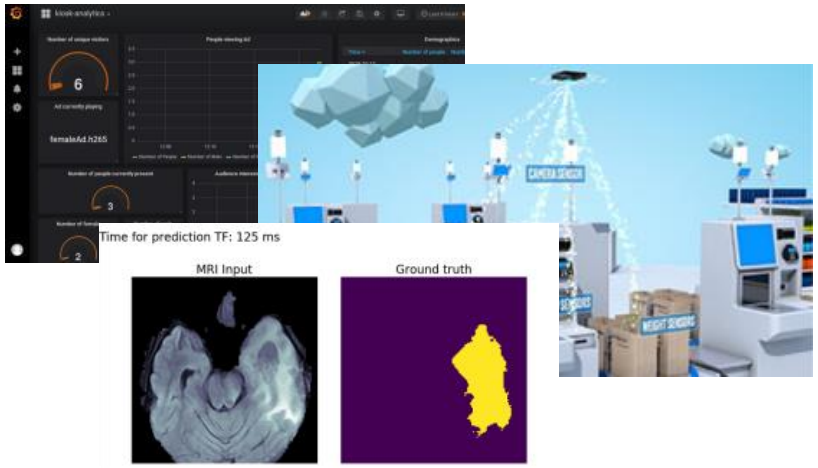
PRE-TRAINED MODELS

https://github.com/openvinotoolkit/open_model_zoo/tree/master/models

Demos Applications

Quickly get started with example demo applications

Take advantage of **pre-built, open-sourced** example implementations with step-by-step guidance and required components list



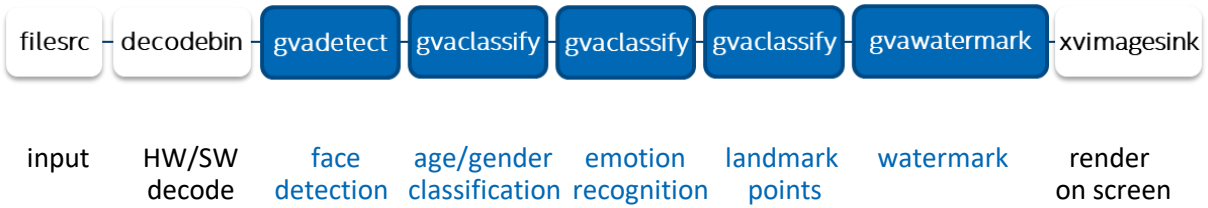
[3D Human Pose Estimation Python* Demo]
[3D Segmentation Python* Demo]
[Action Recognition Python* Demo]
[BERT Question Answering Embedding Python* Demo]
[BERT Question Answering Python* Demo]
[Classification C++ Demo]
[Colorization Demo]
[Crossroad Camera C++ Demo]
[Face Detection MTCNN Python* Demo]
[Formula Recognition Python* Demo]
[G-API Interactive Face Detection Demo]
[Gaze Estimation Demo]
[Gesture Recognition Python* Demo]
[Handwritten Text Recognition Demo]
[Human Pose Estimation C++ Demo]
[Human Pose Estimation Python* Demo]
[Image Deblurring Python* Demo]
[Image Inpainting Python Demo]
[Image Retrieval Python* Demo]
[Image Segmentation C++ Demo]
[Image Segmentation Python* Demo]
[Image Translation Demo]
[Instance Segmentation Python* Demo]
[Interactive Face Detection C++ Demo]
[Machine Translation Python* Demo]
[MonoDepth Python Demo]
[Multi Camera Multi Target Python* Demo]
[Multi-Channel Face Detection C++ Demo]
[Multi-Channel Human Pose Estimation C++ Demo]
[Multi-Channel Object Detection Yolov3 C++ Demo]
[Object Detection C++ Demo]
[Object Detection Python* Demo]
[Pedestrian Tracker C++ Demo]
[Place Recognition Python* Demo]
[Security Barrier Camera C++ Demo]
[Single Human Pose Estimation Demo (top-down pipeline)]
[Smart Classroom C++ Demo]
[Sound Classification Python* Demo]
[Speech Recognition Demo]
[Super Resolution C++ Demo]
[TensorFlow* Object Detection Mask R-CNNs Segmentation C++ Demo]
[Text Detection C++ Demo]
[Text Spotting Python* Demo]
[Text-to-speech Python* Demo]
[Whiteboard Inpainting Demo]

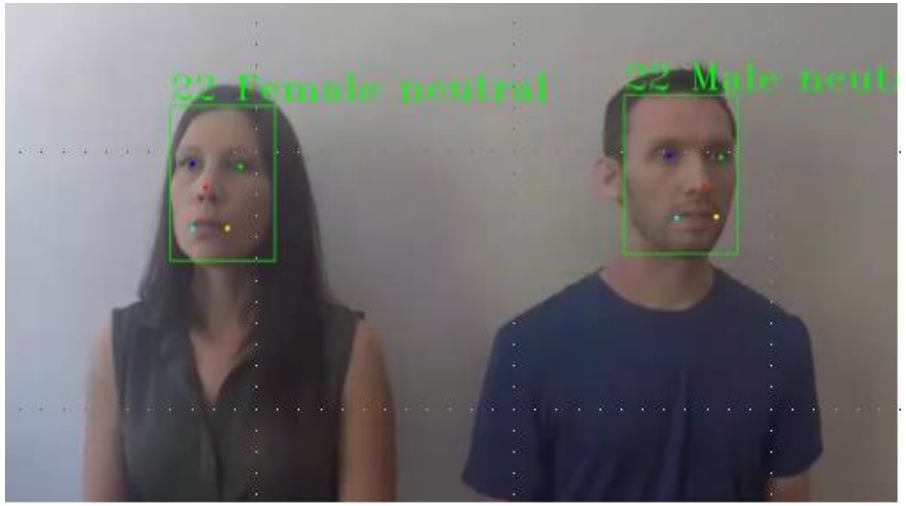
DEMO APPLICATIONS

https://github.com/openvinotoolkit/open_model_zoo/tree/master/demos

Video Analytics Pipeline

Detection and Classification





```
gst-launch-1.0 filesrc location=/path/to/video.mp4 !
decodebin ! \
gvadetect model=face-detection-adas-0001.xml model-proc=face-detection-adas-0001.json ! queue ! \
gvaclassify device=CPU model=age-gender-recognition.xml model-proc=age-gender-recognition.json ! queue ! \
gvaclassify device=GPU model=emotions-recognition.xml model-proc=emotions-recognition.json ! queue ! \
gvaclassify device=MULTI:CPU,GPU model=landmarks-regression.xml model-proc=landmarks-regression.json ! queue ! \
gvawatermark ! ximagesink
```

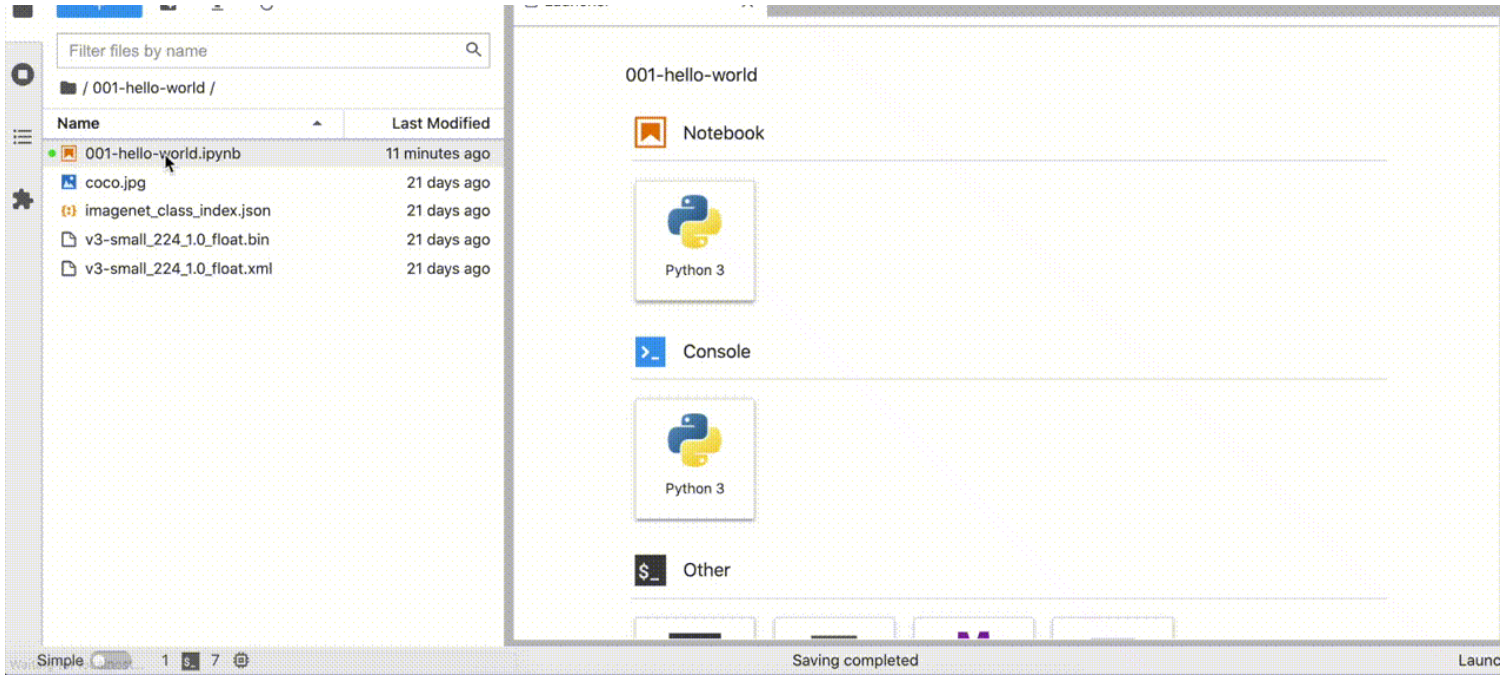
Release 2022.1

OpenVINO Notebooks

Python Demos and Tutorials for OpenVINO

How to get OpenVINO

setup and running in 5 minutes?



HARDWARE SUPPORTED

CPU

GPU

VPU

GNA

OS SUPPORTED

WIN 10

WIN SERVER

LINUX

MAC

Intel® DevCloud for the Edge

Accelerate Test Cycles with the Intel® DevCloud for the Edge

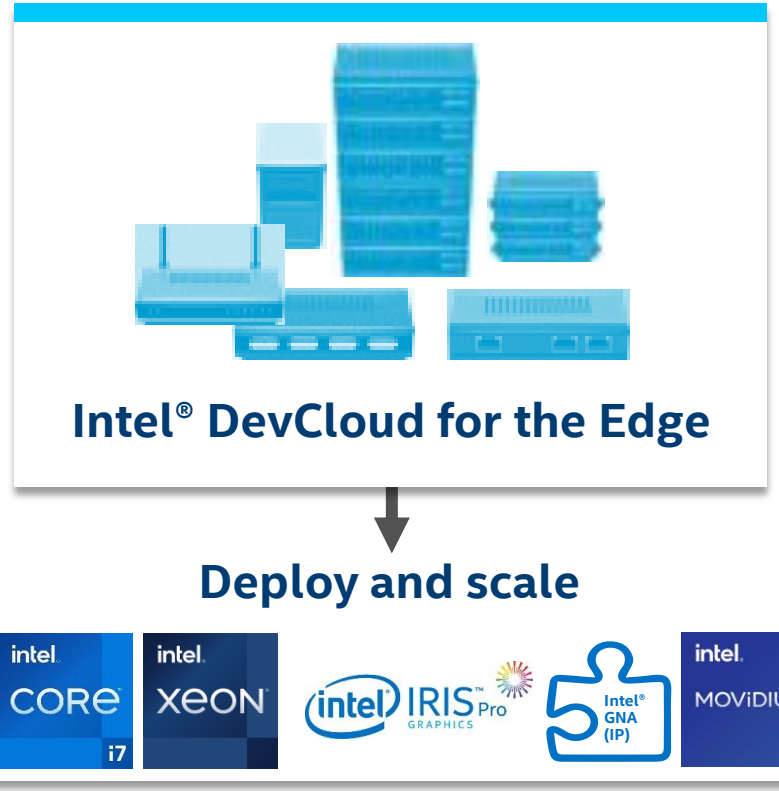
A Development Sandbox for Developers, Researchers, and Startups to Test AI and Vision Workloads Remotely before Deployment.

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

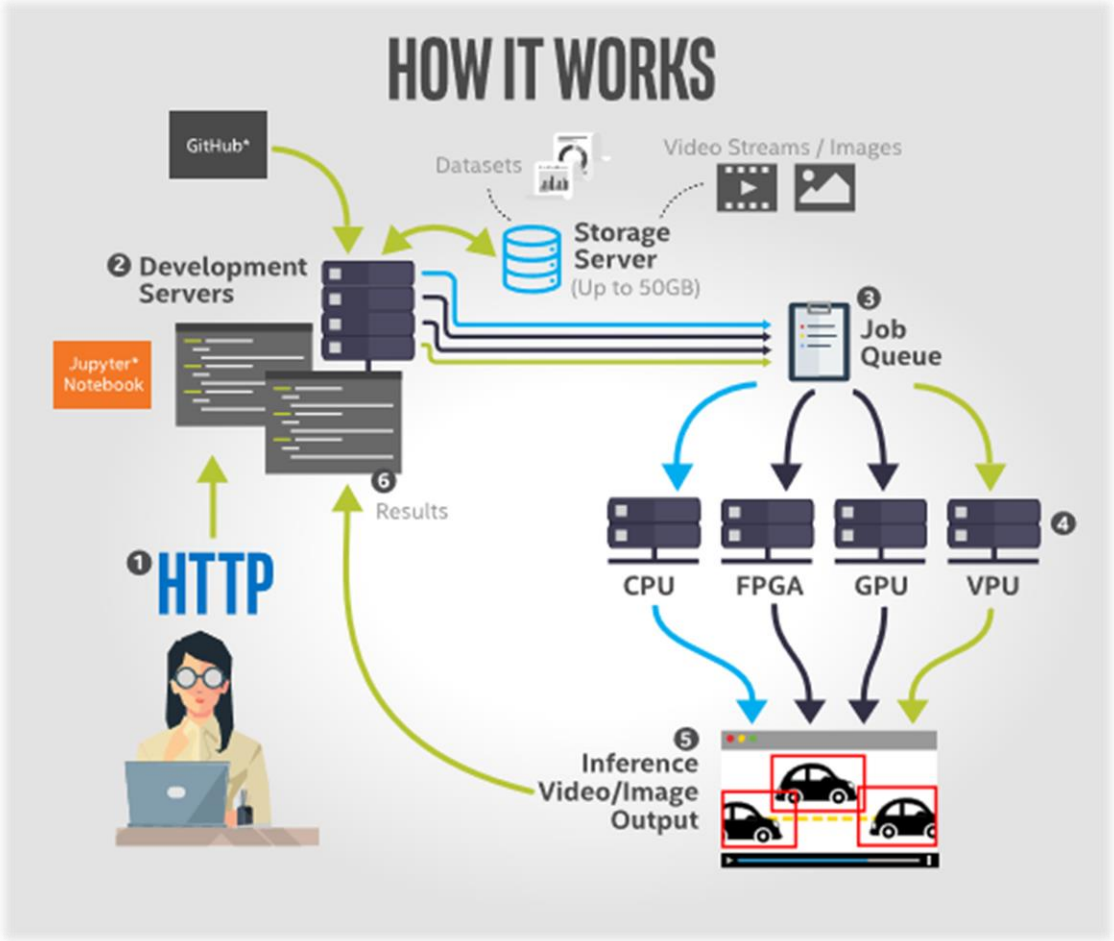
Developers can now graphically analyze models using the DL Workbench on Intel® DevCloud for the Edge (instead of local machine only) to compare, visualize and fine-tune a solution against multiple remote hardware configurations



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

Ready to get started?

Download directly from Intel for free

Intel® Distribution of OpenVINO ™ toolkit
(Recommended)

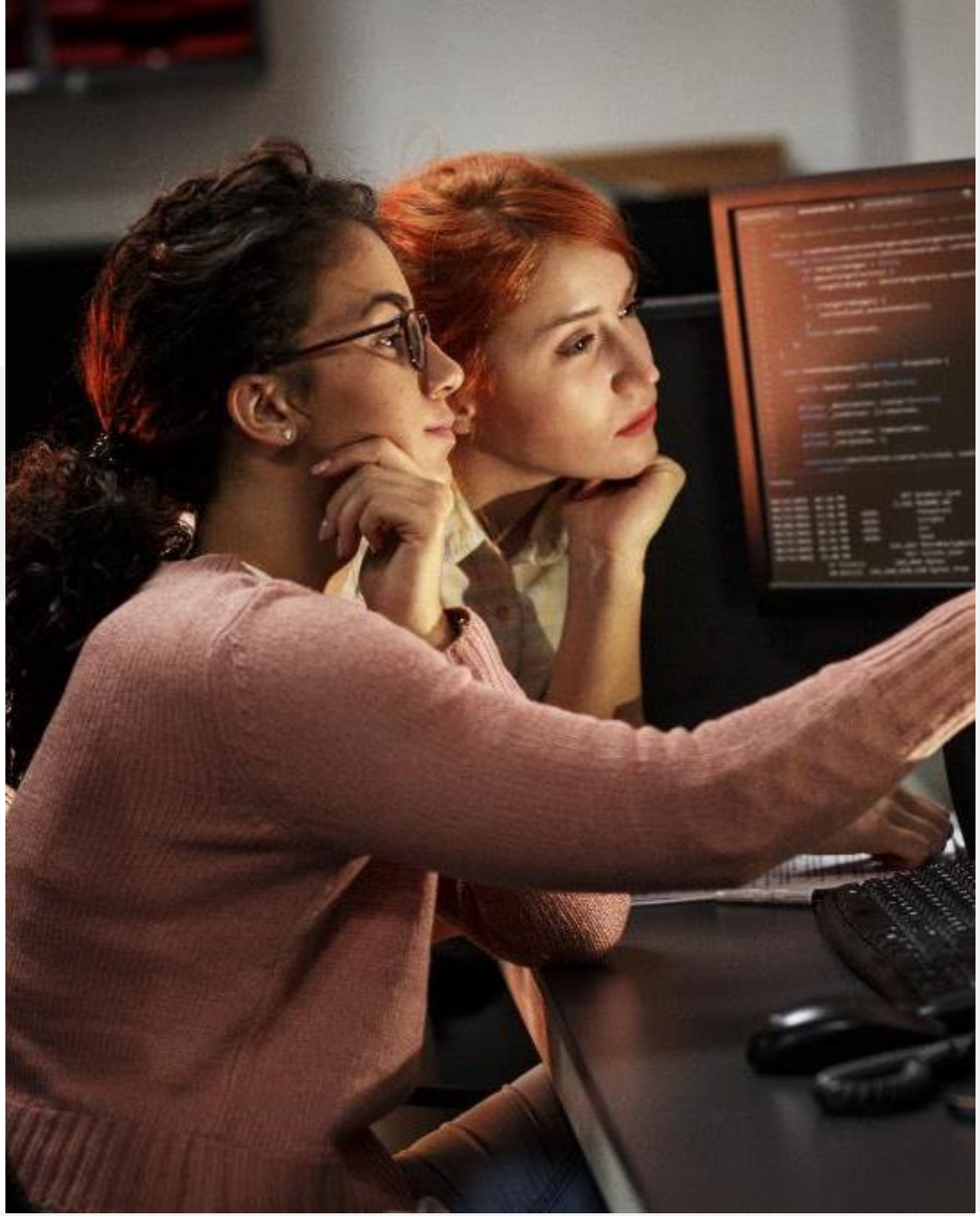
Also available from

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