Outline

▪ Prework – OpenVINO Notebooks installation	20 minutes
▪ General aspects of OpenVINO ▪ Hands-on experience – OpenVINO Notebooks	40 minutes
	10 minutes break

▪ General Aspects of Optimization process
  ▪ Model Optimizer
  ▪ Post training Optimization
  ▪ NNCF with segmentation model
  ▪ Improve performance with AUTO plugin
	80 minutes

	10 minutes break
▪ OpenVINO Training Extensions (OTE) ▪ Object detection with OTE	20 minutes
▪ Anomalib by OpenVINO ▪ End2End experience	40 minutes
	10 minutes wrap

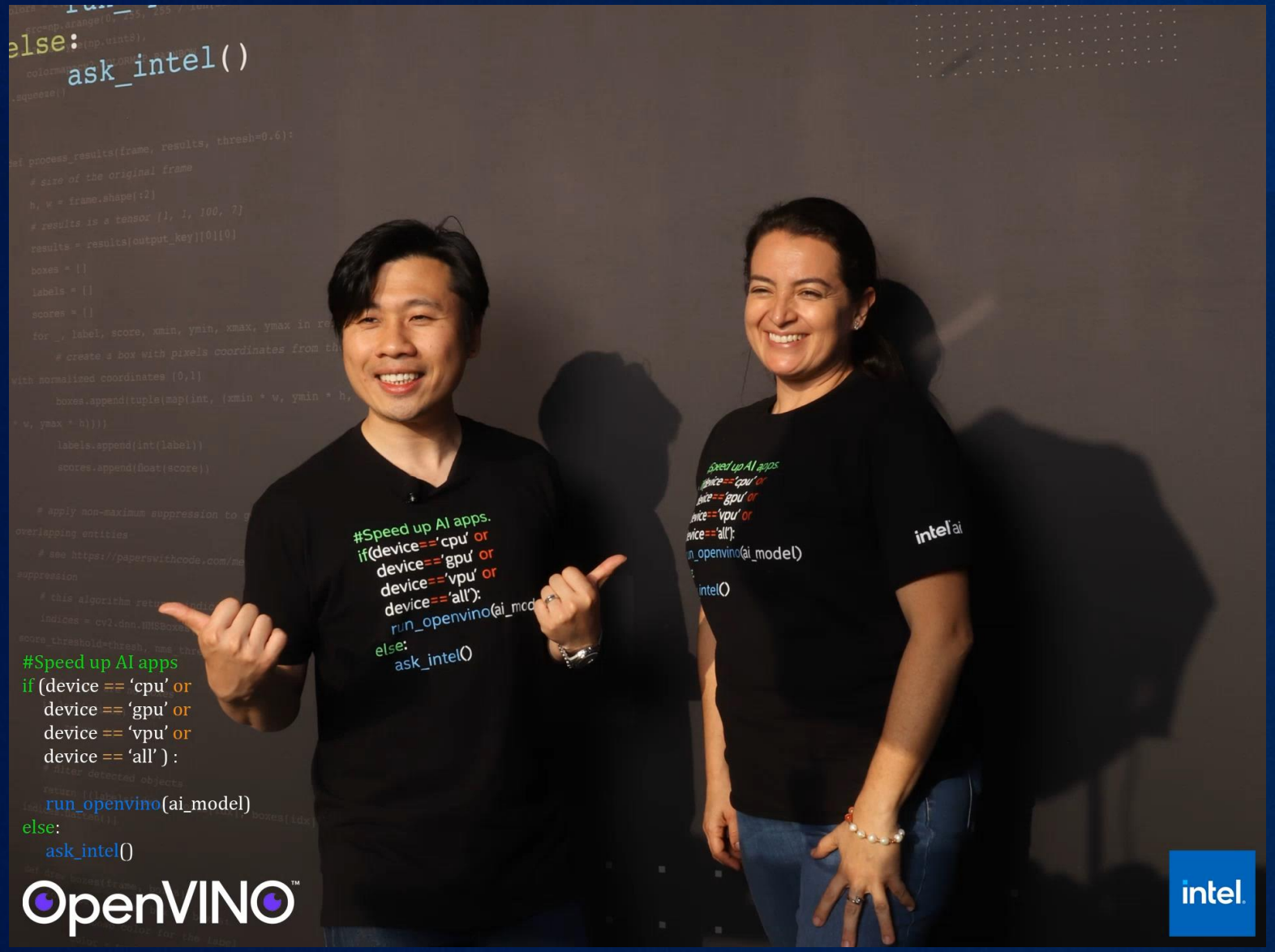
intel

OpenVINO™
CVPR 2022

How to Get Quick and Performant Model for Your Edge Application. From Data To Application.

Paula Ramos, PhD and Raymond Lo, PhD
AI Software Evangelists

Contributors: Helena Kloosterman, Samet Akcay, Zhuo Wu, and Yury Gorbachev



Raymond Lo
.linkedin.com/in/raymondlo84/

Paula Ramos
linkedin.com/in/paula-ramos-41097319/

Pre-Work

Setup and Install

OpenVINO™

Prepare Yourself for Intel's Tutorial in CVPR

June 19th- Afternoon Session

- ☑ Follow the instructions of this Wiki page. https://intel.ly/3aMjE0r
- ☑ Run the notebooks 301 and 401
- ☑ Share screenshots of your results in our repo. https://intel.ly/3MGDRIK

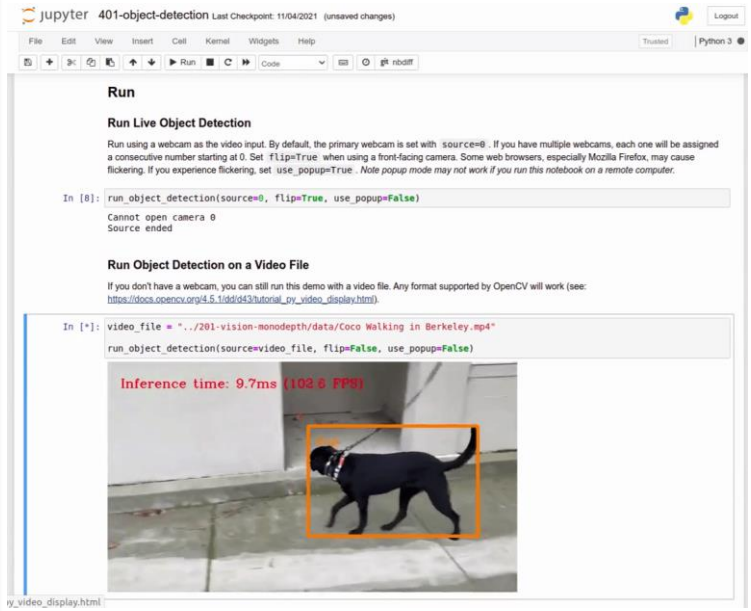
We will have some prizes at the end of the tutorial for those who complete prework.

OpenVINO Notebooks

```
# For Ubuntu
# Step 1: Create the Environment
$ python3 -m venv openvino_env
$ source openvino_env/bin/activate
```

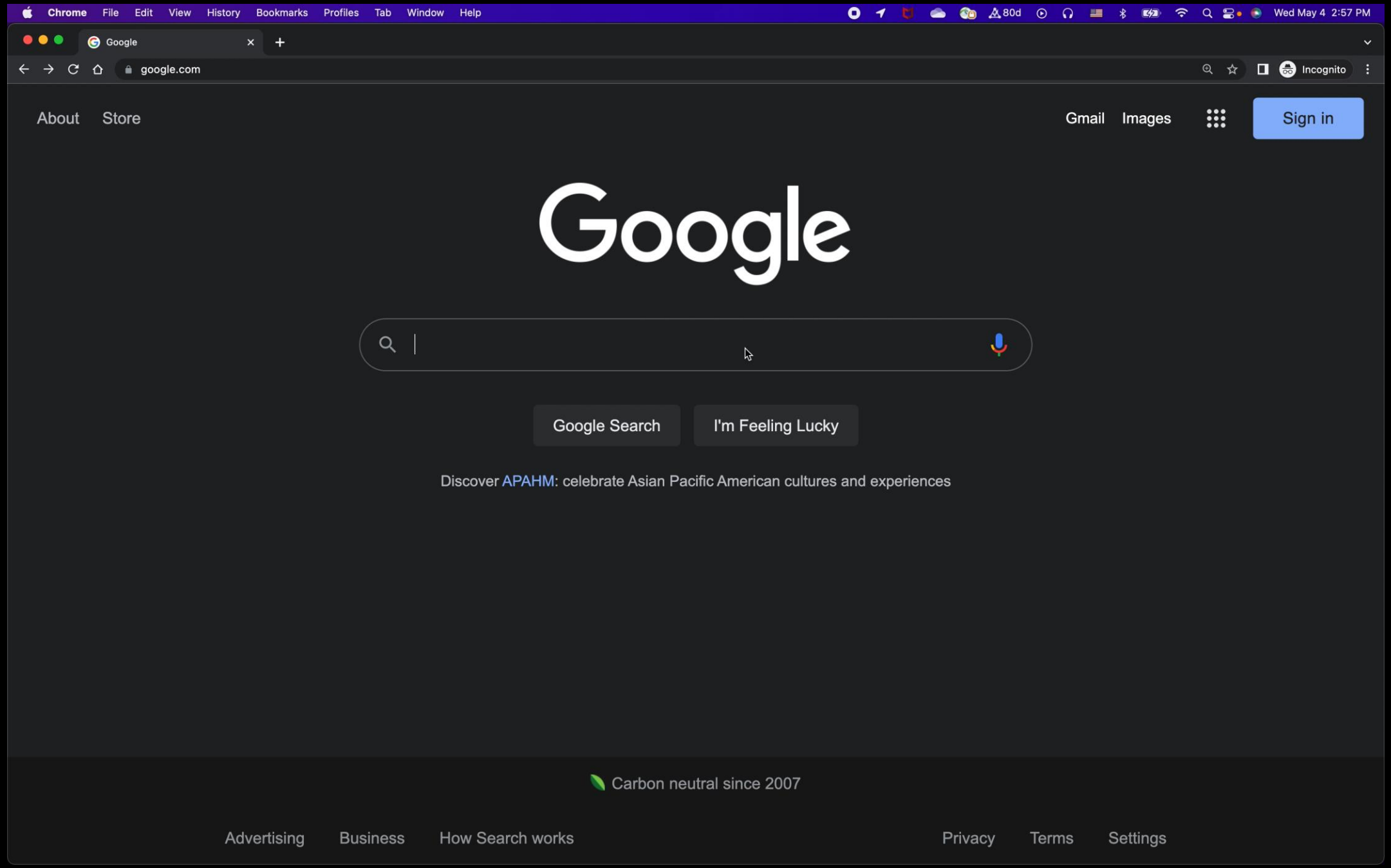
```
# For Windows
# Step 1: Create the Environment
$ python -m venv openvino_env
$ openvino_env\Scripts\activate
```

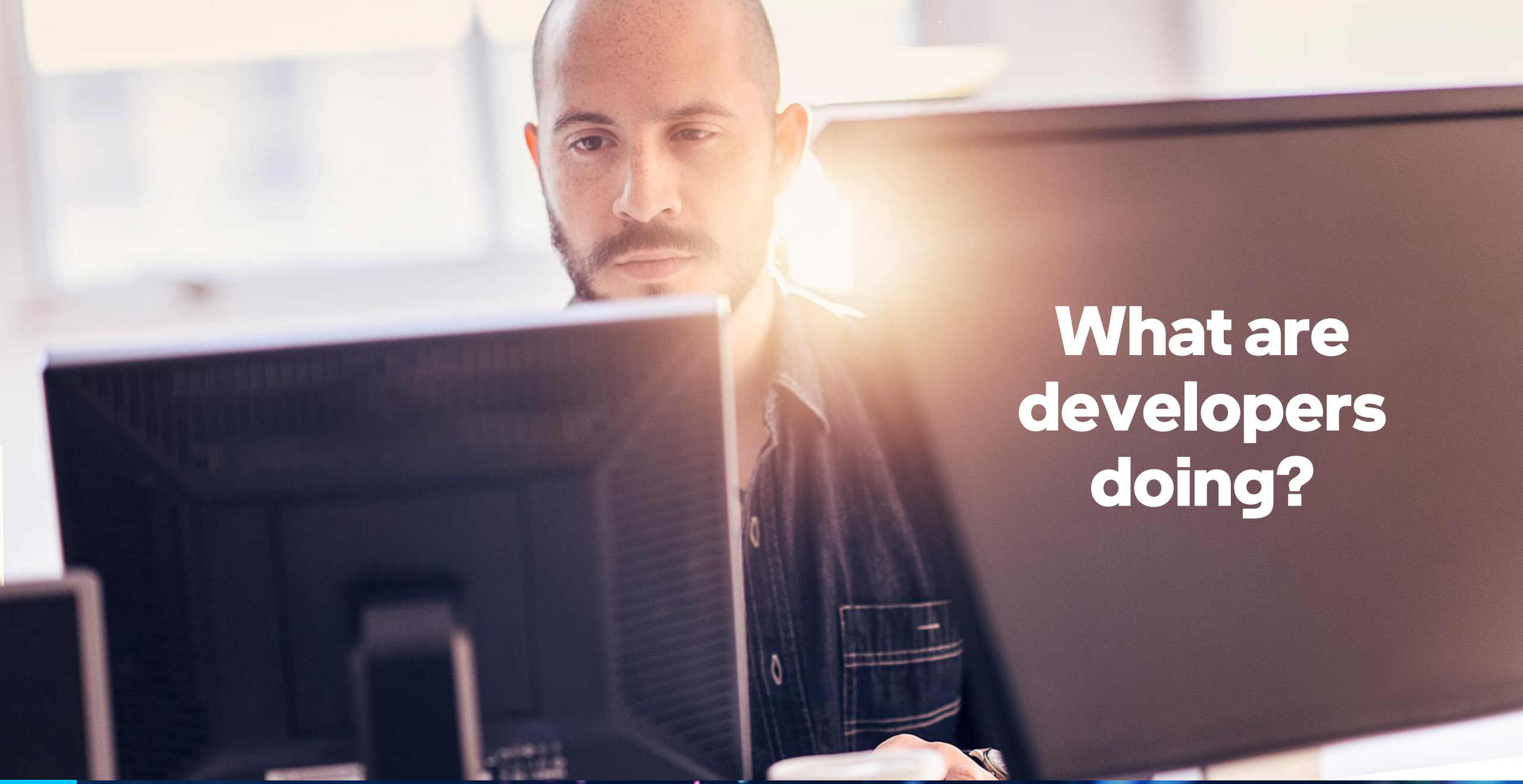
```
# Step 2: Clone the Repository
$ git clone https://github.com/openvinotoolkit/openvino_notebooks.git
$ cd openvino_notebooks

# Step 3: Install requirements
$ python -m pip install --upgrade pip
$ pip install -r requirements.txt

# Step 4: Launch the Notebooks
$ jupyter lab
```



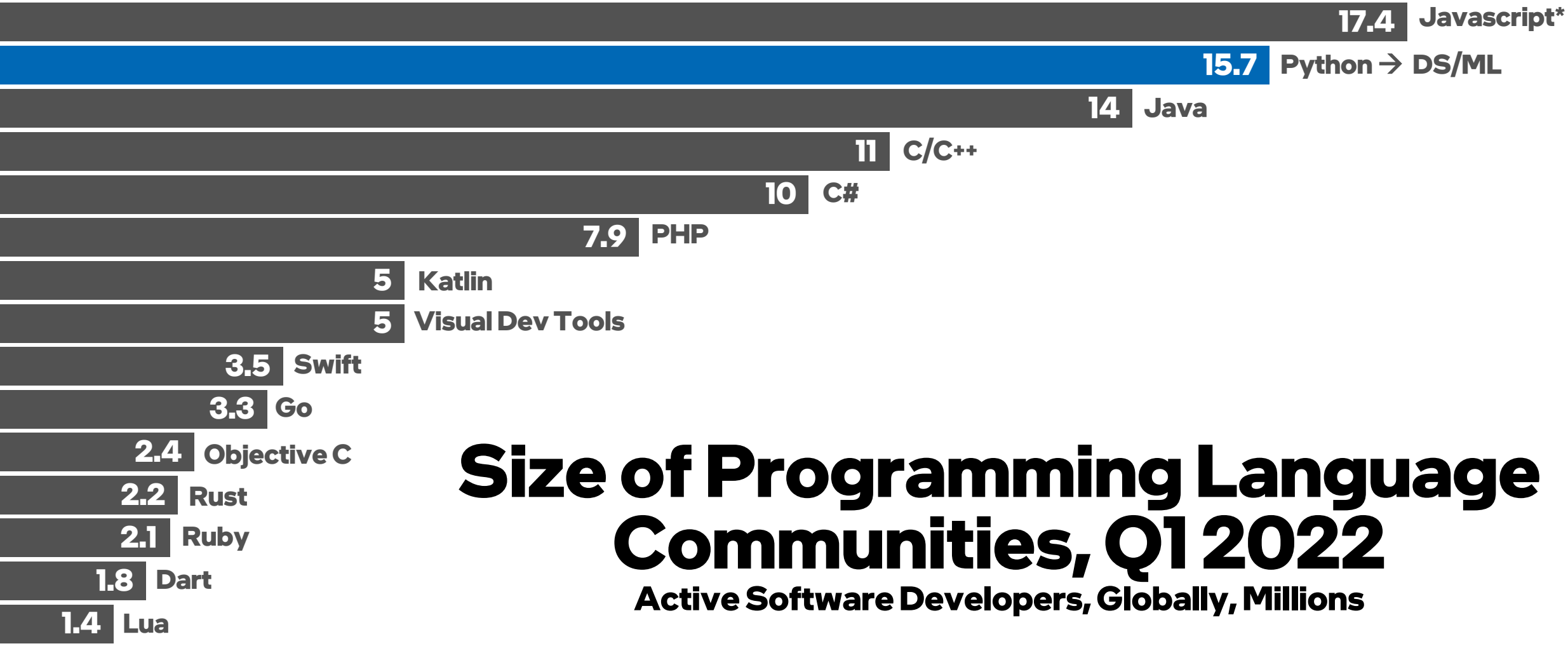

```
For installing just OPENVINO-DEV
$ pip install openvino-dev
```

How many developers are there in the world?


Chrome File Edit View History Bookmarks Profiles Tab Window Help
Wed May 4 2:57 PM
Google
google.com
Incognito
About Store
Gmail Images
Sign in
Google
Google Search
I'm Feeling Lucky
Discover APAHM: celebrate Asian Pacific American cultures and experiences
Carbon neutral since 2007
Advertising Business How Search works
Privacy Terms Settings

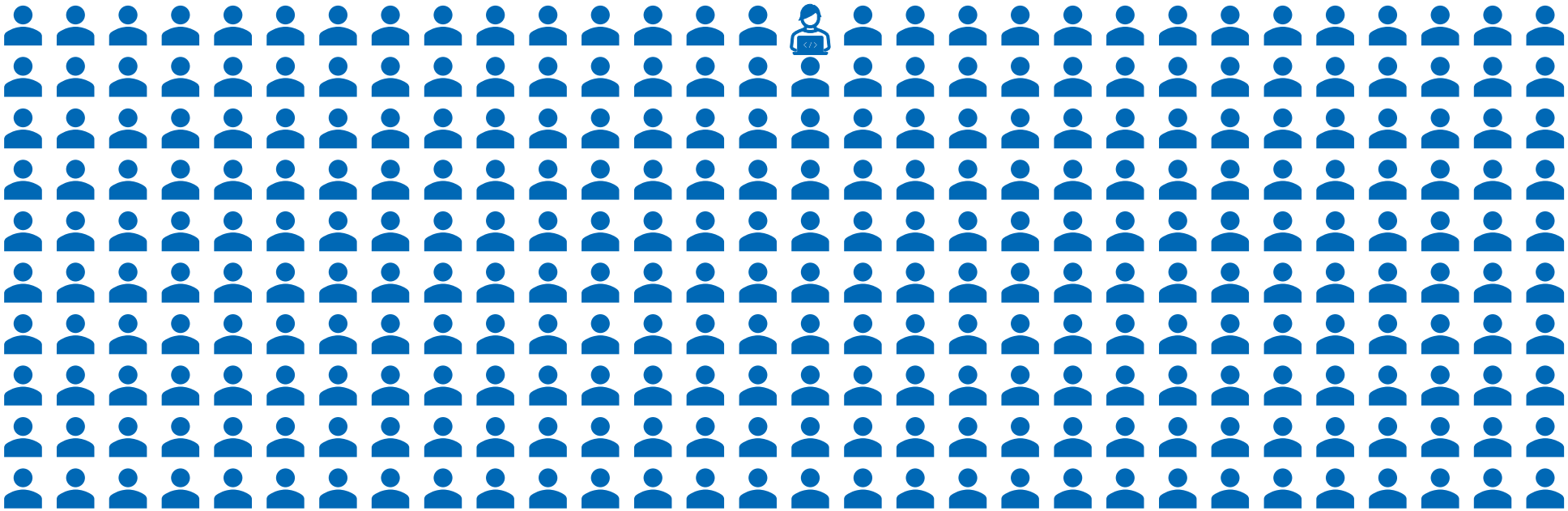


What are developers doing?

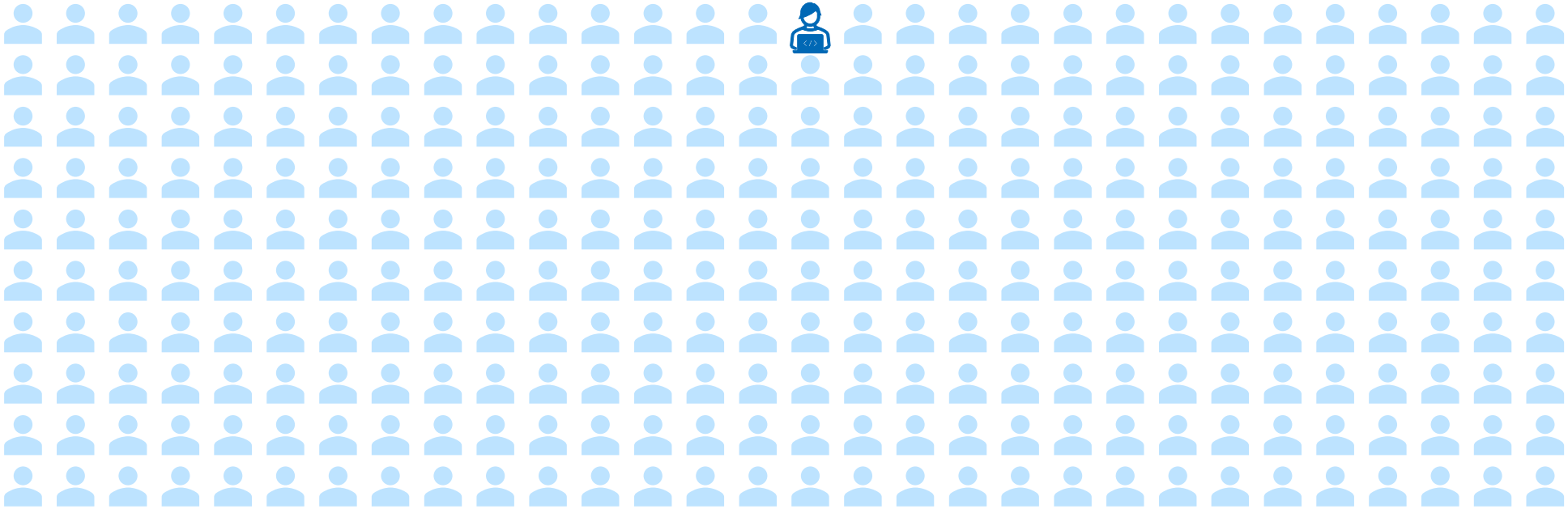


Size of Programming Language Communities, Q1 2022

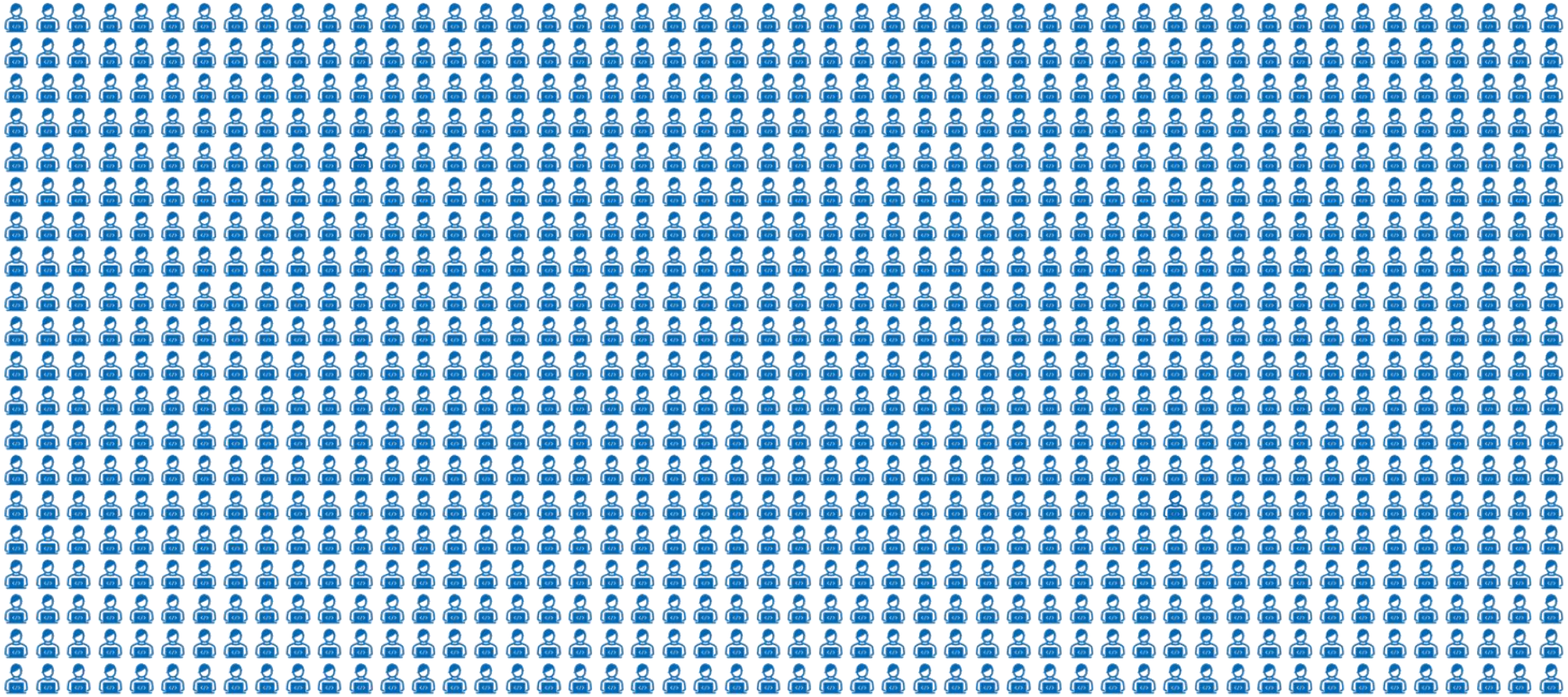
Active Software Developers, Globally, Millions



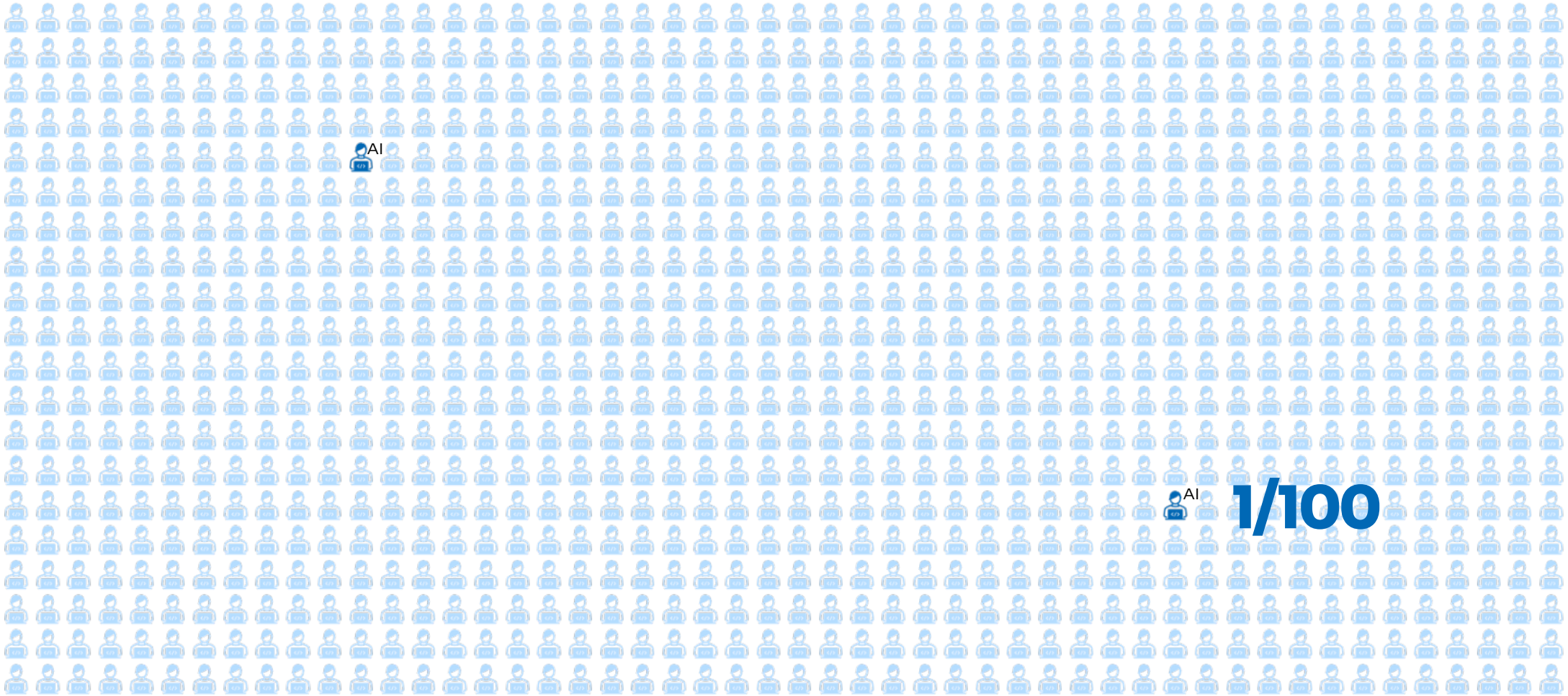
1/300
.35%



Values: https://www.future-processing.com/blog/how-many-developers-are-there-in-the-world-in-2019/



1%



https://www.stateofai2019.com/chapter-6-the-war-for-talent/


Make Technology Accessible
Through Democratization

What is OpenVINO™

OpenVINO™

Visual
+
NLP, Audio...

Inference

and
Neural
Network

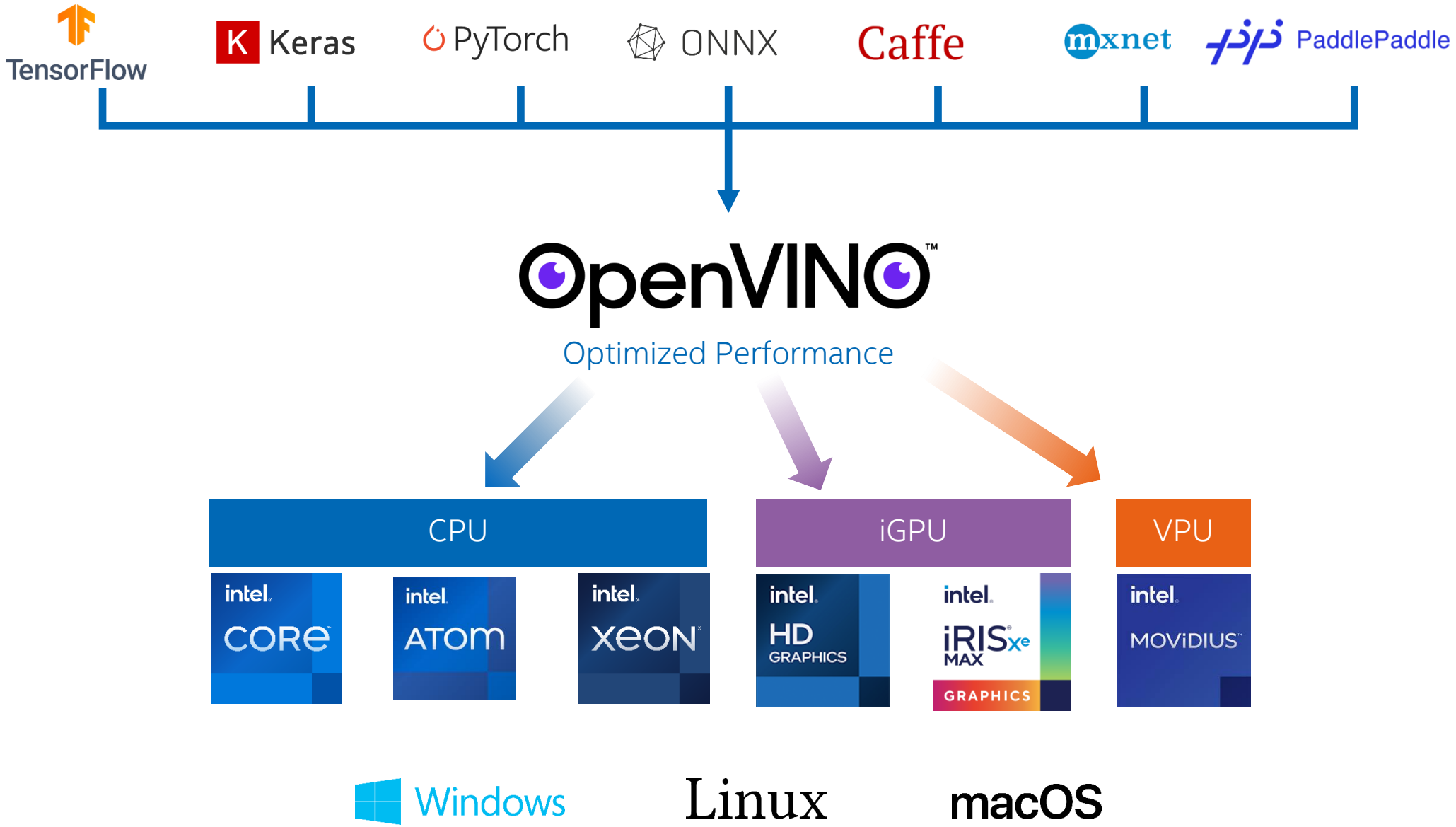
Optimization

Powered by oneAPI

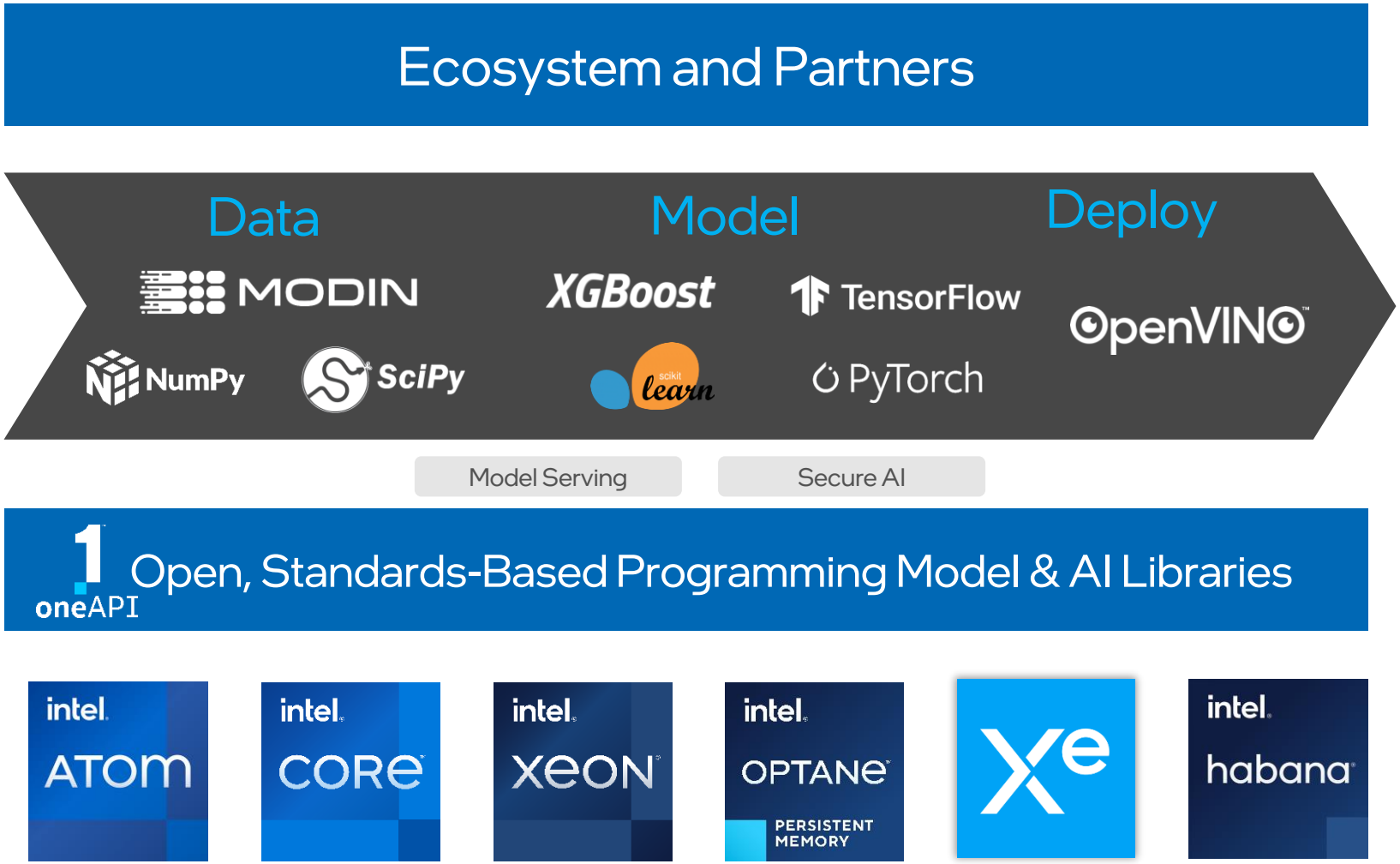
OpenVINO™

DEVELOPER JOURNEY

1 BUILD 2 OPTIMIZE 3 DEPLOY


TensorFlow
Keras
PyTorch
ONNX
Caffe
mxnet
PaddlePaddle
OpenVINO™
Optimized Performance
CPU
iGPU
VPU
intel CORE
intel ATOM
intel XEON
intel HD GRAPHICS
intel iRIS Xe MAX GRAPHICS
intel MOVIDIUS
Windows
Linux
macOS

Intel AI Software Portfolio



Ecosystem Adoption

OpenVINO

intel

Installation Methods

ubuntu

Red Hat

python Package Index

docker

ANACONDA

```
pip install openvino-dev
```

www.openvino.ai

OpenVINO Runtime

```python
from openvino.runtime import Core

img = load_img()

core = Core()

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

output_layer = compiled_model.outputs[0]

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

“dog”

Let's run the demo!

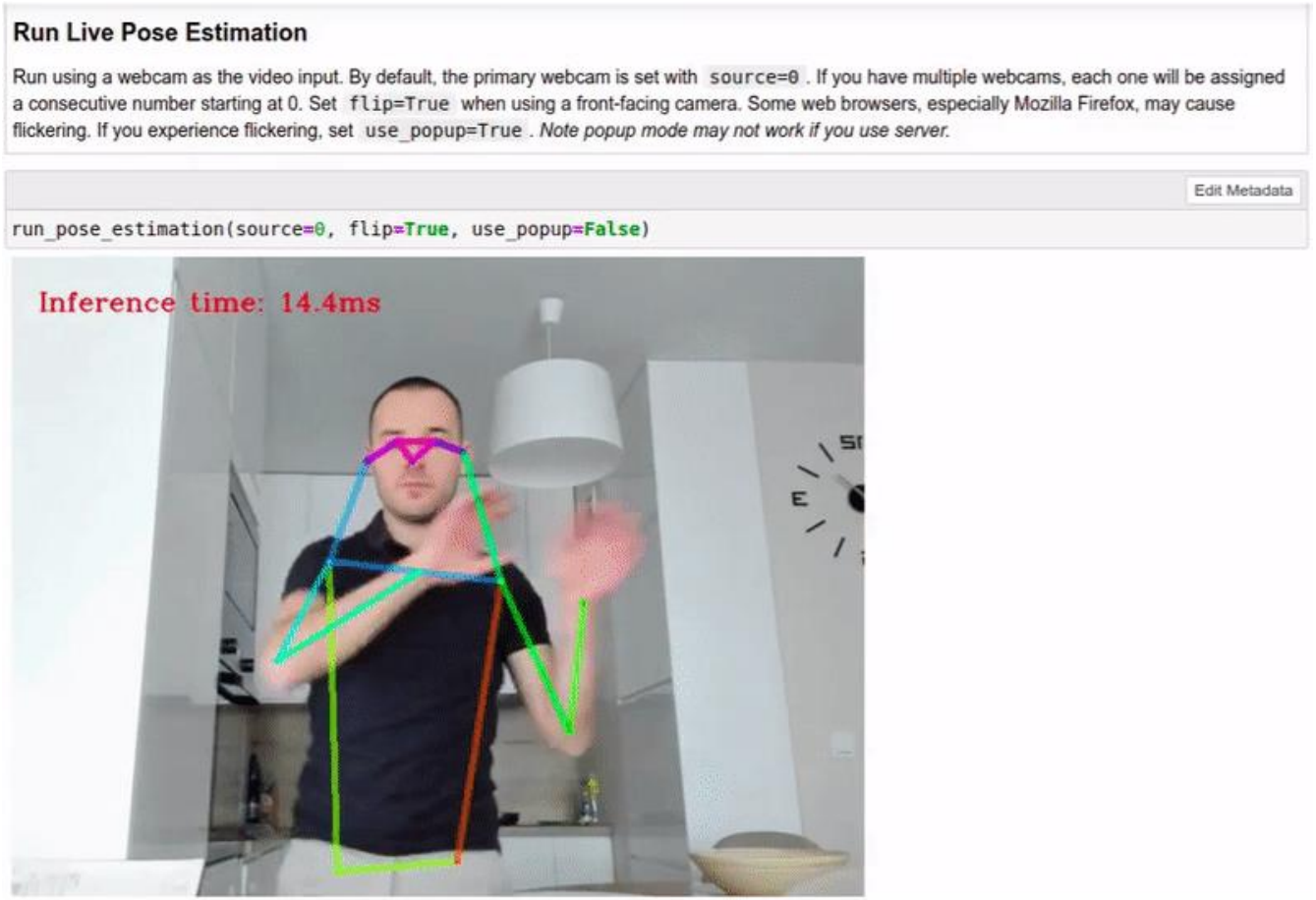
OpenVINO Notebooks

OpenVINO Toolkit offers several tools for making a model run faster and take less memory

Try it yourself

openvino.ai

Exercise 1: Basic with OpenVINO



Task # 1:

Run the human pose estimation notebook and tell us how many FPS you have and what is your CPU.

Share with us your screenshot and result using this Discussion thread, using Task # 1 and your name in the header message.

github.com/openvinotoolkit/openvino_notebooks/discussions/569

OpenVINO™

CVPR 2022

Break: 10 minutes

Optimization Tools

OpenVINO Toolkit offers several tools for making a model run faster and take less memory

OpenVINO™

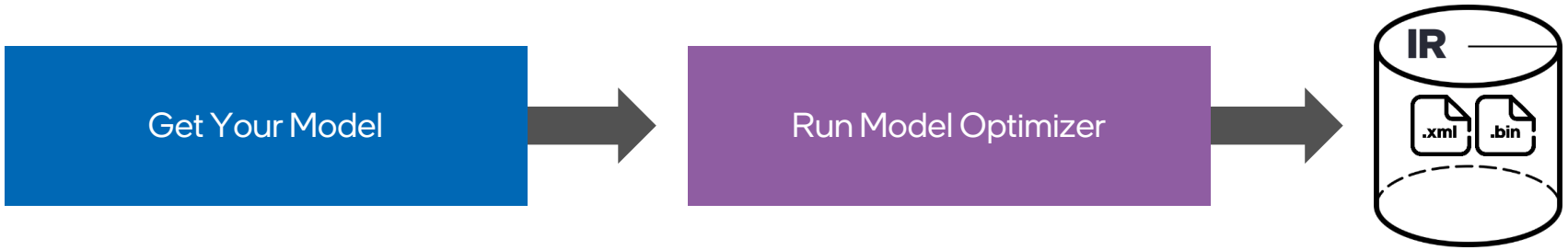
OPTIMIZATION TOOLS

MO
Model Optimizer

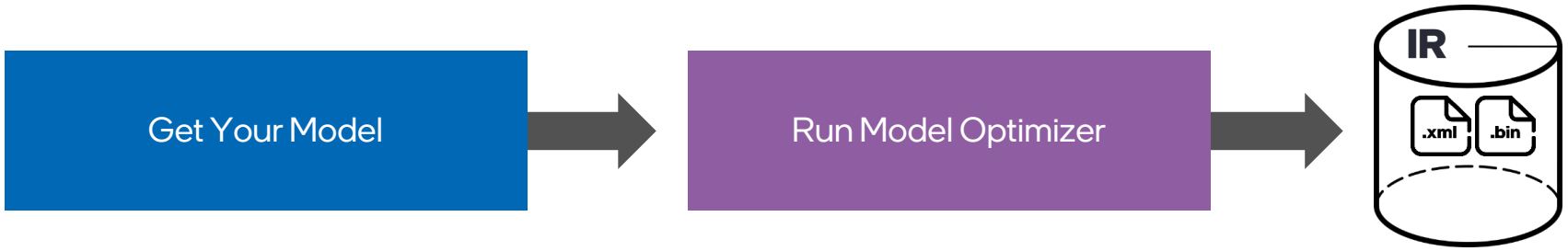
NNCF
Neural Network Compression Framework

POT
Post-Training Optimization Tool

Model Optimizer



Model Optimizer



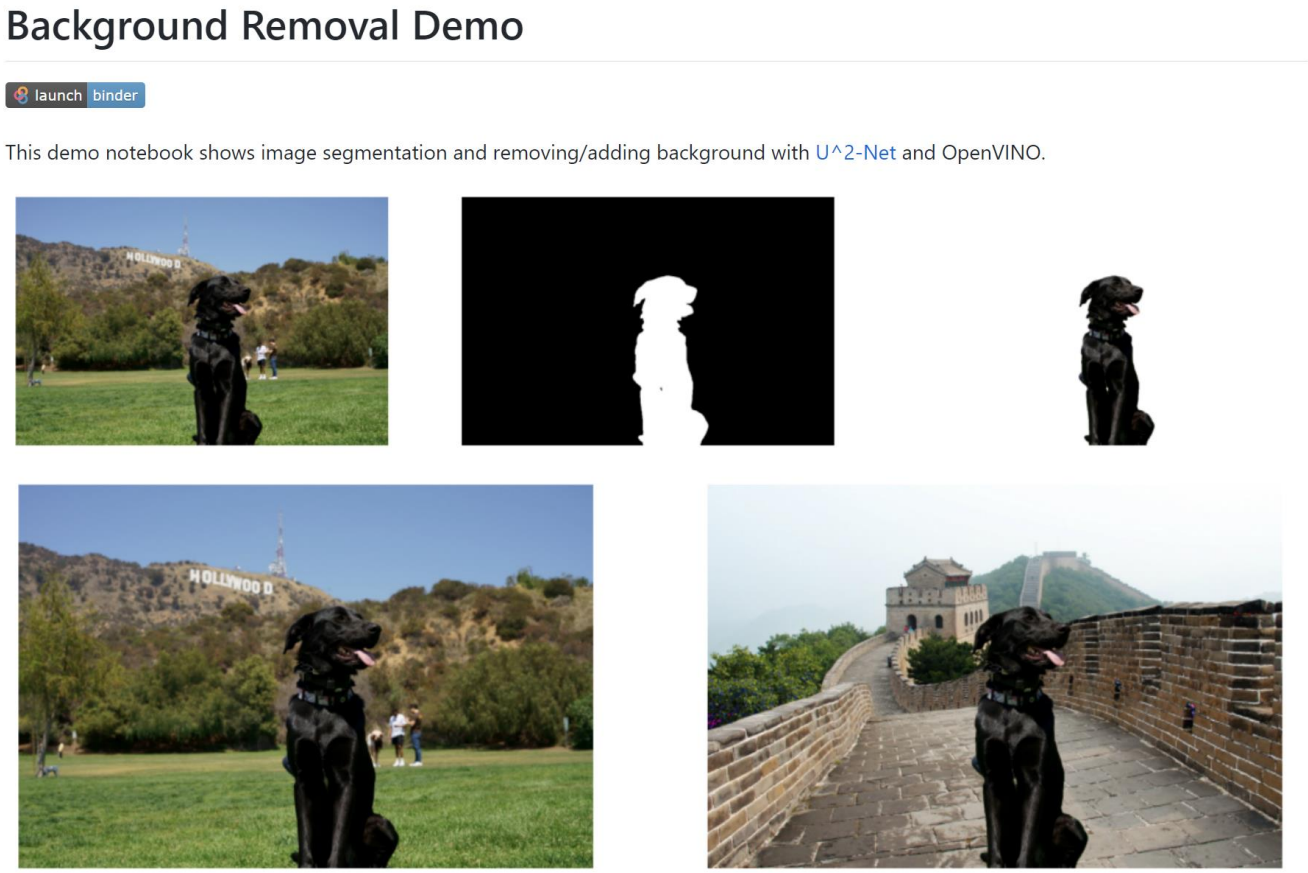

```
$ mo --input_model model.onnx
     --data_type FP32
```

Neural Network

(any format)

Intermediate Representation (IR)

Exercise 2:
Background Removal with OpenVINO IR



Task #2:

Run the background removal notebook and modify the model precision to FP32

Upload an image of your own and run the inference and compare the performance

Share with us your screenshot and result using this Discussion thread, using Task # 2 and your name in the header message.

github.com/openvinotoolkit/openvino_notebooks/discussions/569

OpenVINO™

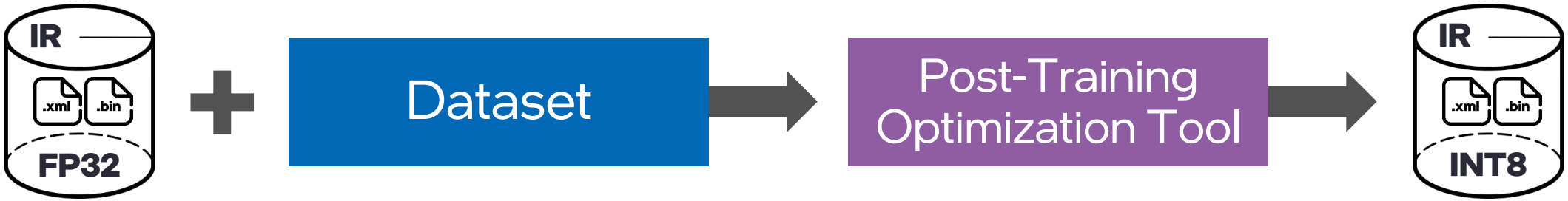
OPTIMIZATION TOOLS

MO
Model Optimizer

NNCF
Neural Network Compression Framework

POT
Post-Training Optimization Tool

Post-Training Optimization Tool



Default Quantization

Accuracy Aware Quantization

Performance/Mixed preset

Exercise 3:
POT Quantization with a Classification Model

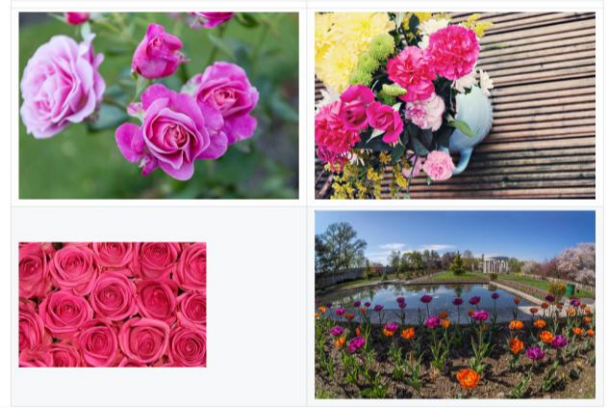
Post-Training Quantization with TensorFlow Classification Model

This example demonstrates how to quantize the OpenVINO model that was created in 301-tensorflow-training-openvino.ipynb, to improve inference speed. Quantization is performed with Post-Training Optimization Tool (POT). A custom dataloader and metric will be defined, and accuracy and performance will be computed for the original IR model and the quantized model.

Preparation

The notebook requires that the training notebook has been run and that the Intermediate Representation (IR) models are created. If the IR models do not exist, running the next cell will run the training notebook. This will take a while.

```
from pathlib import Path

import tensorflow as tf

model_xml = Path("model/flower/flower_ir.xml")
dataset_url = (
    "https://storage.googleapis.com/download.tensorflow.org/exam
)
data_dir = Path(tf.keras.utils.get_file("flower_photos", origin=

if not model_xml.exists():
    print("Executing training notebook. This will take a while..
    %run 301-tensorflow-training-openvino.ipynb
```



Task #3:

During the prework you ran the 301 notebooks. Run the 301-tensorflow-training-openvino-pot.ipynb.

Share with us your result using this Discussion thread, using Task # 3 and your name in the header message.

github.com/openvinotoolkit/openvino_notebooks/discussions/569

OpenVINO™

OPTIMIZATION TOOLS

MO
Model Optimizer

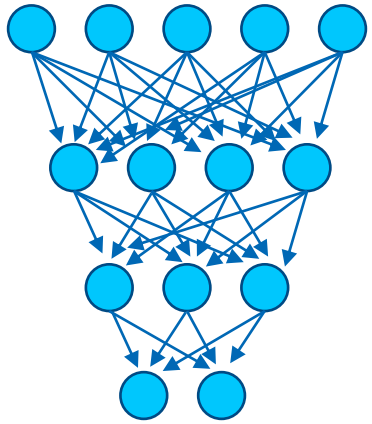
NNCF
Neural Network Compression Framework

POT
Post-Training Optimization Tool

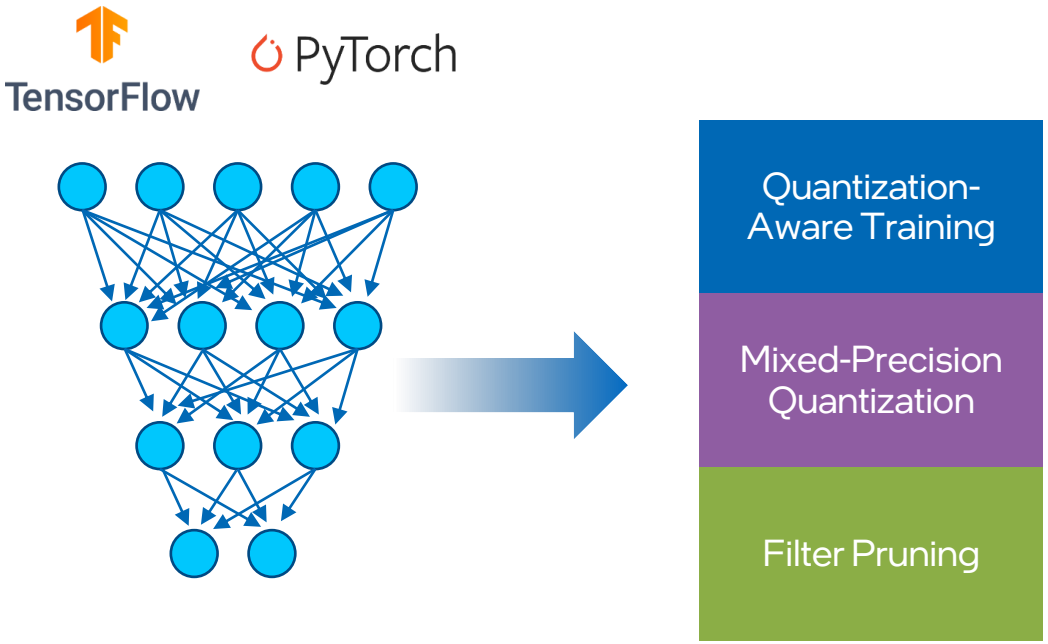
Neural Network Compression Framework

TensorFlow

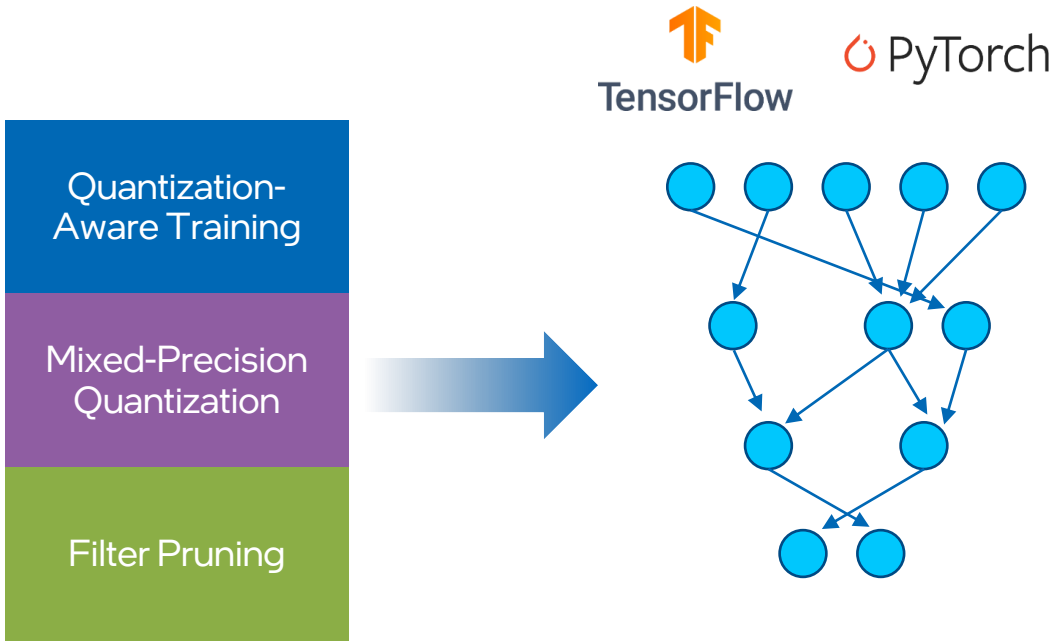
PyTorch



Neural Network Compression Framework (NNCF)



Neural Network Compression Framework (NNCF)



Exercise 4:

Quantize a Segmentation Model and Show Live Inference

Task #4:

Run the 110-ct-segmentation-quantize-nncf.ipynb, in the Quantization Config section Change the *preset* option on nncf_config to "mixed".

Describe if the performance is better or not for the new configuration, just using CPU.

Share with us your result using this Discussion thread, using Task # 4 and your name in the header message.

github.com/openvinotoolkit/openvino_notebooks/discussions/569

What About the Hardware?

Could we improve the performance of the AI models using our own laptop or edge device?

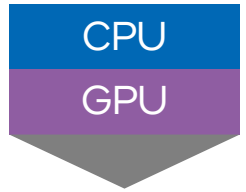
Supported Devices

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

Supported Devices

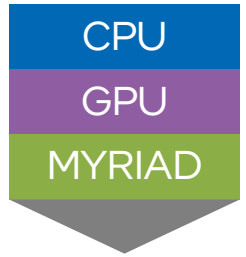

```
compiled_model = core.compile_model(model=model, device_name="CPU")
```

Supported Devices



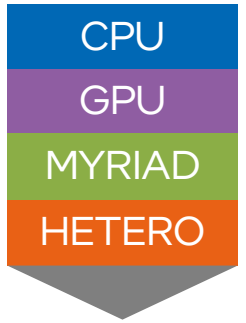

```
compiled_model = core.compile_model(model=model, device_name="CPU")
```

Supported Devices

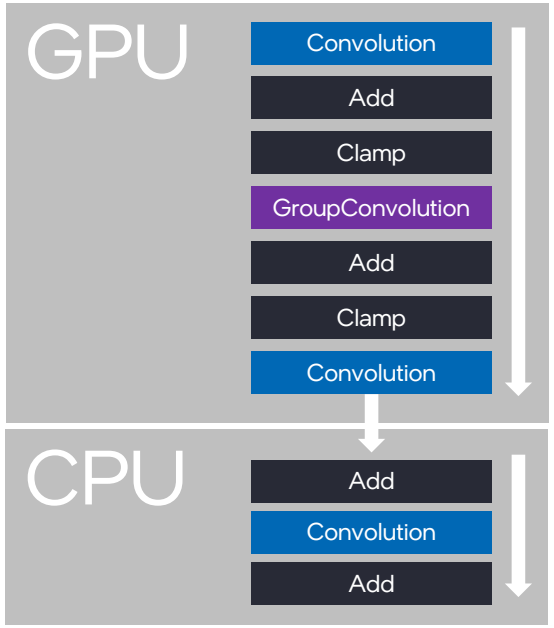



```
compiled_model = core.compile_model(model=model, device_name="CPU")
```

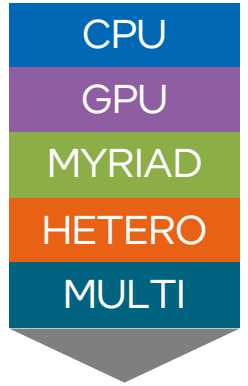
Supported Devices



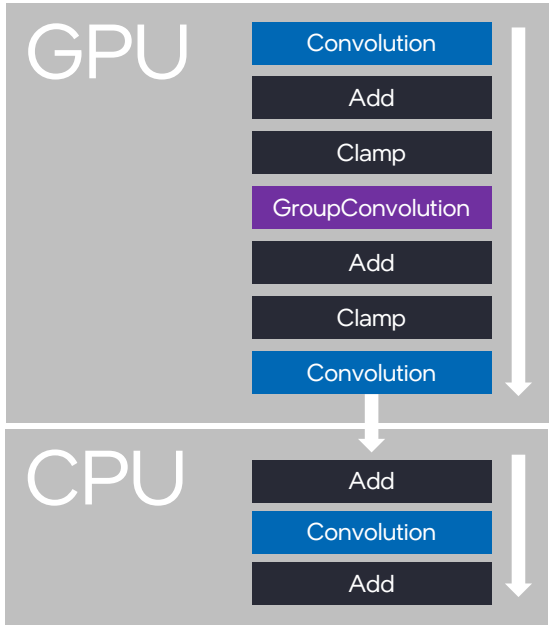

```
compiled_model = core.compile_model(model=model, device_name="CPU")
```

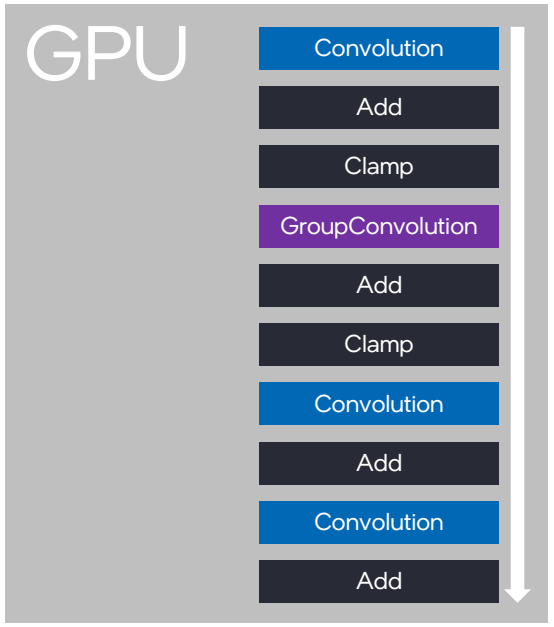


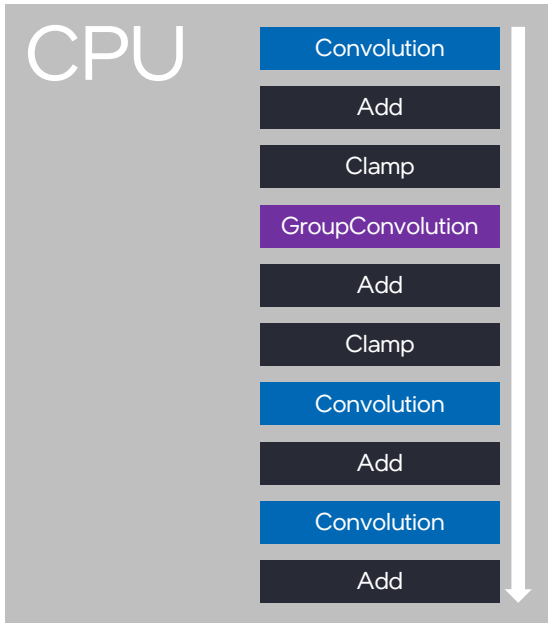
Supported Devices




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

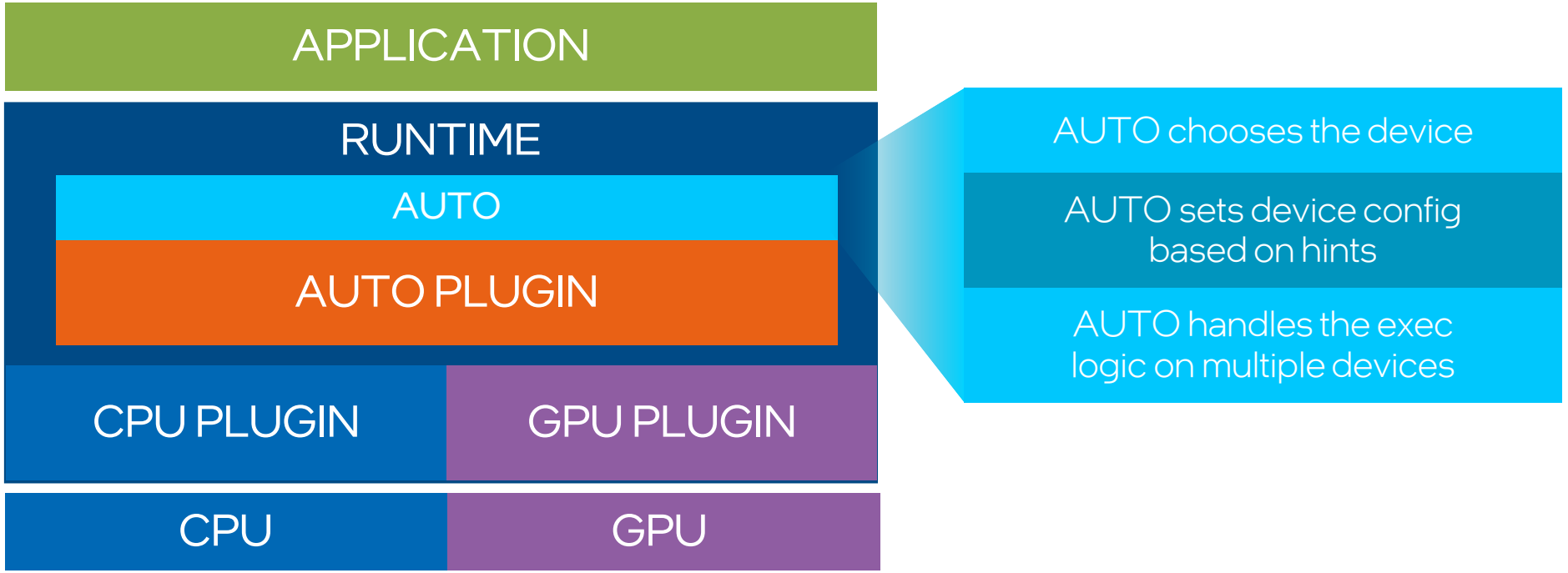






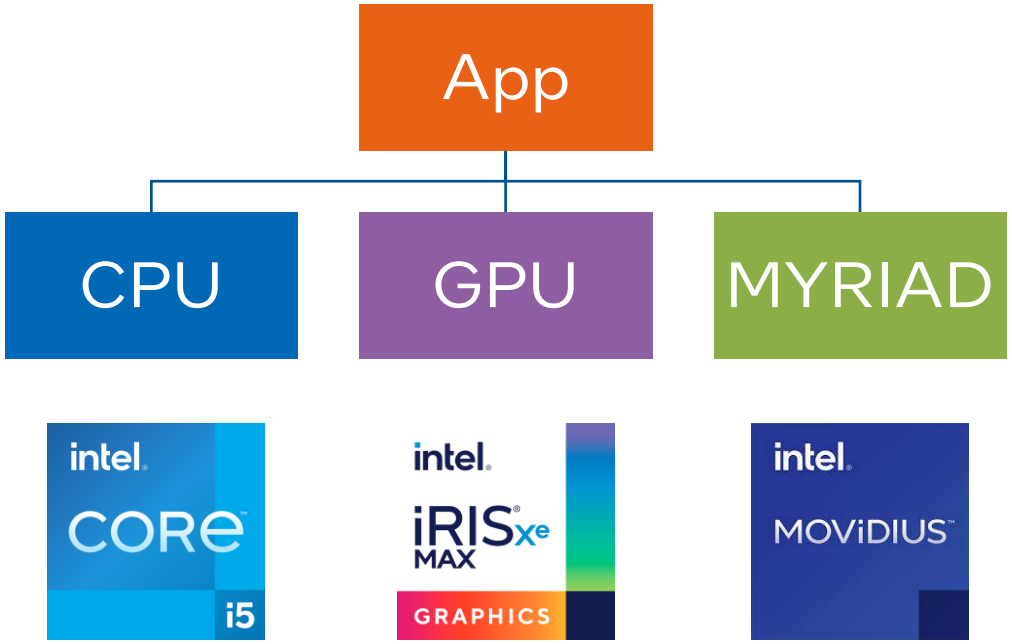
AUTO Device


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



Development scenario

Inference on multiple devices




```
if "GPU" in available_devices:
    ...
elif "MYRIAD" in available_devices:
    ...
else:
    ...
```

Alice: "How do you simplify a unified app for different platforms?"

Development scenario

Inference performance preference

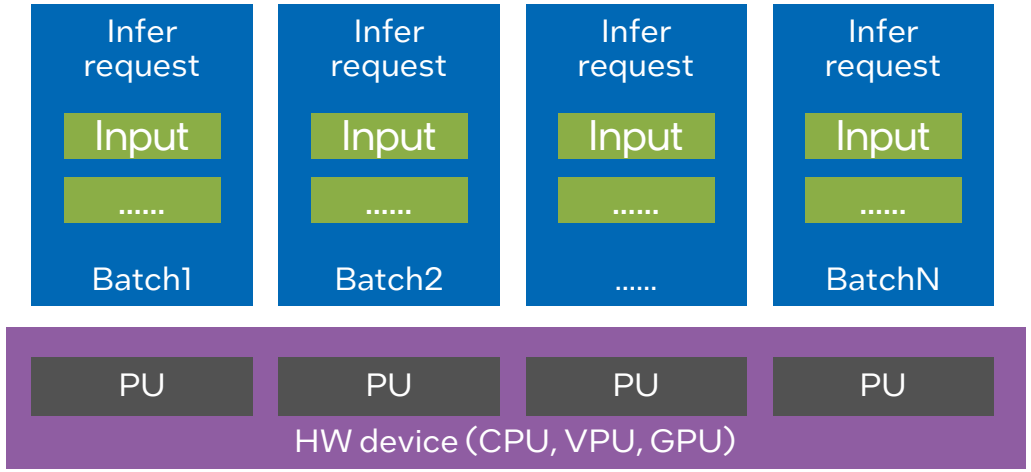
Throughput

Latency

Cathy and Dave: "How do you reach app performance targets without learning and tuning device configurations?"

Performance hints

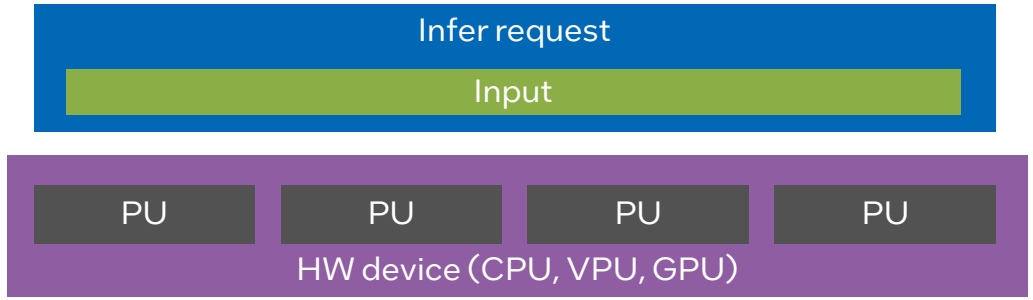
Throughput



Multiple streams
Infer concurrently
Auto batching

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

Latency



Single stream
Infer one by one

```
core.compile_model(model=model, device_name="AUTO",
        config={"PERFORMANCE_HINT":"LATENCY"})
```

PU = Processing unit

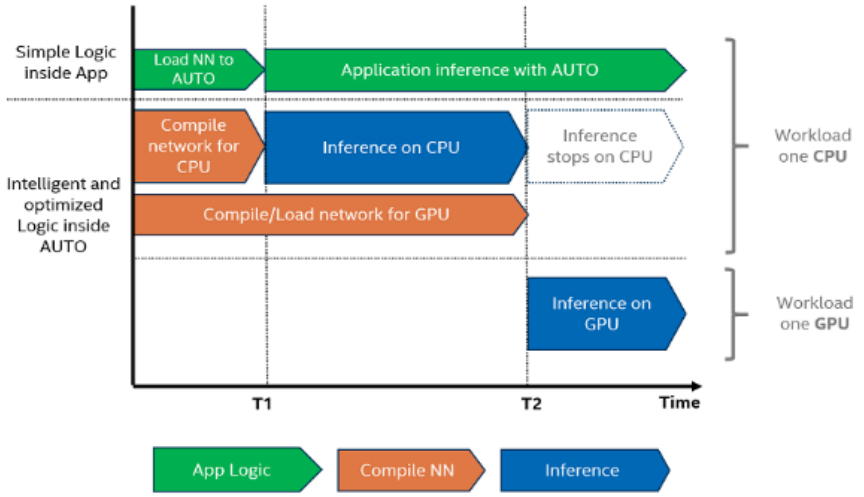
Exercise 5:
Auto Device Plugin with OpenVINO

Automatic Device Selection with OpenVINO™

The Auto device (or AUTO in short) selects the most suitable device for inference by considering the model precision, power efficiency and processing capability of the available compute devices. The model precision (i.e. FP32, FP16, INT8, etc.) is the first consideration to filter out the devices that cannot run the network efficiently.

Next, if dedicated accelerators are available, these devices are preferred (e.g. integrated and discrete GPU or VPU). CPU is used as the default "fallback device". Please note that AUTO makes this selection only once at the model load time.

When using accelerator device like GPUs, loading models to these devices may take a long time. To address this challenge for applications that require fast first inference response, AUTO starts inferencing immediately on the CPU and then transparently shifts inferencing to the GPU once it is ready. This dramatically reduces the time to first inference.



Task # 5:

Let's share your findings about latency vs throughput here!

Describe your findings and share with us your result using this Discussion thread, using Task # 5 and your name in the header message.

github.com/openvinotoolkit/openvino_notebooks/discussions/569

If You Are Interested in Using OpenVINO on Your Deployments

Take a look of the Open Model Zoo

Open Model Zoo

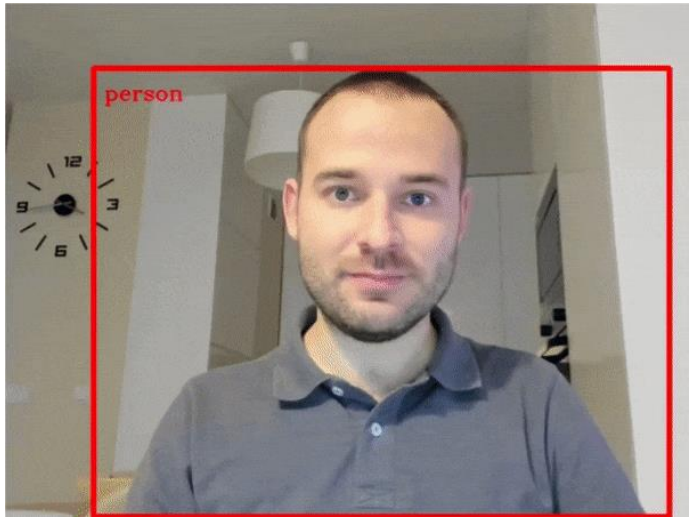
270+

Pre-Trained + Optimized Models

Open Model Zoo

270+

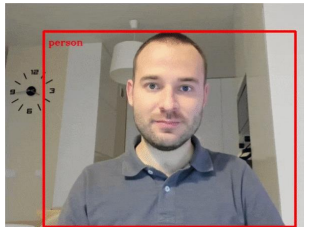
Pre-Trained + Optimized Models

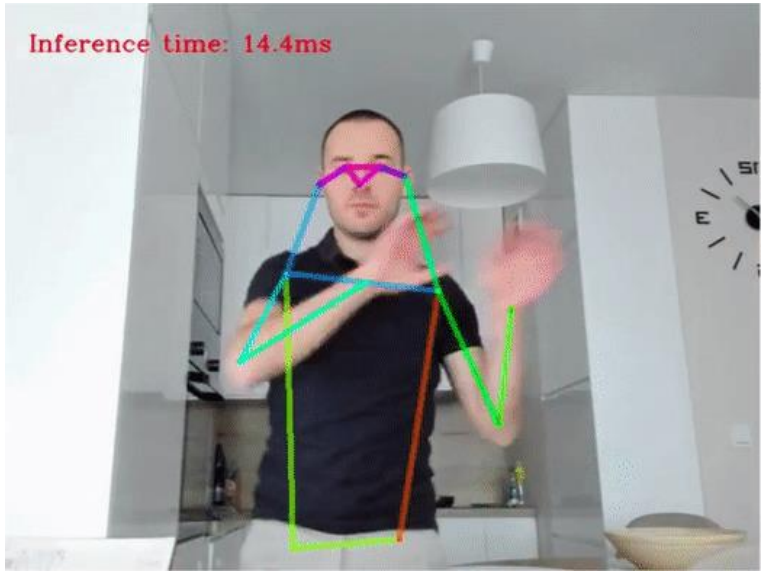


Open Model Zoo

270+

Pre-Trained + Optimized Models

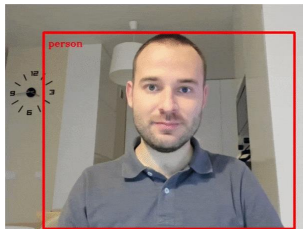


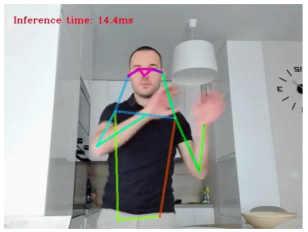


Open Model Zoo

270+

Pre-Trained + Optimized Models

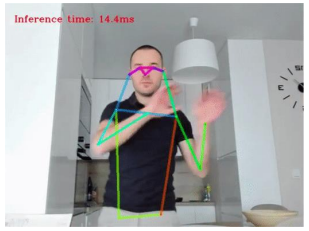


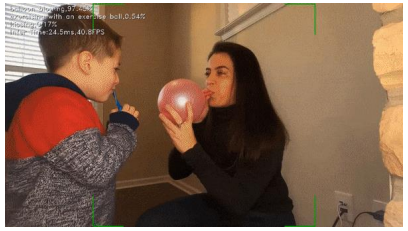


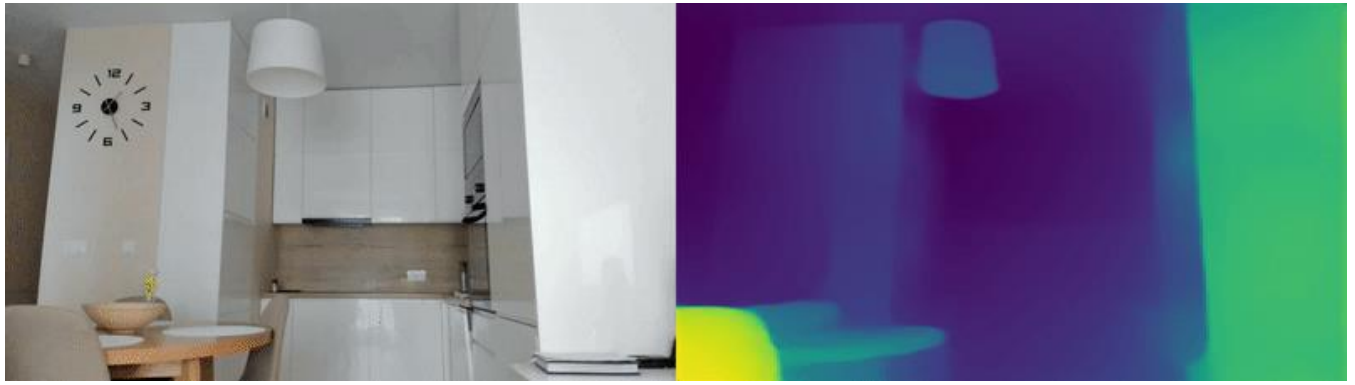
Open Model Zoo

270+

Pre-Trained + Optimized Models



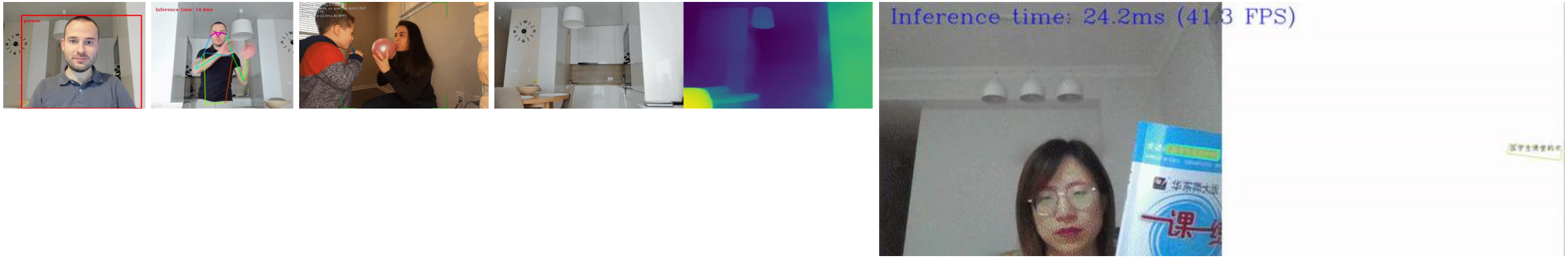




Open Model Zoo

270+

Pre-Trained + Optimized Models



Open Model Zoo

270+

Pre-Trained + Optimized Models

OpenVINO™

CVPR 2022

Break: 10 minutes

What is OTE?

OpenVINO Training Extensions

OpenVINO Training Extensions

Semantic Segmentation

Rotated Object Detection

Object Counting

Object Detection

Image Classification

Anomaly classification/detection/ segmentation

Computer Vision Framework

Installation Process

Ubuntu 18.04 or 20.04

Python 3.8 or higher

CUDA Toolkit 11.1 - for training on CUDA-compatible GPU

OTE - Components

OTE
ALGORITHMS
CV Task

OTE - Components

OTE ALGORITHMS
CV Task

OTE CLI
Operate deep learning models

OTE - Components

OTE ALGORITHMS
CV Task

OTE CLI
Operate deep learning models

OTE SDK
Interface

OTE - Components

OTE ALGORITHMS
CV Task

OTE CLI
Operate deep learning models

OTE SDK
Interface

DEPLOYMENT
Demo + Model

OpenVINO Training Extensions

Models pre-trained with larger datasets

Simple development interface

Optimization with NNCF or POT

OpenVINO Training Extensions

Models pre-trained with larger datasets

Simple development interface

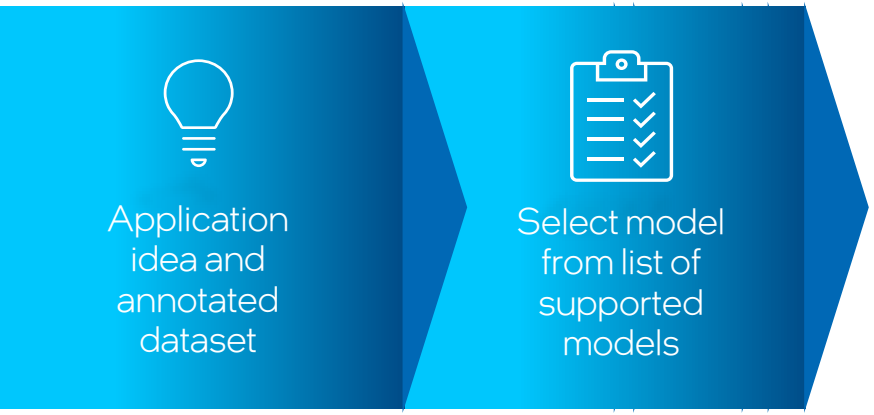
Optimization with NNCF or POT

OpenVINO Training Extensions

Models pre-trained with larger datasets

Simple development interface

Optimization with NNCF or POT

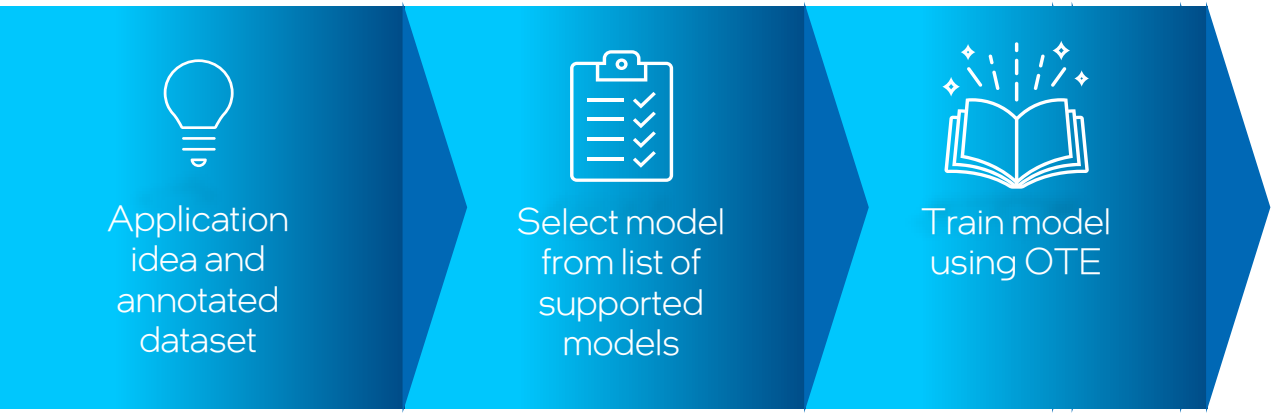


OpenVINO Training Extensions

Models pre-trained with larger datasets

Simple development interface

Optimization with NNCF or POT

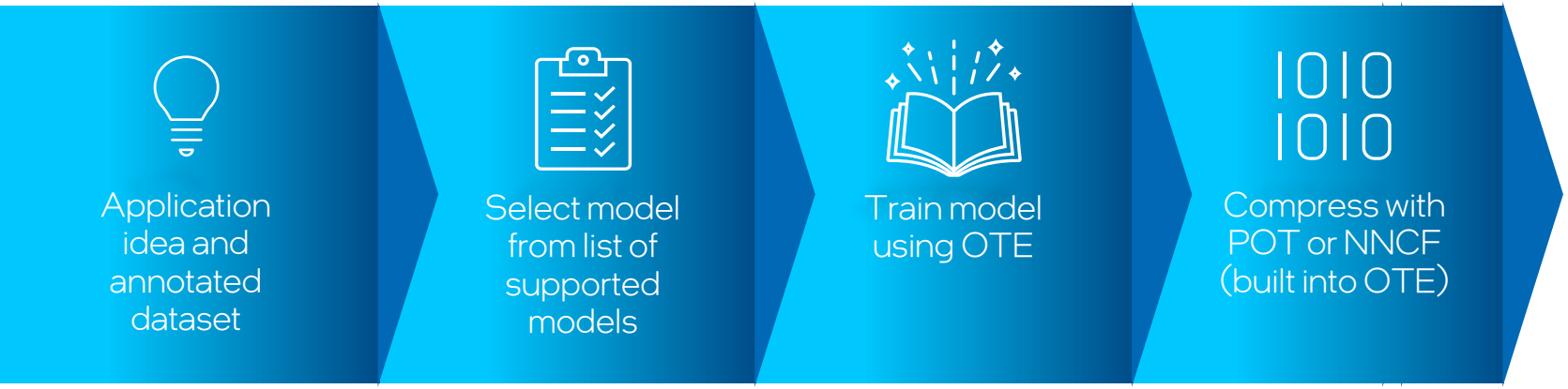


OpenVINO Training Extensions

Models pre-trained with larger datasets

Simple development interface

Optimization with NNCF or POT



OpenVINO Training Extensions

Models pre-trained with larger datasets

Simple development interface

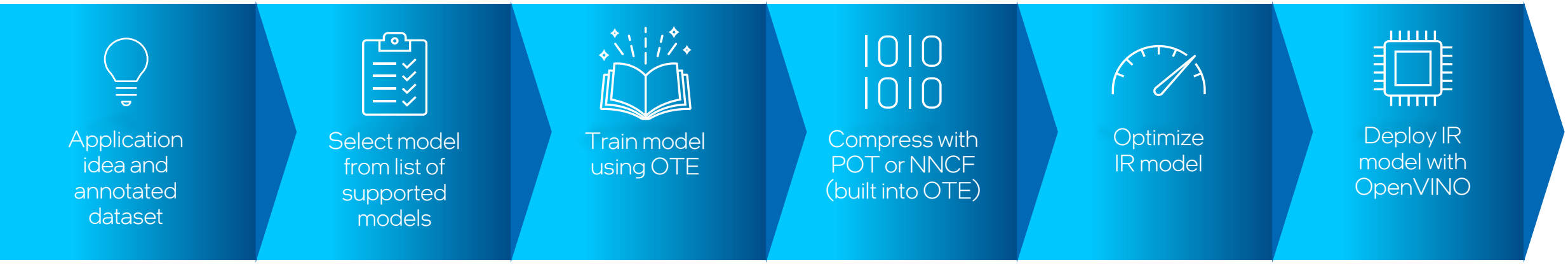
Optimization with NNCF or POT

OpenVINO Training Extensions

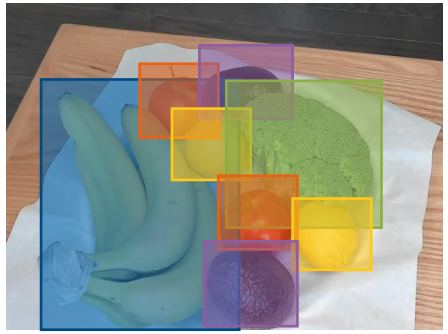
Models pre-trained with larger datasets

Simple development interface

Optimization with NNCF or POT



Exercise 6:
Object Detection with OTE



jupyter train Last Checkpoint: 05/27/2022 (autosaved) Logout

File Edit View Insert Cell Kernel Help Not Trusted Python 3 (ipykernel)

Run Markdown

Object Detection

Import everything

In [1]:
```
import os
import sys

import cv2
import numpy as np

from ote_sdk.configuration.helper import create as create_parameters_from_parameters_schema
from ote_sdk.entities.inference_parameters import InferenceParameters
from ote_sdk.entities.label_schema import LabelSchemaEntity
from ote_sdk.entities.model import ModelEntity
from ote_sdk.entities.resultset import ResultSetEntity
from ote_sdk.entities.subset import Subset
from ote_sdk.entities.task_environment import TaskEnvironment
from ote_sdk.usecases.tasks.interfaces.export_interface import ExportType

from ote_cli.datasets import get_dataset_class
from ote_cli.registry import Registry
from ote_cli.utils.importing import get_impl_class
```

Register templates

In [2]:
```
templates_dir = '../../external'
registry = Registry(templates_dir)
print(registry)
```

```
- id: Custom_Semantic_Segmentation_Lite-HRNet-18-mod2_OCR
  name: Lite-HRNet-18-mod2 OCR
  path: /home/paula/ote/training_extensions/external/mmsegmentation/configs/custom-sematic-segmentation/ocr-lite
-hrnet-18-mod2/template.yaml
  task_type: SEGMENTATION
- id: Custom_Semantic_Segmentation_Lite-HRNet-18_OCR
  name: Lite-HRNet-18 OCR
  path: /home/paula/ote/training_extensions/external/mmsegmentation/configs/custom-sematic-segmentation/ocr-lite
-hrnet-18/template.yaml
  task_type: SEGMENTATION
- id: Custom_Semantic_Segmentation_Lite-HRNet-x-mod3_OCR
  name: Lite-HRNet-x-mod3 OCR
  path: /home/paula/ote/training_extensions/external/mmsegmentation/configs/custom-sematic-segmentation/ocr-lite
-hrnet-x-mod3/template.yaml
  task_type: SEGMENTATION
- id: Custom_Semantic_Segmentation_Lite-HRNet-s-mod2_OCR
  name: Lite-HRNet-s-mod2 OCR
  path: /home/paula/ote/training_extensions/external/mmsegmentation/configs/custom-sematic-segmentation/ocr-lite
-hrnet-s-mod2/template.yaml
```

Load model template and its hyper parameters

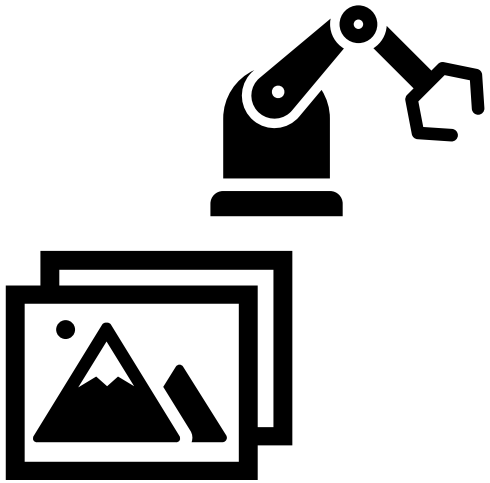
In [3]:
```
model_template = registry.get('Custom_Object_Detection_Gen3_SSD')
hyper_parameters = model_template.hyper_parameters.data
```

Get dataset instantiated

In [4]:
```
Dataset = get_dataset_class(model_template.task_type)

dataset = Dataset(
    train_subset={'ann_file': '../../data/fruits/annotations/instances_NONE.json',
                  'data_root': '../../data/fruits/images/NONE'},
    val_subset={'ann_file': '../../data/fruits/annotations/instances_NONE.json',
                'data_root': '../../data/fruits/images/NONE'}
```

What is Anomalib?

Anomalib is a library for unsupervised anomaly detection from data collection to deployment.



No-defect

Defect

Anomalib

Factory data is unbalanced, lots of good, not enough defect

Anomalib

README.md

ANOMALiB

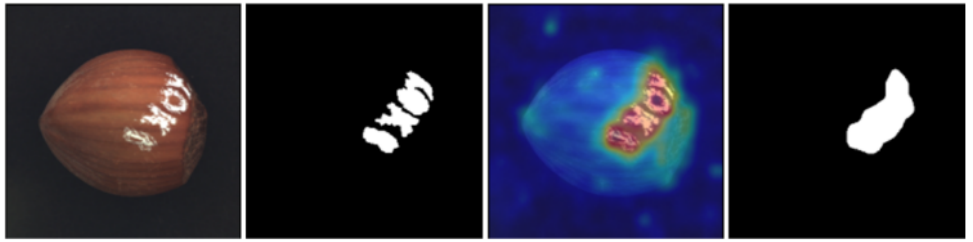
A library for benchmarking, developing and deploying deep learning anomaly detection algorithms

Key Features • Getting Started • Docs • License

python 3.7+ | pytorch 1.8.1+ | openvino 2021.4.2 | code style black | Nightly-regression Test failing | Pre-merge Checks failing | Build Docs passing

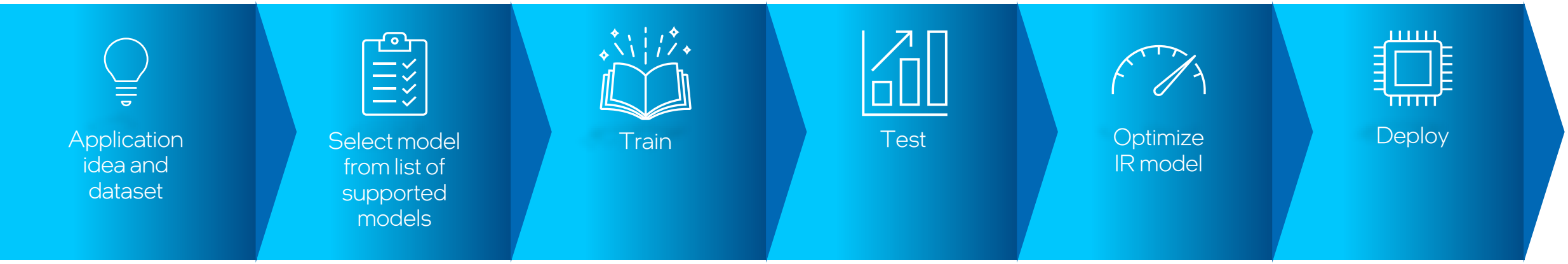
Introduction

Anomalib is a deep learning library that aims to collect state-of-the-art anomaly detection algorithms for benchmarking on both public and private datasets. Anomalib provides several ready-to-use implementations of anomaly detection algorithms described in the recent literature, as well as a set of tools that facilitate the development and implementation of custom models. The library has a strong focus on image-based anomaly detection, where the goal of the algorithm is to identify anomalous images, or anomalous pixel regions within images in a dataset. Anomalib is constantly updated with new algorithms and training/inference extensions, so keep checking!



Anomalib

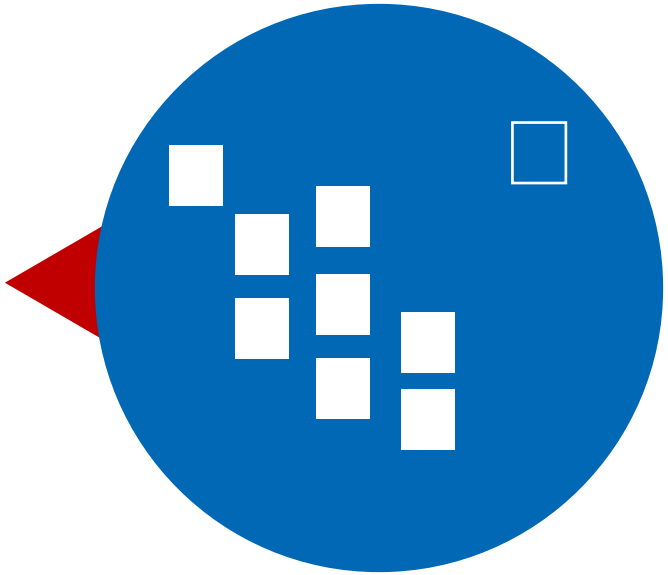
A unified library with a set of components and tools that could be used in anomaly detection research and production.



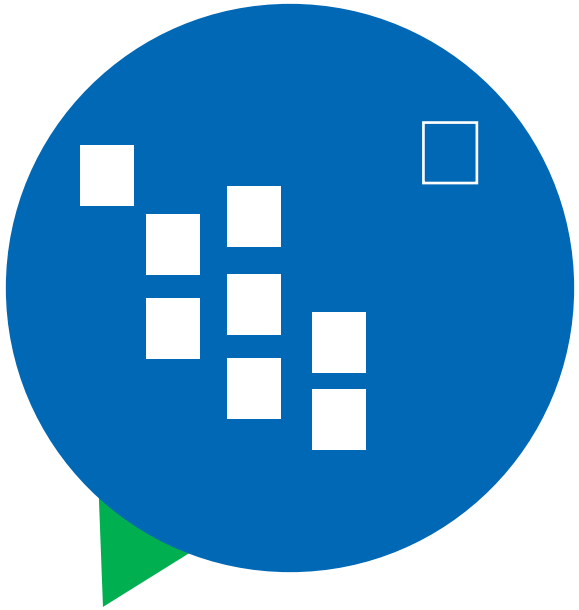
Anomalib
Components

ALGORITHMS

Ready to use state-of-the-art algorithms with no need for defective training data or labeling.



Anomalib
Components



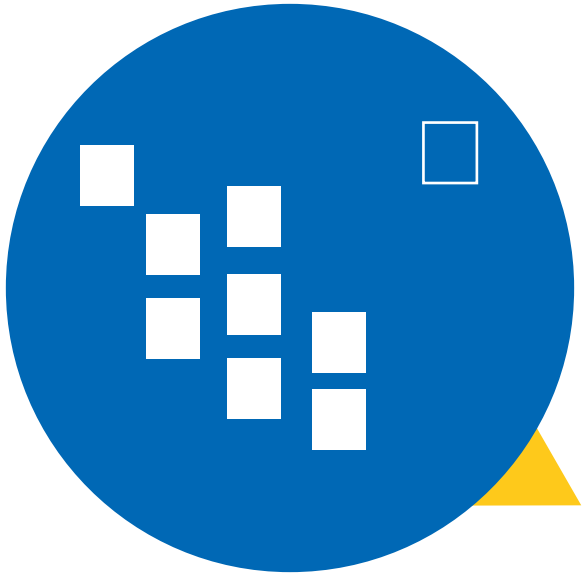
ALGORITHMS

Ready to use state-of-the-art algorithms with no need for defective training data or labeling.

MODULES

Number of modules to design custom algorithms for specific use-cases

Anomalib
Components

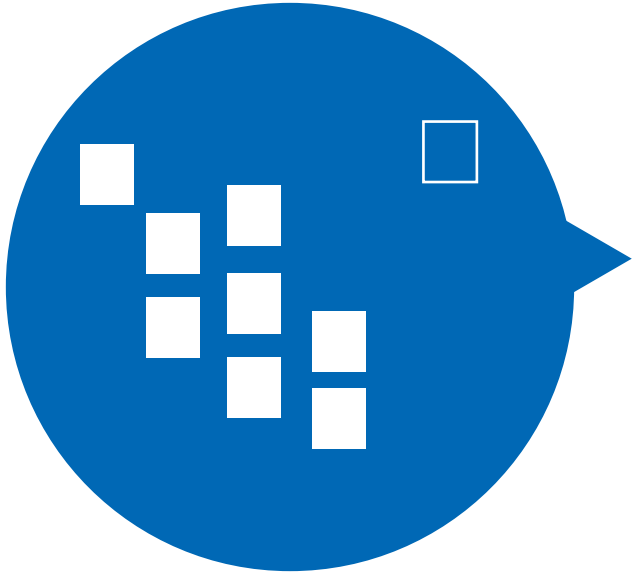


ALGORITHMS
Ready to use state-of-the-art algorithms with no need for defective training data or labeling.

MODULES
Number of modules to design custom algorithms for specific use-cases

TOOLS
CLI-based, configurable entry-points training, testing, inference, hyperparameter optimization Logging and benchmarking.

Anomalib
Components



ALGORITHMS
Ready to use state-of-the-art algorithms with no need for defective training data or labeling.

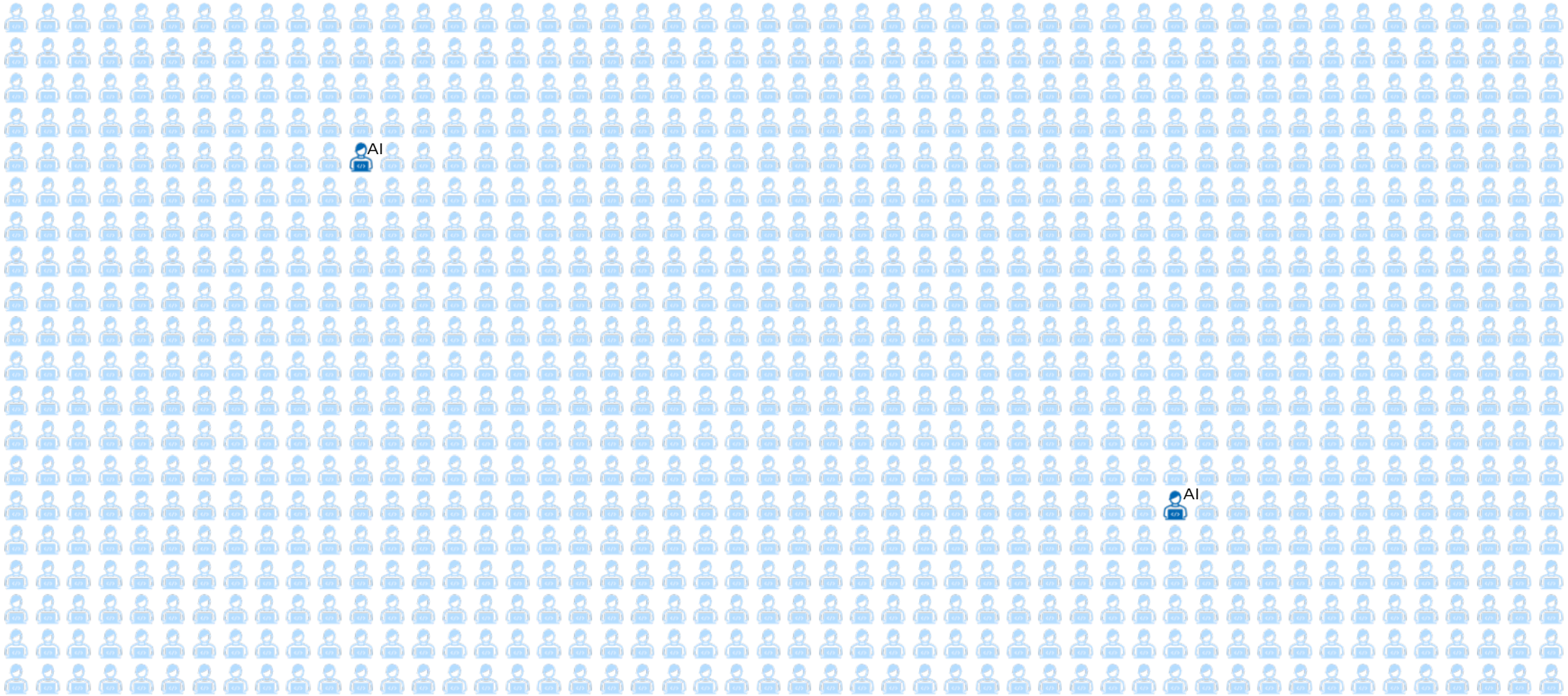
DEPLOYMENT
Supports real-time deployment by using OpenVINO model-optimization and quantization features

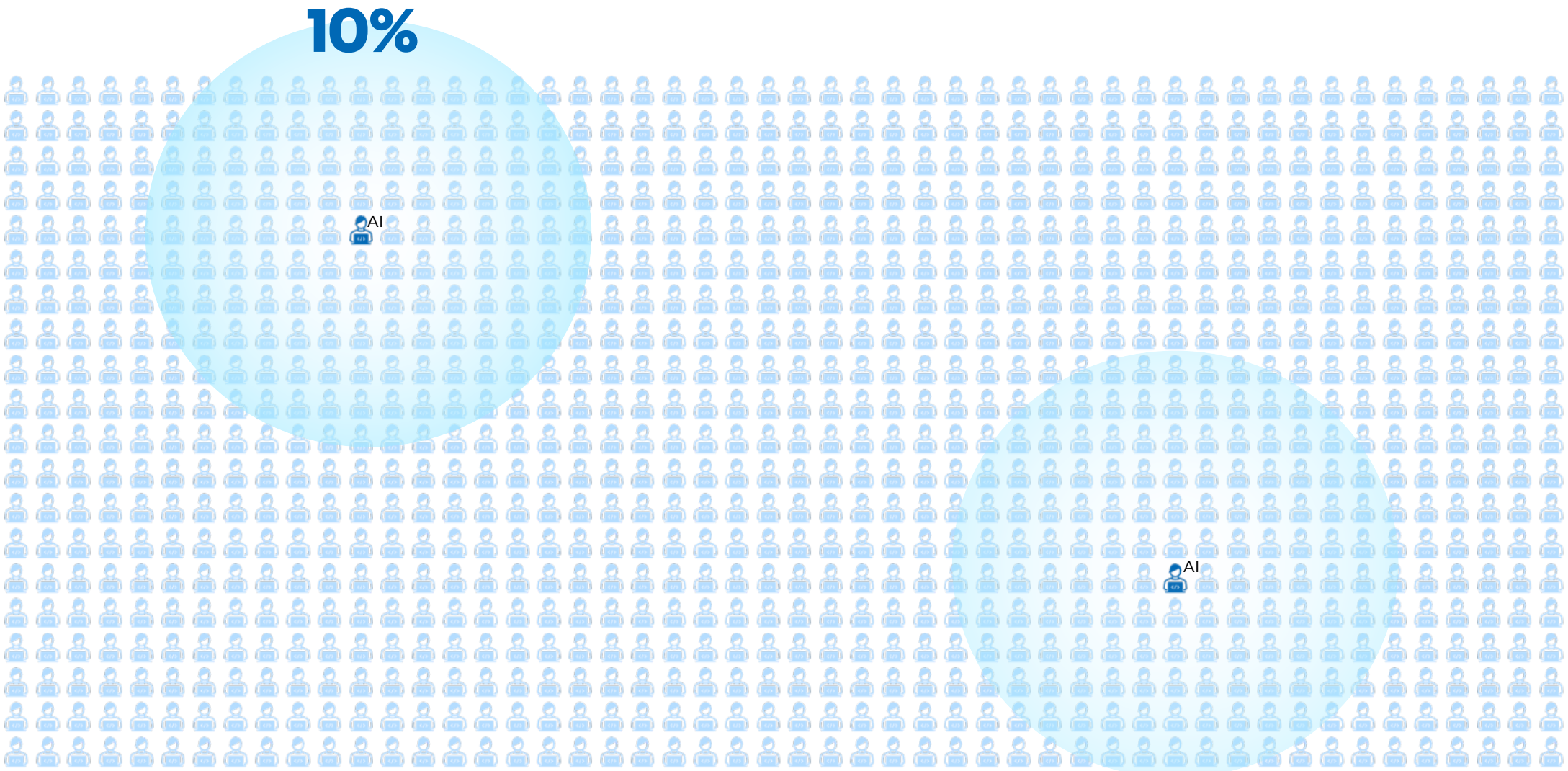
MODULES
Number of modules to design custom algorithms for specific use-cases

TOOLS
CLI-based, configurable entry-points training, testing, inference, hyperparameter optimization Logging and benchmarking.

Exercise 7: Use Anomalib E2E

1%

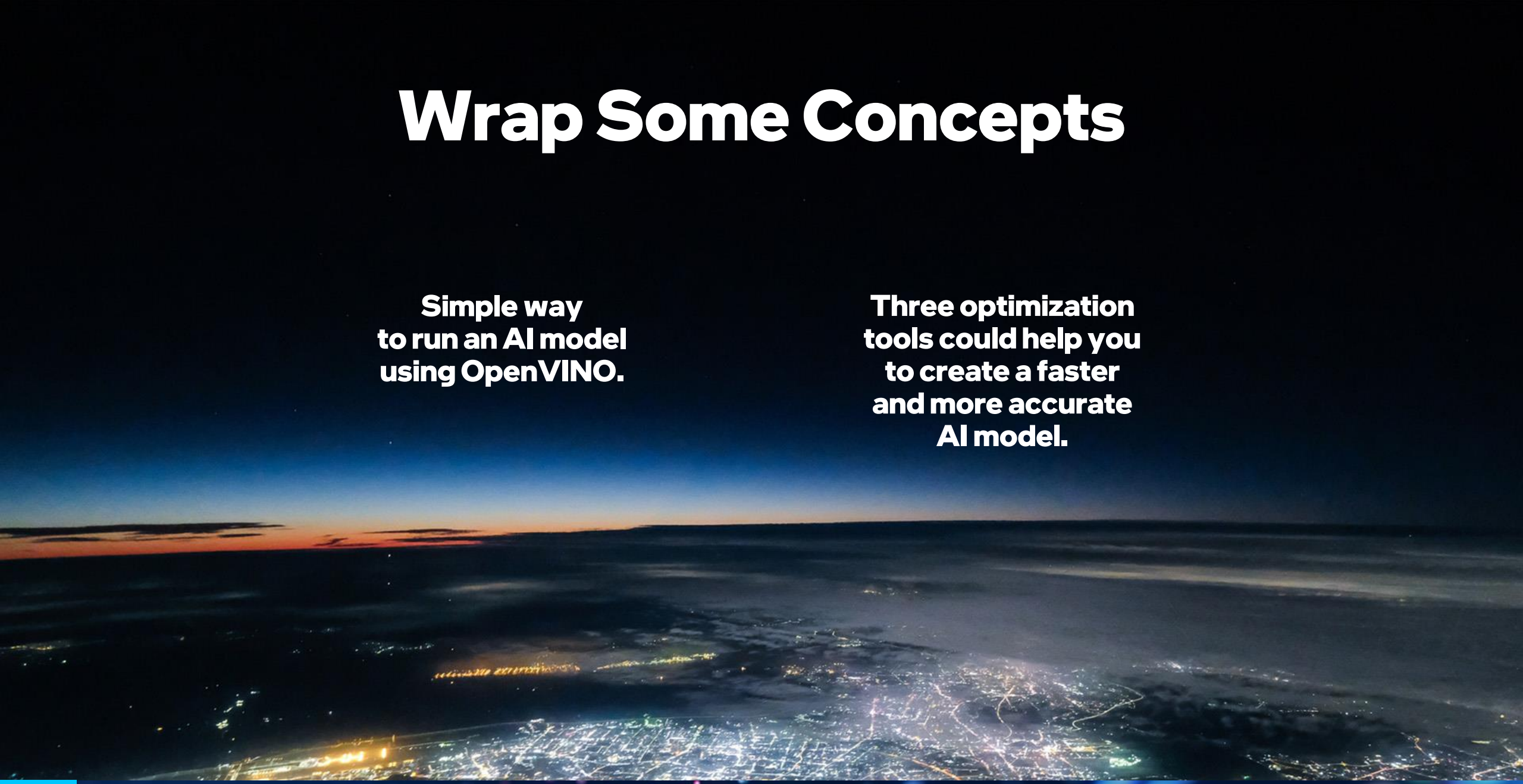



10%
AI
AI

Wrap Some Concepts

**Simple way
to run an AI model
using OpenVINO.**

Wrap Some Concepts



Simple way to run an AI model using OpenVINO.

Three optimization tools could help you to create a faster and more accurate AI model.

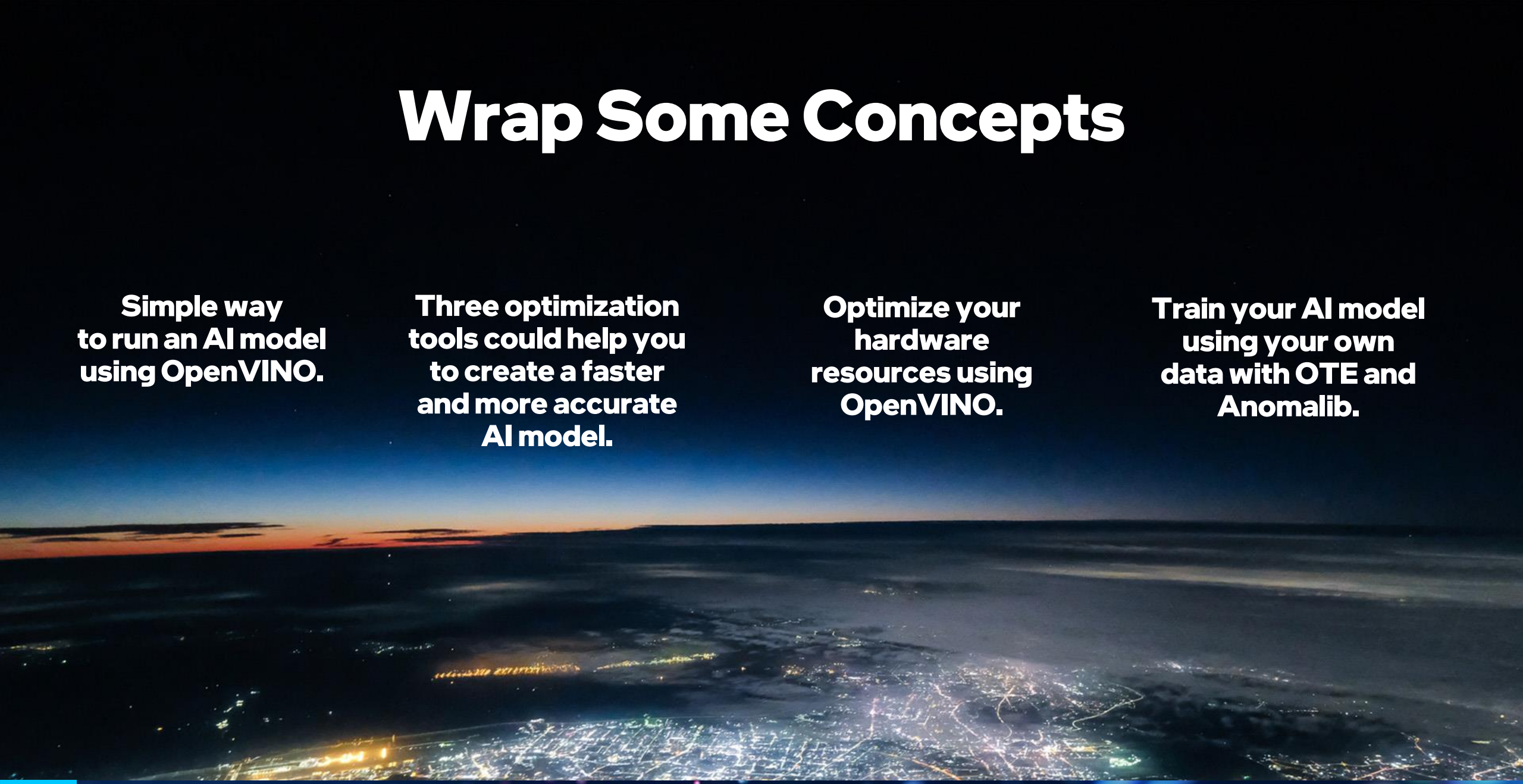
Wrap Some Concepts

Simple way to run an AI model using OpenVINO.

Three optimization tools could help you to create a faster and more accurate AI model.

Optimize your hardware resources using OpenVINO.

Wrap Some Concepts



Simple way to run an AI model using OpenVINO.

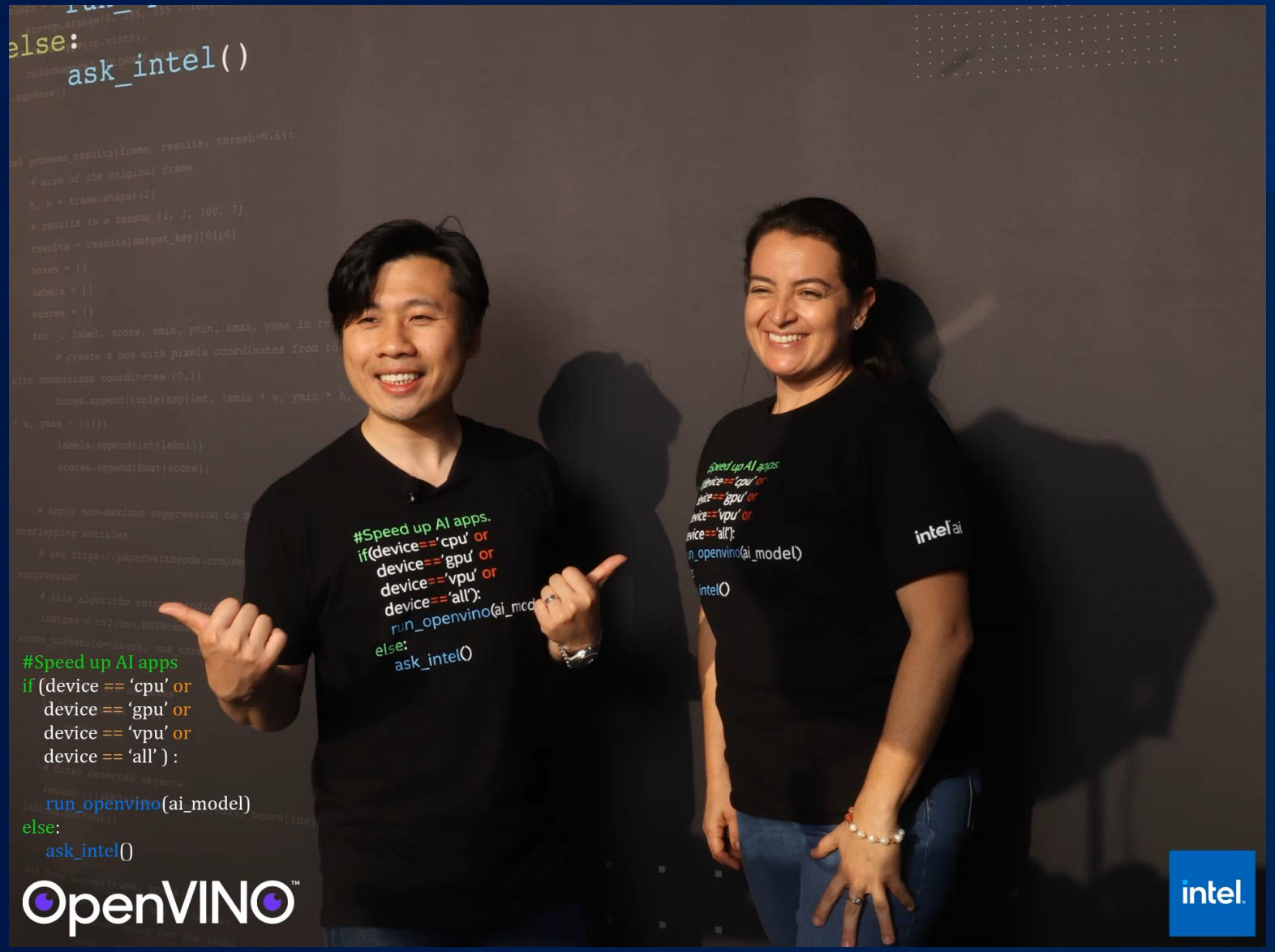
Three optimization tools could help you to create a faster and more accurate AI model.

Optimize your hardware resources using OpenVINO.

Train your AI model using your own data with OTE and Anomalib.

Call for Action

Let’s stay connected



Raymond Lo
.linkedin.com/in/raymondlo84/

Paula Ramos
linkedin.com/in/paula-ramos-41097319/

OpenVINO™

CVPR 2022

How to Get Quick and Performant Model for Your Edge Application.

From Data To Application.

Paula Ramos and Raymond Lo (Speakers)

AI Software Evangelists

Helena Kloosterman, Samet Akcay, Zhuo Wu, and Yury Gorbachev (Contributors)

Notices and disclaimers

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.

Intel® technologies may require enabled hardware, software, or service activation. No product or component can be absolutely secure. Your costs and results may vary.

Performance varies by use, configuration and other factors. Learn more on the Performance Index site.

No product or component can be absolutely secure.

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