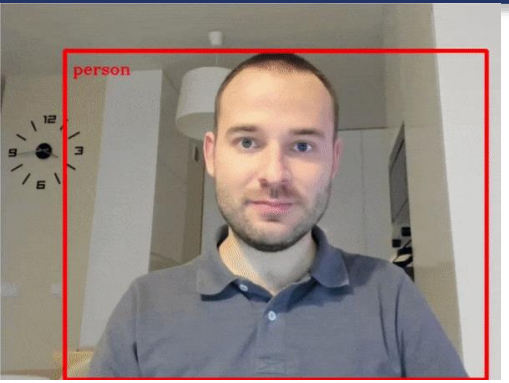
Unit 4

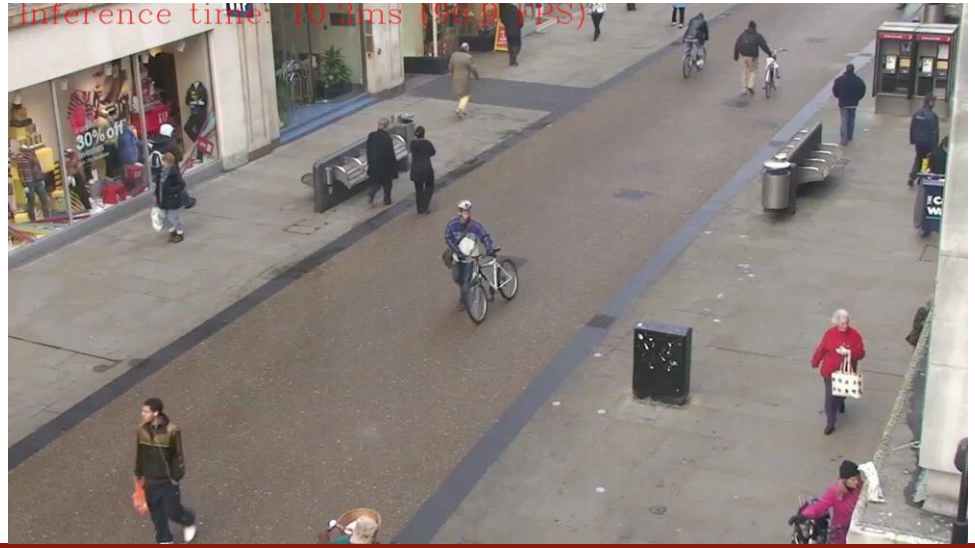
Computer Vision with Open VINO

Disclaimer

The content is curated from online/offline resources and used for educational purpose only

Computer Vision with Open VINO





Learning Objectives

- Introduction to Intel OpenVINO
- OpenVINO Toolkit and components
- Model Optimizer
- Hands on application of OpenVINO

Intel OpenVINO Toolkit

OpenVINO stands for Open Visual Inferencing and Neural Network Optimization.

- Designed to speed up networks used in visual inferencing tasks like image classification and object detection.
- DNNs used for solving visual tasks these days are Convolutional Neural Networks (CNN).
- OpenVINO speeds up computation by first optimizing the neural network model in a hardware agnostic way using a model optimizer followed by hardware-specific acceleration accomplished using the OpenVINO Inference Engine for the particular hardware.

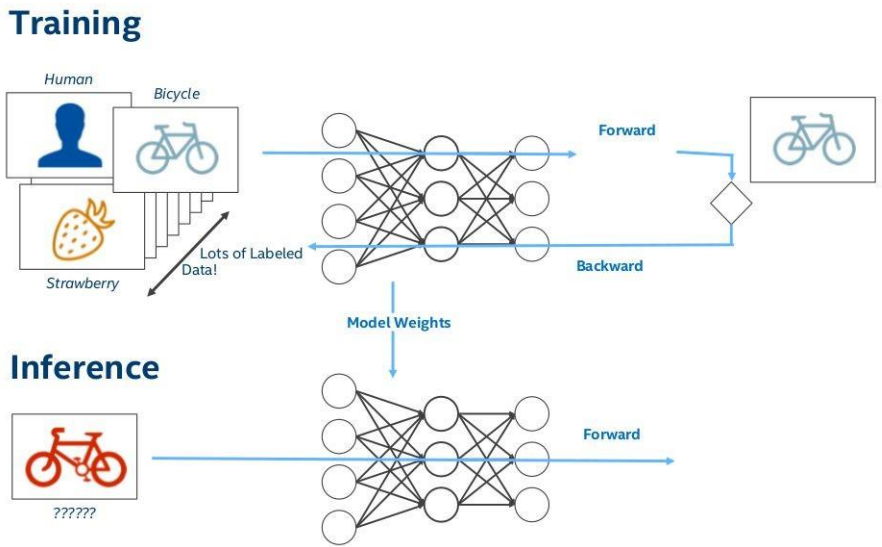
Why OpenVINO ?

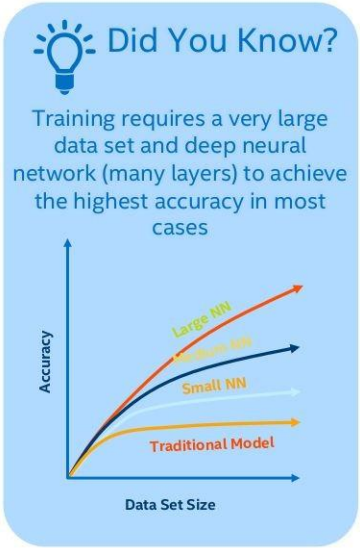
Practical	Advantage of built-in Intel processors
Efficient	Optimize your AI model for production
Effective	Improves hardware utilization
Adaptive	Change the h/w easily
Resourceful	More than 40 pre-ttained models
Open	Opensource in nature

Click here

Reference link

Deep Learning: Training vs. Inference

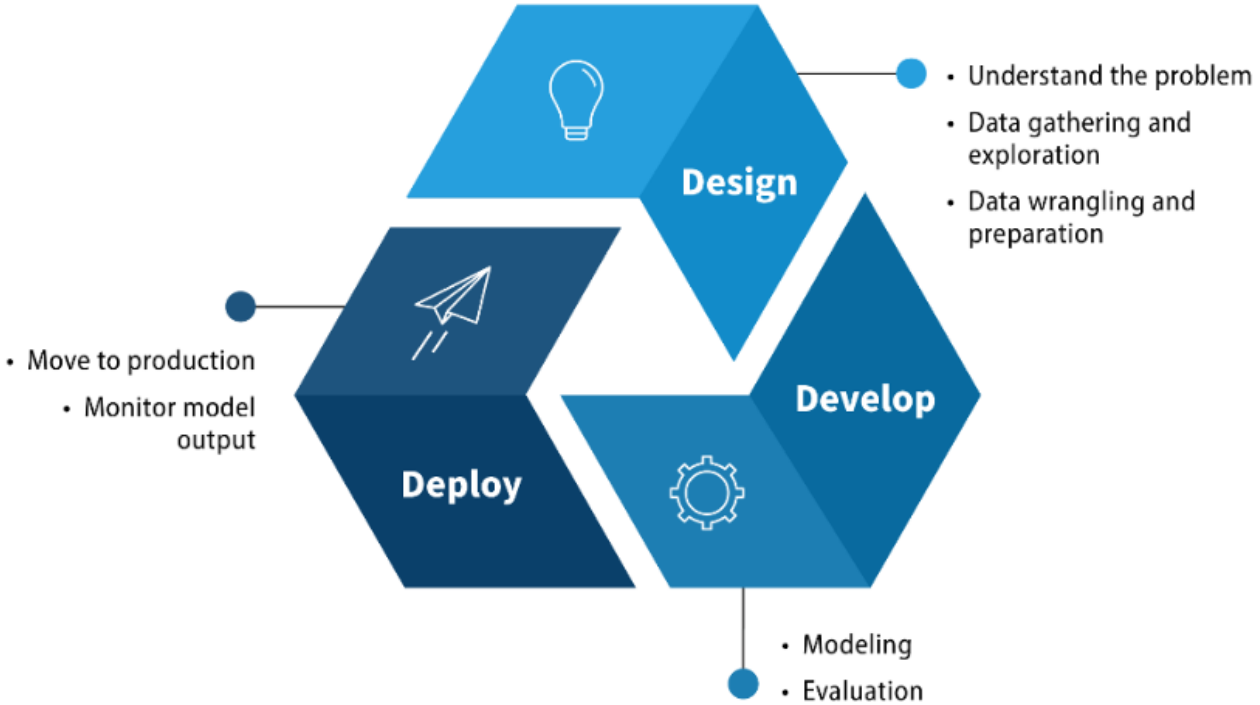




Click here

Reference link

Artificial Intelligence Development Cycle



Click here

Reference link

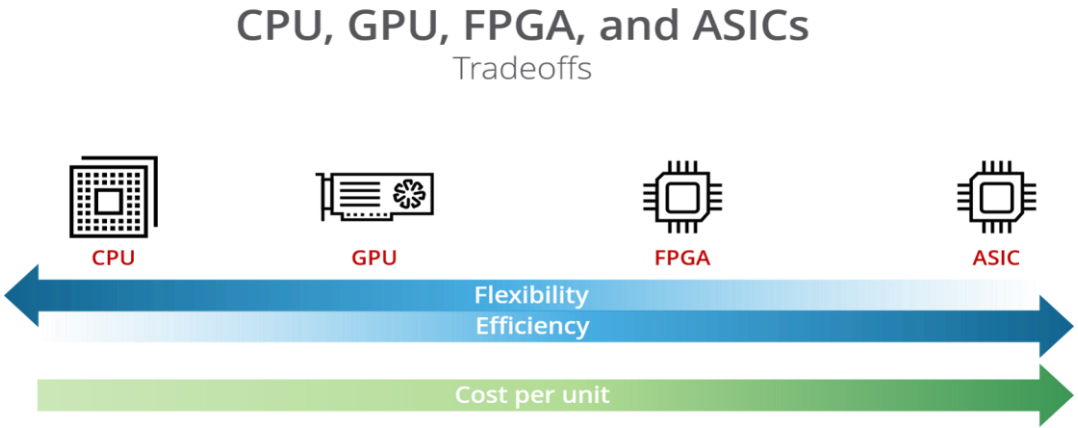
Choosing the “Right” Hardware

Power/Performance Efficiency Varies

- Running the right workload on the right piece of hardware = Higher efficiency.
- Hardware acceleration is a must.
- Heterogenous computing.

Tradeoffs

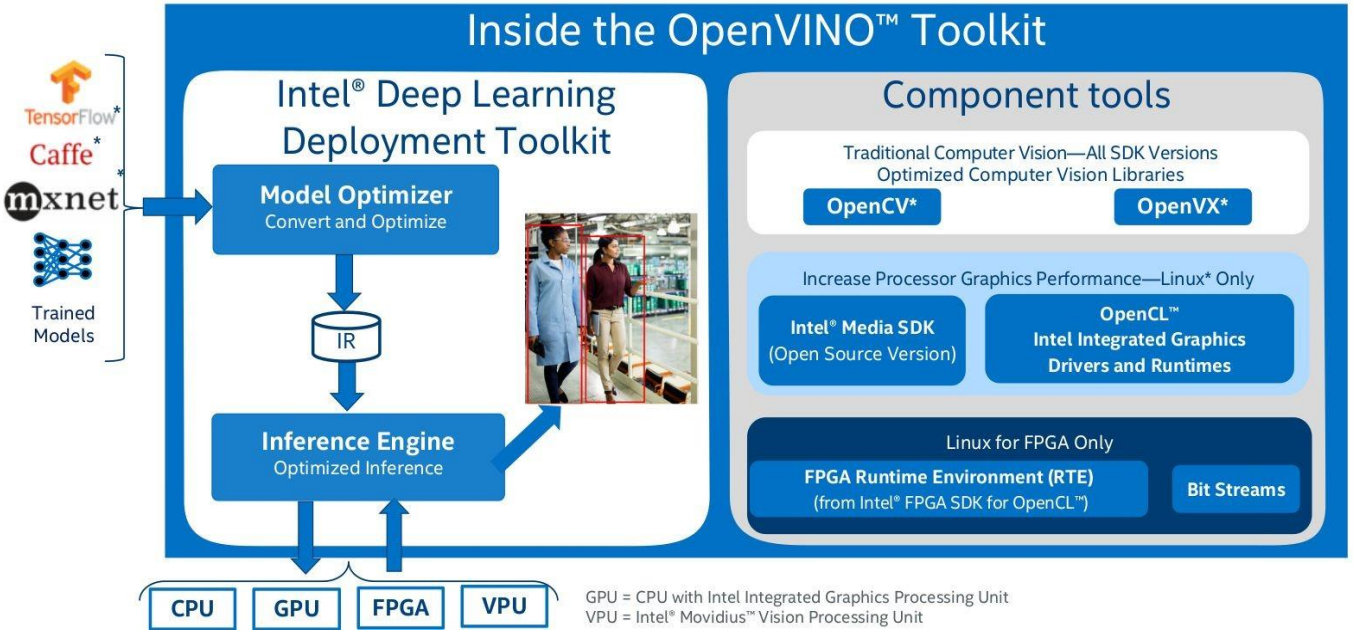
- Power/performance
- Price
- Software flexibility, portability



Click here

Reference link

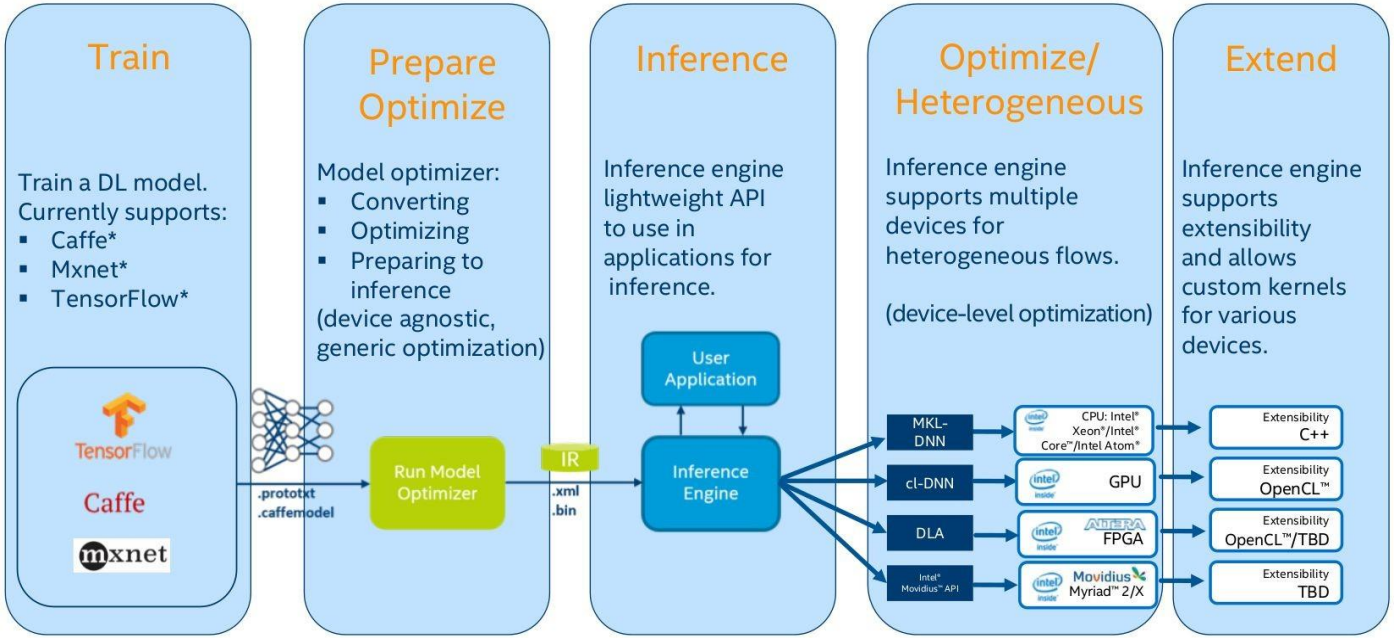
OpenVINO Toolkit and Components



Click here

Reference link

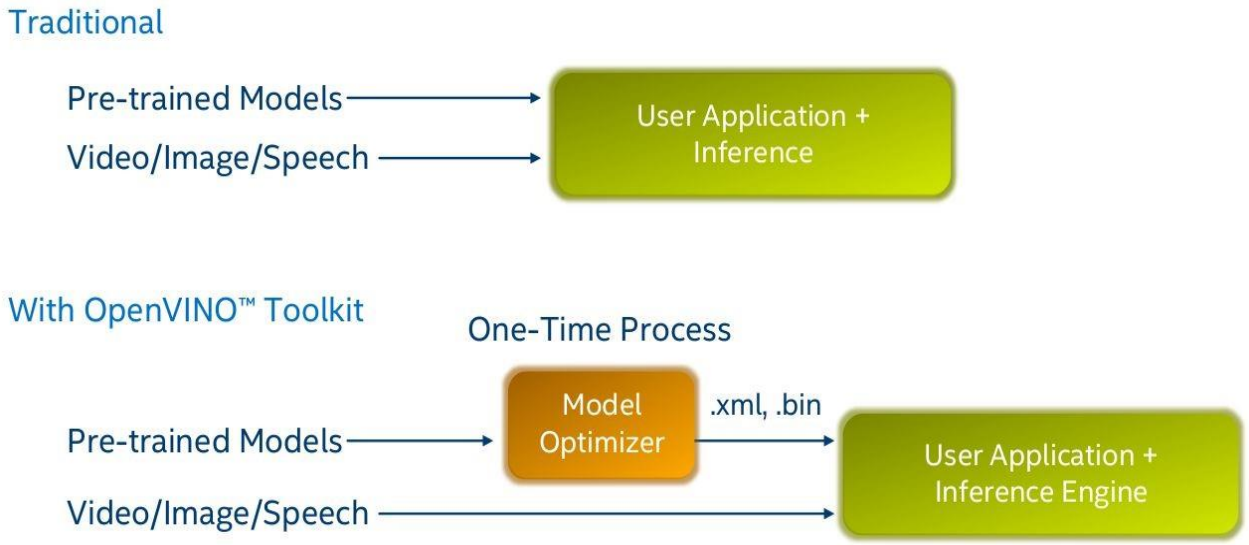
Computer Vision Application Development - OpenVINO Toolkit



Click here

Reference link

Deep Learning Application Deployment

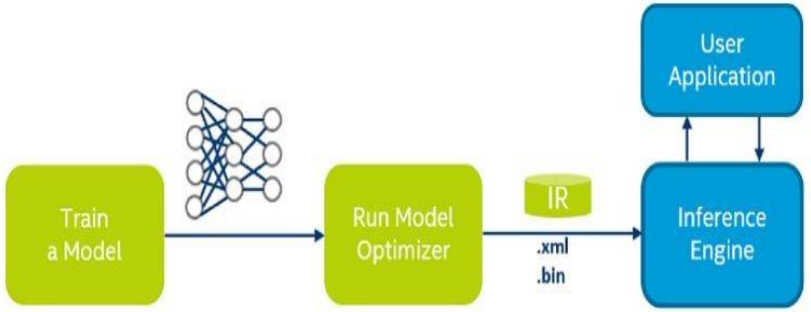


Click here

Reference link

Model Optimizer

- The Model Optimizer is a Python*-based cross-platform command line tool for importing trained models from popular deep learning frameworks such as Caffe*, TensorFlow*, Apache MXNet*, ONNX* and Kaldi*.
- It facilitates the transition between the training and deployment environment, performs static model analysis, and adjusts deep learning models for optimal execution on end-point target devices.
- The Inference Engine API offers a unified API across a number of supported Intel® platforms.



Click here

Reference link

Model Optimizer

- Model Optimizer process assumes you have a network model trained using a supported deep learning framework.
- When you run a pre-trained model through the Model Optimizer, your output is an Intermediate Representation (IR) of the network. The Intermediate Representation is a pair of files that describe the whole model:
    - .xml: Describes the network topology
    - .bin: Contains the weights and biases binary data

A Brief About OpenVINO Intermediate Representation

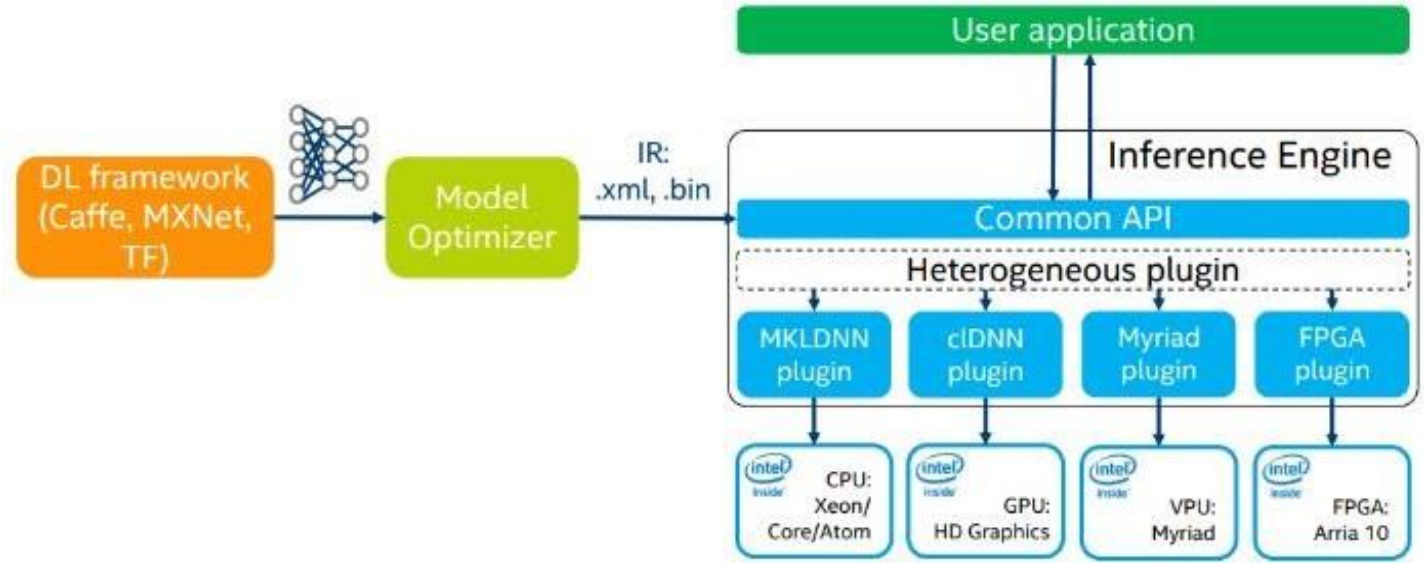
- The OpenVINO Toolkit represents neural network models with the help of two files:
- An XML (.xml) file - this file contains the neural network topology, more commonly known as the architecture.
- A binary (.bin) file - it contains the weights of the neural network model.
- This representation is called the OpenVINO Intermediate Representation (IR).
- Okay, so what's there in an XML file?
- The XML file has different tags to represent the neural network operations and the data flow between them. For example, the <layer> tag is meant for operations like convolution or max-pooling.

A Sample .XML file

- The code shows a small part of the Tiny YOLOv2 XML file. One of the <layer> tags, as you can see, contains the convolution operation.
- Similarly, in the rest of the topology, the other <layer> tags may contain pooling or activation operations.
- The different sub-tags represent the data type, as well as the input and output dimensions.
- The XML file does not contain any model weights. It only has the topology for the corresponding binary (.bin) file that contains the model weights.

```
<net name="yolo-v2-tiny-ava-0001" version="10">
<layers>
<layer id="0" name="data" type="Parameter" version="opset1">
<data element_type="f32" shape="1,3,416,416"/>
<output>
<port id="0" names="data:0" precision="FP32">
<dim>1</dim>
<dim>3</dim>
<dim>416</dim>
<dim>416</dim>
</port>
</output>
</layer>
...
<layer id="5" name="yolov2/darknet_model/conv1/Conv2D/Transpose2101_const" type
    <data element_type="f32" offset="24" shape="16,3,3,3" size="1728"/>
    <output>
    <port id="0" names="yolov2/darknet_model/conv1/W/read:0" precision="FP32">
    <dim>16</dim>
    <dim>3</dim>
    <dim>3</dim>
    <dim>3</dim>
   </port>
   </output>
</layer>
…
</meta_data>
</net>
```

OpenVINO Inference Engine : Hardware Specific Optimizations



Click here

Reference link

Operations of Model Optimizer

- Reshaping
- Batching
- Modifying the network structure
- Standardizing and Scaling
- Quantization

Intel® Distribution of OpenVINO™ Toolkit

A Linux build environment needs these components:

- OpenCV 3.4 or higher
- GNU Compiler Collection (GCC)* 3.4 or higher
- CMake* 2.8 or higher
- Python* 3.5 or higher

NOTE - Only proceed with the installation when you have all the pre required softwares installed on your machine.

Model Optimizer Guide

- Configure your model optimizer (if you have not done that already) for different frameworks by executing ‘install_prerequisites.sh’ present in
  - /opt/intel/openvino/deployment_tools/model_optimizer/install_prerequisites
- cd /opt/intel/openvino/deployment_tools/model_optimizer
- python3 mo.py --input_model <INPUT_MODEL> - to optimize the
  - <INPUT MODEL>
- For example, to optimize the alexnet model based on caffe framework, execute
  - python3 mo.py --input_model alexnet.caffemodel
- As a result of executing the above command, two files - ‘alexnet.xml’ and ‘alexnet.bin’ will be created in your working directory.
- Download the model from internet separately

Model Optimizer Guide

Converting a caffe model: A caffe model has 2 associated files,

1. **.prototxt** - The definition of CNN goes in here. This file defines the layers in the neural network, each layer's inputs, outputs and functionality.

2. **.caffemodel** - This contains the information of the trained neural network (trained model). download both the files and use model optimizer to convert:

https://github.com/BVLC/caffe/wiki/Model-Zoo

python3 mo.py --input_model /home/suryender/Downloads/age_net.caffemodel **--input_proto** /home/suryender/Downloads/deploy_age.prototxt **--output_dir**

Lab 1 : Face Detection using OpenVINO on R-PI

Quiz

Question 1: What does OpenVINO stand for?

a) Open Virtual Intelligence and Neural Optimization

b) Open Visual Inference and Neural Operations

c) Open Vision Intelligence and Neural Optimization

d) Open Visual Inference and Neural Network Optimization

Answer: d) Open Visual Inference and Neural Network Optimization

Quiz

Question 2: What is the primary purpose of the OpenVINO Toolkit?

a) Data collection and preparation for AI models

b) Training deep learning models from scratch

c) Accelerating deployment of trained models on various hardware

d) Developing graphical user interfaces for AI applications

Answer: c) Accelerating deployment of trained models on various hardware

Quiz

Question 3: Which component of OpenVINO converts trained models into Intermediate Representation (IR)?

a) Model Zoo

b) Inference Engine

c) Deep Learning Workbench

d) Model Optimizer

Answer: d) Model Optimizer

Quiz

Question 4: What is the role of the Inference Engine in OpenVINO?

a) It handles data collection for AI models

b) It trains deep learning models

c) It optimizes models for deployment on specific hardware

d) It manages efficient execution of optimized models on different hardware architectures

Answer: d) It manages efficient execution of optimized models on different hardware architectures

Quiz

Question 5: Which of the following is a benefit of using the OpenVINO Toolkit?

a) It only supports CPUs for deployment

b) It's only suitable for cloud-based AI applications

c) It allows for efficient deployment on various hardware architectures, including edge devices

d) It's limited to models trained using PyTorch only

Answer: c) It allows for efficient deployment on various hardware architectures, including edge devices

Reference

- "Release Notes for Intel Distribution of OpenVINO toolkit 2022". March 2022.
- "OpenVINO Toolkit: Welcome to OpenVINO".
- "Introduction to Intel Deep Learning Deployment Toolkit – OpenVINO Toolkit".
- Wilbur, Marcia. "Use the Model Downloader and Model Optimizer for the Intel® Distribution of OpenVINO™ Toolkit on Raspberry Pi*"

Thank you...!