Easily speed up Deep Learning inference – Write once deploy anywhere!

Vladimir Kilyazov

AI Software Solutions Engineer

intel

OpenVINO™

1 oneAPI Powered by oneAPI

Notices and Disclaimers

Performance varies by use, configuration, and other factors. Learn more at intel.com/PerformanceIndex.

Performance results are based on testing as of dates shown in configurations and may not reflect all publicly available updates. See backup for configuration details. No product or component can be absolutely secure.

Your costs and results may vary.

Intel® technologies may require enabled hardware, software, or service activation.

Intel® optimizations, for Intel® compilers or other products, may not optimize to the same degree for non-Intel products.

Intel does not control or audit third-party data. You should consult other sources to evaluate accuracy.

Results have been estimated or simulated.

Intel is committed to respecting human rights and avoiding complicity in human rights abuses.
See Intel's Global Human Rights Principles. Intel® products and software are intended only to be used in applications that do not cause or contribute to a violation of an internationally recognized human right.

Performance varies by use, configuration and other factors. Learn more at www.Intel.com/PerformanceIndex.

System Board	1. Intel® Prototype RVP DDR5 ADL	2. iEi FLEX-BX210AI
CPU	Core™ i9-12900HK @ 2.5 GHz.	Core™ i9-10900TE @ 1.8 GHz.
Sockets / Physical cores	1 / 6 perf + 8 efficiency	1 / 10
HyperThreading / Turbo Setting	Enabled / On	Enabled / On
Memory	2 x 8 GB DDR5 @ 4800 MHz	2 x 8 GB DDR4 @ 2400 MHz
OS	UB-20.04 LTS	UB-18.04 LTS
Kernel	5.15.0-1003-intel-iotg	5.4.0-42-generic
Software	Intel® Distribution of OpenVINO™ Toolkit R3 2022.1	Intel® Distribution of OpenVINO™ Toolkit 2022.1
BIOS	ADLPFWI1.R00.2411.A02.2110081023	AMI Z667AR10.BIN
BIOS release date	October 8, 2021	July 15, 2020
BIOS Setting	Select optimized default settings, save & exit	Select default settings, save & exit
Benchmark Date **Benchmarked by**	May 12, 2022 Intel Corporation	March 17, 2022 Intel Corporation
Precision and Batch Size	Int 8 / Batch 1	Int 8 / Batch 1
Workload: Model / image size	Efficientdet-d0, 512x512; Inception-V4, 299x299 Resnet-50, 224x224; Yolo-V3-tiny, 416x416	Efficientdet-d0, 512x512; Inception-V4, 299x299 Resnet-50, 224x224; Yolo-V3-tiny, 416x416
Inference priority	Throughput	Throughput
Power (TDP Link)/socket	45W	35W

Performance varies by use, configuration and other factors. Learn more at www.Intel.com/PerformanceIndex (Events → Intel® Innovation 2022 Press Briefings)

System Board	Intel® Internal RVP 2S Server Board	Intel® Internal RVP 2S Server Board
CPU	Xeon® Gold 6346 @ 3.10 GHz.	Xeon® Gold 6336Y
Sockets / Physical cores	2 / 16	2 / 24
HyperThreading / Turbo Setting	Enabled / On	Enabled / On
Memory	128 GB DDR4 @ 3200 MHz	128 GB DDR4 @ 3200 MHz
OS	UB-20.04 LTS	UB-20.04 LTS
Kernel	5.10.0-generic	5.15.0-generic
Software	Intel® Distribution of OpenVINO™ Toolkit 2022.2	DeepStream 6.1.1 from nvcr.io/nvidia/deepstream:6.1-devel container
GPU	1x Flex 170, 512 EU, Agama-devel-419.38	1x NVIDIA A10, RT cores: 72
Workload: **Codec, resolution, frame rate** **Model, size (HxW), BS**	HEVC, 1080P, 30 fps Resnet-50, 224x224, 64	H.265, 1080P, 25 fps Resnet-50, 224x224, 64
TDP/socket	150W	150W
Benchmark Date **Benchmarked by**	Sep 23, 2022 Intel Corporation	Sep 23, 2022 Intel Corporation

Compounding effect of hardware and software configuration

See the compounding effect

System board	1. Purley E63448-400, Intel® Internal Reference System	2. Intel® Server Board S2600STB	3. Intel Internal Reference System
CPU	Intel® Xeon® Silver 4116 @ 2.1 GHz	Intel® Xeon® Silver 4216 CPU @ 2.10 GHz	Intel® Xeon® Silver 4316 CPU @ 2.30 GHz
Sockets, physical cores/socket	2, 12	2, 16	2, 20
Hyperthreading/turbo setting	Enabled/On	Enabled/On	Enabled/On
Memory	12x 16 GB DDR4 2400 MHz	12x 64 GB DDR4 2400 MHz	16 x32GB DDR4 2666 MHz
OS	UB-16.04.3 LTS	UB-18.04 LTS	UB-20.04 LTS
Kernel	4.4.0-210-generic	4.15.0-96-generic	5.13.0-rc5-intel-next+
Software	Intel® Distribution of OpenVINO™ Toolkit R5 2018	Intel® Distribution of OpenVINO™ Toolkit R3 2019	Intel® Distribution of OpenVINO™ Toolkit 2021.4.1
BIOS	PLYXCRB1.86B.0616.D08.2109180410	—	WLYDCRB1.SYS.0020.P93.2103190412
BIOS release date	September 18, 2021	—	March 19, 2021
BIOS setting	Select optimized default settings, save, and exit	Select optimized default settings, save, and exit	Select optimized default settings, change power policy to “performance,” save, and exit
Test date	October 8, 2021	September 27, 2019	September 6, 2021
Precision and batch size	FP32/Batch 1	int8/Batch 1	int8/Batch 1
Workload: Model/image size	MobileNet-SSD/300x300	MobileNet-SSD/300x300	MobileNet-SSD/300x300
Number of inference requests	24	32	10
Number of execution streams	24	32	10
Power (TDP link)/socket	170W	200W	300W

Compounding Effect of Hardware and Software

See the compounding effect slide

System board	Intel® Server Board S2600STB	M50CYP2SB1U Coyote Pass	Intel Corporation / Archer City
CPU	Intel® Xeon® Platinum 8270 CPU @ 2.7 GHz	Intel® Xeon® Platinum 8380 CPU @ 2.3 GHz	Intel® Xeon® Platinum 8490H @ 1.9 GHz
Sockets, physical cores/socket	2, 26	2, 40	2, 60
Hyperthreading/turbo setting	Enabled/On	Enabled/On	Enabled/On
Memory	12x 16 GB DDR4 2933 MHz	16 x16GB DDR4 3200 MHz	16x16 GB DDR5 4800 MHz
OS	UB-18.04 LTS	UB-22.04 LTS	UB-22.04 LTS
Kernel	5.3.0-24-generic	5.19.0-38-generic	5.19.0-41-generic
Software	Intel® Distribution of OpenVINO™ Toolkit 2021.4	Intel® Distribution of OpenVINO™ Toolkit 2022.3	Intel® Distribution of OpenVINO™ Toolkit 2023.0
BIOS	SE5C620.86B.02.01. 0013.121520200651	SE5C620.86B.01.01.0006.2207150335	EGSDREL1.SYS.9409.P31.2302280828
BIOS release date	12/15/2020	7/15/2022	2/28/2023
BIOS setting	Select optimized default settings, save, and exit	Select optimized default settings, save, and exit	Select optimized default settings, save, and exit
Test date	6/18, 2021	6/20/2023	5/25/2023
Precision and batch size	int8/Batch 1	int8/Batch 1	Int8/Batch 1
Number of inference requests	52	80	120
Number of execution streams	52	80	120
Power (TDP)/socket	205W	270W	350W

Workloads (model: input HxW):
Inception-v4: (299x299); Resnet-50: (224x224); Unet-camvid-onnx-0001: (368x480); Yolo-v3-tiny: (416x416)

Challenges in Deep Learning

Development and deployment challenges in deep learning

Maximizing trained performance

Varied HW acceleration capabilities require specific tuning when deploying

Low-performing, lower- accuracy models deployed

Integration challenges

No way to streamline end-to-end development workflow

Slow time to solution and time to market

No one size fits all

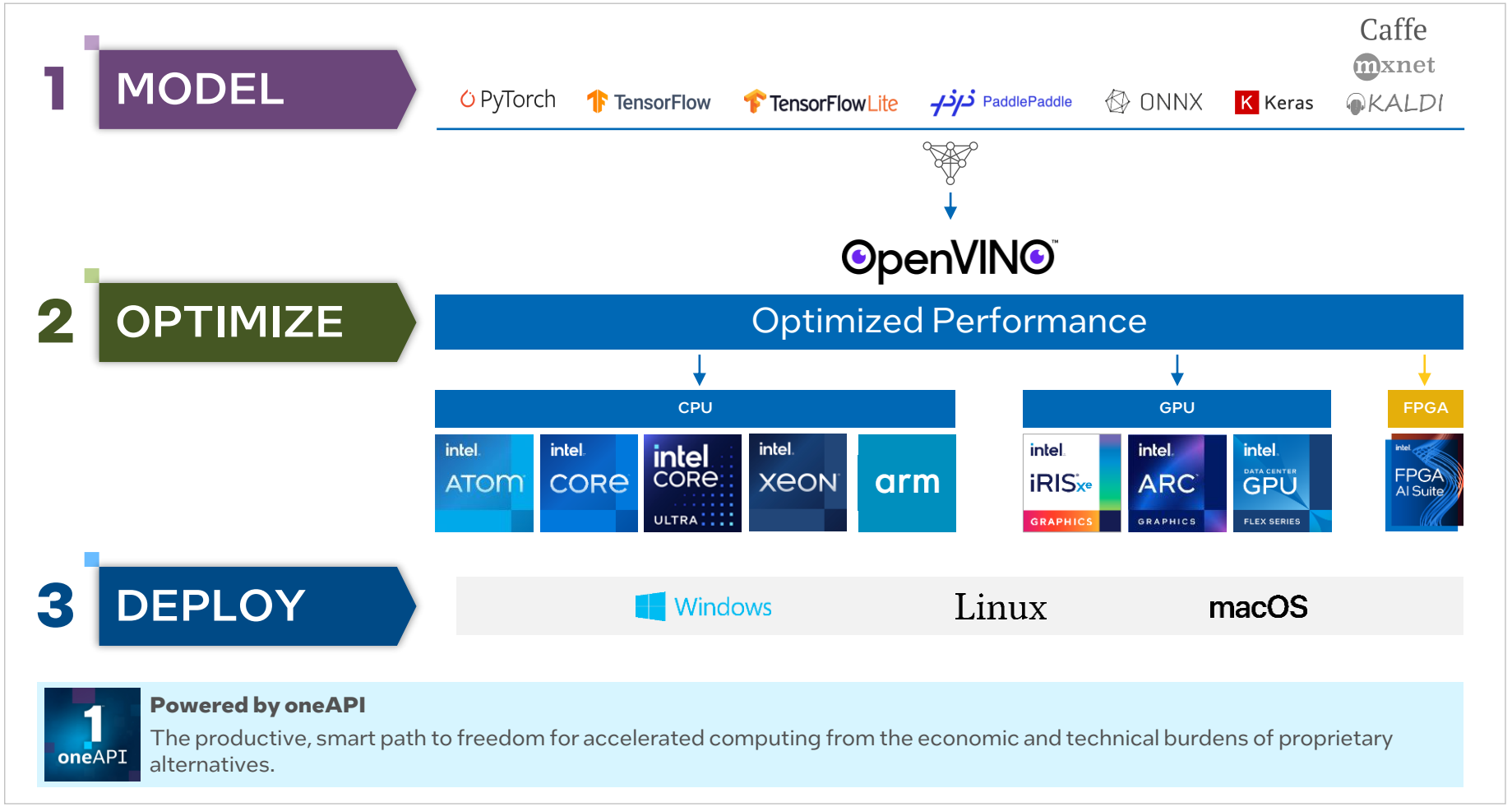
Diverse requirements for myriad use cases require unique approaches

Inability to meet use-case-specific requirements

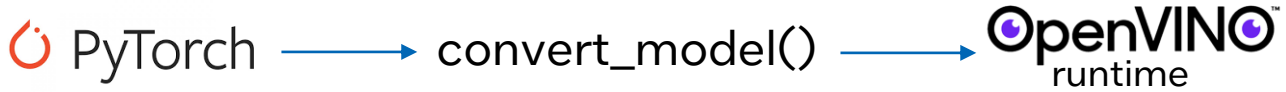
OpenVINO™ Toolkit Overview

Fast, accurate results with high-performance, deep learning inference

Convert and optimize models, and deploy across a mix of hardware and environments, on-premises and on-device, in the browser or in the cloud



OpenVINO™ - PyTorch Framework Improvements



Key Benefits

- **Support for models trained using PyTorch 1.x**
- **Easy to use:** Load PyTorch model directly to convert_model() API and infer using OpenVINO™ APIs.
- **Support for all Intel devices:** Intel® CPU, iGPU, dGPU and NPU.
- **No explicit model conversion required:** Inline conversion of PyTorch model to OpenVINO™ IR
- **No offline MO step needed**

Example Code

```python
from torchvision.models import resnet50
import torch
# Import OV compile and convert
from openvino import compile_model, convert_model

# PyTorch model load
example_inputs = [torch.zeros(1, 3, 224, 224)]
model = resnet50(pretrained=True)

# Convert model
ov_model = convert_model(model, example_input=example_inputs)
#  Compile model and run inference as usual OV Model
compiled_model = compile_model(ov_model)
# return result in OpenVINO format
result = compiled_model(example_inputs)
```

PyTorch (torch.compile) with OpenVINO™ backend

```python
import torch
import torchvision.models as models
import openvino.frontend.pytorch.torchdynamo.backend
model = models.resnet50(pretrained=True)
input = torch.rand((1,3,224,224))
model = torch.compile(model, backend='openvino')
pred = model(input)
```

Key Benefits

- **Support for PyTorch 2.0+**
- **Stay in the PyTorch API:** Leverage OpenVINO while using PyTorch APIs for inferencing.
- **Support for all Intel devices:** Intel® CPU, iGPU, dGPU and NPU.
- **Platform Support:** Linux and Windows OS

Key Features

- **Graph Partitioning:** OpenVINO™ unsupported operators fallback to PyTorch on CPU
- **Model Caching:** Improvement in model loading time on GPU.
- **Significant optimizations for Stable Diffusion and tested/validated thoroughly**

AUTO Plugin Capabilities

CPU | GPU | NPU | HETERO | MULTI | AUTO

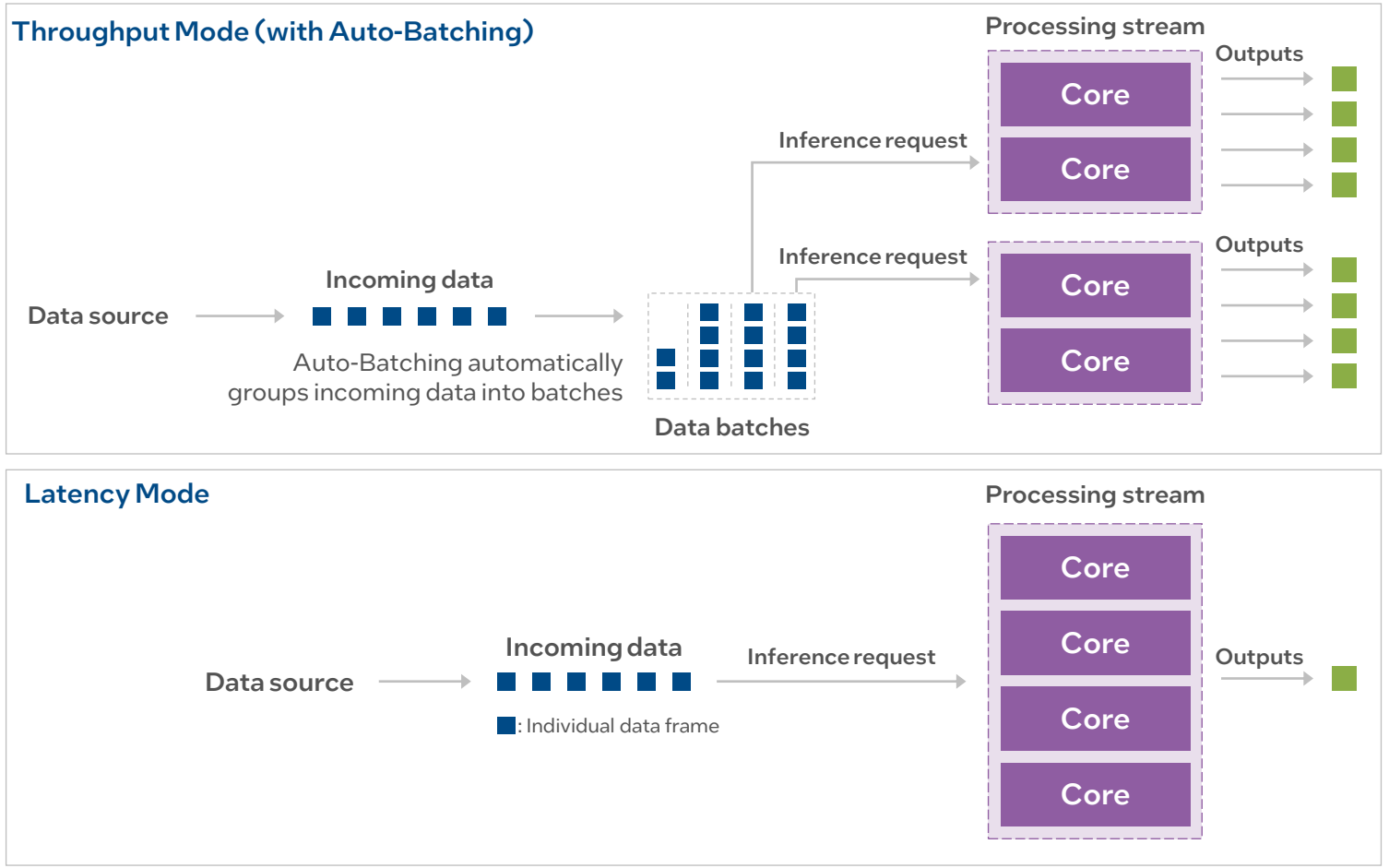
The AUTO plugin automatically detects processing resources and maximizes inference performance.

It does not disrupt the workloads switching from CPU to GPU, ensuring maximum efficiency of resources.

The AUTO plugin uses performance hints that prioritize either latency or throughput and load balances across compute within both the CPU and discrete.

Learn more at doc.openvino.ai.

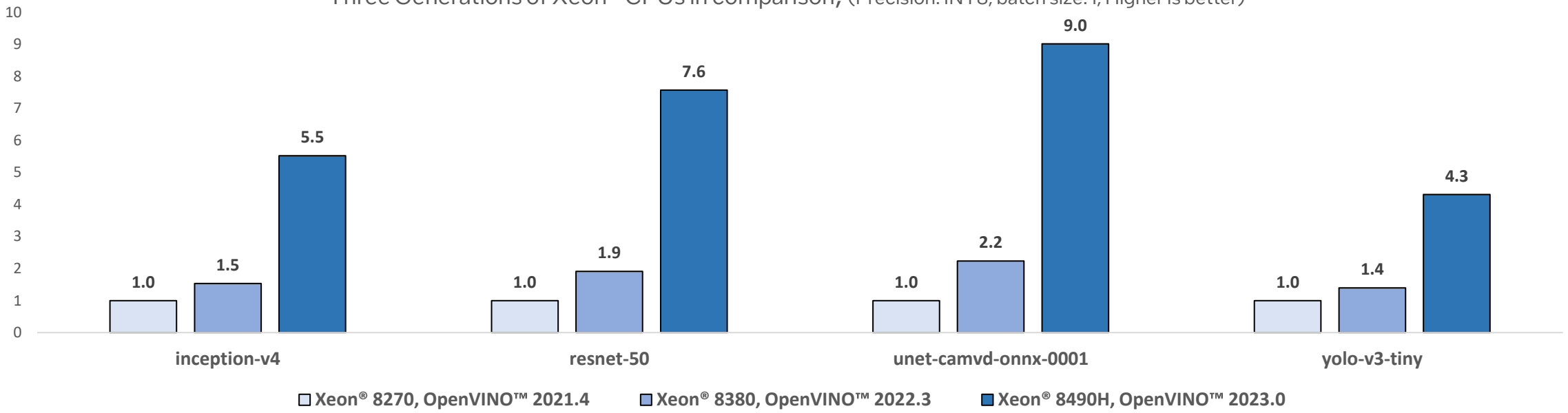
```
compiled_model = core.compile_model(model=model, device_name="AUTO")
```

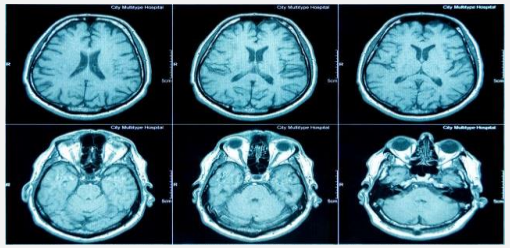


Compounding Effect of Hardware and Software

OpenVINO™ using Intel AMX improves AI inference performance exponentially.

Three Generations of Xeon® CPUs in comparison, (Precision: INT8, batch size: 1, Higher is better)

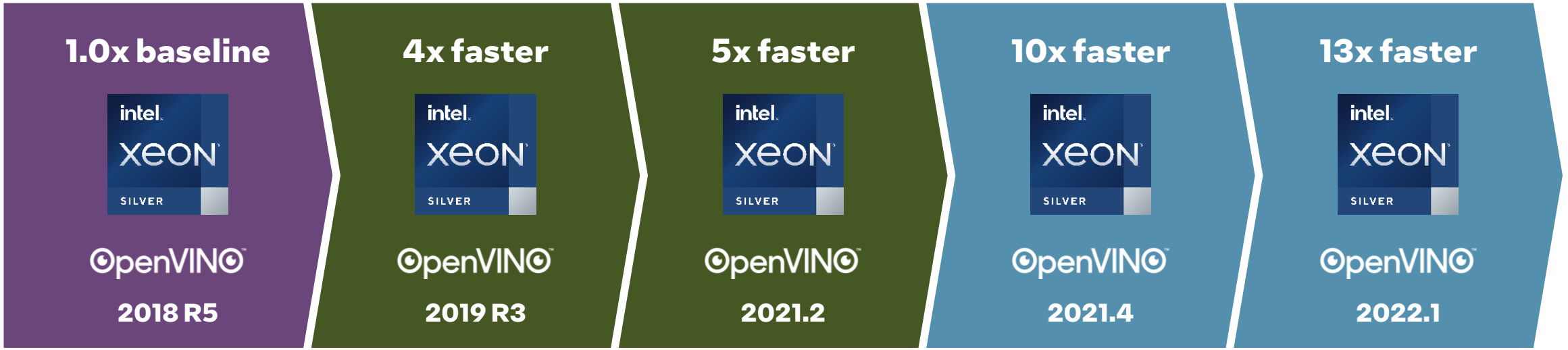




See backup for system configuration details, workloads and pricing Results may vary.
AI workloads cover image classification, high-res semantic segmentation and object detection.
For workloads and configurations see this slide 8.

Object Detection + Intel® Xeon® Scalable Processors

Compelling AI inference performance increases over time using the mobilenet-ssd model

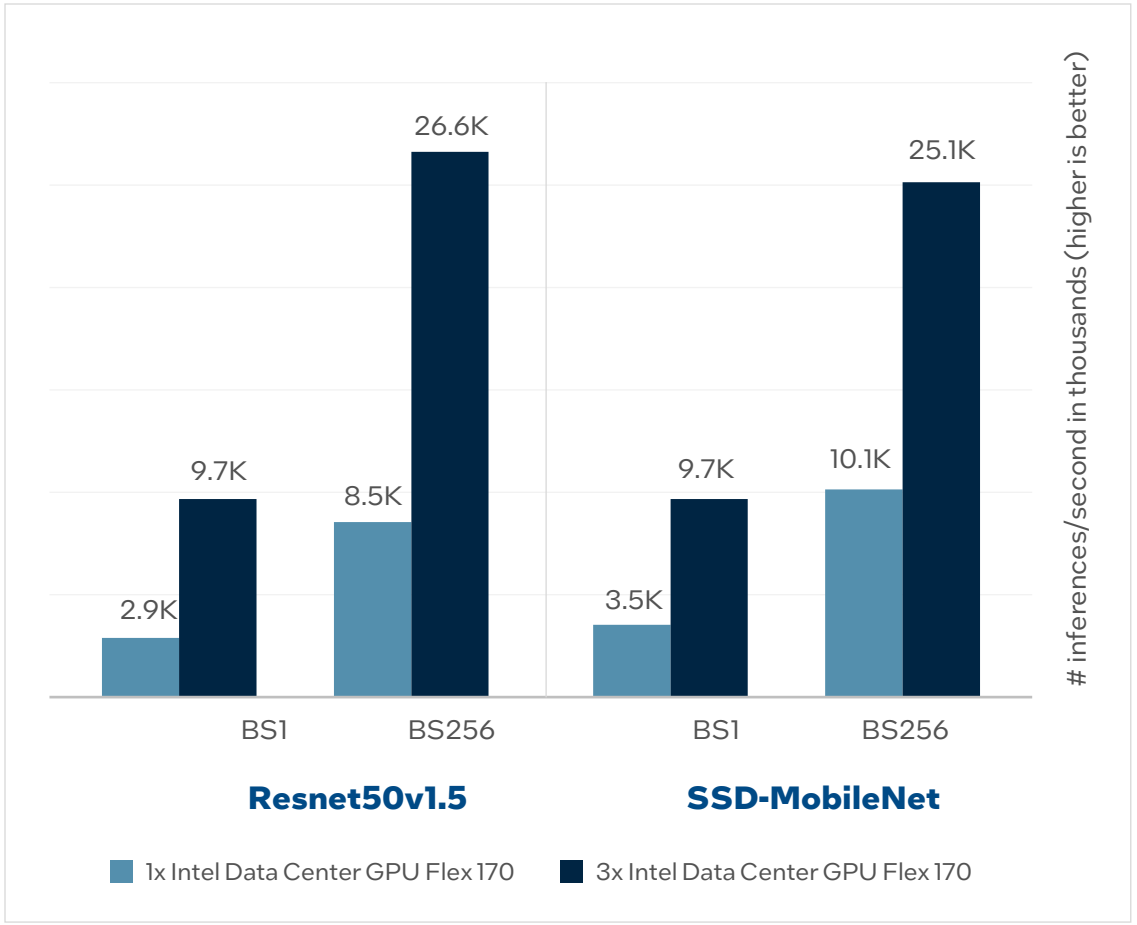


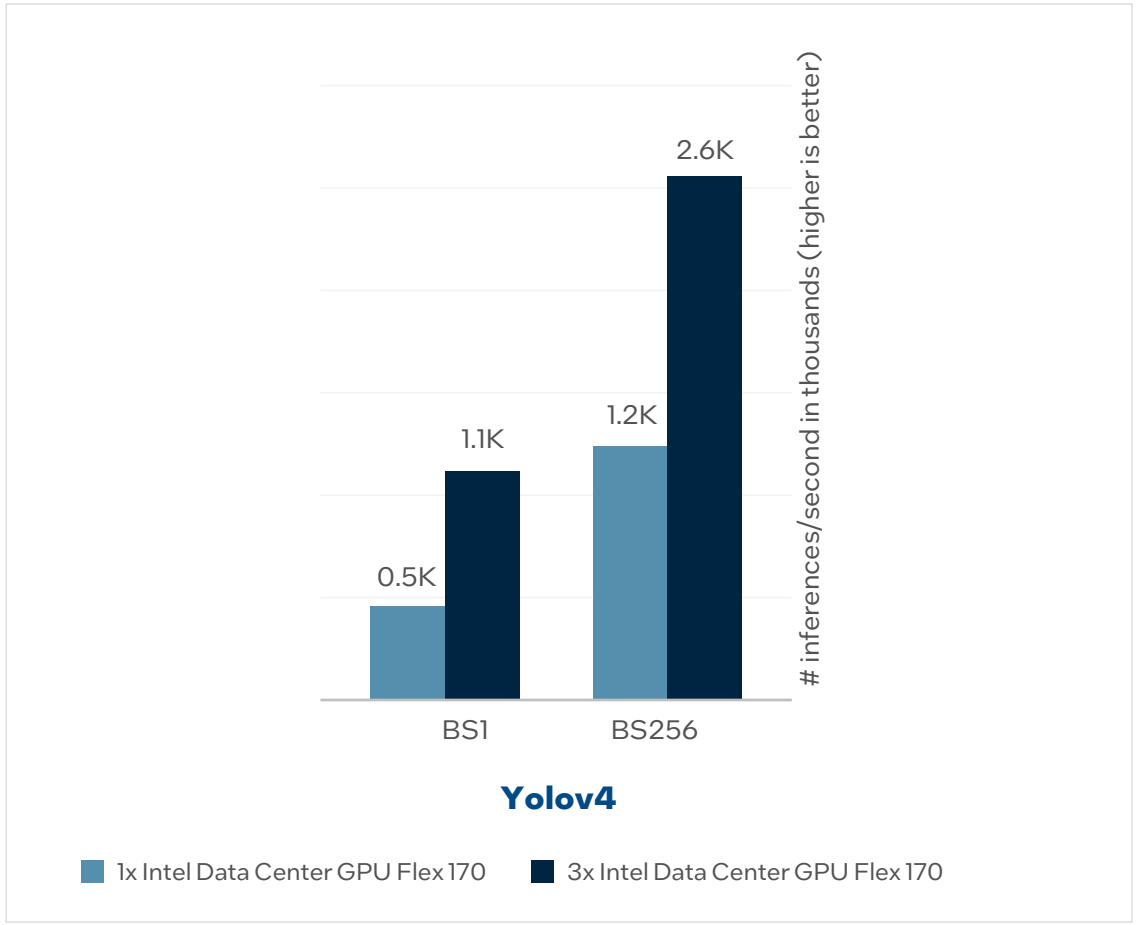
4116 1st Gen Intel® Xeon® Scalable processor with FP32	4216 2nd Gen Intel Xeon Scalable processor with Intel® Deep Learning Boost (int8 VNNI)	4216R 2nd Gen Intel Xeon Scalable processor with Intel Deep Learning Boost (int8 VNNI)	4316 3rd Gen Intel® Xeon® Scalable processor with Intel Deep Learning Boost (int8 VNNI)	4316 3rd Gen Intel Xeon Scalable processor with Intel Deep Learning Boost (int8 VNNI)

See here for workloads and configurations. Results may vary.

1. 2018 R5 obtained on system configuration 1
2. 2019 R3 obtained on system configuration 2
3. OV-2021.2 obtained on system configuration 3
4. OV-2021.4.1 and OV-2022.1 obtained on system configuration 4
5. For workloads and configurations see this slide 7.

Achieve Higher FPS for AI Inferencing

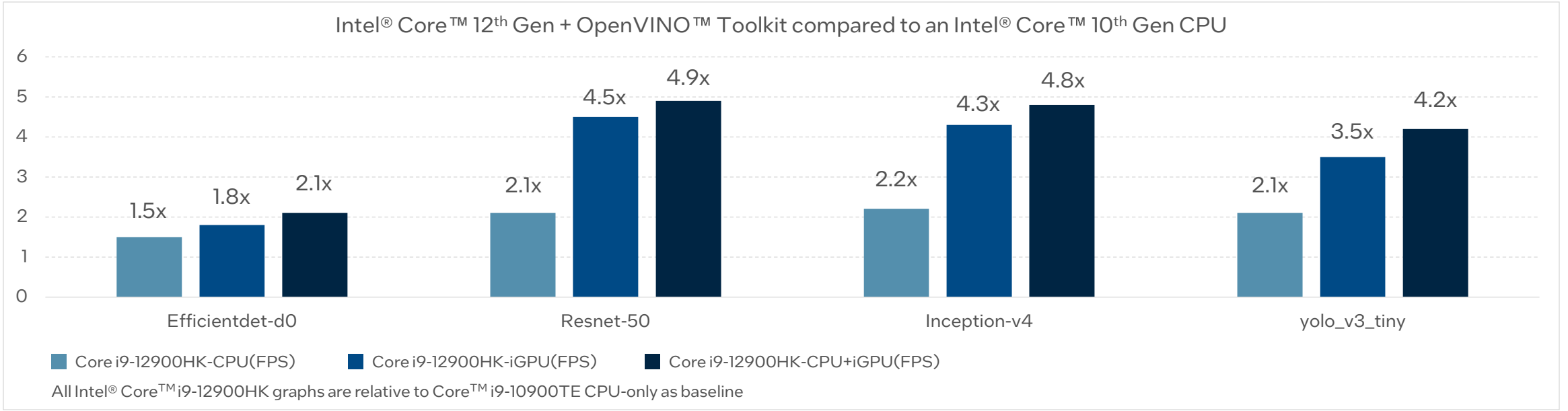




Based on OV 2022.3. For workloads and configurations visit www.Intel.com/PerformanceIndex. Click on the Events tab and and Intel® Innovation 2022. Results may vary.
For workloads and configurations see this slide 7.

Compounding Effect of Hardware and Software

Use Intel® Iris® Xe Graphics + CPU combined for maximum inferencing



See backup for system configuration details, workloads and pricing Results may vary
System 1) Core i9-12900HK
System 2) Core i9-10900TE
For workloads and configurations see this slide 5.

OpenVINO™ Toolkit Developer Journey

1 | MODEL

PyTorch | TensorFlow | TensorFlow Lite | PaddlePaddle | ONNX | Keras

Open Model Zoo
280+ open source and optimized pretrained models

Intel Optimum
Use OpenVINO as an extension in Hugging Face transformer models and gain model compression and performance benefits

intel GETi
Build computer vision models in a fraction of the time and with less data.

2 | OPTIMIZE

OpenVINO Model Converter
Convert trained model from supported frameworks

IR Data

Read, load, infer

OpenVINO format (intermediate representation file) (.pb, .tflite, .onnx,)

Direct model conversion for TensorFlow and PyTorch
For select models, you can skip steps to get to deployment faster

Model Compression with NNCF
Neural Network Compression Framework provides quantization aware training, model pruning and sparsity along with post-training optimization

Jupyter Notebooks
Get sample code on the latest models to help get your application into production faster

3 | DEPLOY

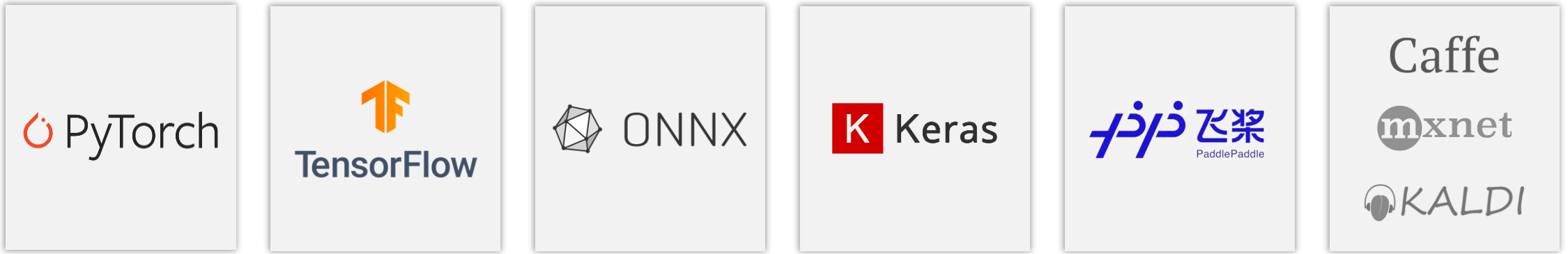
OpenVINO™ Model Server
Serve models over gRPC, REST, or C API endpoints

OpenVINO™ Runtime
Common Python, C and C++ APIs that abstracts low-level programming for each device below

Model selection

Expedite the model training process

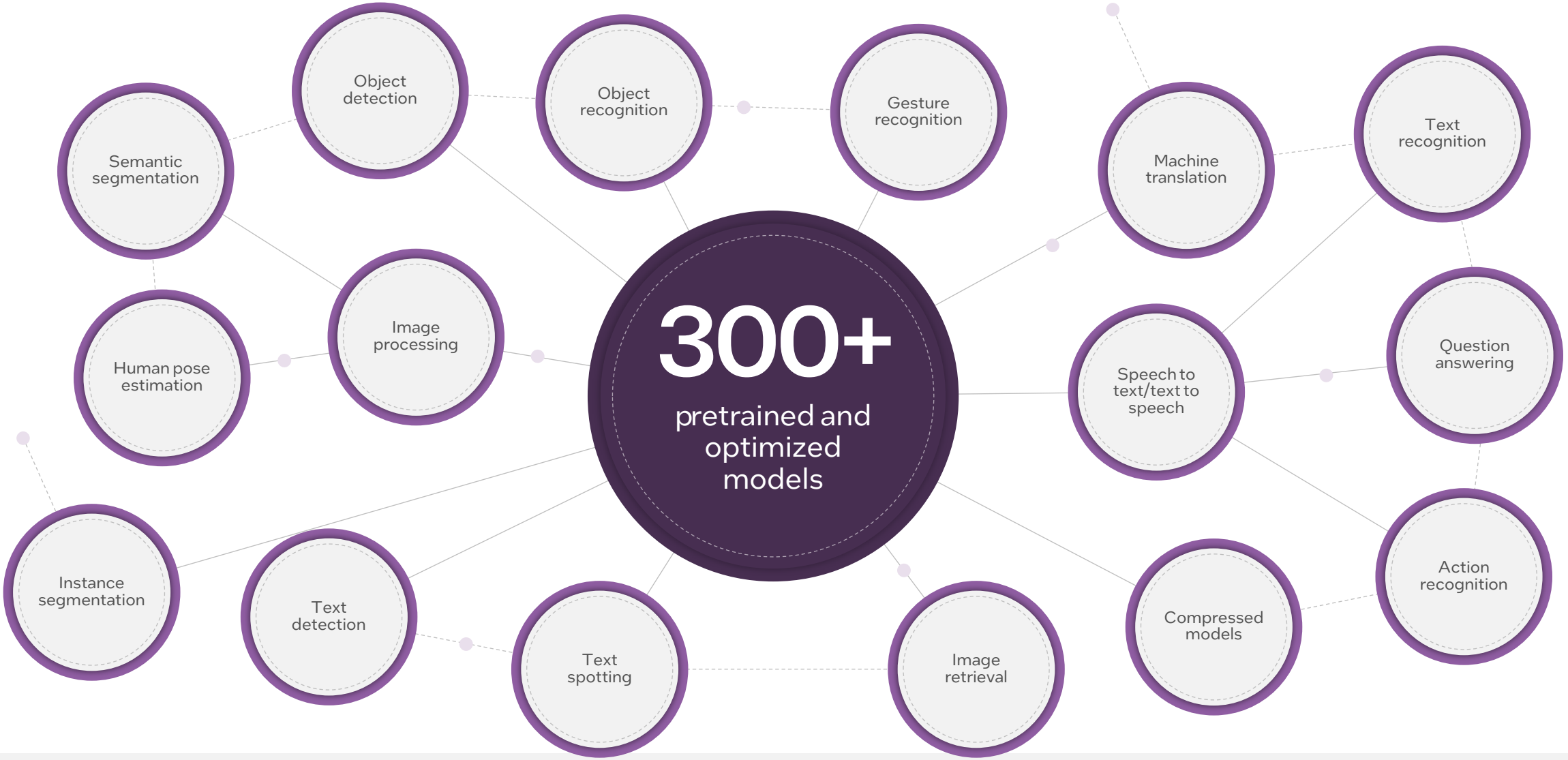
Leverage the expansive variety of open-source pre-trained models from the following model zoos



Supported frameworks and formats: https://docs.openvino.ai/latest/openvino_docs_MO_DG_prepare_model_Supported_Frameworks_Layers.html#doxid-openvino-docs-m-o-d-g-prepare-model-supported-frameworks-layers

Convert models with Model Optimizer: https://docs.openvino.ai/latest/openvino_docs_MO_DG_Deep_Learning_Model_Optimizer_DevGuide.html

Model selection

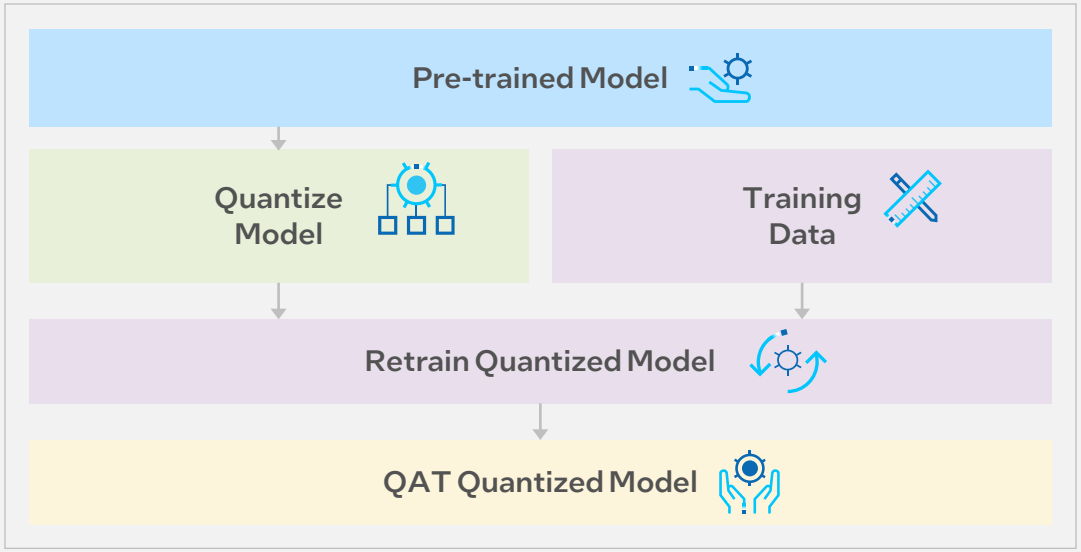


Model compression

Quantization Paths

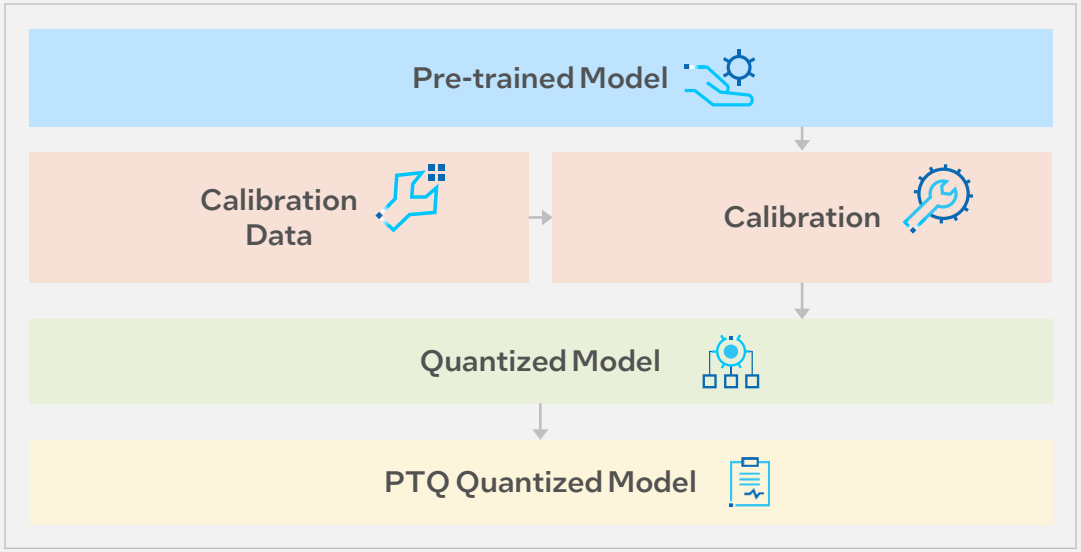
Training Aware Quantization (QAT)

Incorporates quantization-aware techniques during training to optimize the model for lower precision. Aims to preserve accuracy but requires complex and time-consuming model retraining.



Post Training Quantization (PTQ)

Quantizes a pre-trained model after training, reducing model precision to improve memory usage and inference speed while potentially sacrificing some accuracy.



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

Development guide: https://docs.openvino.ai/latest/openvino_docs_MO_DG_Deep_Learning_Model_Optimizer_DevGuide.html

Model optimization

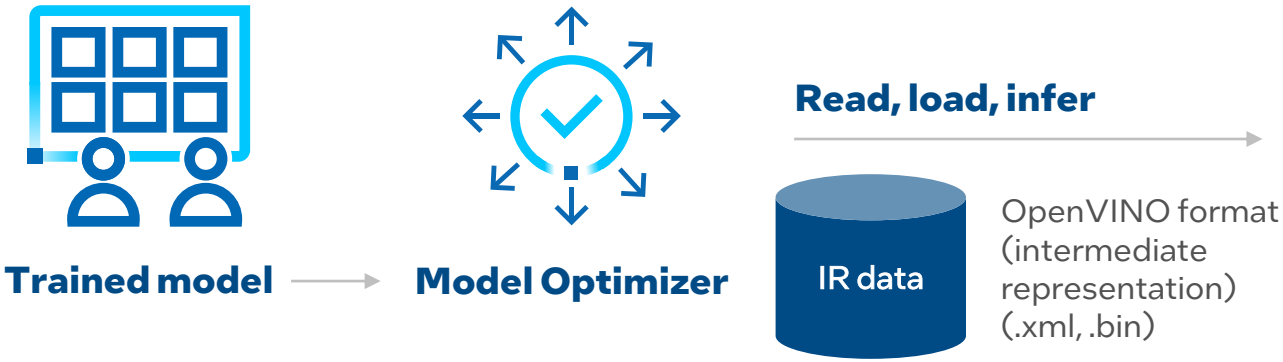
Model Optimizer

A Python-based tool to import trained models and convert them to intermediate representation (IR)

- Creates smaller disk footprint for ease of deployment
- Reduces time to download and convert
- Runs faster on certain hardware, reducing latency and providing more efficient inference
- Optimizes for performance or space with conservative topology transformations
- Offers hardware-agnostic optimizations

Optimization techniques available are:

- Linear operation fusing
- Stride optimizations
- Group convolutions fusing



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

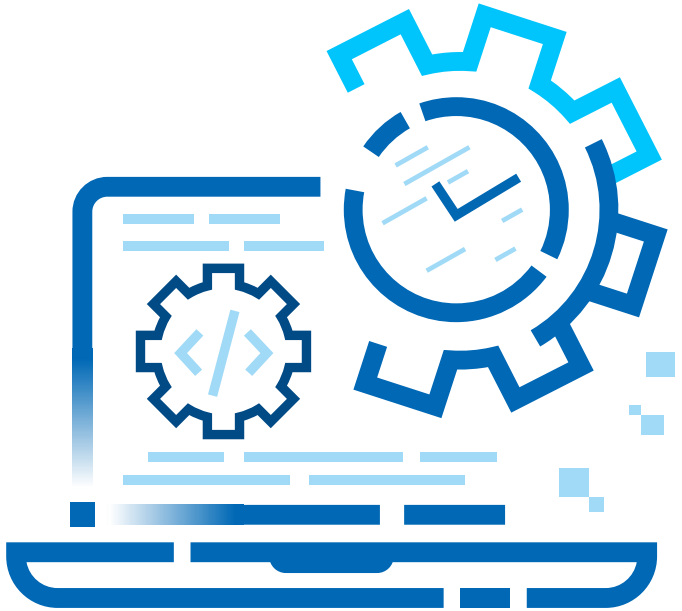
Note: OpenVINO IR, ONNX, PaddlePaddle, TensorFlow and TensorFlow Lite models can be used as input to OpenVINO Runtime. Caffe, MXNet and PyTorch must be converted to IR or ONNX before loading

Development guide: https://docs.openvino.ai/latest/openvino_docs_MO_DG_Deep_Learning_Model_Optimizer_DevGuide.html

Model deployment

Runtime

- High-level C, C++, and Python inference runtime API
- Interface is implemented as dynamically loaded plugins for each hardware type
- Delivers superior performance for each type without requiring users to implement and maintain multiple code pathways



Development guide: https://docs.openvino.ai/latest/openvino_docs_OV_UG_OV_Runtime_User_Guide.html#

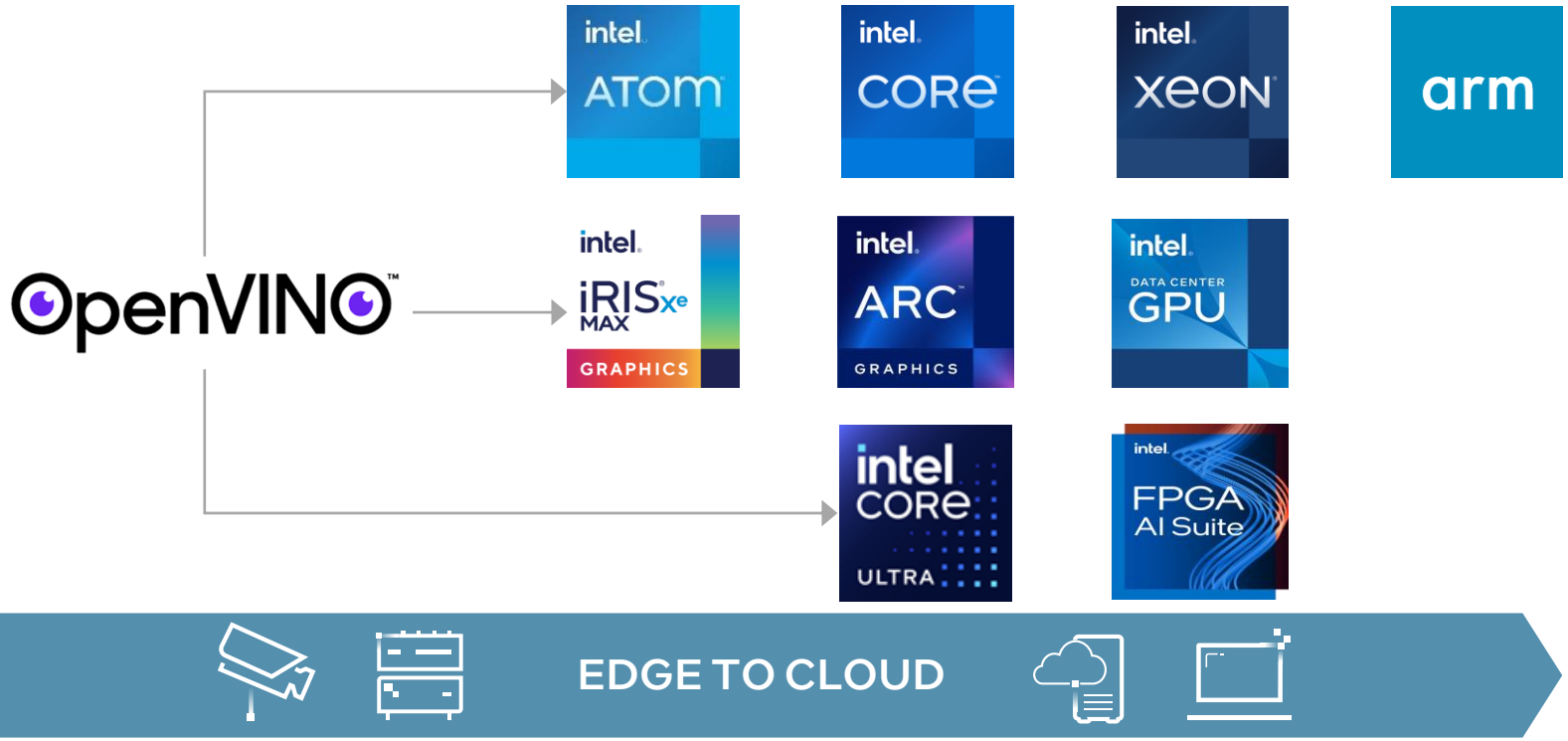
Model deployment

Write once, deploy anywhere

Common high-level inference runtime for cross-platform flexibility

Write once, and deploy across different platforms with the same API and framework-independent execution.

Full environment utilization, or multidevice plugin, across available hardware for superior performance results.

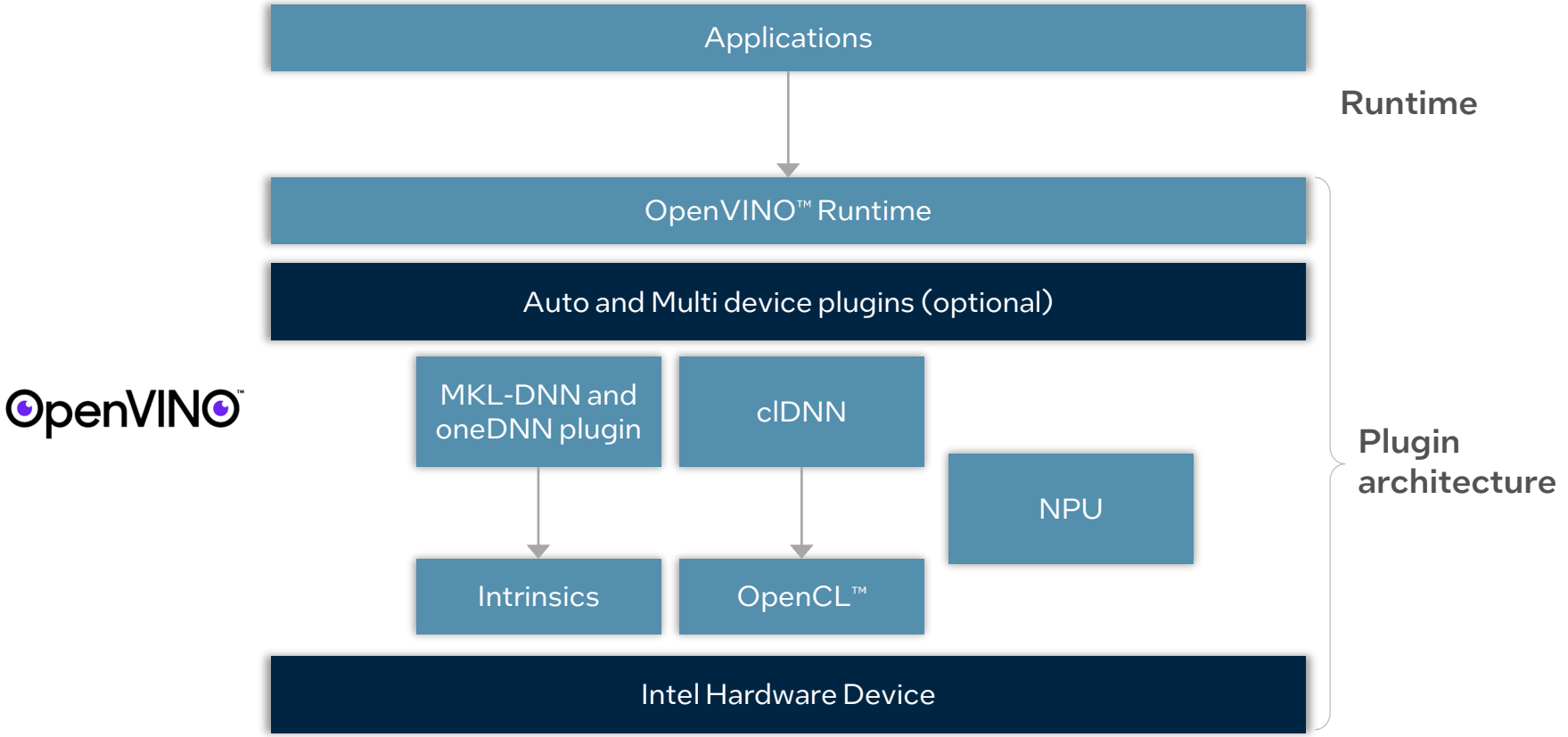


For more details on supported platforms, see system requirements : https://www.intel.com/content/www/us/en/developer/tools/openvino-toolkit/system-requirements.html

Model deployment

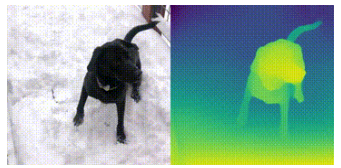
Write once, deploy anywhere

OpenVINO® Runtime architecture



GitHub Jupyter Notebooks

Valuable tutorials for computer vision and natural language processing

<Hello World/>	**Hello World**	▪ Basic introduction to OpenVINO's Python API ▪ Inference on a mobilenetv3 image classification model
TensorFlow	**Tensorflow Training**	▪ End-to-end training to deployment workflow starting with TensorFlow's Flowers classification demo
{API}	**OpenVINO API**	▪ Load Runtime and Show Info ▪ Loading a Model ▪ Getting Information about a Model ▪ Doing Inference on a Model ▪ Reshaping and Resizing
Model Optimizer & Runtime	**Model Tools**	▪ Download a model, ▪ Convert to OpenVINO's IR format ▪ Get model information, ▪ Benchmark model
TensorFlow	**Tensorflow to OpenVINO**	▪ Demonstrates how to convert TensorFlow models to OpenVINO IR
	**Mono-Depth**	▪ Demonstrates Monocular Depth Estimation with MidasNet model ▪ Users can upload their own videos and images, input data will be resized
PyTorch	**PyTorch to OpenVINO**	▪ Convert PyTorch models to OpenVINO IR ▪ Uses Model Optimizer to convert the open source fastseg semantic segmentation model
	**Background Removal**	▪ Background removal in images ▪ The open source U^2-Net model is converted from PyTorch

New Notebooks with 2023.2 Release

Several Jupyter notebooks have been updated to demonstrate the conversion and optimization of PyTorch models *without ONNX conversion*:

- PyTorch to OpenVINO - Convert PyTorch models in formats *torch.nn.Module* and *torch.jit.ScriptModule* to OpenVINO IR
- Post-Training Quantization of PyTorch models with NNCF - Apply int8 quantization to PyTorch models
- Quantization of Image Classification Models - Apply int8 quantization to a MobileNet V2 PyTorch model
- Visual Question Answering and Image Captioning using BLIP and OpenVINO - Optimize the BLIP PyTorch model
- Text-to-Image Generation and Infinite Zoom with Stable Diffusion v2 and OpenVINO™ - Optimize the models in the Stable Diffusion 2.0 pipeline
- Object masks from prompts with SAM and OpenVINO™ - Optimize the PyTorch-based Segment Anything Model (SAM)
- Optimizing PyTorch models with Neural Network Compression Framework of OpenVINO™ by 8-bit quantization - Quantization Aware Training (QAT) with PyTorch models

A few new notebooks were added to show how to convert and optimize models, including those from TensorFlow Hub, TorchVision, and Hugging Face Hub:

- TorchVision Zoo with OpenVINO™ - Download and optimize pre-trained models directly from PyTorch
- Hugging Face Model Hub with OpenVINO™ - Learn how to download and optimize pre-trained models from Hugging Face hub
- TensorFlow Hub models + OpenVINO - Download and optimize pre-trained models directly from TensorFlow Hub
- Convert Detectron2 Models to OpenVINO - Optimize the popular Facebook Research model for object detection and segmentation
- Convert TensorFlow Object Detection and Instance Segmentation Models to OpenVINO™ - Optimize Faster R-CNN with Resnet-50 V1 from TensorFlow Hub
- Visual-language assistant with LLaVA and OpenVINO - End-to-end multi-modal demo using LLaVA (Large Language and Vision Assistant)
- Subject-driven image generation and editing using BLIP Diffusion and OpenVINO - Optimize BLIP-Diffusion for zero-shot subject-driven image generation
- SoftVC VITS Singing Voice Conversion and OpenVINO™ - Optimize SoftVC and VITS for voice conversion using audio input
- Object segmentations with FastSAM and OpenVINO™ - Optimize Fast Segment Anything Model (FastSAM) for object segmentation
- Image Generation with DeciDiffusion - Optimize DeciDiffusion 1.0 for text-to-image generation
- Document Visual Question Answering Using Pix2Struct and OpenVINO - Demonstration of multi-modal question answering using OCR and language models

Gen AI notebooks with optimized performance right out of the box:

- Create an LLM-powered Chatbot using OpenVINO – Running chatbot such as Llama2 on CPUs and GPUs with the int8 weight compression, and impressively it would run on laptops with only 24GB of RAM.
- Image generation with Latent Consistency Model and OpenVINO – achieve remarkable generative images with much lower computer resources
- Generate creative QR codes with ControlNet QR Code Monster and OpenVINO™ - create your own graphical QR code with ControlNet and Stable Diffusion.

Ready to get Started?

Choose and download free directly from Intel

Intel® Distribution of OpenVINO™ Toolkit

Also available from these sources:

Intel® Developer Cloud | PIP | Docker Hub | Dockerfile | Anaconda Cloud | YUM | APT | Conan | Homebrew | vcpkg

Build from source:

GitHub | Gitee (for China)


intel