intel

OpenVINO™ DEVCON Workshop Series 2024

Model Development and Deployment with the OpenVINO™ Ecosystem

Our speakers

Paula Ramos
AI Software Evangelist

Samet Akcay
AI Research Engineer
& Scientist

Today

01
Computer Vision Solutions

02
OpenVINO Toolkit

03
Anomalib

04
CVPR Challenge

Computer Vision Solutions

AI is Everywhere – from Edge to Cloud

Edge

- Real time data processing
- Wider reach
- Data sovereignty
- Cost efficiency

Frictionless Retail

Traffic Monitoring

Defect Detection

Depth Camera

IP Camera

NVR

Sensor

OpenVINO™ 1 oneAPI

Choose Your Compute

Light AI

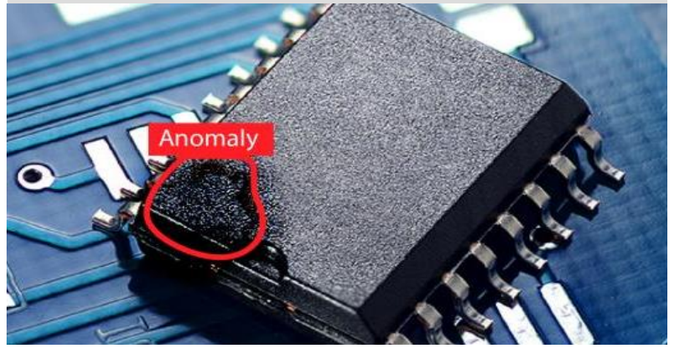
Efficiency for sub-100W designs

CPU AI, built-in GPU, built-in NPU

intel. CORE + intel. ATOM

+

intel. iRIS Xe GRAPHICS



Medium AI

Scale up perf/W for diverse system designs

Discrete GPU & built-in CPU AI

intel. XEON + intel. CORE

+

intel. ARC GRAPHICS + intel. iRIS Xe GRAPHICS

Heavy AI

Optimize for peak perf and density

Optimize with high-end Discrete GPU

intel. XEON + intel. CORE

+

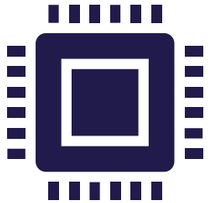
intel. DATA CENTER GPU FLEX SERIES + intel. ARC GRAPHICS

Large
model size

Large memory
footprint

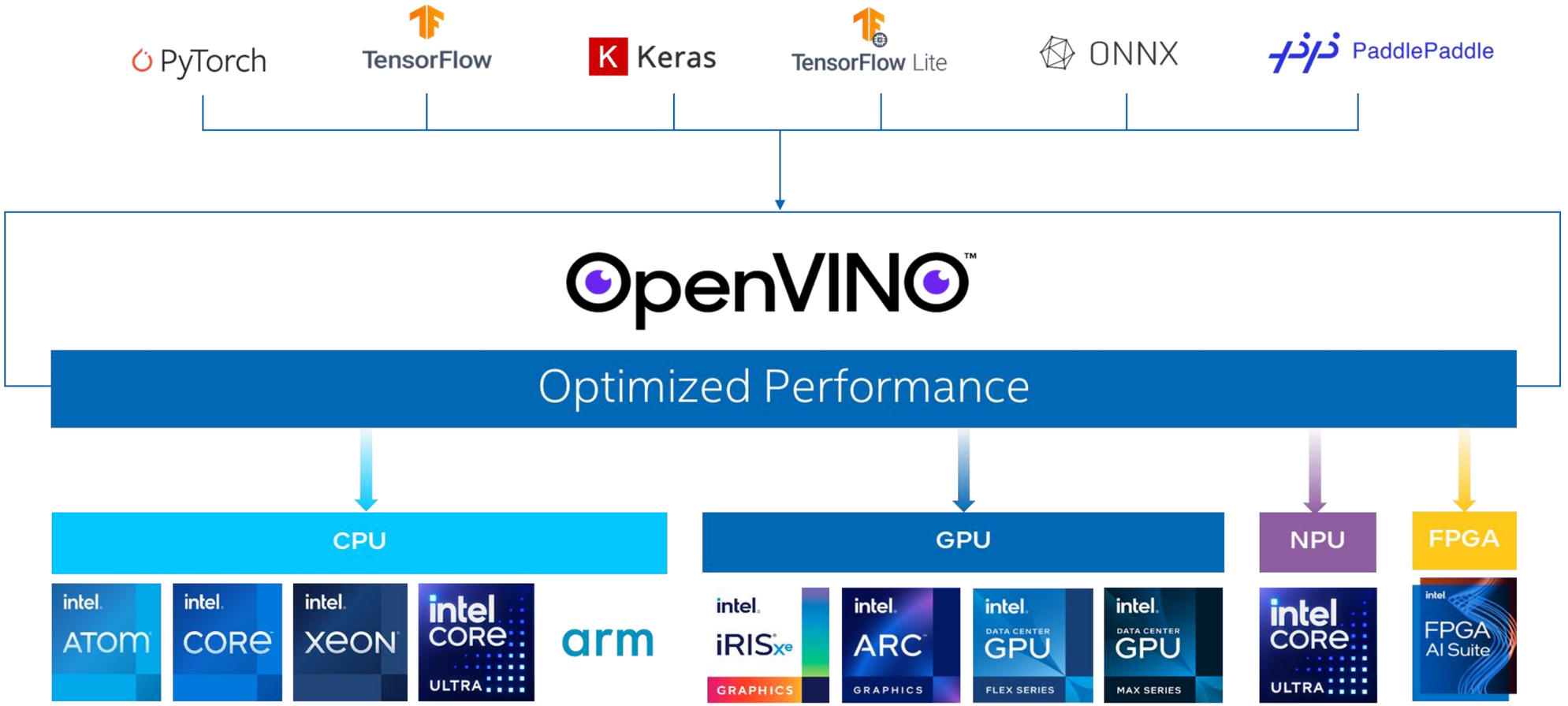
Slow inference
speed

Difficulty training
+ optimizing



No flexibility to
run workloads
on different HW

An Open Standard for Building AI at the Edge



Powered by oneAPI:

Windows Linux macOS

Benefits of Building Applications with OpenVINO™

Build and deploy AI applications in simple steps

Faster inference speed

Maximize AI performance across CPU, GPU, NPU

Smaller model & binary size

Reduce memory footprint

Ability to scale to many nodes with serving

What is going on with the rest of the workflow?

OpenVINO™

Intel open-source library

Real-world datasets
are highly imbalanced

Abnormalities are often unknown and might evolve/change

Anomaly Detection
Anomalib

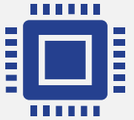
Anomalib is a library to design, develop and deploy DL anomaly detection algorithms

Key Features

SOTA Anomaly Detection Tasks and Models

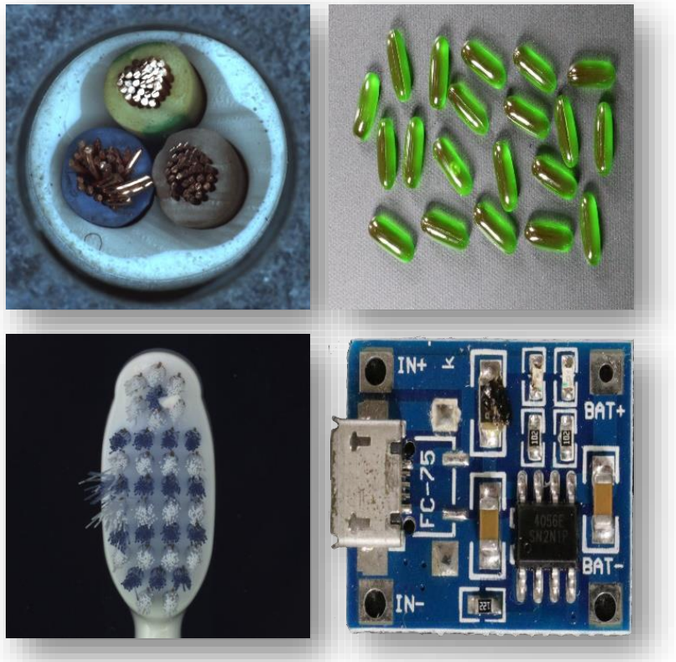
Modular API for customization

Fully configurable CLI for end-to-end machine learning life cycle

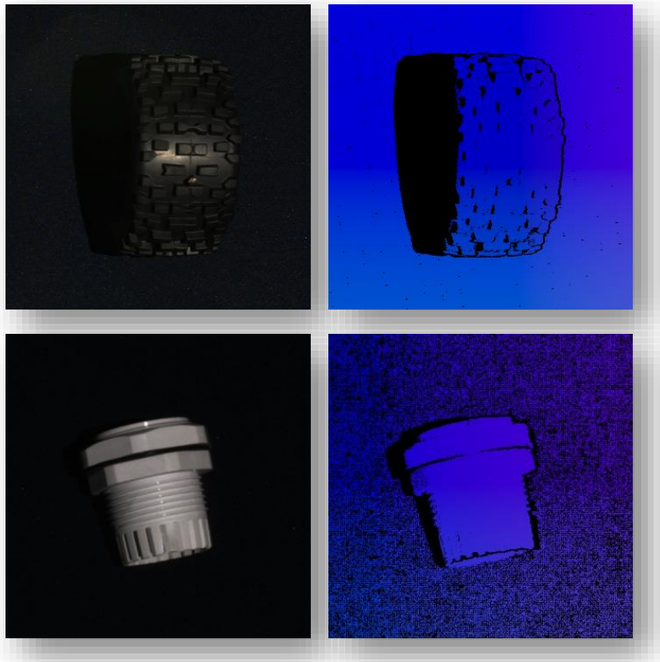


Various inference interfaces for deployment

Supported Domains



Image



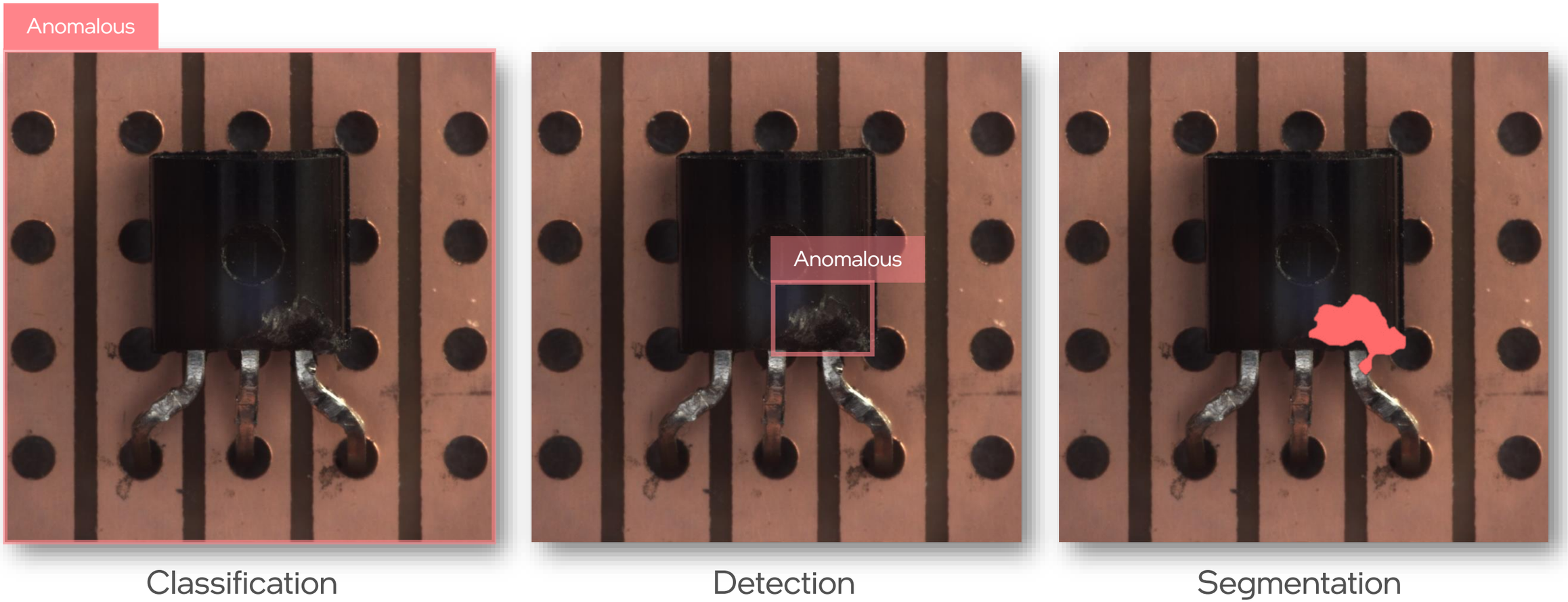
Depth

Video

P. Bergmann *et al.* The MVTec Anomaly Detection Dataset: A Comprehensive Real-World Dataset for Unsupervised Anomaly Detection, IJCV 2021
Zou, Y., Jeong, J., Pemula, L., Zhang, D., & Dabeer, O. SPot-the-Difference Self-supervised Pre-training for Anomaly Detection and Segmentation, ECCV 2022
P. Bergmann, X. Jin, D. Sattlegger, C. Steger: The MVTec 3D-AD Dataset for Unsupervised 3D Anomaly Detection and Localization, VISAPP 2022

W. Liu and W. Luo, D. Lian and S. Gao. "Future Frame Prediction for Anomaly Detection -- A New Baseline." CVPR 2018.
Lu, C., Jianping S., and Jiaya J., Abnormal event detection at 150 fps in matlab, ICCV 2013

Supported Tasks



P. Bergmann *et al.* The MVTec Anomaly Detection Dataset: A Comprehensive Real-World Dataset for Unsupervised Anomaly Detection, IJCV 2021

High-Level Workflow

Training

Trained on normal images only

Validation/Test

Evaluated on normal and anomalous images

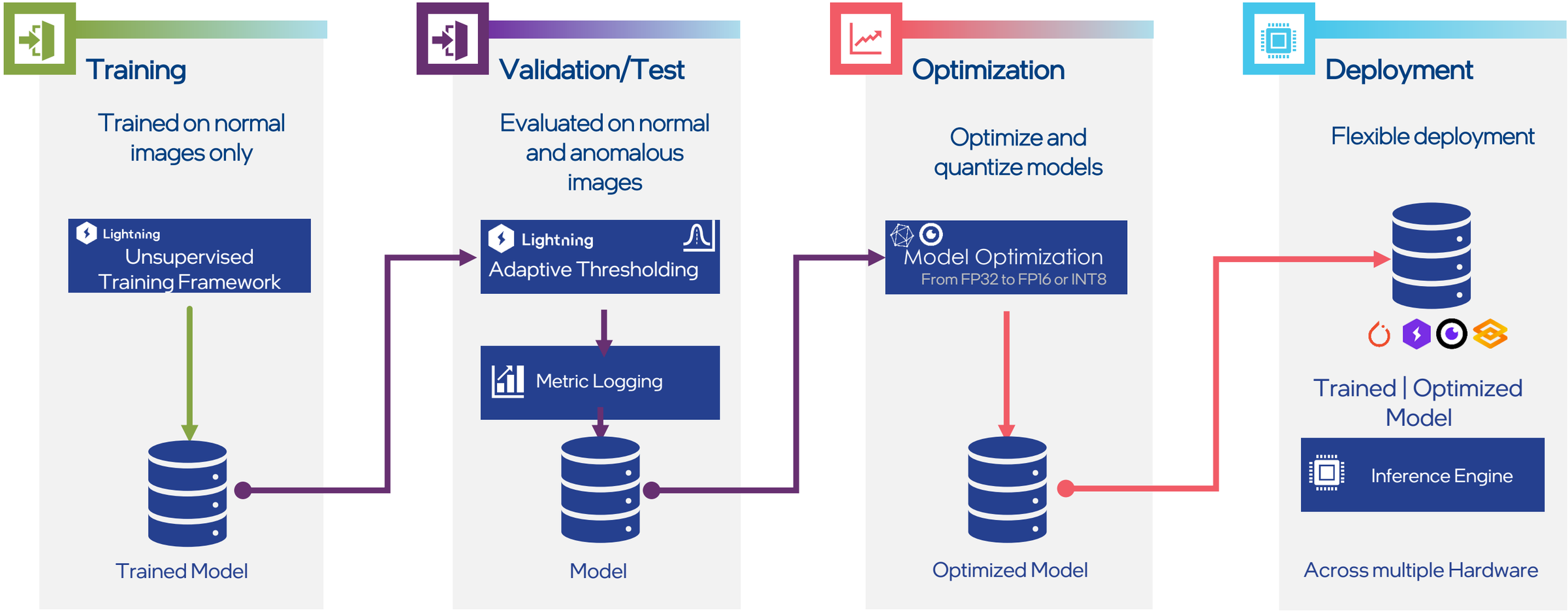
Optimization

Optimize and quantize models

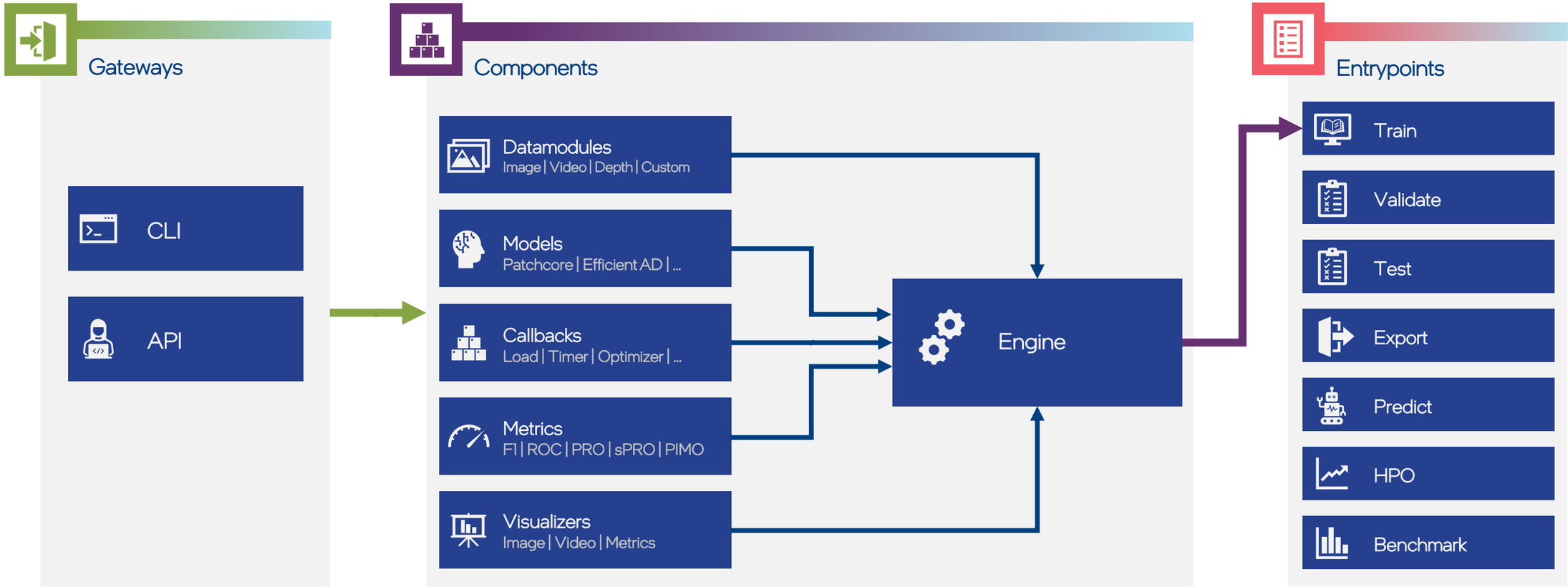
Deployment

Flexible deployment

High-Level Workflow



High-Level Architecture



Entry points - Brief

API – Entrypoints

```
# Train the model
>>> engine.train(...)

# Validate the model
>>> engine.validate(...)

# Test the model
>>> engine.test(...)

# Export the model
>>> engine.export(...)

# Make a prediction
>>> engine.predict(...)
```

CLI – Entrypoints

```
# Train the model
› anomalib train ...

# Validate the model
› anomalib validate ...

# Test the model
› anomalib test ...

# Export the model
› anomalib export ...

# Make a prediction
› anomalib predict ...
```

Learn and build step-by-step

01 Installation

02 Data Acquisition

03 Training

04 Testing

05 Optimization

06 Deployment

01 Installation

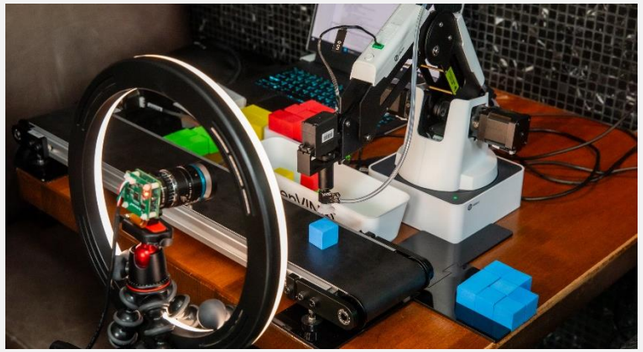
Easy Installation

Hardware-agnostic, configurable installation

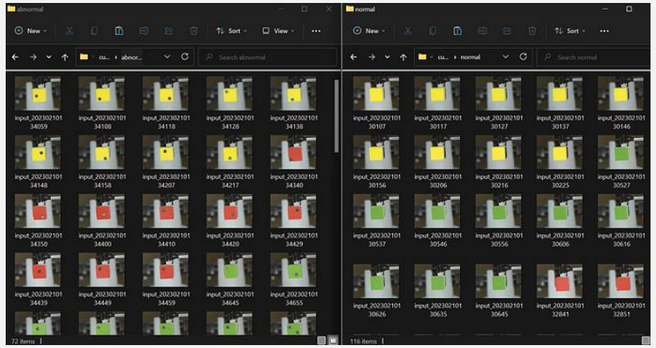
```
❯ pip install anomalib

❯ anomalib install -h
╭─ Arguments ──────────────────────────────────────────────────────────────────╮
│ Usage: anomalib [options] install [-h]                                       │
│                 [--option {full,core,dev,logge                               │
│                 [-v]                                                         │
│                                                                              │
│ Options:                                                                     │
│     -h, --help Show this help message and exit.                              │
│     --option {full,core,dev,loggers,notebooks,openvino}                      │
│         Install the full or optional-dependencies                            │
│         (type: None, default: full)                                          │
│     -v, --verbose Set Logger level to INFO (default: False                   │
│                                                                              │
╰──────────────────────────────────────────────────────────────────────────────╯
```

02 Data Acquisition

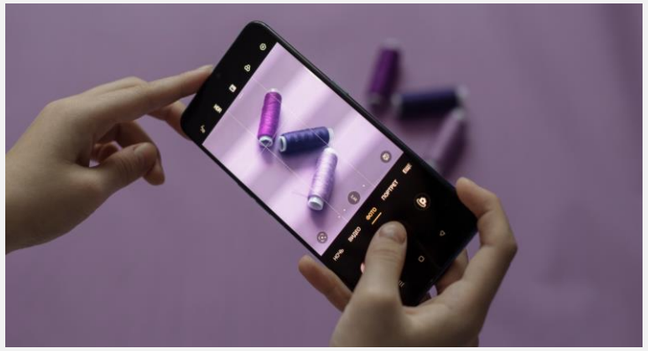
Option 1:
Create a dataset using Dobot Magician



Option 2:
Download images
-Our dataset
-MVTec



Option 3:
Create a dataset with you own phone or camera



02 Data Acquisition

Datamodule API

Public benchmark support

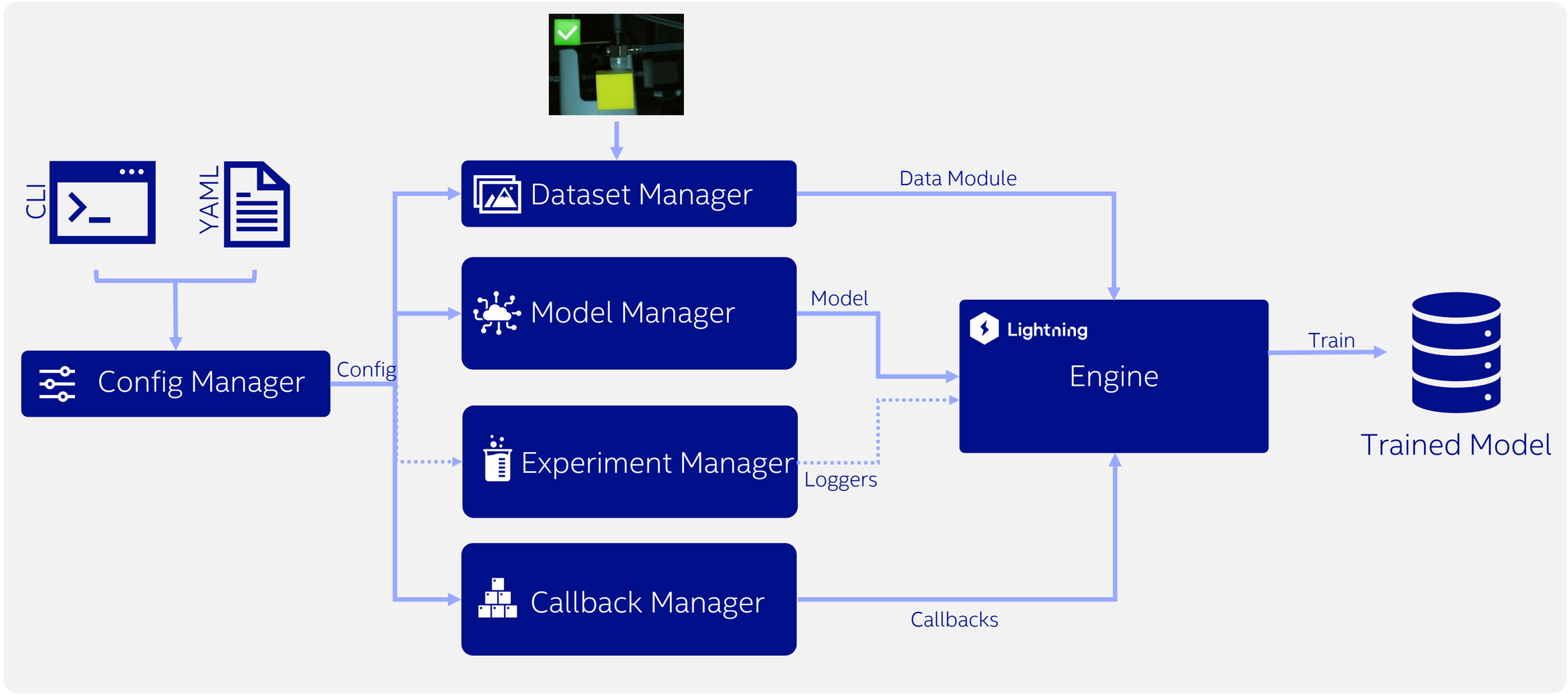
```
>>> from anomalib.data import MVTec, BTech, Visa,
...

>>> datamodule = MVTec()
>>> datamodule.setup()


>>> i, train_data =
next(enumerate(datamodule.train_dataloader()))
>>> print(train_data.keys())
dict_keys(['image_path', 'label', 'image’])

>>> print(train_data["image"].shape)
torch.Size([32, 3, 256, 256])

>>> i, val_data =
next(enumerate(datamodule.val_dataloader()))
>>> i, test_data =
next(enumerate(datamodule.test_dataloader()))
```

Datamodule API

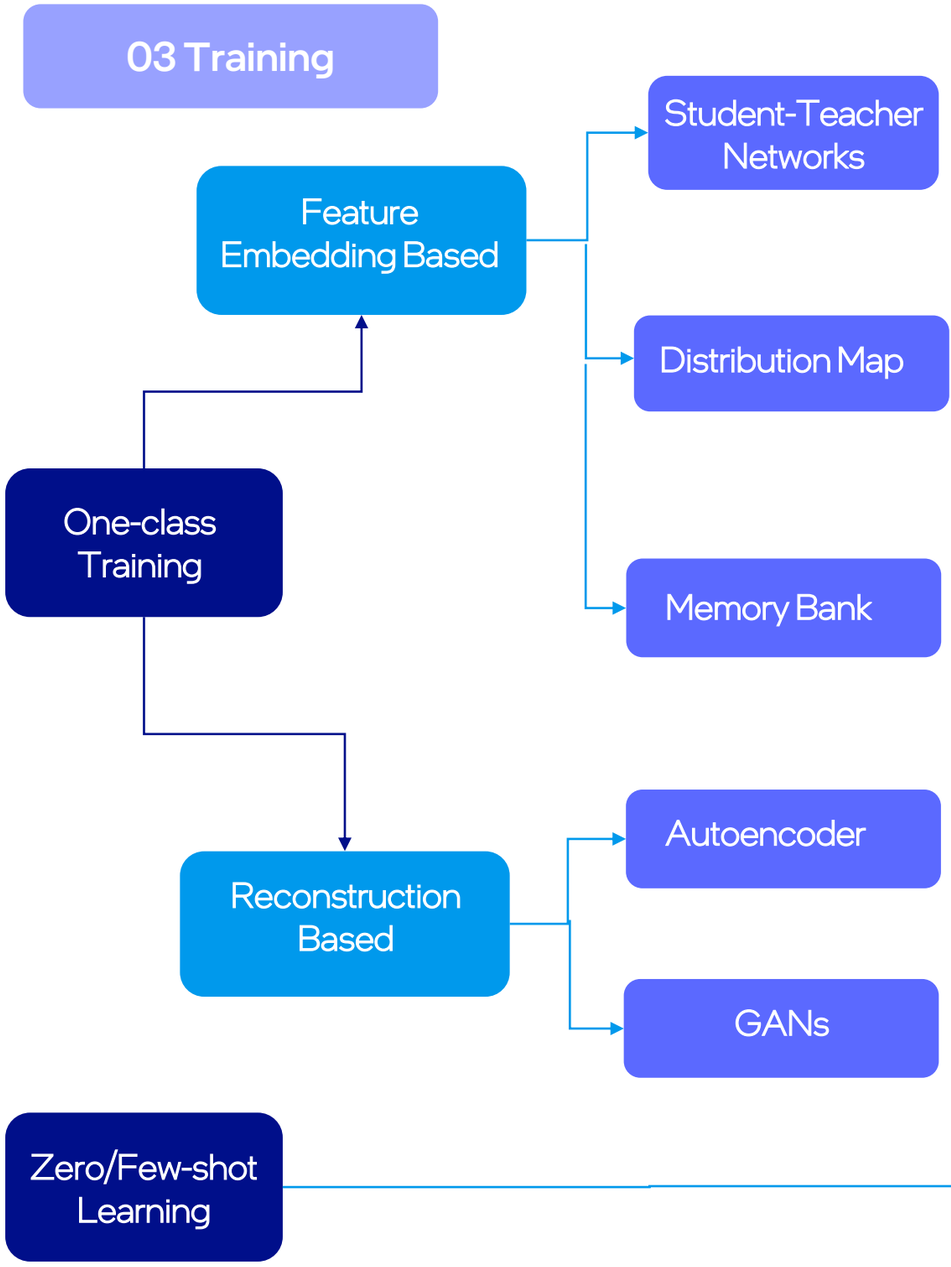
Custom data support

```
>>> from anomalib.data import Folder
>>> datamodule = Folder(
...     name="my_custom_dataset",
...     root="datasets/hazelnut",
...     normal_dir="good",
...     abnormal_dir="bad",
...     task="classification"
... )
>>> datamodule.setup()


>>> i, train_data =
next(enumerate(datamodule.train_dataloader()))

>>> print(train_data["image"].shape)
torch.Size([32, 3, 256, 256])
```

03 Training


CLI
YAML
Config Manager
Config
Dataset Manager
Model Manager
Experiment Manager
Callback Manager
Data Module
Model
Loggers
Callbacks
Lightning
Engine
Train
Trained Model

03 Training



- EfficientAD: Accurate Visual Anomaly Detection at Millisecond-Level Latencies, ArXiv 23
- Anomaly Detection via Reverse Distillation from One-Class Embedding, CVPR 22
- Student-Teacher Feature Pyramid Matching for Unsupervised Anomaly Detection, BMVC 21

- CFLOW-AD: Real-Time Unsupervised Anomaly Detection with Localization via Conditional Normalizing Flows, WACV 22
- Fully Convolutional Cross-Scale-Flows for Image-based Defect Detection, WACV 22
- FastFlow: Unsupervised Anomaly Detection and Localization via 2D Normalizing Flows, ArXiV 21

- Patchcore: Towards Total Recall in Industrial Anomaly Detection, CVPR 22
- PaDiM: a Patch Distribution Modeling Framework for Anomaly Detection, ICPR 21
- Attribute-based Representations for Accurate and Interpretable Video Anomaly Detection, ArXiV 23
- Coupled-hypersphere-based Feature Adaptation for Target-Oriented Anomaly Localization, IEEE Access 22

- DRÆM – A discriminatively trained reconstruction embedding for surface anomaly detection, ICCV 21
- Self-Supervised Predictive Convolutional Attentive Block for Anomaly Detection, CVPR 22
- EfficientAD: Accurate Visual Anomaly Detection at Millisecond-Level Latencies, ArXiv 23
- DSR – A dual subspace re-projection network for surface anomaly detection, ECCV 22

- GANomaly: Semi-Supervised Anomaly Detection via Adversarial Training, ACCV 18

- WinCLIP: Zero-/Few-Shot Anomaly Classification and Segmentation, CVPR 23

* This is a growing list as anomalib is constantly updated with newer models

Entry points

API - Train

```
# Import the required modules
>>> from anomalib.data import MVTec
>>> from anomalib.models import Patchcore
>>> from anomalib.engine import Engine

# Initialize the datamodule, model and engine
>>> datamodule = MVTec()
>>> model = Patchcore()
>>> engine = Engine()

# Train the model
>>> engine.train(datamodule=datamodule, model=model)
```

CLI - Train

```
# Get help about the training arguments, run:
❯ anomalib train -h

# Train by using the default values.
❯ anomalib train \
      --data anomalib.data.MVTec \
      --model Patchcore

# Train by overriding arguments.
❯ anomalib train \
       --model Patchcore \
       --data anomalib.data.MVTec \
       --data.category transistor

# Train using YAML config file.
❯ anomalib train \
       --config patchcore.yaml
```

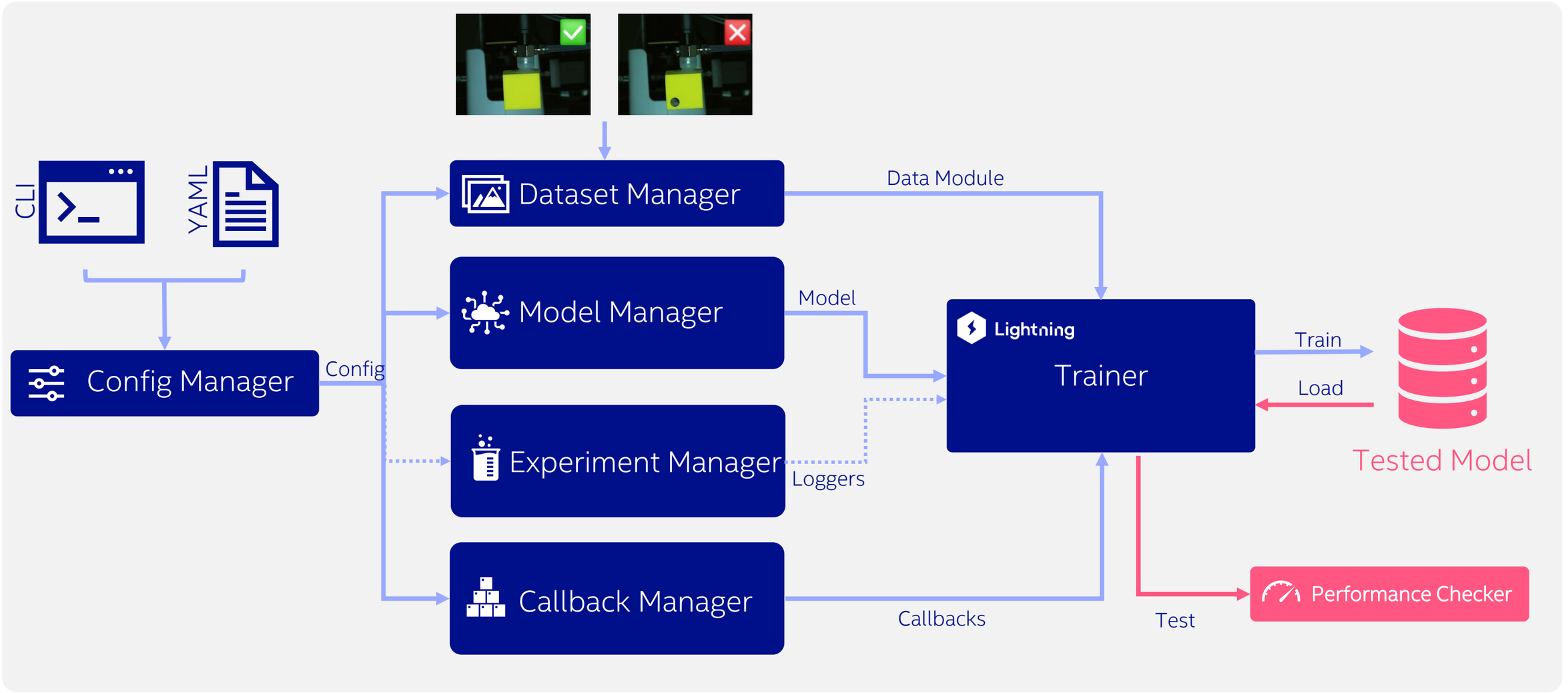
Metrics

Metrics API

Image and Pixel-wise metric support via torchmetrics

```
# Import the required modules
>>> from anomalib.data import MVTec
>>> from anomalib.models import Patchcore
>>> from anomalib.engine import Engine

# Customize the image and/or pixel metrics
>>> datamodule = MVTec()
>>> model = Patchcore()
>>> engine = Engine(pixel_metrics=["F1Score", "AUROC", "AUPRO"])

# Train the model
>>> engine.train(datamodule=datamodule, model=model)
```

04 Testing

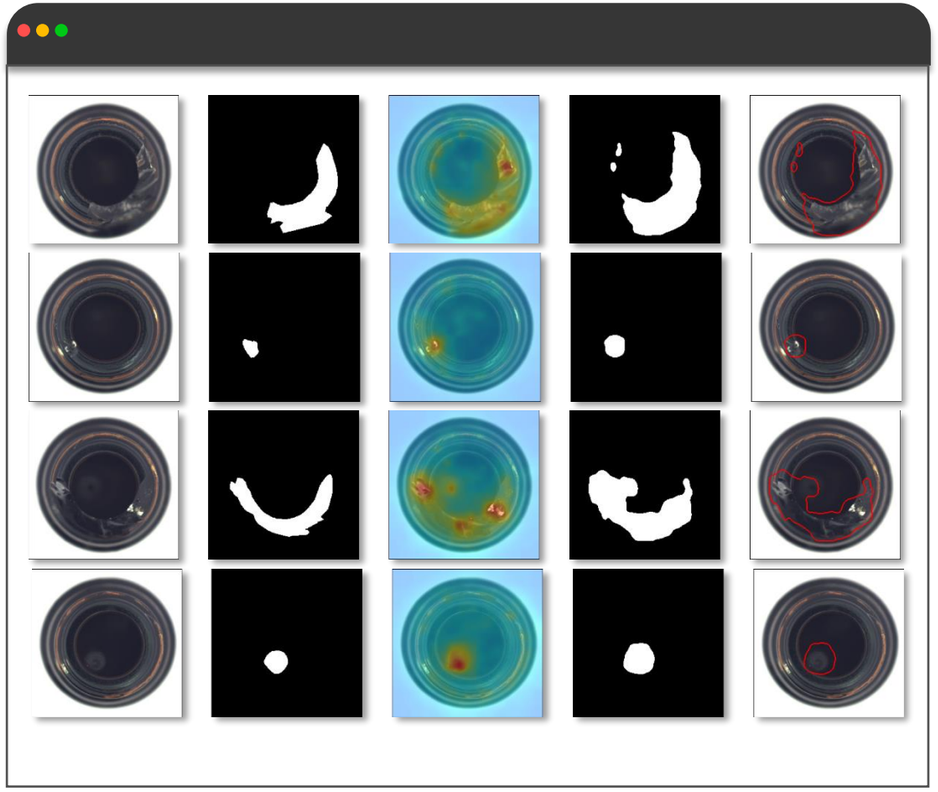

CLI
YAML
Config Manager
Config
Dataset Manager
Model Manager
Experiment Manager
Callback Manager
Data Module
Model
Loggers
Callbacks
Lightning
Trainer
Train
Load
Tested Model
Test
Performance Checker

03 Training

Modular API

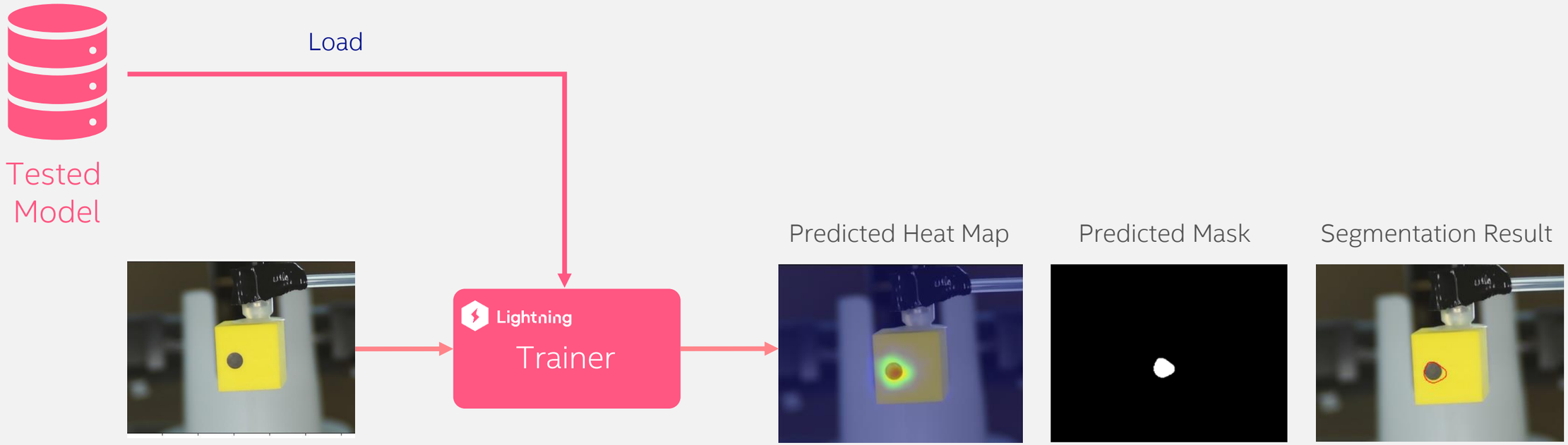
```python
# Import data, model and trainer
from anomalib.data import MVTec
from anomalib.models import Padim
from anomalib.trainer import AnomalibTrainer


# Initialize the data, model and trainer
datamodule = MVTec("./MVTec/", "bottle", image_size=256)
model = Padim(input_size=(256, 256))
trainer = AnomalibTrainer(pixel_metrics=["AUROC"], max_epochs=1)


# Train the model
trainer.fit(model, datamodule=datamodule)
```

```
~/Pr/anomalib  feature/custom_loops !1 ?2
❯
```



* This is a growing list as anomalib is constantly updated with newer models

04 Testing


Load
Tested Model
Lightning
Trainer
Predicted Heat Map
Predicted Mask
Segmentation Result

Entry points

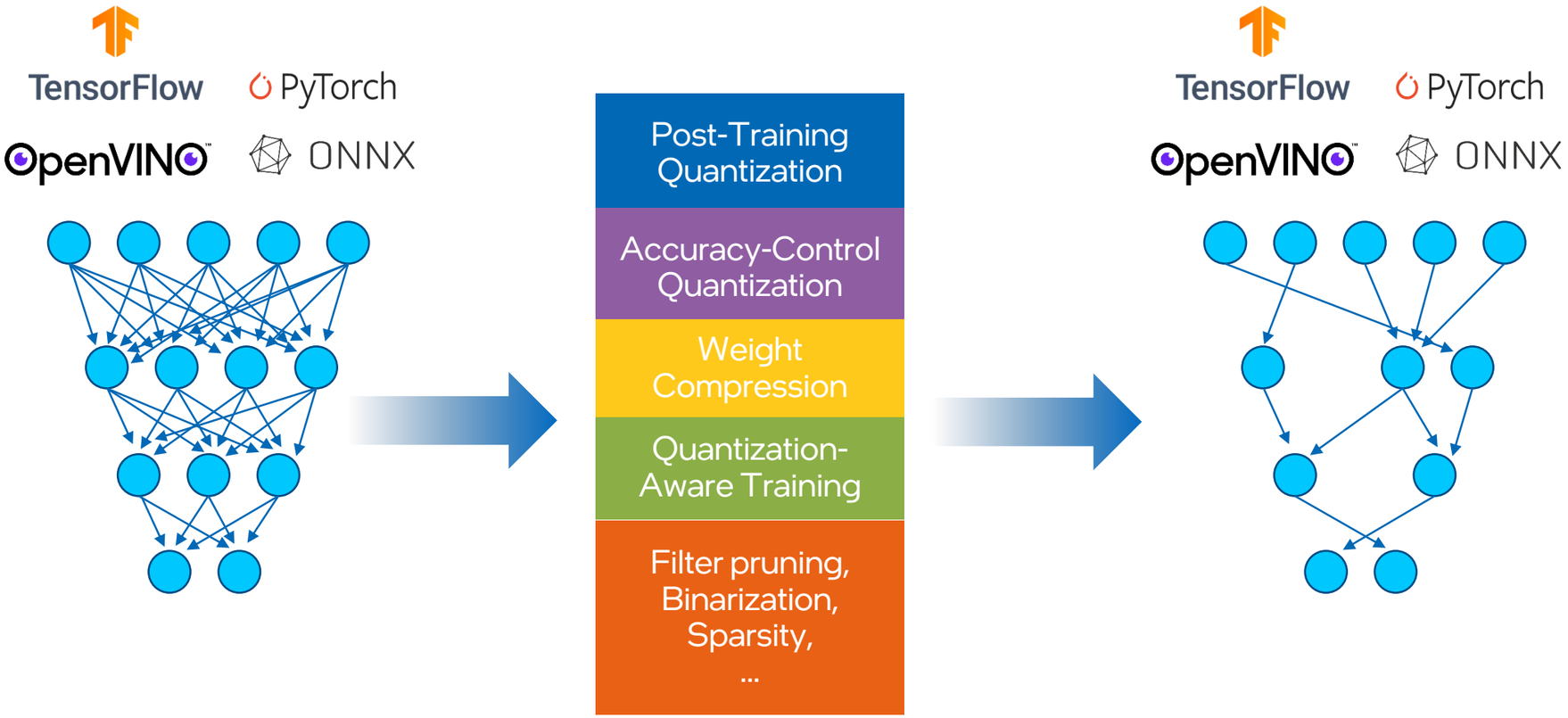
API – Test / Export

```
# Training
...

# Test the model
>>> engine.test(datamodule=datamodule, model=model)

# Export the model
>>> model_path = engine.export(
...     model=model, # model from the previous step
...     export_type="openvino", # "torch", "onnx", "openvino"
... )
```

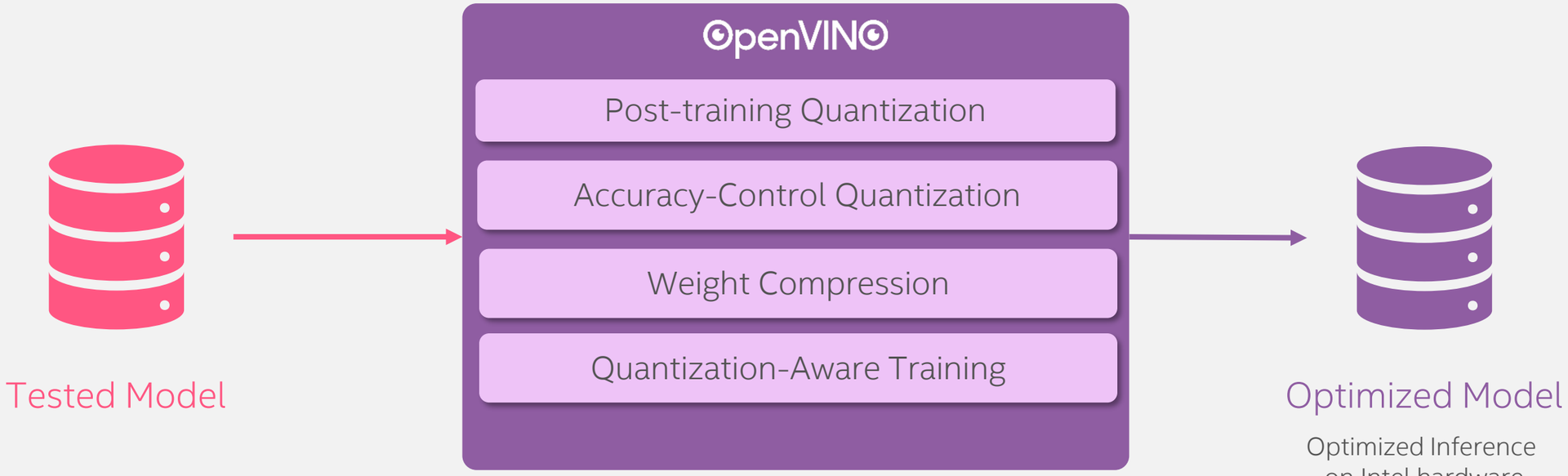
CLI – Test / Export

```
# Get help about the training arguments, run:
> anomalib test -h

# Train by using the default values.
> anomalib test \
      --data anomalib.data.MVTec \
      --model Patchcore

# Train by overriding arguments.
> anomalib export \
       --model Patchcore \
       --data anomalib.data.MVTec \
       --export_type openvino \
```

Neural Network Compression Framework (NNCF)



05 Optimization



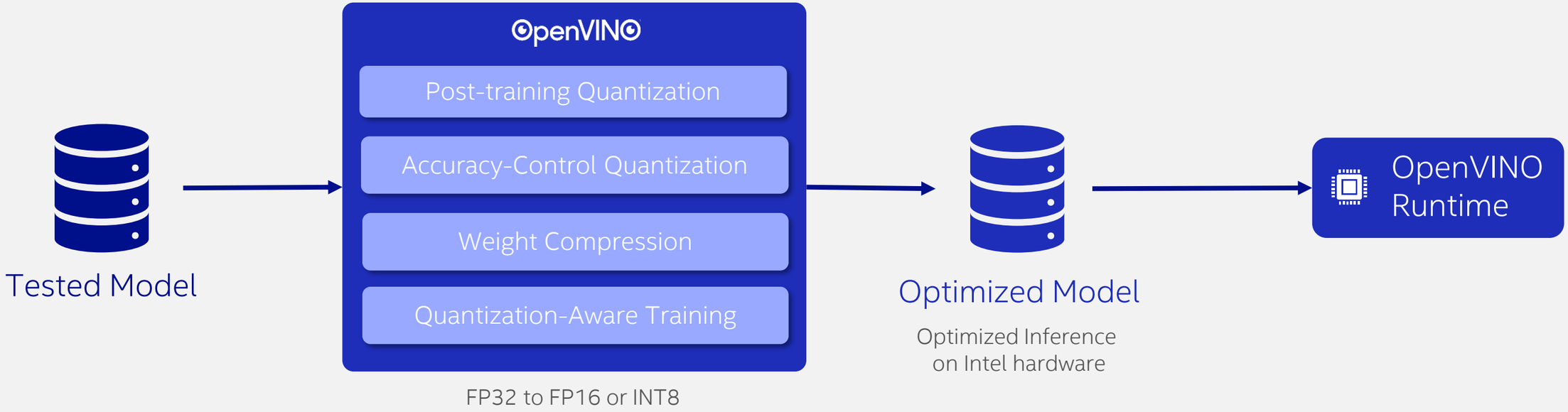
Neural Network Compression Framework (NNCF)

- Post-Training Quantization
- Accuracy-Control Quantization
- Weight Compression
- Quantization-Aware Training
- Filter pruning, Binarization, Sparsity, ...

API – Model Compression and quantization

```
# Post-Training Quantization
>>> engine.export(model, ExportType.OPENVINO, export_dir,
compression_type=CompressionType.INT8_PTQ, datamodule=datamodule)

# Accuracy-Control Quantization
>>> engine.export(model, ExportType.OPENVINO, export_dir,
compression_type=CompressionType.INT8_ACQ, datamodule=datamodule),
metric="F1Score")

# Weight Compression
>>> engine.export((model, ExportType.OPENVINO, export_dir,
compression_type=CompressionType.FP16, datamodule=datamodule)

>>> engine.export((model, ExportType.OPENVINO, export_dir,
compression_type=CompressionType.INT8, datamodule=datamodule)
```

06 Deployment

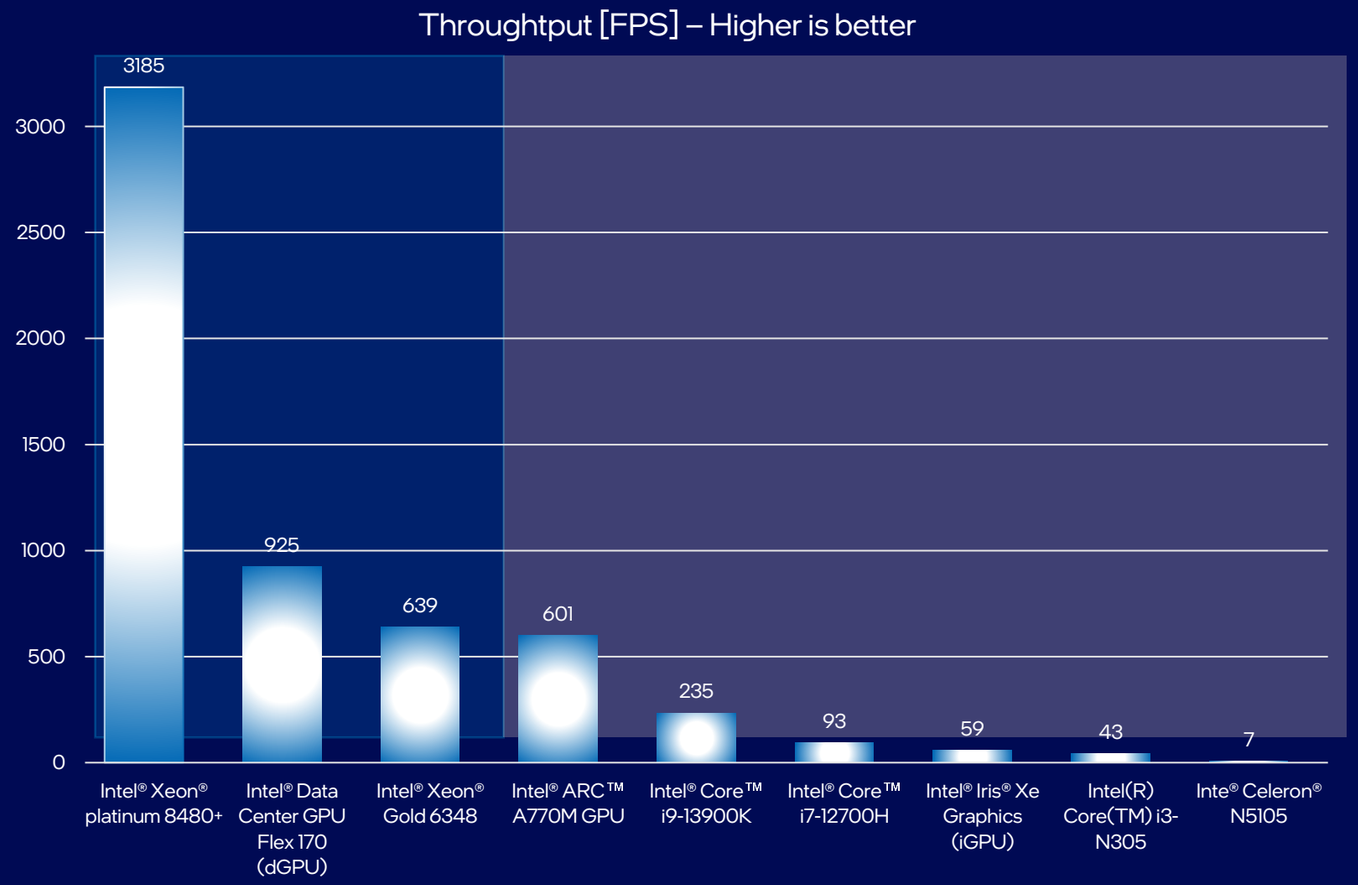


intel

Demo using Anomalib Notebooks

OpenVINO™

Optimized Performance



Inferencing results on Padim model Trained using Anomalib

Results may vary

Platform Configurations for Performance Benchmarks

System Board	Intel Corporation D50DNP1SBB	Intel Corporation M50CYP2SBSTD	Intel® Client Systems NUC13RNGi9	Intel® Client Systems NUC12SNKi72	AAEON UPN-ADLN01 V1.0 220950173	AIxBoard Aixboard-jspl
CPU	Intel(R) Xeon(R) Platinum 8480+	Intel(R) Xeon(R) Gold 6348 CPU @ 2.60GHz	13th Gen Intel(R) Core(TM) i9-13900K @ 5.4 GHz	12th Gen Intel® Core™ i7-12700H @ 2.30 GHz	Intel® Core™ i3-N305 @ 3.80 GHz	Intel® Celeron® N5105 @ 2.00GHz
Sockets / Physical cores	1 / 56 (112 Threads)	2 / 28 (56 Threads)	1 / 24 (32 Threads)	1 /14 (20 Threads)	1 / 8 (8 Threads)	1 / 4
HyperThreading / Turbo Setting	Enabled / On	Enabled / On	Enabled / On	Enabled / On	Disabled	Disabled
Memory	512 GB DDR4 @ 4800 MHz	256 GB DDR4 @ 3200 MHz	64 GB DDR5 @ 4800 MHz	64 GB DDR4 @ 3200 MHz	16GB DDR5 @4800 MHz	8GB LPDDR4 @ 2933MHz
OS	Ubuntu 22.04.2 LTS	Ubuntu 22.04.2 LTS	Ubuntu 22.04.2 LTS	Windows 11 Enterprise v22H2	Ubuntu 22.04.2 LTS	Ubuntu 20.04.6 LTS
Kernel	5.15.0-72-generic	5.15.0-57-generic	5.15.0-73-generic	22621.1702	5.15.0-1028-intel-iotg	5.15.0-1016-intel-iotg
Software	OpenVINO 2023.0.0	OpenVINO 2022.3.0	OpenVINO 2022.3.0	OpenVINO 2022.3.0	OpenVINO 2022.3	OpenVINO 2022.3
BIOS	Intel Corp. SE5C7411.86B.9525.D13.2302071333	Intel Corp. SE5C620.86B.01.01.0007.2210270543	Intel Corp. SBRPL579.0053.2022.1125.0101	Intel Corp. SNADL357.0053.2022.1102.1218	American Megatrends International, LLC. UNADAM10	AIXBoard Aix-jspl-v1
BIOS Release Date	02/07/2023	10/27/2022	11/25/2022	11/02/2022	12/15/2022	03/06/2023
GPU	N/A	1x Intel® Data Center GPU Flex 170	1x Intel® Arc A770 16GB, 512 EU 1x Intel® UHD Graphics 770	1x Intel® Arc A770™ 16GB, 512 EU 1x Intel® Iris® Xe Graphics	N/A	N/A
Workload: Codec, resolution, frame rate Model, size (HxW), BS	Padim Model – Backbone ResNet-18, input size [256, 256], batch 32 FP32	Padim Model – Backbone ResNet-18, input size [256, 256], batch 32 FP32 (CPU) FP16 (GPU)	Padim Model – Backbone ResNet-18, input size [256, 256], batch 32 FP32 (CPU)	Padim Model – Backbone ResNet-18, input size [256, 256], batch 32 FP32 (CPU) FP16 (GPU)	Padim Model – Backbone ResNet-18, input size [256, 256], batch 32 FP32	Padim Model – Backbone ResNet-18, input size [256, 256], batch 32 FP32
TDP	350W	235W	150W	45W	15W	10W
Benchmark Date Benchmarked by	May 31, 2023 Intel Corporation	May 29, 2023 Intel Corporation	May 29, 2023 Intel Corporation	May 29, 2023 Intel Corporation	May 29, 2023 Intel Corporation	May 29, 2023 Intel Corporation

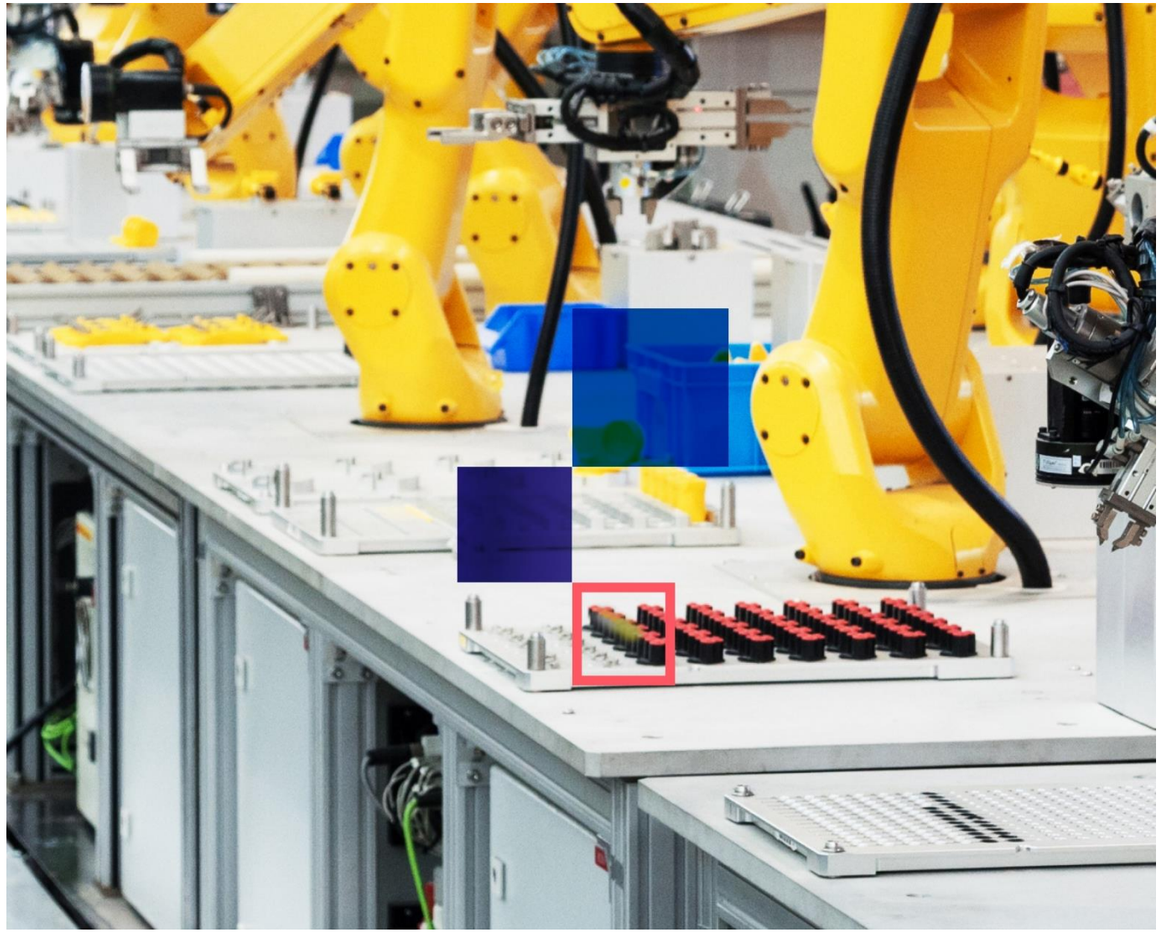
VAND 2.0 Challenge at CVPR

Sponsored by Intel

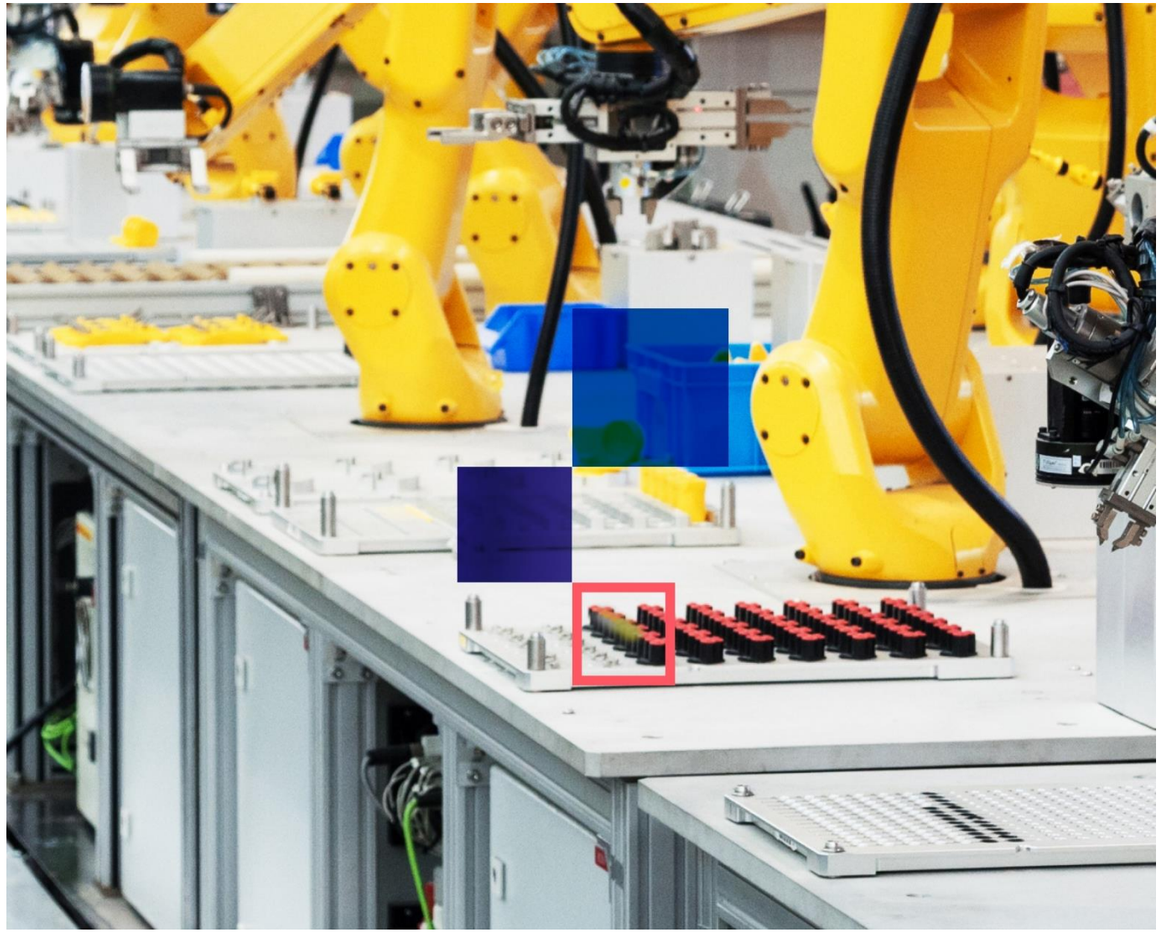
The Challenge

This year our challenge aims to bring visual anomaly detection closer to industrial visual inspection, which has wide real-world applications. This challenge consists of two categories:

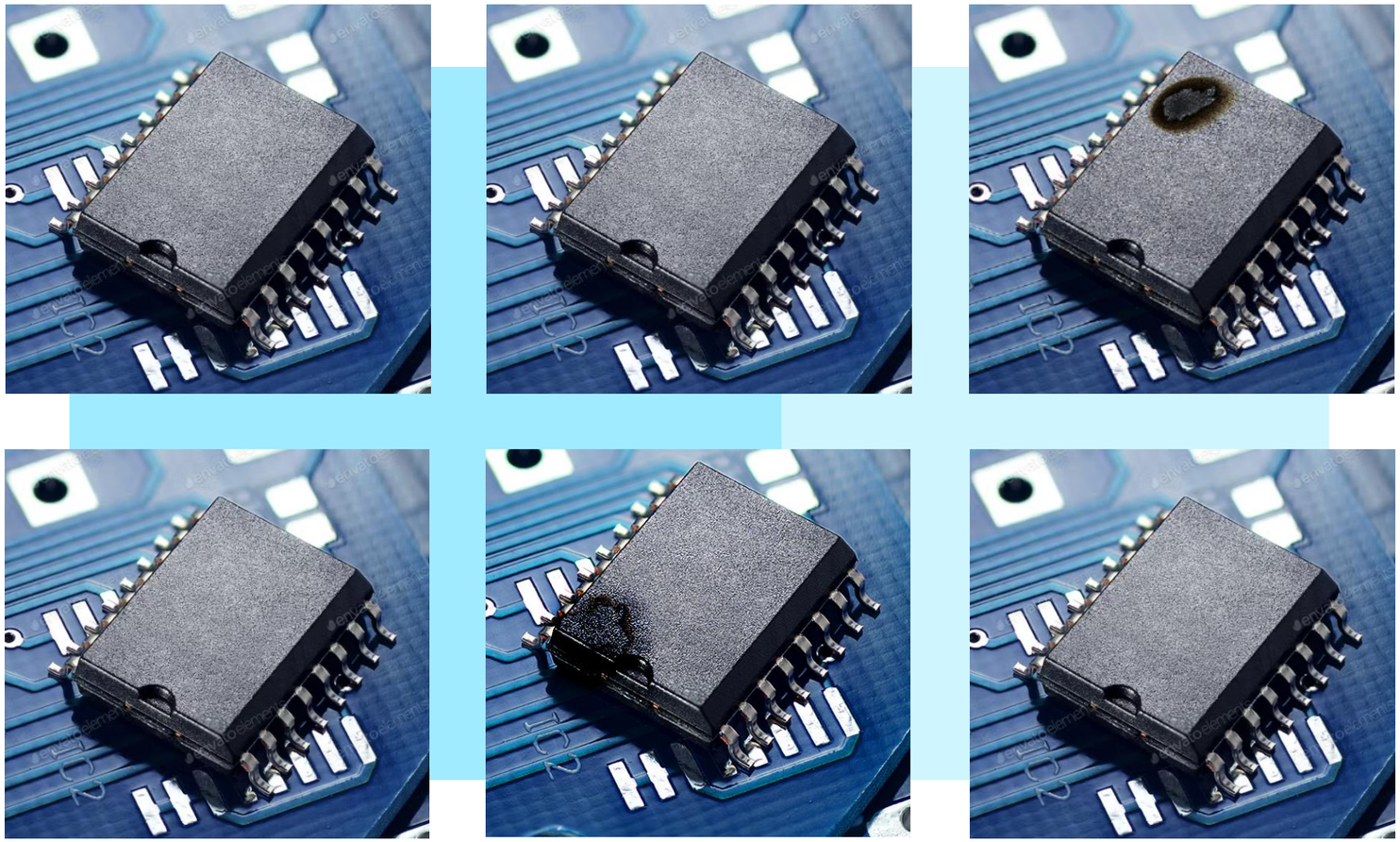
- **Category 1** — Adapt & Detect: Robust Anomaly Detection in Real-World Applications
- **Category 2** — VLM Anomaly Challenge: Few-Shot Learning for Logical and Structural Detection



420 participants 62 submissions



What / Where are the defects? – Category 1

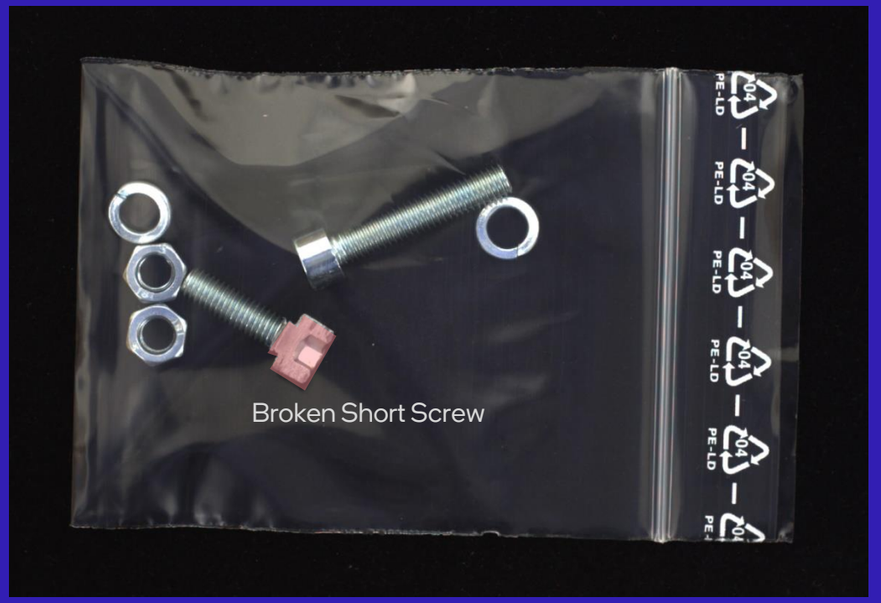


What / Where are the defects?

What / Where are the defects?

Where are the defects?

Where are the defects?



The Leaderboard

F1-Max[1, 2] will be the main evaluation metric for the challenge

- Category 1 final score is the harmonic mean of the Image F1 and Pixel F1 scores.
- Category 2 final score is the Image F1 score.

Category I – Adapt & Detect

Place	Entrant	Project	Image Score	Pixel Score	Final Score
1	Babar Hussain, Qiang Liu, Dahai Yu	ARNet for Robust Anomaly Detection	0.966	0.656	0.811
2	Canhui Tang, M. Callman, Sanping Zhou	CanhuiTang_submission_v2	0.947	0.598	0.773
3	Yukino Tsuzuki	Ensemble PatchCore	0.944	0.450	0.697

Category II – VLM Anomaly Challenge

Place	Entrant	Project	AUFC	Final Score
1	Zhaopeng Gu	AnomalyMoE	0.8184	0.8184
2	Nestor Bao	RJVoyagers_2	0.7958	0.7958
3	saidinesh pola	Category-2 VAND2.0 fewshot classification	0.7867	0.7867

[1] https://www.hackster.io/contests/openvino2024/
[2] https://sites.google.com/view/vand-2-0-cvpr-2024/challenge


Winners

The Prizes

Grand Prize for Each Category

- **The newest Intel AI PC:** a laptop perfect for on-the-go AI boosting performance.
- **An invitation to present their solutions during our workshop!**

Second Place Prize for Each Category

- **An Intel® ARC™ Discrete Graphics Card:** Accelerate your system's performance.

Challenge Special Thanks

Dick Ameln, MSc

AI Research Engineer/Scientist

Utrecht, NL

Ashwin Vaidya, MSc

AI Research Engineer/Scientist

Groningen, NL

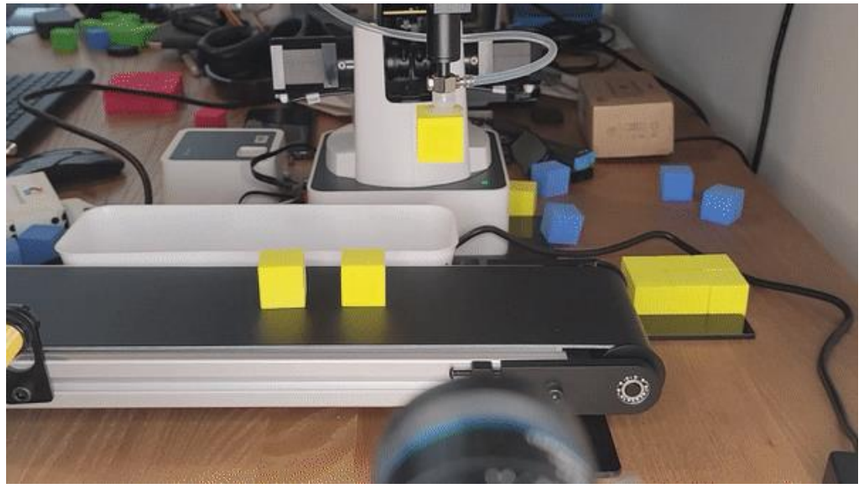
Don’t forget to install Anomalib and OpenVINO for your next Anomaly detection project

Anomalib Repo

OpenVINO Notebooks

Summary of Defect Detection with Anomalib

Solution Demo



Solution Flow

1. Training
2. Testing/Validating
3. Optimization
4. Deployment

Download from our GitHub repo

Solution Highlights

- Unsupervised learning for imbalanced data set (PaDiM model)
- Optimized inference performance (OpenVINO)

OpenVINO Features

- Reduce model size and improve inference speed while maintaining accuracy (quantization)
- Deploy to various intel and third-party hardware accelerators with minimal code changes

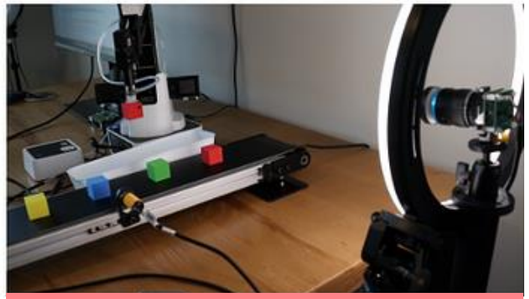
Anomalib Features

- Ready-to-use deep learning anomaly detection algorithms and benchmark datasets.
- PyTorch Lightning based model implementations to reduce boilerplate code and limit the implementation efforts to the bare essentials.
- A set of inference tools for quick and easy deployment of the standard or custom anomaly detection models.

Build real-world AI solutions with Edge AI Reference Kits

Smart Meter Scanning with OpenVINO

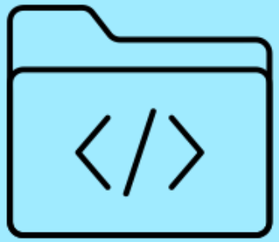
Intelligence Queue Management with OpenVINO



Defect Detection with Anomalib and OpenVINO

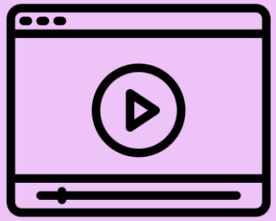
Explore more

Jumpstart with rich materials and resources



Source Codes

Jupyter Notebooks



Videos

Articles

Follow our AI software evangelists around the world

Ria Cheruvu

USA
(Phoenix, Arizona)

MSc in Data Science

Exp. 5+ years as
AI SW Lead Architect
and Research Engineer

Paula Ramos

USA
(Raleigh, North Carolina)

PhD in Engineering

Exp. 17+ years
as researcher

Dmitriy Pastushenkov

EMEA
(Karlsruhe, Germany)

MSc in
Computer Science

Exp. 17+ years as Software
Engineer and Architect

Adrian Boguszewski

EMEA
(Swindon, United Kingdom)

MS in Computer Science

Exp. 5+ years as
DL Engineer

Anisha Udayakumar

APJ
(Bengaluru, India)

BTech in Civil Engineering

Exp. 5+ years as
Innovation Consultant

Zhuo Wu

PRC
(Shanghai, China)

PhD in Electronics

Exp. 15+ years as
researcher and professor

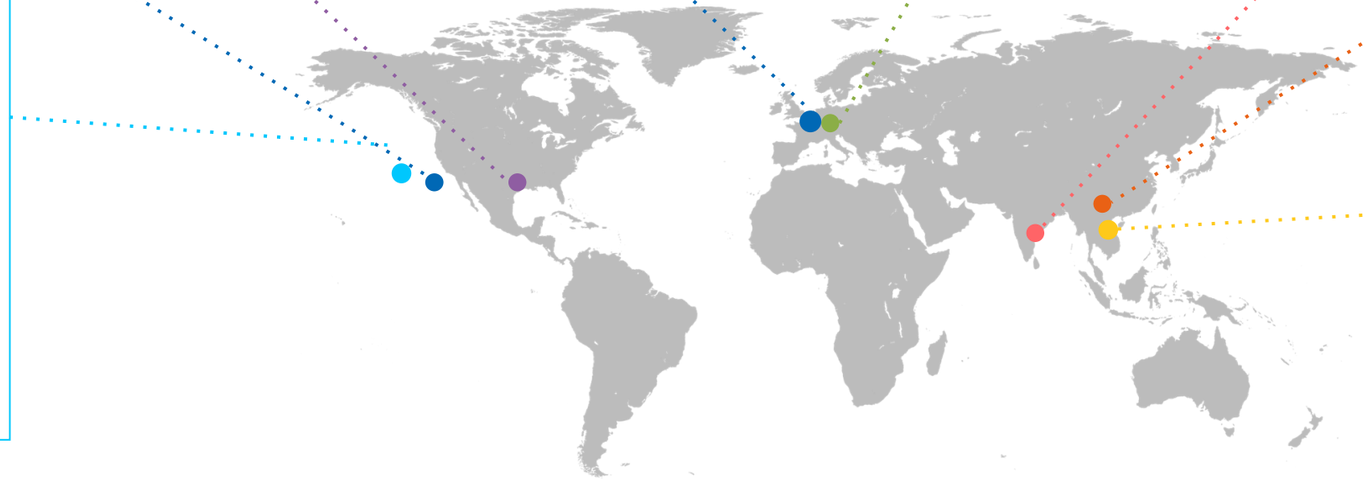
Raymond Lo

Global Lead

USA
(Santa Clara, California)

PhD Engineering

Exp. 9+ as Entrepreneur Executive
and Evangelist



Ethan Yang

PRC
(Shanghai, China)

MSc in Electronics

Exp 5+ years as
AI Solution Engineer

intel innovation

AI: The New Age
Solving the World's Toughest Challenges, Together.

Calling All Developers & Technologists!

From front-end, web, app devs to back-end, full-stack, database & DevOps to data scientists & researchers, and more: Learn, collaborate, and solve at Intel Innovation – an event for developers by developers.

- Hear from leading industry luminaries, technologists & start-up entrepreneurs in the field of AI.
- Get the latest AI development tools, hands-on experience & join on-site Hackathons to optimize your AI code & workflows.
- Learn the breadth of future technology advancements in AI through keynotes, sessions, birds of a feathers, and hands-on labs.
- Share unique ideas and perspectives and collaborate with your peers.

Save the Date:
September 24-25, 2024
San Jose Convention Center, CA

Opt-In for Early Access When Registration Opens!

www.intel.com/innovation

intel

Thank You

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

Samet Akcay
linkedin.com/in/sametakcay
Samet.Akcay@intel.com

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.