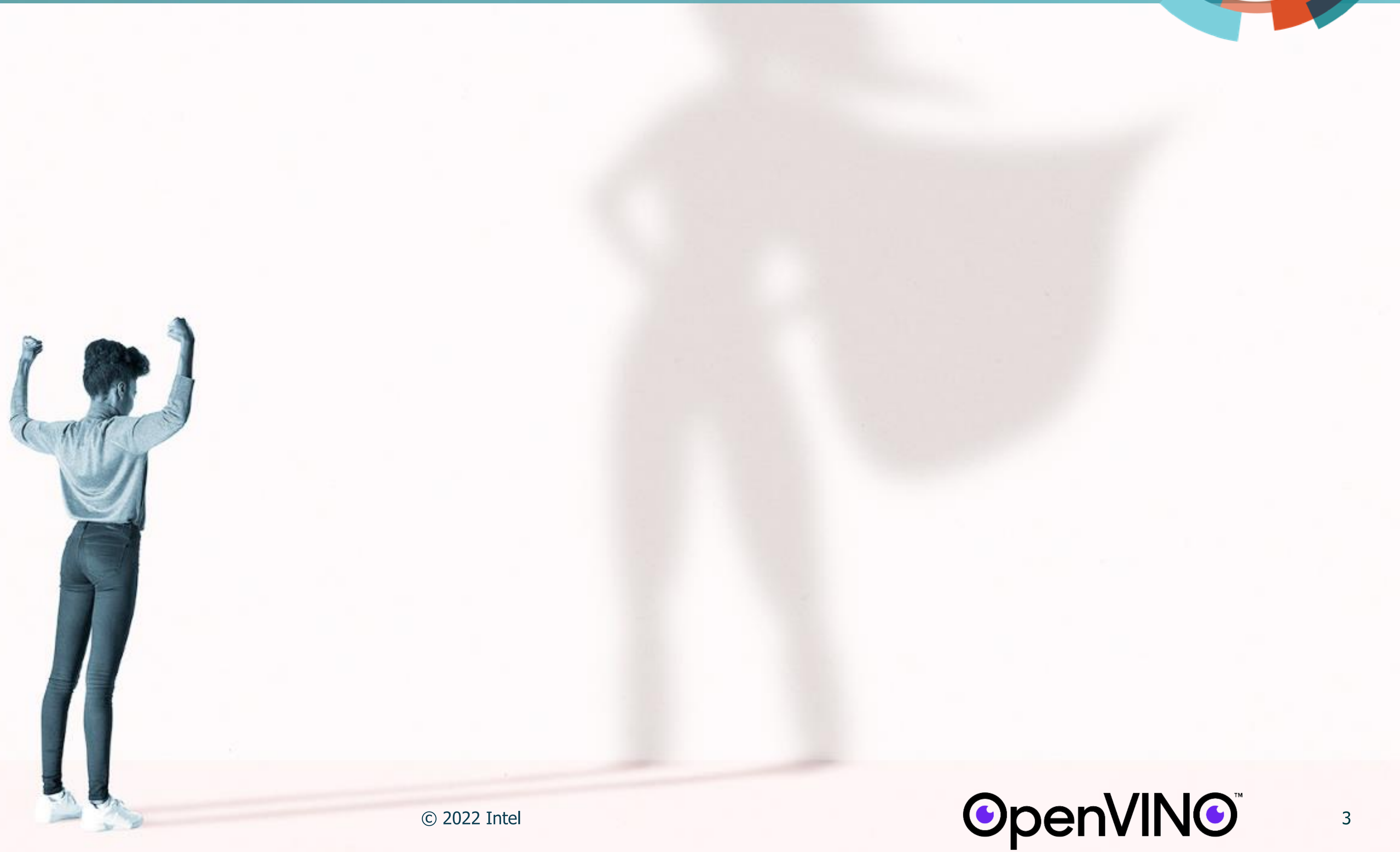
Optimization Techniques with OpenVINO™ to Enhance Performance on Your Existing Hardware

Nico Galoppo, Principal Engineer
Ryan Loney, Technical Product Manager

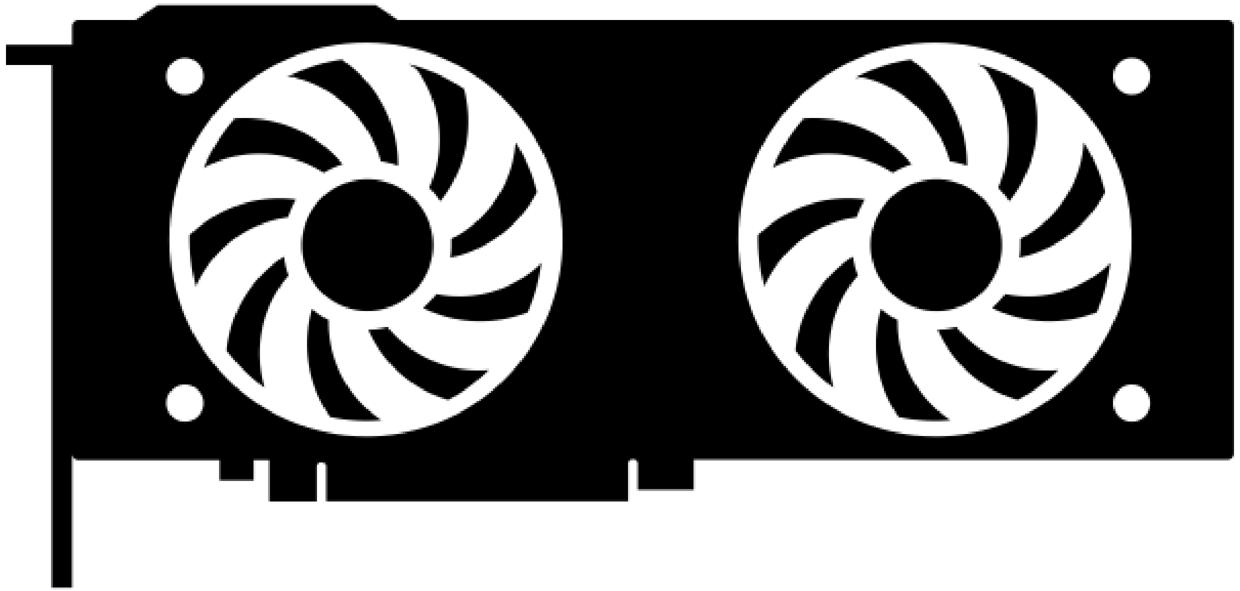
The Challenge

The Challenge

How do I Deploy My Great Neural Network?

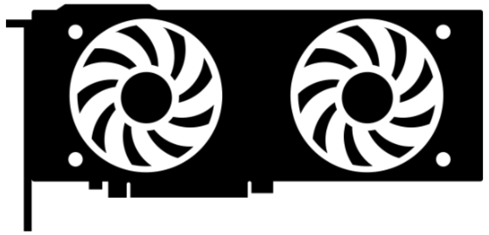


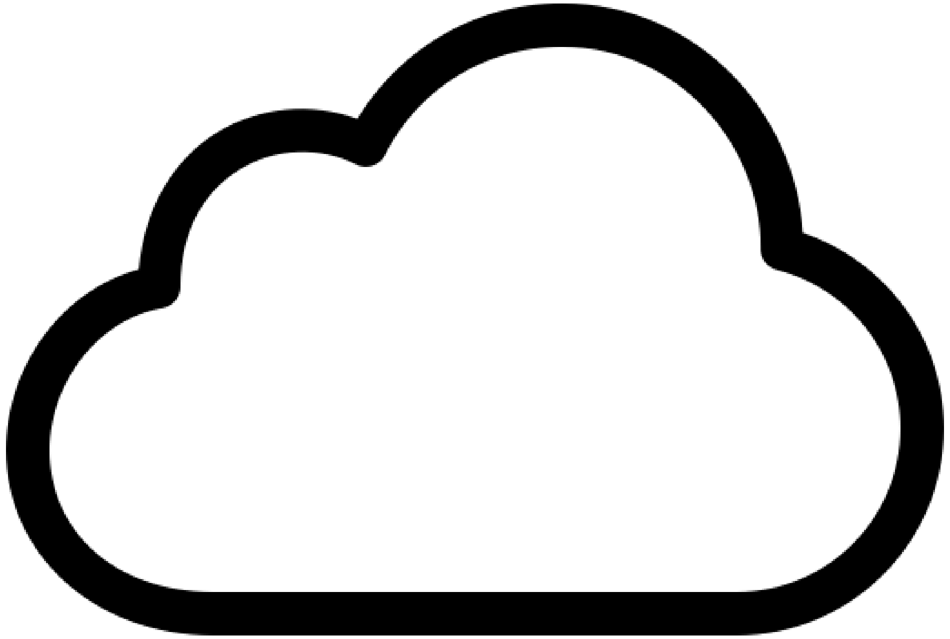
GPU?



Cloud?

GPU?

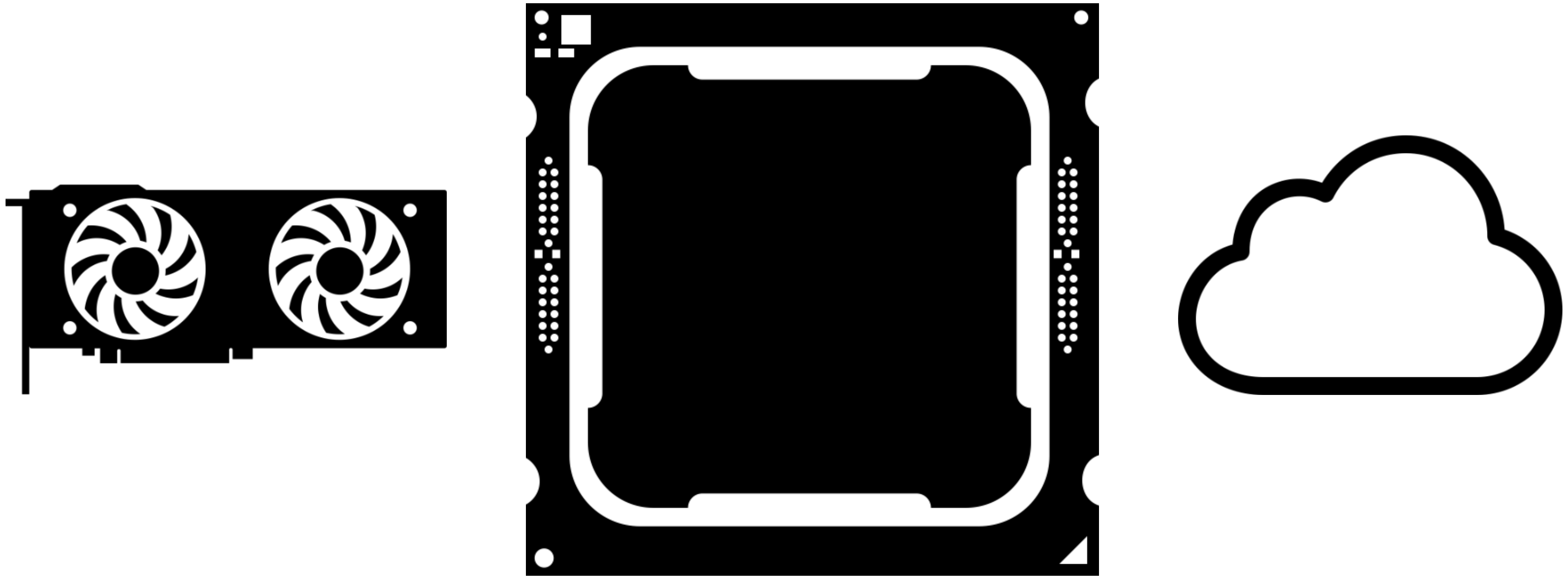




GPU?

Cloud?

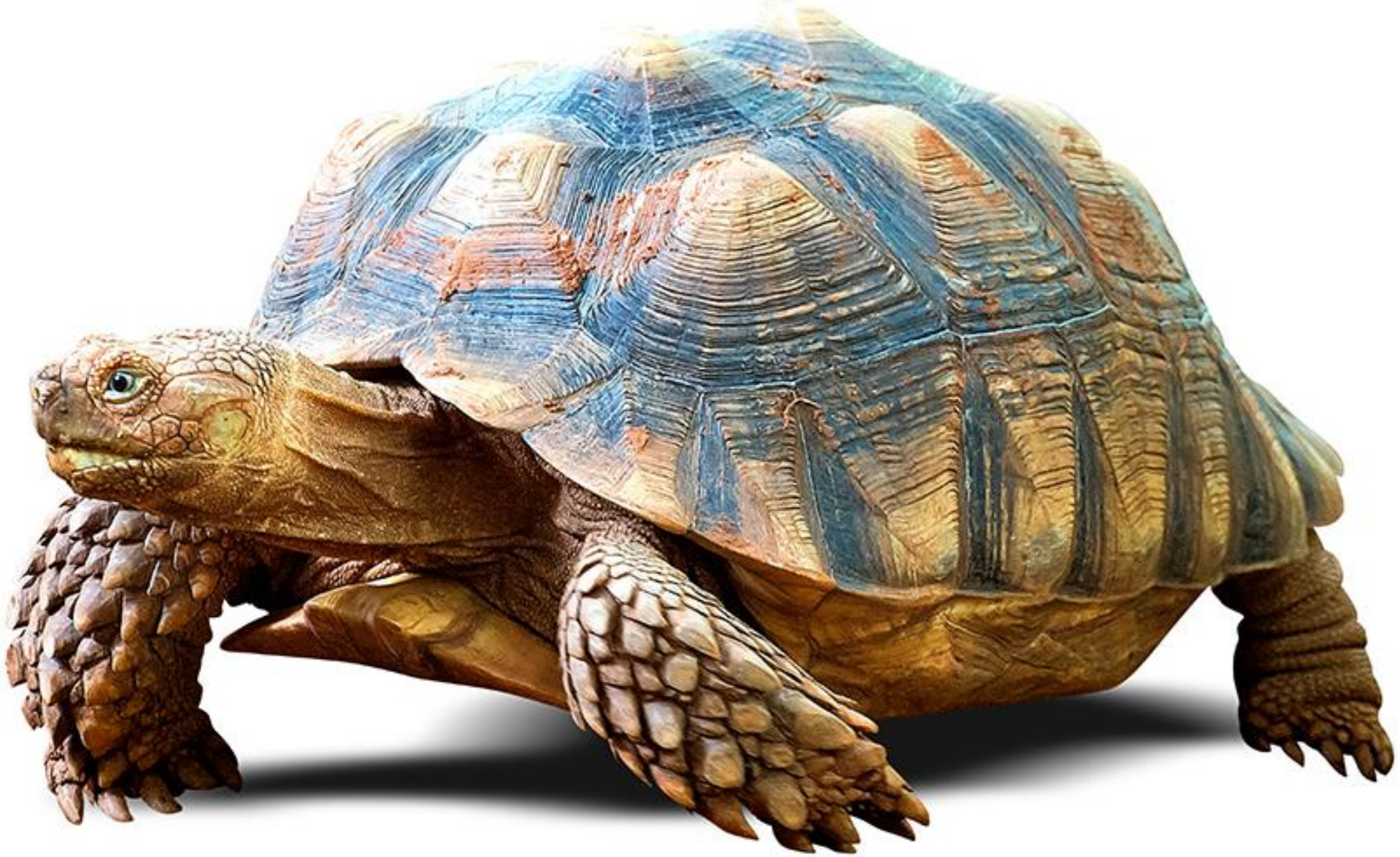
Maybe just CPU?



“But running on CPU is slow…”

Almost Every Deep Learning Engineer

Is it?



The Solution

OpenVINO™

Visual

+

NLP, Audio…

Inference

and **N**eural Network

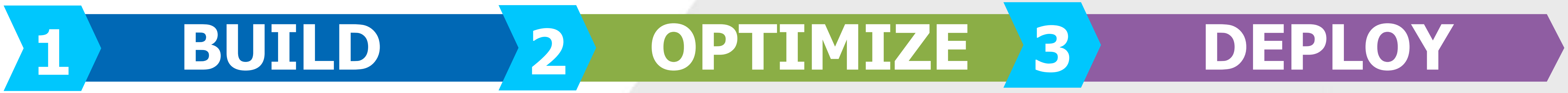
Optimization

Built on the foundation of

OpenVINO™ oneAPI

Developer Journey

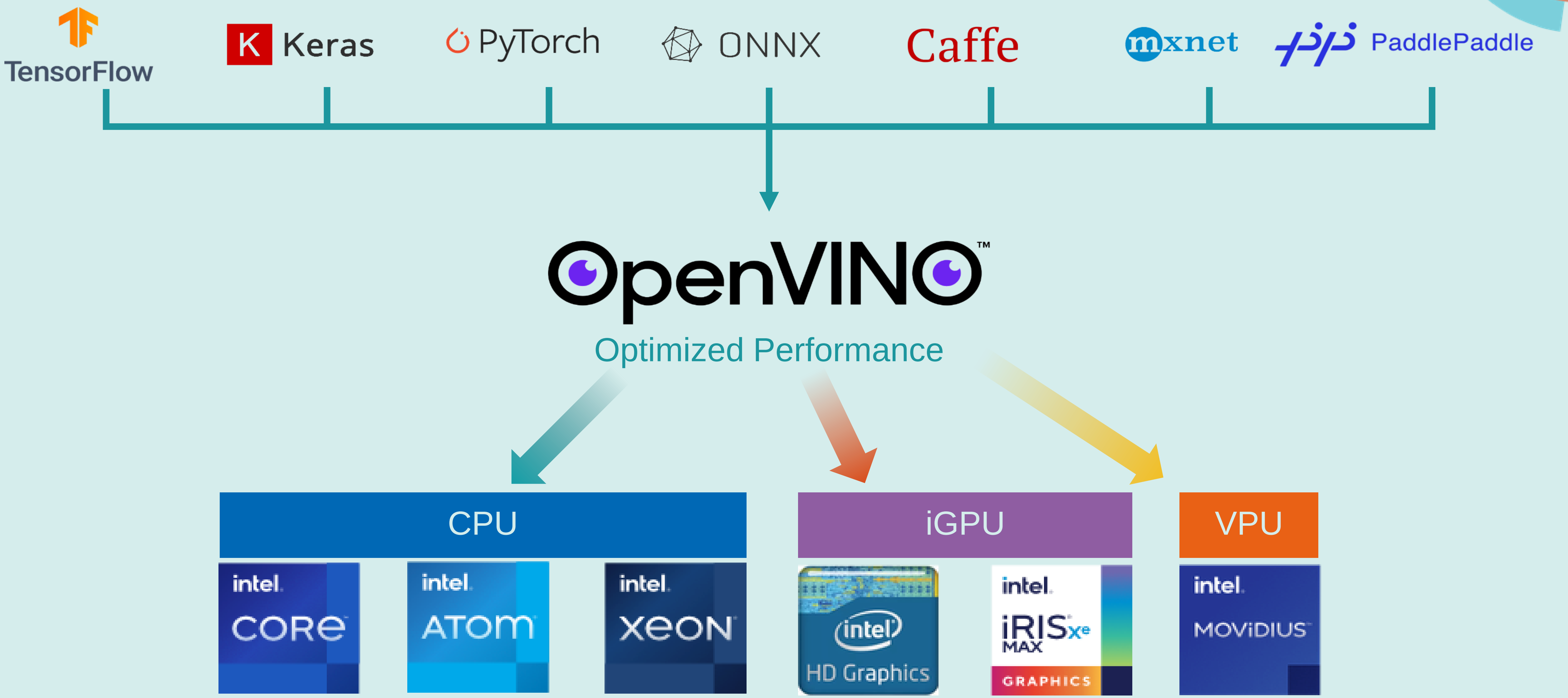
OpenVINO™



1 BUILD

2 OPTIMIZE

3 DEPLOY



Windows Linux macOS

Installation Methods

ubuntu

Red Hat

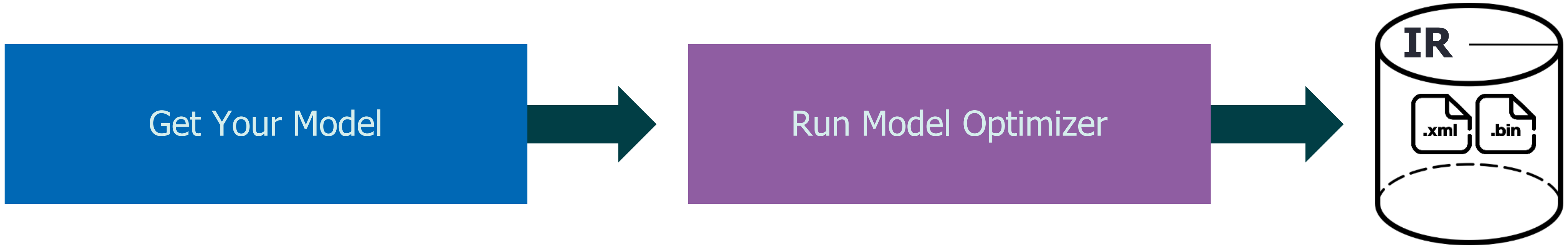
python Package Index

docker

ANACONDA

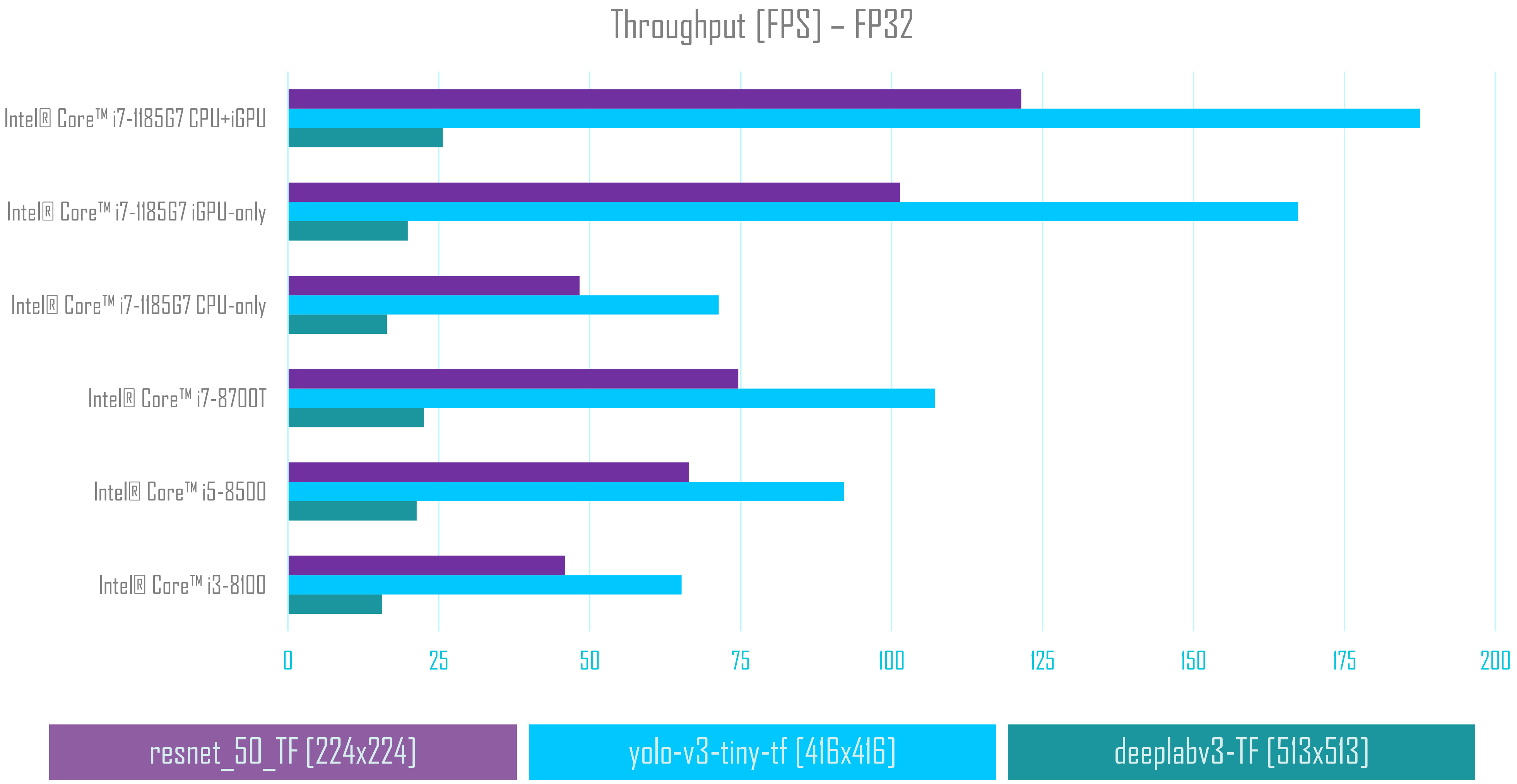
www.openvino.ai

Model Optimizer



For workloads and configurations visit www.intel.com/PerformanceIndex. Results may vary.

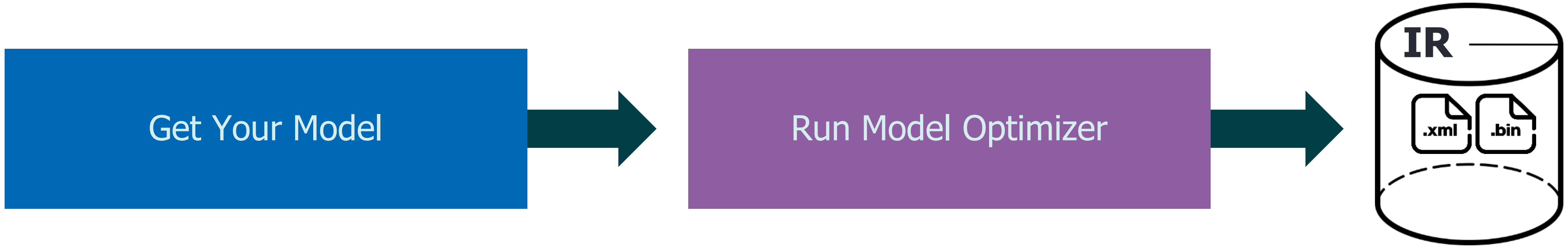
Performance



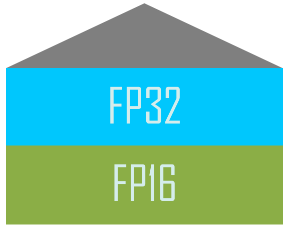
```
$ benchmark_app -m model_path -d device
```

For workloads and configurations please scan QR code. Results may vary.

Model Optimizer

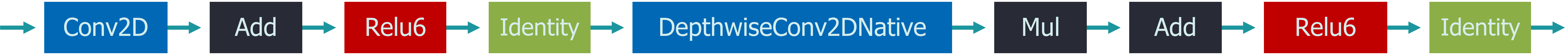


```
$ mo --input_model model.onnx
--data_type FP32
```



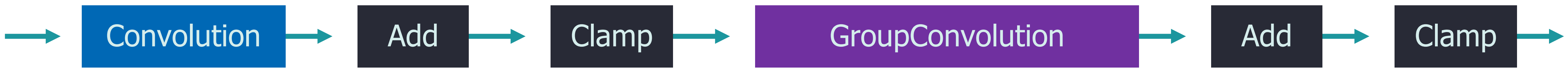
For workloads and configurations visit www.intel.com/PerformanceIndex. Results may vary.

Neural Network (any format)

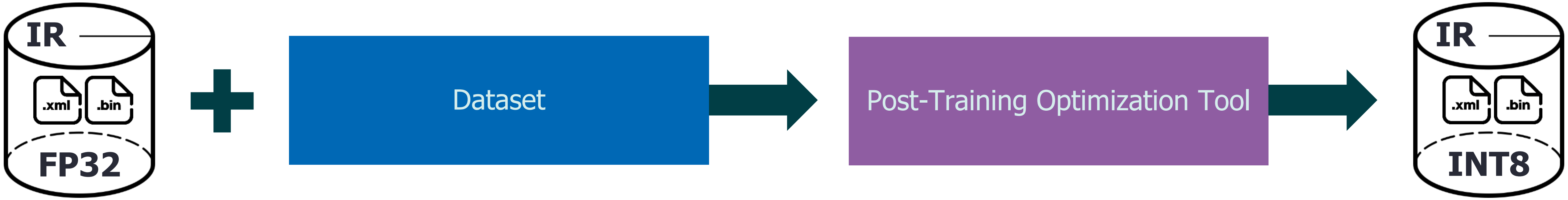


Conv2D → Add → Relu6 → Identity → DepthwiseConv2DNative → Mul → Add → Relu6 → Identity

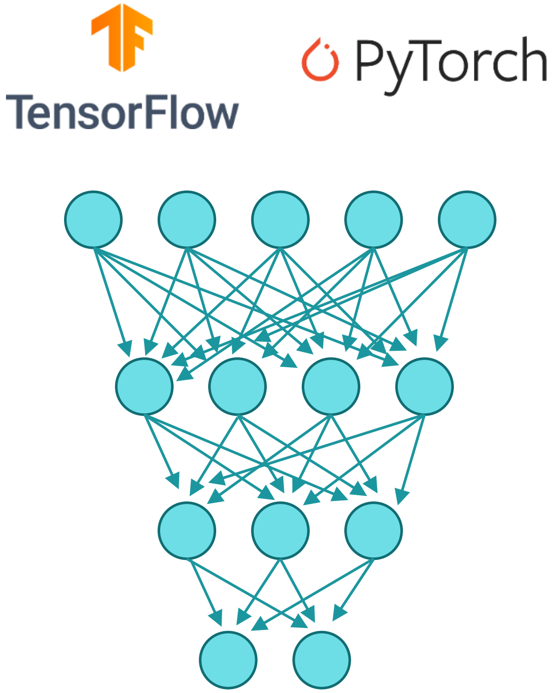
Intermediate Representation (IR)



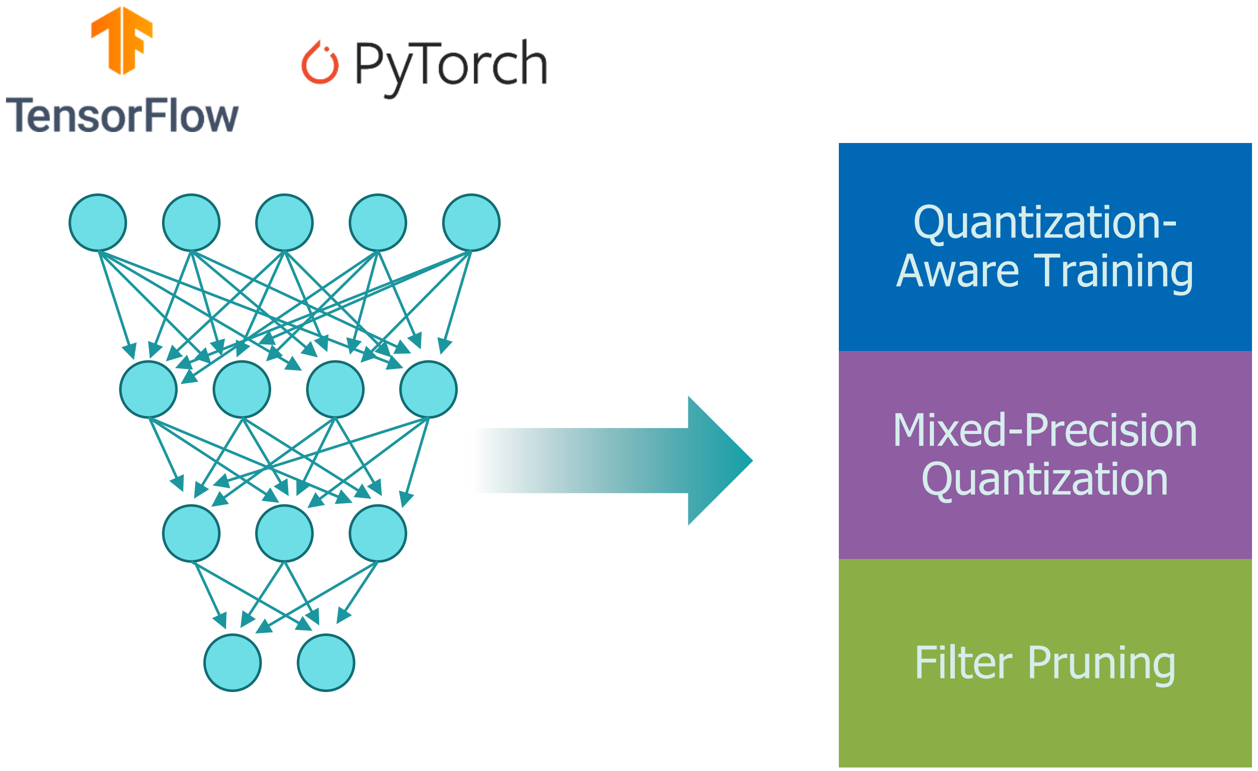
Post-Training Optimization Tool (POT)



Neural Network Compression Framework (NNCF)



Neural Network Compression Framework (NNCF)

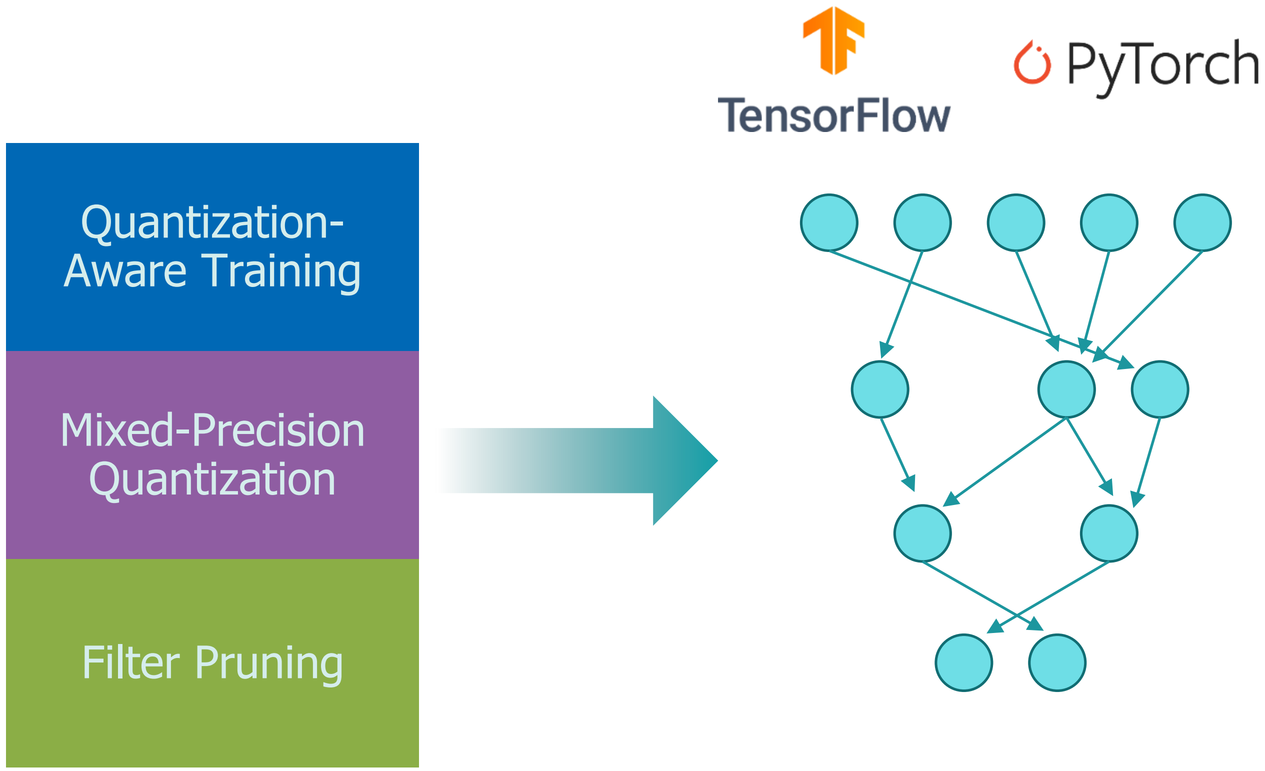


Quantization-Aware Training

Mixed-Precision Quantization

Filter Pruning

Neural Network Compression Framework (NNCF)



Quantization-Aware Training

Mixed-Precision Quantization

Filter Pruning

OpenVINO Runtime

```python
from openvino.runtime import Core

img = load_img()

core = Core()
model = core.read_model(model="model.xml", weights="model.bin")
compiled_model = core.compile_model(model=model, device_name="CPU")

output_layer = compiled_model.outputs[0]

result = compiled_model([img])[output_layer]
```

„dog"

© 2022 Intel

Supported Devices

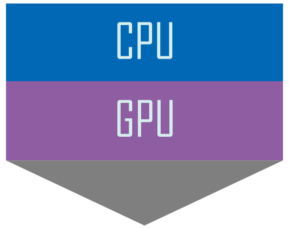
```
compiled_model = core.compile_model(model=model, device_name="CPU")
```

Supported Devices

CPU

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

Supported Devices

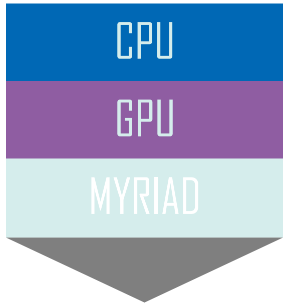


CPU

GPU

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

Supported Devices



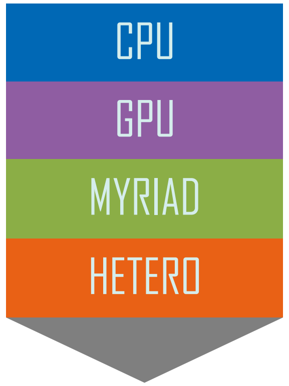
```
compiled_model = core.compile_model(model=model, device_name="CPU")
```

Supported Devices

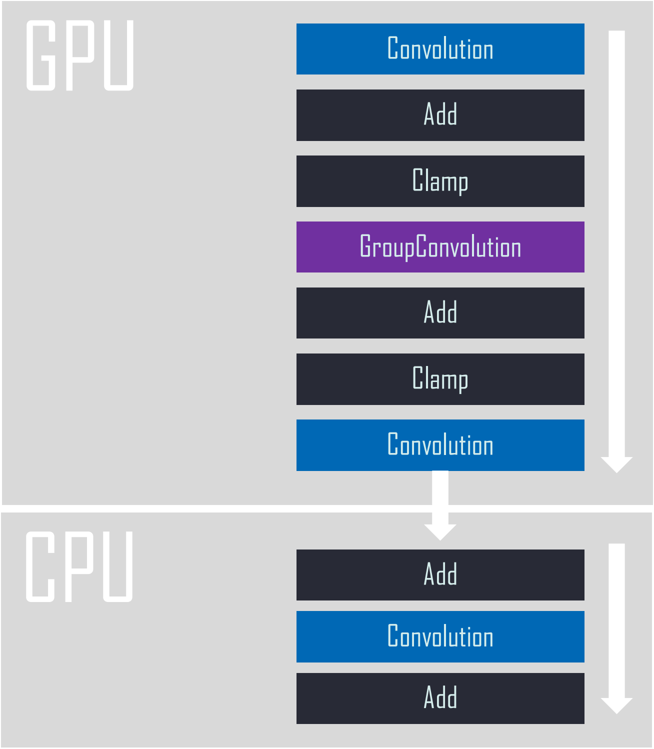
intel CORE i5

intel iRIS xe MAX GRAPHICS

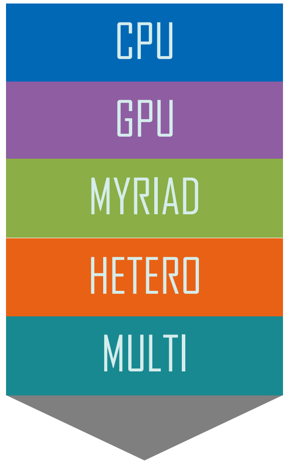
intel MOVIDIUS



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



Supported Devices



CPU

GPU

MYRIAD

HETERO

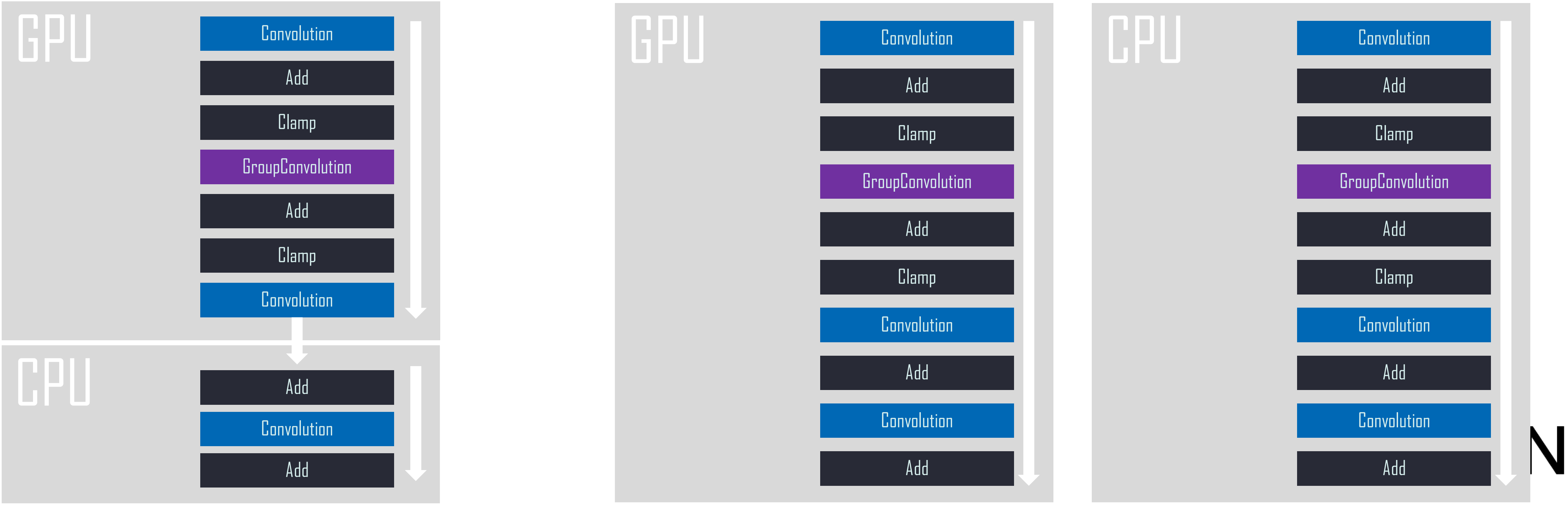
MULTI

intel. CORE i5

intel. iRIS Xe MAX GRAPHICS

intel. MOVIDIUS

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



GPU

- Convolution
- Add
- Clamp
- GroupConvolution
- Add
- Clamp
- Convolution

CPU

- Add
- Convolution
- Add

GPU

- Convolution
- Add
- Clamp
- GroupConvolution
- Add
- Clamp
- Convolution
- Add
- Convolution
- Add

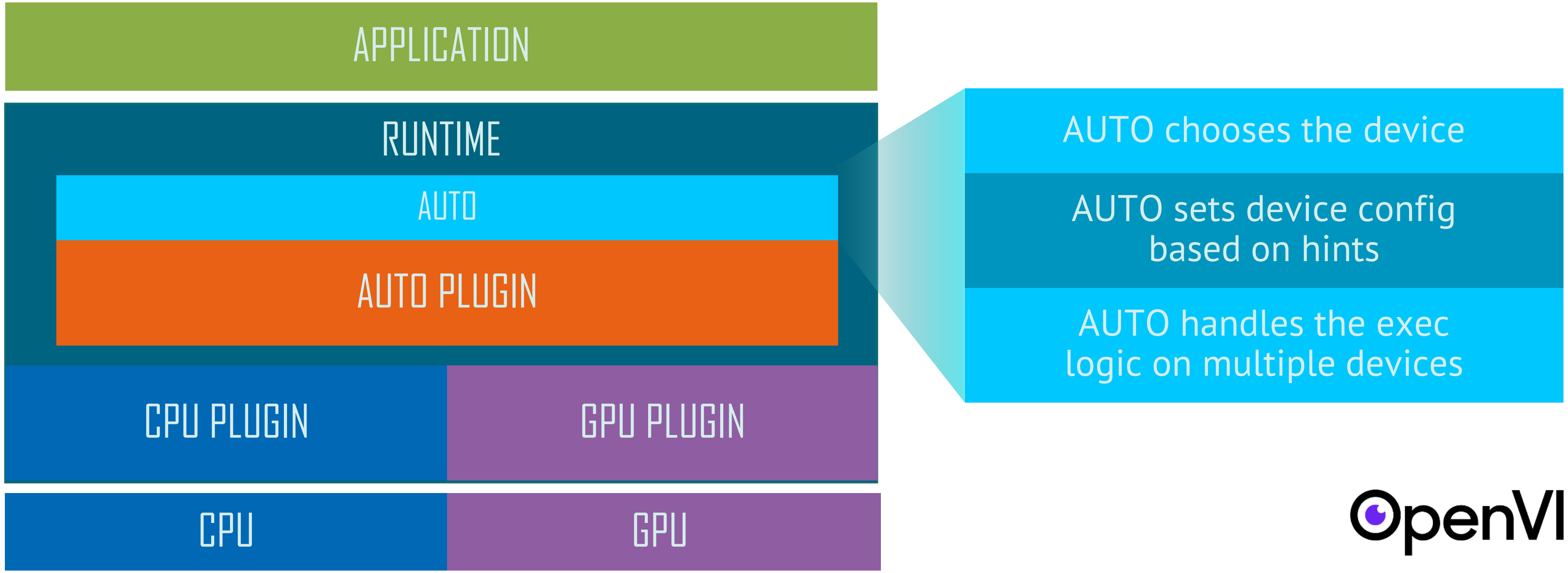
CPU

- Convolution
- Add
- Clamp
- GroupConvolution
- Add
- Clamp
- Convolution
- Add
- Convolution
- Add

Supported Devices

CPU
GPU
MYRIAD
HETERO
MULTI
AUTO

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



APPLICATION

RUNTIME

AUTO

AUTO PLUGIN

CPU PLUGIN

GPU PLUGIN

CPU

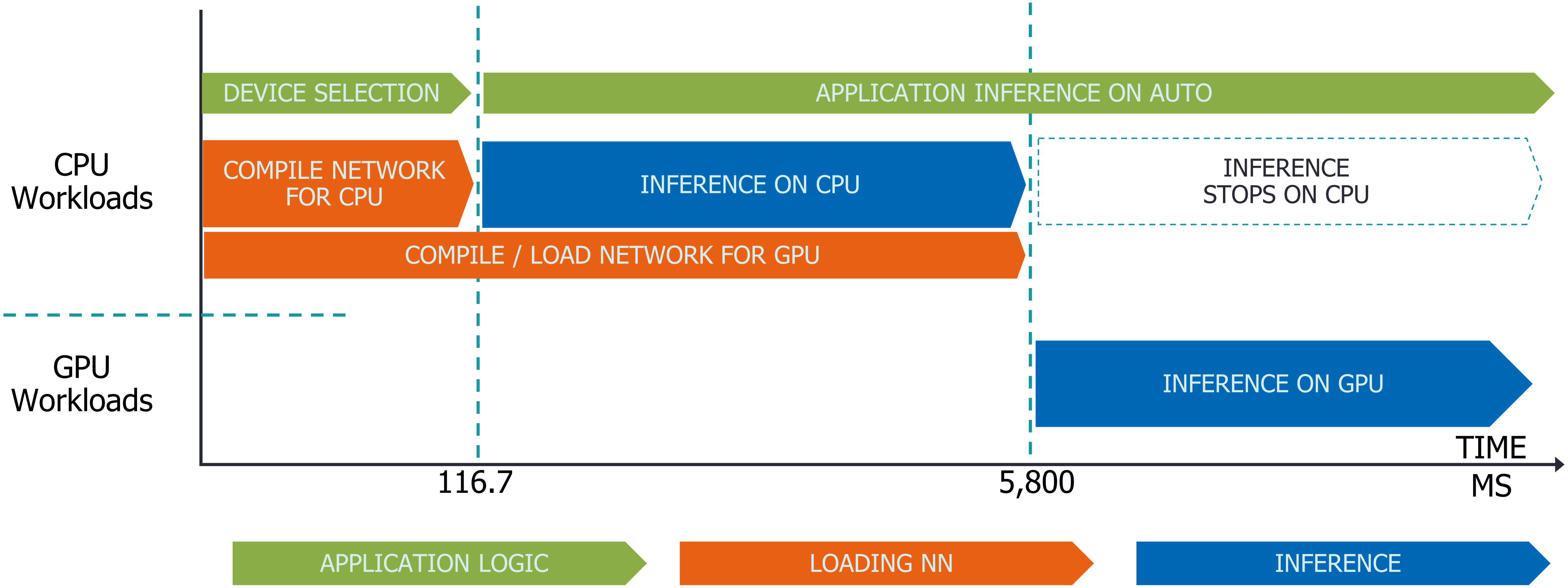
GPU

AUTO chooses the device

AUTO sets device config based on hints

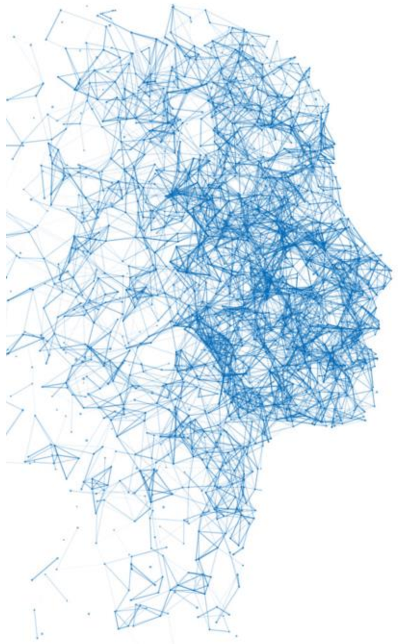
AUTO handles the exec logic on multiple devices

AUTO Device

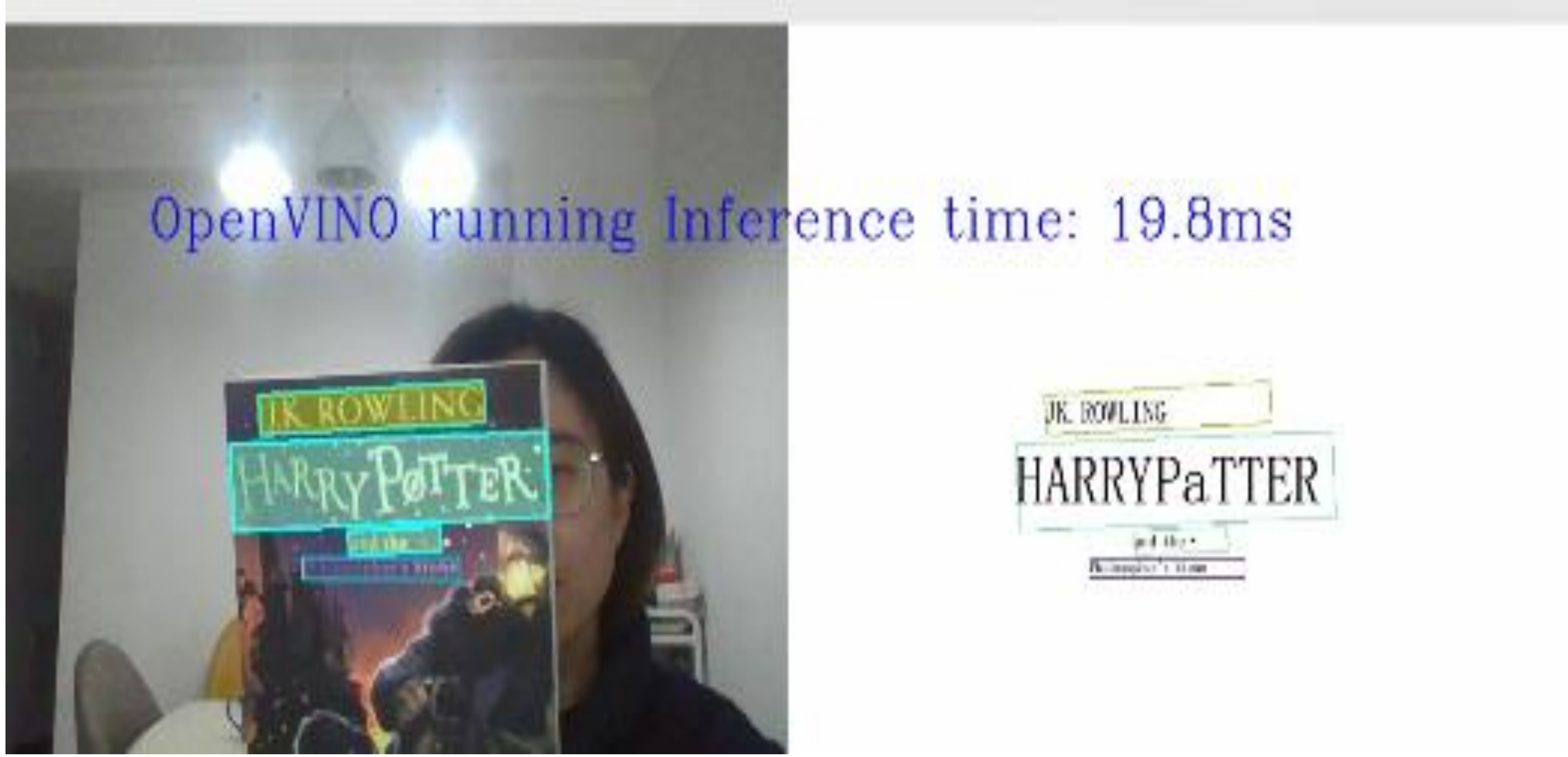


DEVICE SELECTION
APPLICATION INFERENCE ON AUTO
CPU Workloads
COMPILE NETWORK FOR CPU
INFERENCE ON CPU
INFERENCE STOPS ON CPU
COMPILE / LOAD NETWORK FOR GPU
GPU Workloads
INFERENCE ON GPU
TIME
MS
116.7
5,800
APPLICATION LOGIC
LOADING NN
INFERENCE

Input Data with Variable Shape?



“What is the weather going to be like today?”

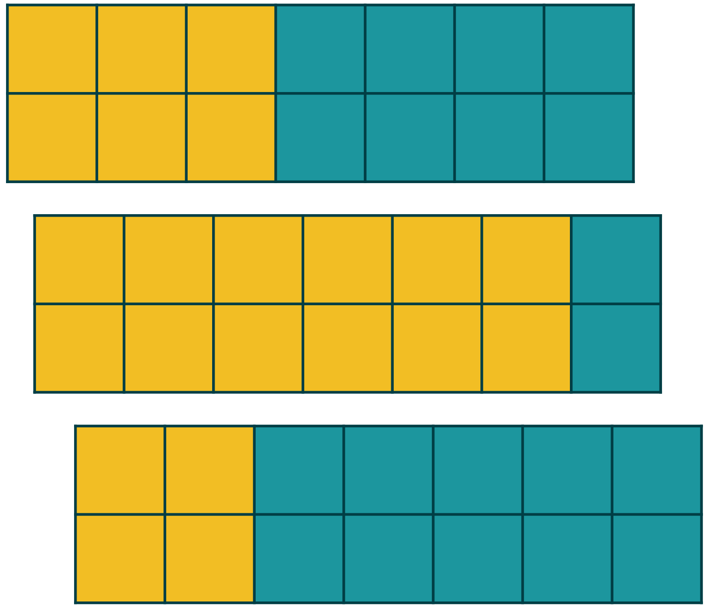


OpenVINO running Inference time: 19.8ms

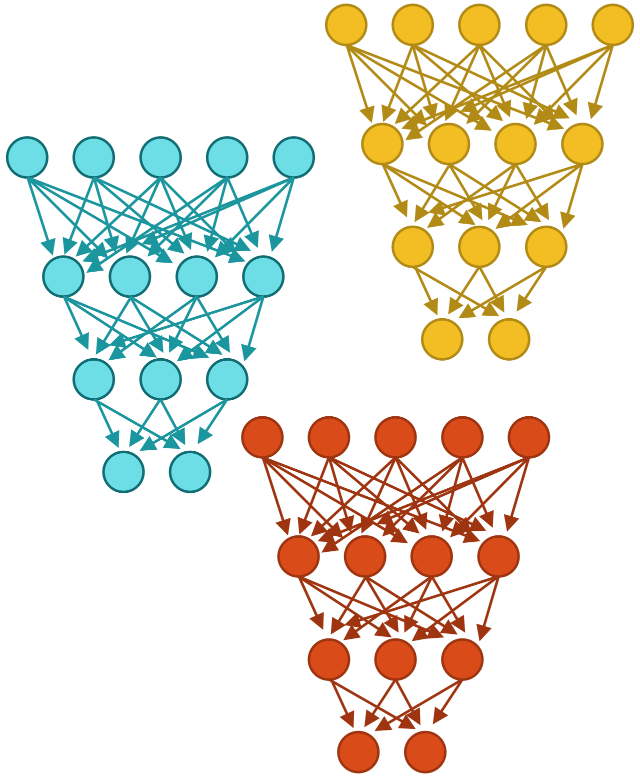
JK. ROWLING

HARRYPaTTER

How to handle variable input shapes?



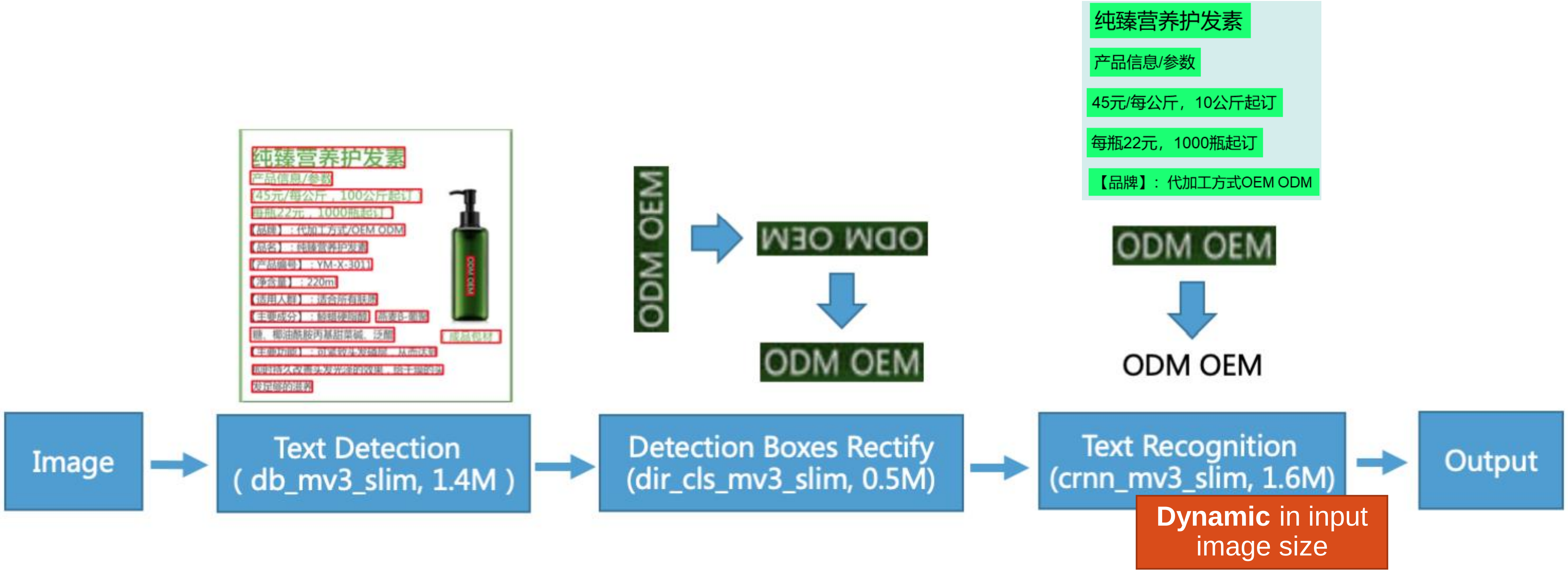
Padding



Multiple Precompiled Models

Inefficient and Cumbersome!

Native Dynamic Shapes Support in OpenVINO™



纯臻营养护发素

产品信息/参数

45元/每公斤，10公斤起订

每瓶22元，1000瓶起订

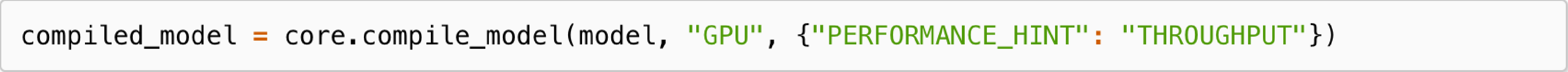
【品牌】：代加工方式OEM ODM

Image → Text Detection (db_mv3_slim, 1.4M) → Detection Boxes Rectify (dir_cls_mv3_slim, 0.5M) → Text Recognition (crnn_mv3_slim, 1.6M) → Output

Dynamic in input image size

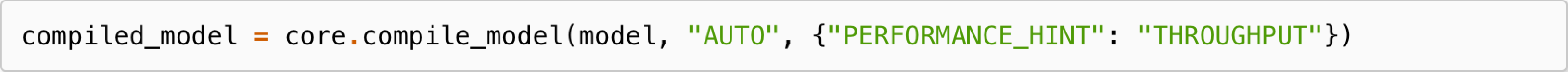
Portable Performance Hints

- Set-and-forget optimization knob – choose **latency** or **throughput.**
- Completely portable between the devices



```
compiled_model = core.compile_model(model, "GPU", {"PERFORMANCE_HINT": "THROUGHPUT"})
```

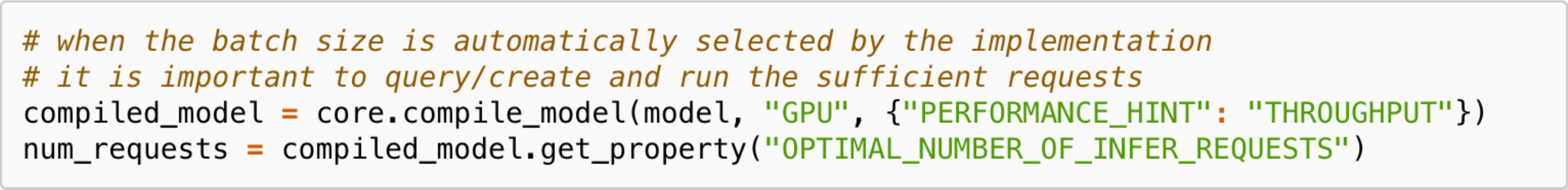
- Throughput hint drives device-specific optimizations
- Even works for AUTO!



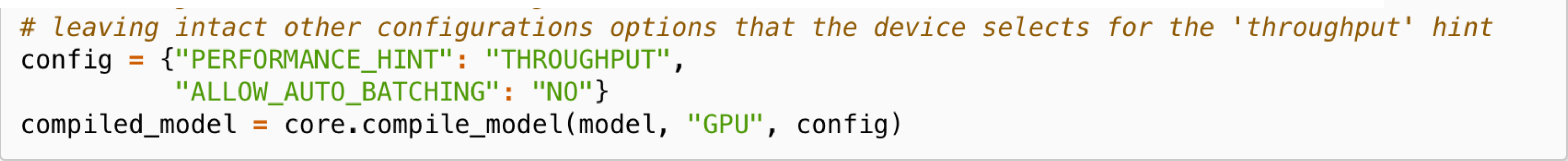
```
compiled_model = core.compile_model(model, "AUTO", {"PERFORMANCE_HINT": "THROUGHPUT"})
```

Automatic Batching

- How do you choose a good batch size?
- Let the OpenVINO™ runtime decide for you!
- No need to batch requests, runtime will do that too!
- Batching can improve throughput with select devices and models.



```
# when the batch size is automatically selected by the implementation
# it is important to query/create and run the sufficient requests
compiled_model = core.compile_model(model, "GPU", {"PERFORMANCE_HINT": "THROUGHPUT"})
num_requests = compiled_model.get_property("OPTIMAL_NUMBER_OF_INFER_REQUESTS")
```



```
# leaving intact other configurations options that the device selects for the 'throughput' hint
config = {"PERFORMANCE_HINT": "THROUGHPUT",
          "ALLOW_AUTO_BATCHING": "NO"}
compiled_model = core.compile_model(model, "GPU", config)
```

Open Model Zoo

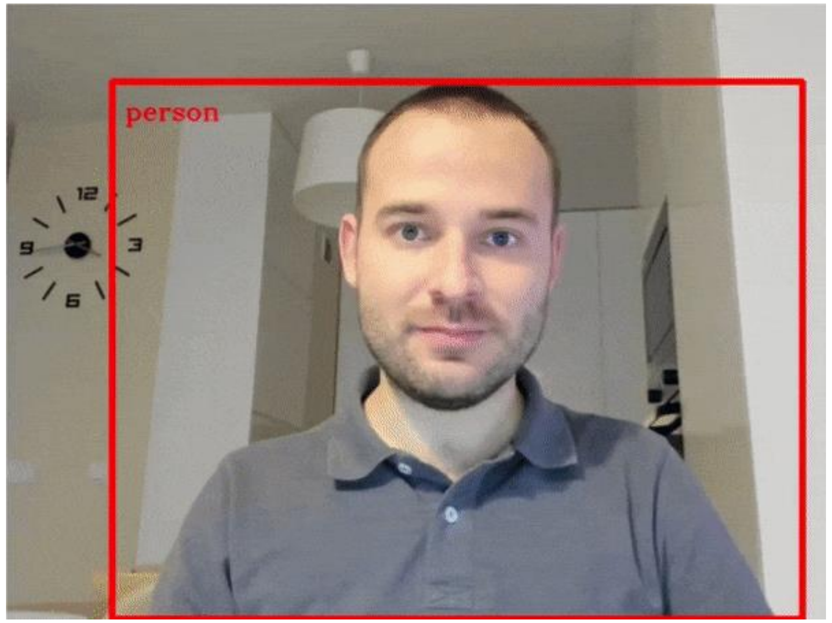
300+

Pre-Trained + Optimized Models

Open Model Zoo

300+

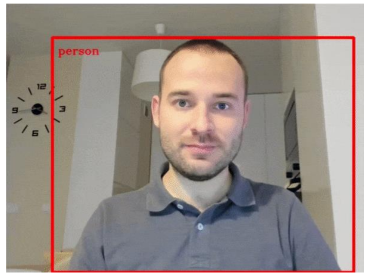
Pre-Trained + Optimized Models

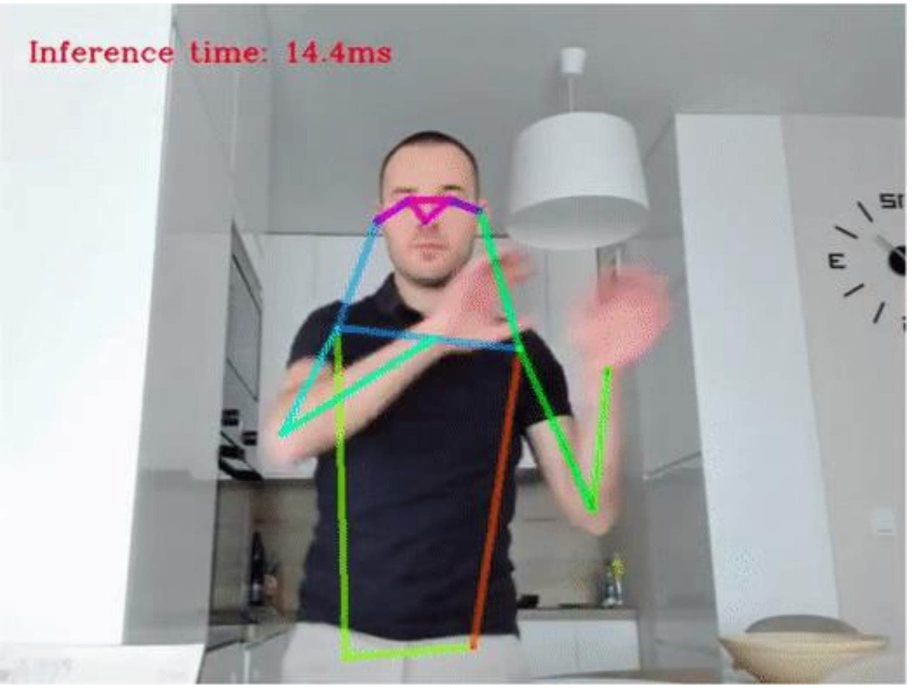


Open Model Zoo

300+

Pre-Trained + Optimized Models

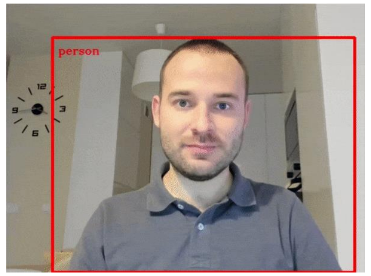


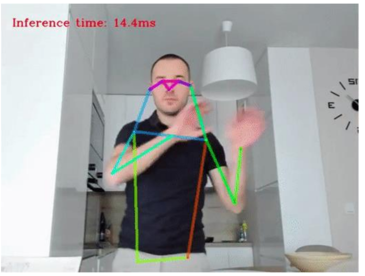


Open Model Zoo

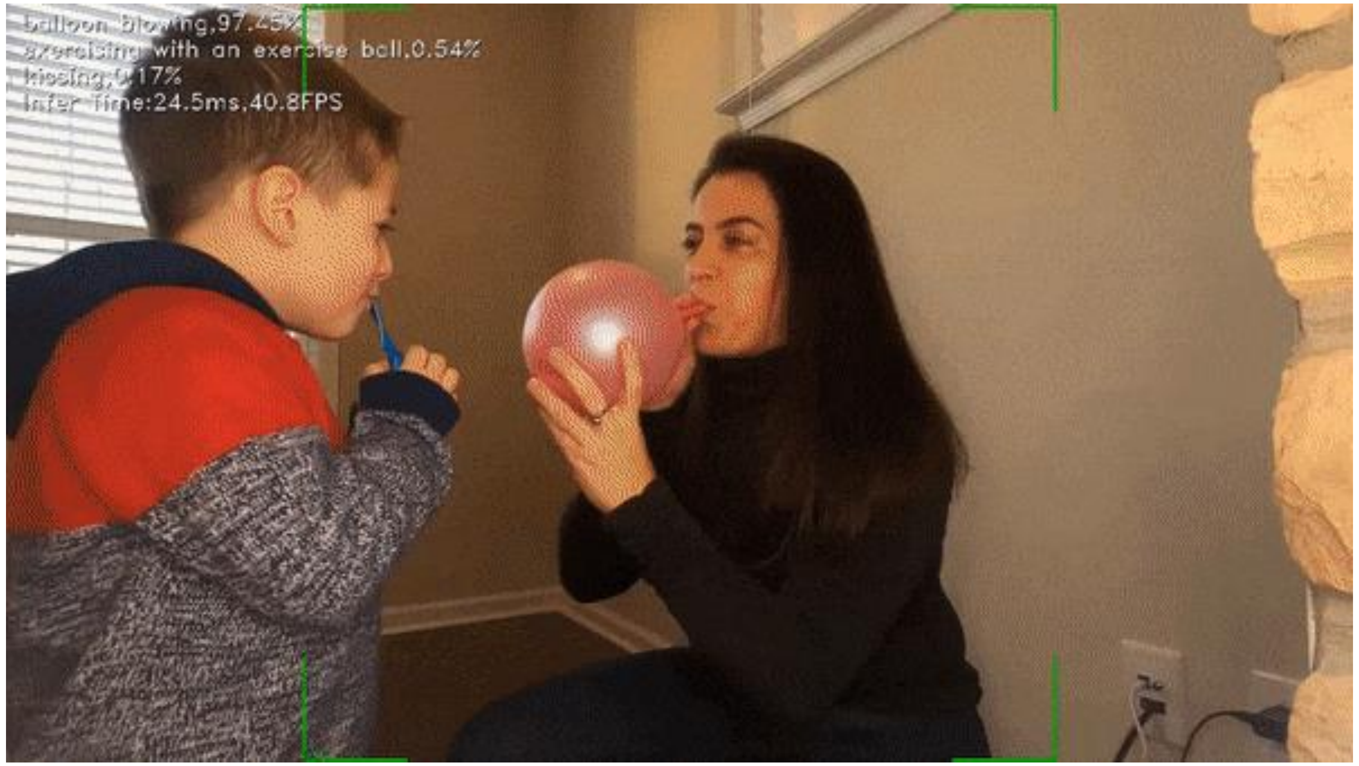
300+

Pre-Trained + Optimized Models





```
Inference time: 14.4ms
```

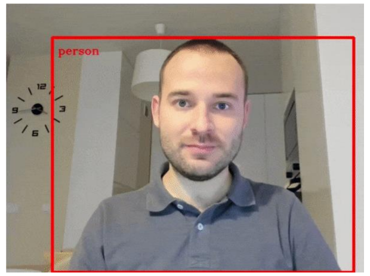


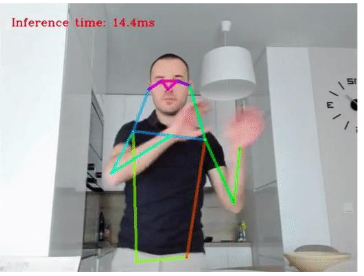
```
balloon blowing,97.45%
exercising with an exercise ball,0.54%
kissing,0.17%
Infer time:24.5ms,40.8FPS
```

Open Model Zoo

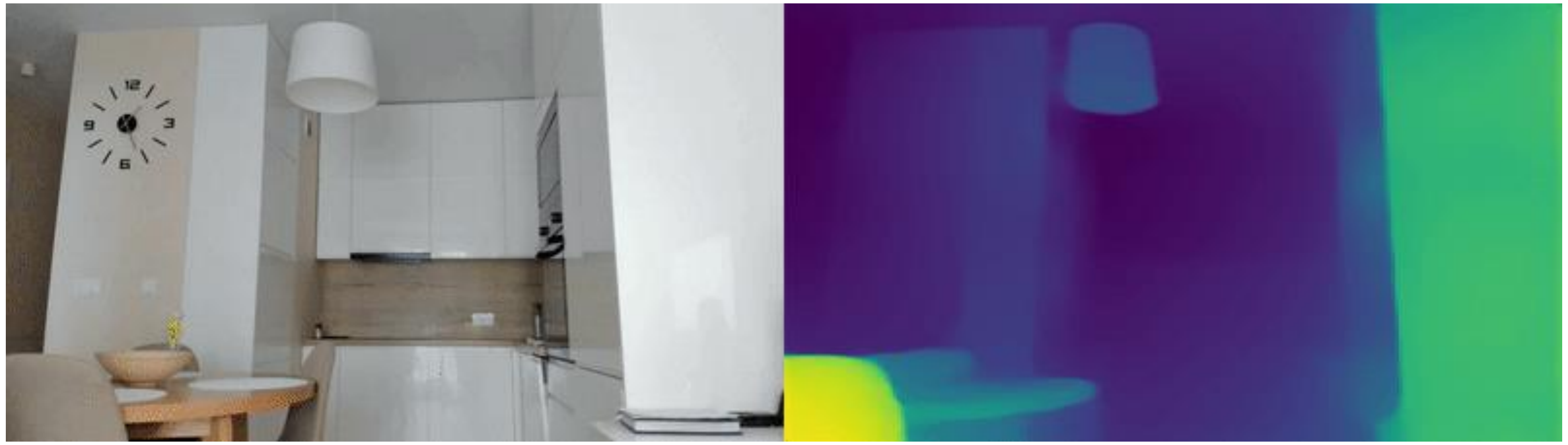
300+

Pre-Trained + Optimized Models





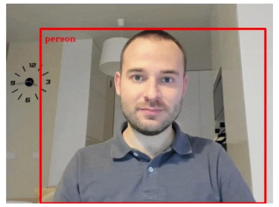
balloon blowing,97.45%
exercising with an exercise ball,0.54%
kissing,0.17%
Infer Time:24.5ms,40.8FPS

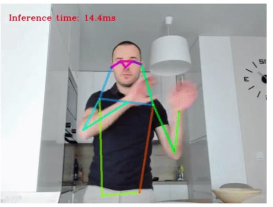


Open Model Zoo

300+

Pre-Trained + Optimized Models



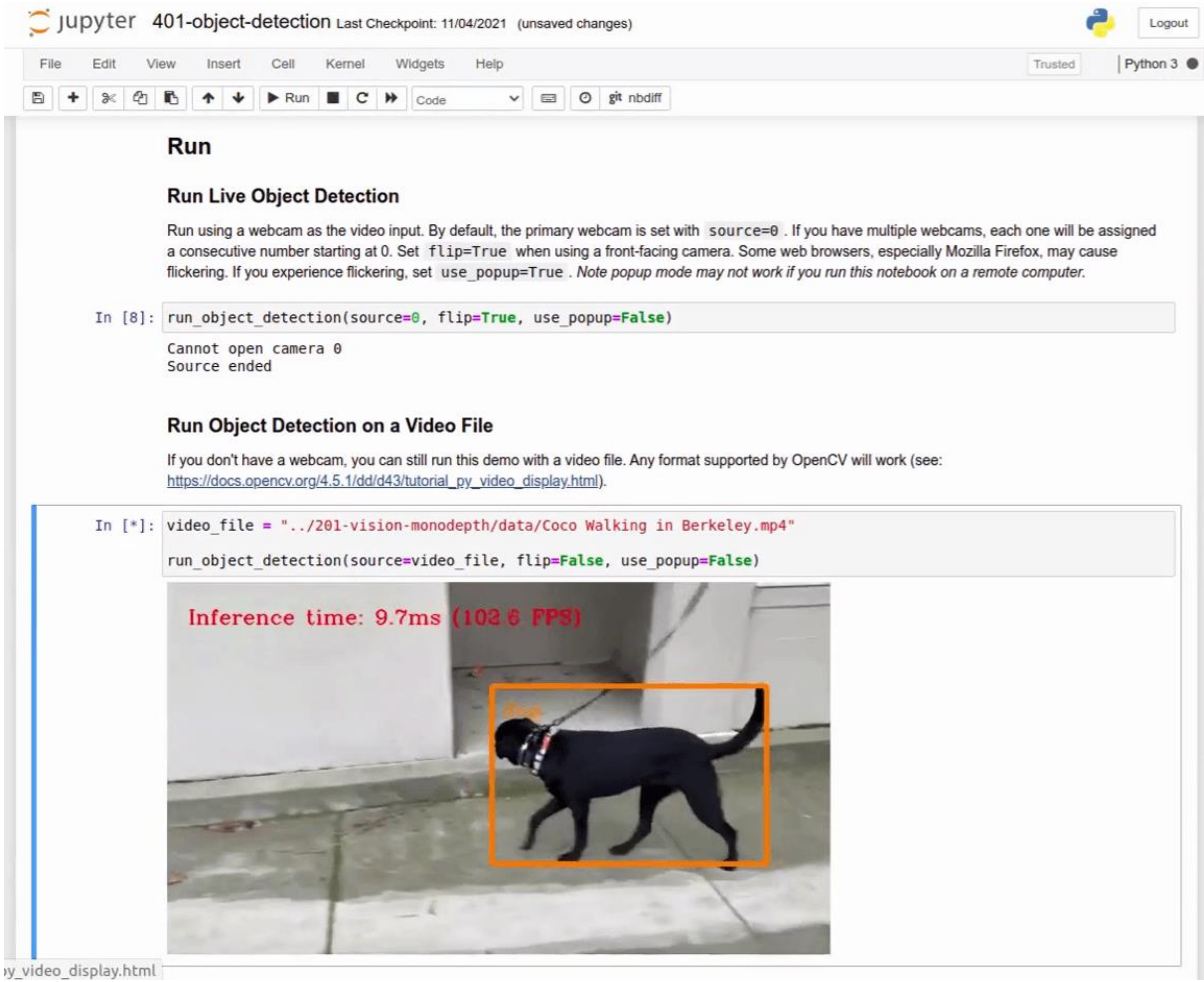


Open Model Zoo

300+

Pre-Trained + Optimized Models

Jupyter Notebooks



Run

Run Live Object Detection

Run using a webcam as the video input. By default, the primary webcam is set with `source=0`. If you have multiple webcams, each one will be assigned a consecutive number starting at 0. Set `flip=True` when using a front-facing camera. Some web browsers, especially Mozilla Firefox, may cause flickering. If you experience flickering, set `use_popup=True`. *Note popup mode may not work if you run this notebook on a remote computer.*

```
In [8]: run_object_detection(source=0, flip=True, use_popup=False)
```

```
Cannot open camera 0
Source ended
```

Run Object Detection on a Video File

If you don't have a webcam, you can still run this demo with a video file. Any format supported by OpenCV will work (see: https://docs.opencv.org/4.5.1/dd/d43/tutorial_py_video_display.html).

```
In [*]: video_file = "../201-vision-monodepth/data/Coco Walking in Berkeley.mp4"

        run_object_detection(source=video_file, flip=False, use_popup=False)
```

Try it yourself!

Learn more and download at openvino.ai

Complete the Intel® Edge AI Certification!

- Supercharge your career
- Approx 20 hours of video + quizzes + coding

Deep Dive Session

Thursday, May 19

Intel AI Developer Expo—Let's Build Something Wonderful Together

Session: **3:00 – 5:30 pm**
Reception: **5:30 - 7:30 pm**

Location: **Room 209/210**

Afternoon snack and post-session Reception will be provided

Thank you!

Platform Configurations for Performance Benchmarks

Configuration	Intel® Core™ i7-1185G7	Intel® Core™ i3-8100	Intel® Core™ i5-8500	Intel® Core™ i7-8700T
Motherboard	Intel Corporation internal/ Reference Validation Platform	GIGABYTE* Z390 UD	ASUS* PRIME Z370-A	GIGABYTE* Z370M DS3H-CF
CPU	Intel® Core™ i7-1185G7 @ 3.00GHz	Intel® Core™ i3-8100 CPU @ 3.60GHz	Intel® Core™ i5-8500 CPU @ 3.00GHz	Intel® Core™ i7-8700T CPU @ 2.40GHz
Hyper Threading	ON	OFF	OFF	ON
Turbo Setting	ON	OFF	ON	ON
Memory	2 x 8 GB DDR4 3200MHz	4 x 8 GB DDR4 2400MHz	2 x 16 GB DDR4 2666MHz	4 x 16 GB DDR4 2400MHz
Operating System	Ubuntu* 18.04 LTS	Ubuntu* 18.04 LTS	Ubuntu* 18.04 LTS	Ubuntu* 18.04 LTS
Kernel Version	5.8.0-05-generic	5.3.0-24-generic	5.3.0-24-generic	5.3.0-24-generic
BIOS Vendor	Intel Corporation	American Megatrends Inc.*	American Megatrends Inc.*	American Megatrends Inc.*
BIOS Version	TGLSFWI1.R00.3425. A00.2010162309	F8	2401	F14c
BIOS Release	16-Oct-20	24-May-19	12-Jul-19	23-Mar-21
BIOS Settings	Default Settings	Select optimized default settings, set OS type to “other”, save and exit	Select optimized default settings, save and exit	Select optimized default settings, set OS type to “other”, save and exit
Batch size	1	1	1	1
Precision	FP32	FP32	FP32	FP32
Number of concurrent inference requests	4	4	3	4
Test Date	18-Jun-21	18-Jun-21	18-Jun-21	18-Jun-21
Rated maximum TDP/socket in Watt	28	65	65	35

Notices and Disclaimers

Performance varies by use, configuration and other factors. Learn more at www.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.

Intel technologies may require enabled hardware, software or service activation.
Your costs and results may vary.

Intel is committed to respecting human rights and avoiding complicity in human rights abuses. See Intel's Global Human Rights Principles. Intel's 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.