



OpenVINO™ Toolkit Overview

**Alexander Nesterov,
Deep Learning Software Engineer**

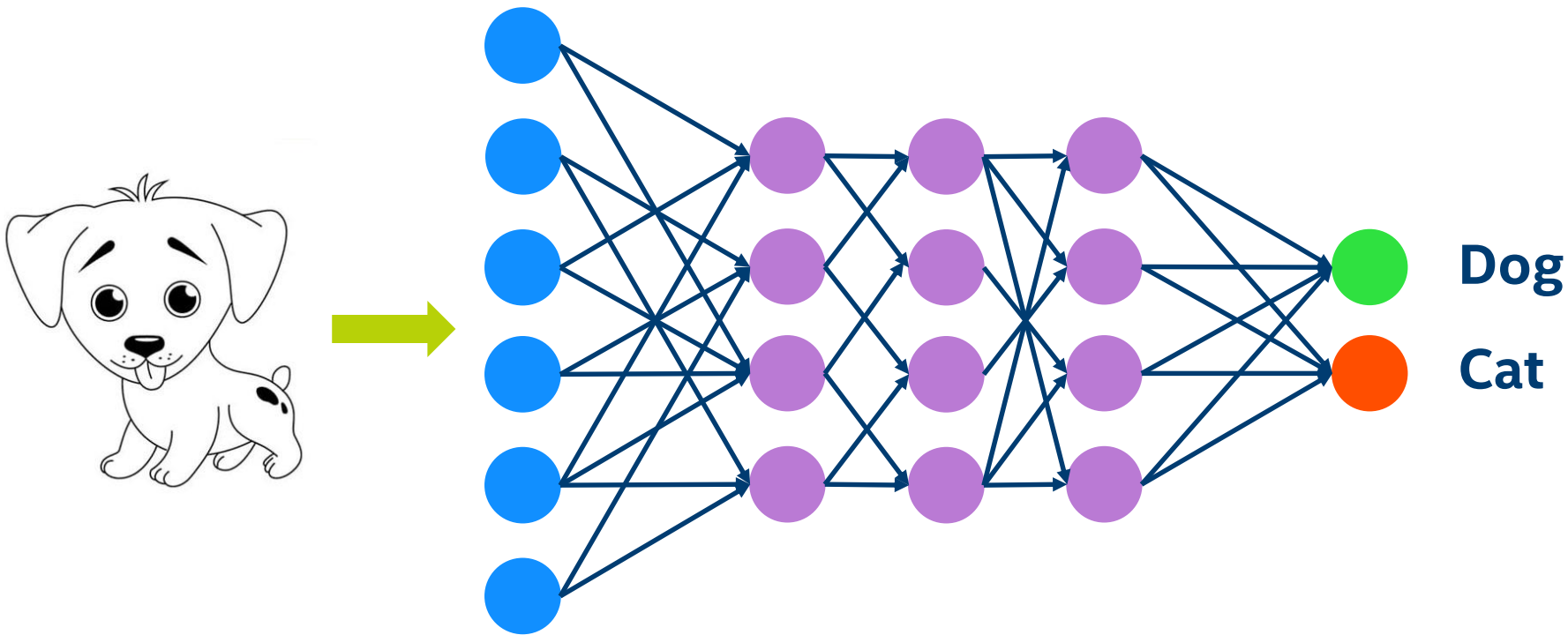
Internet of Things Group

What is OpenVINO Toolkit?

Different applications

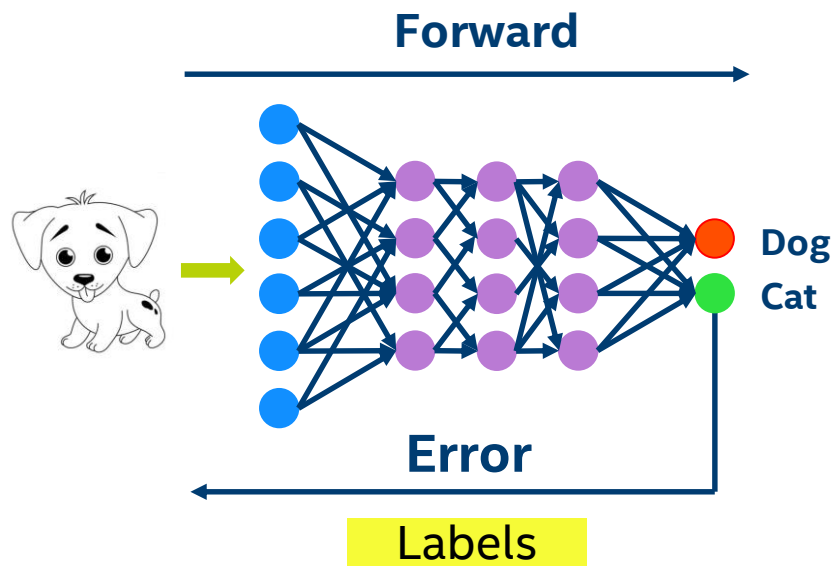
- Solutions for emulate human vision
 - ✓ Convolution neural networks
 - ✓ Traditional computer vision
- Supports heterogeneous execution
 - ✓ CPU, iGPU, FPGA, VPU
- Easy-to-use library of pre-optimized kernels

Deep neural network

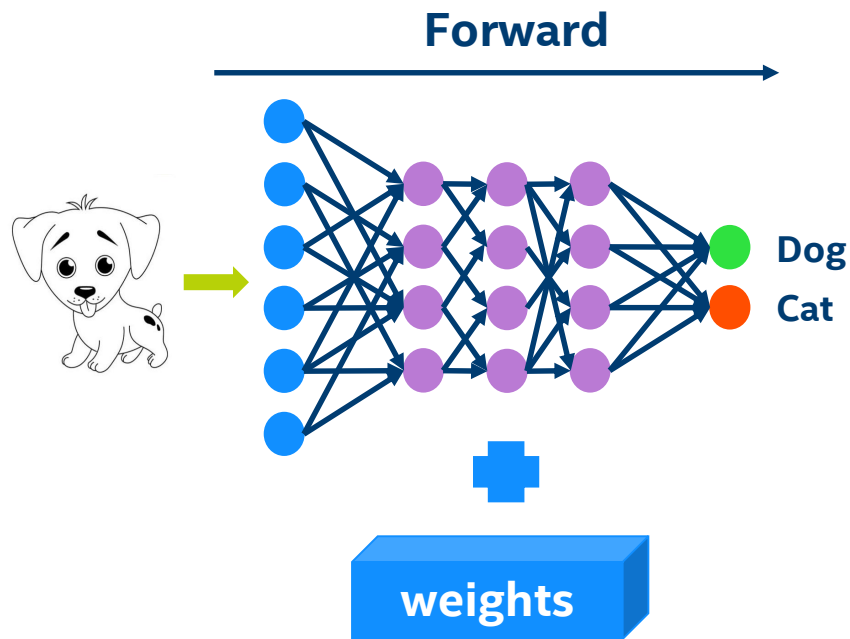


What can be done with DNN?

Learning

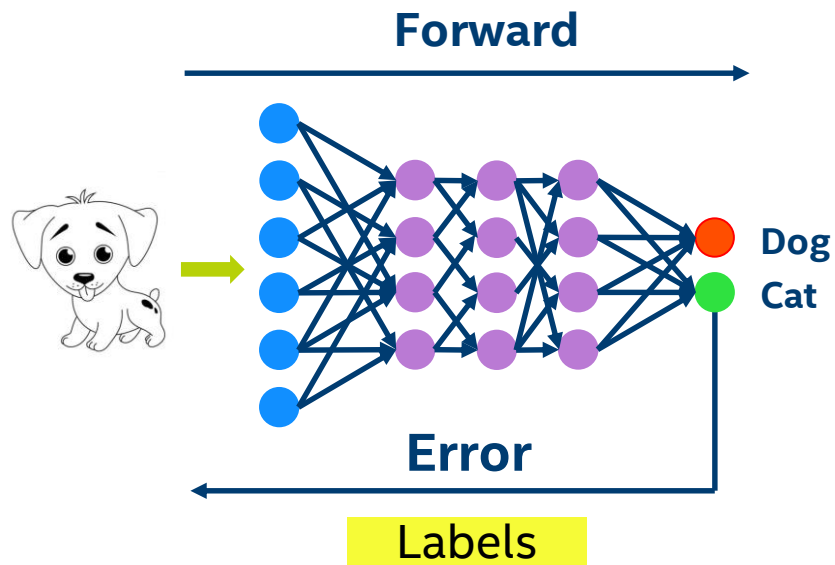


Inference

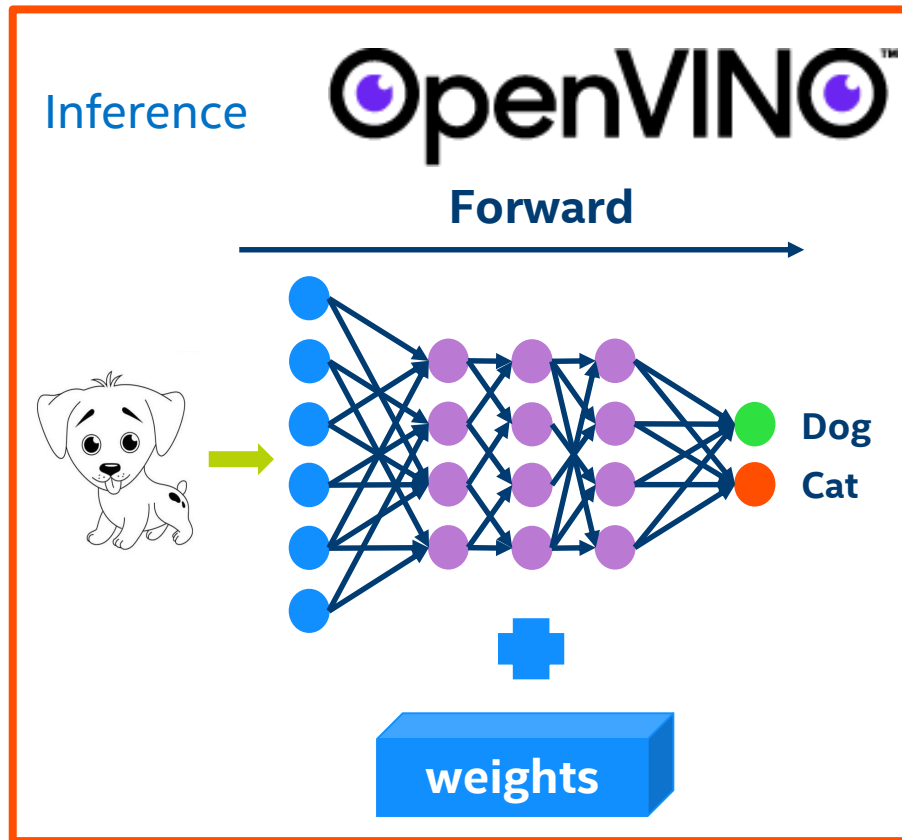


What can be done with DNN?

Learning



Inference



OpenVINO™

Deep
Learning
Deployment
Toolkit

Drivers
and
runtimes for
OpenCL

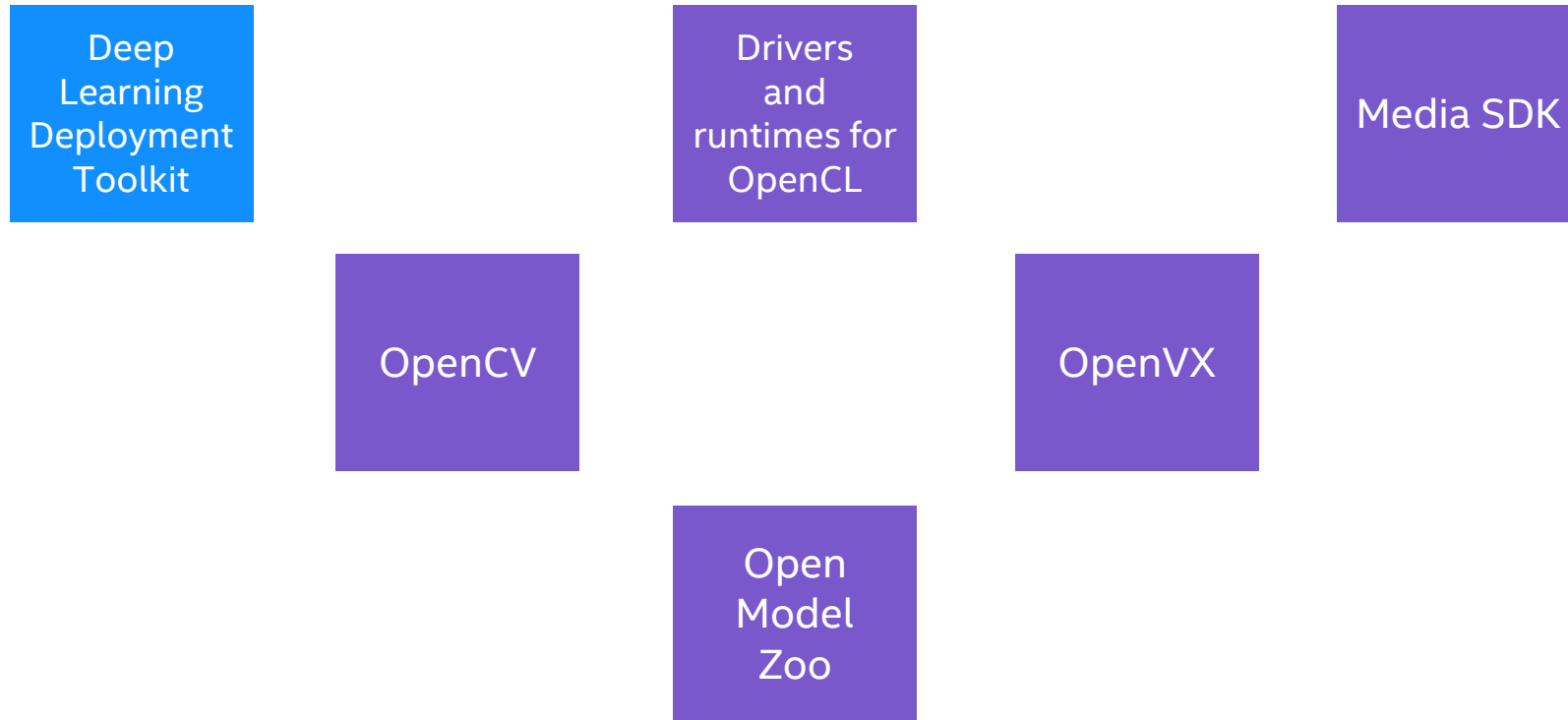
Media SDK

OpenCV

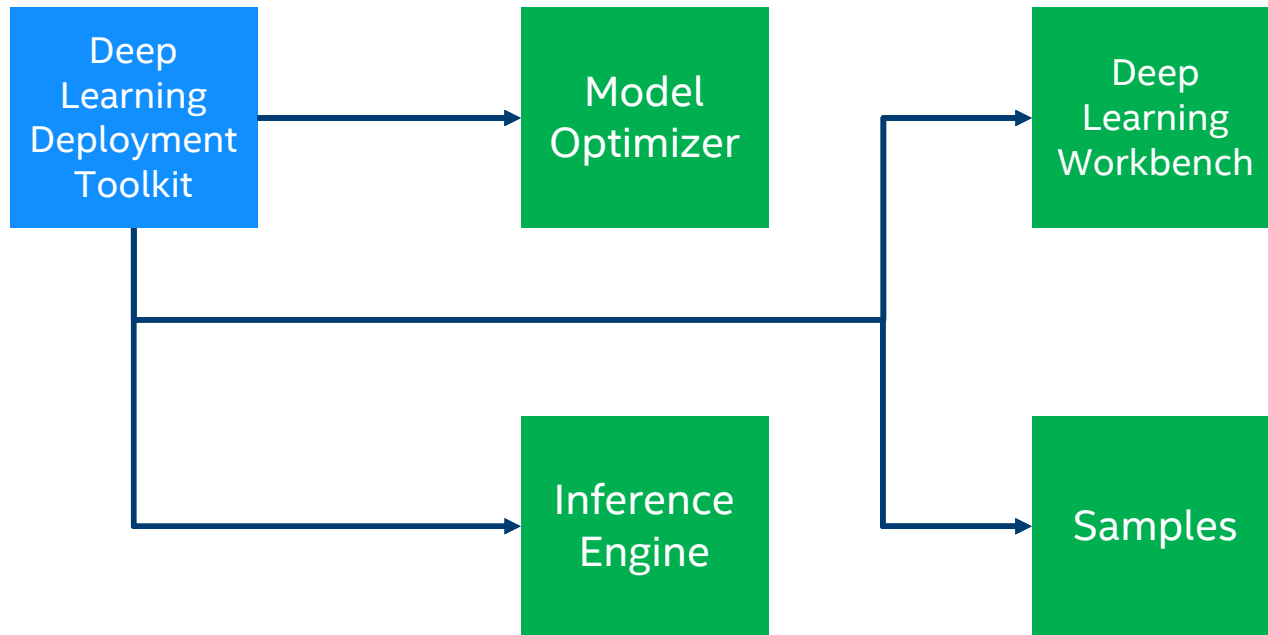
OpenVX

Open
Model
Zoo

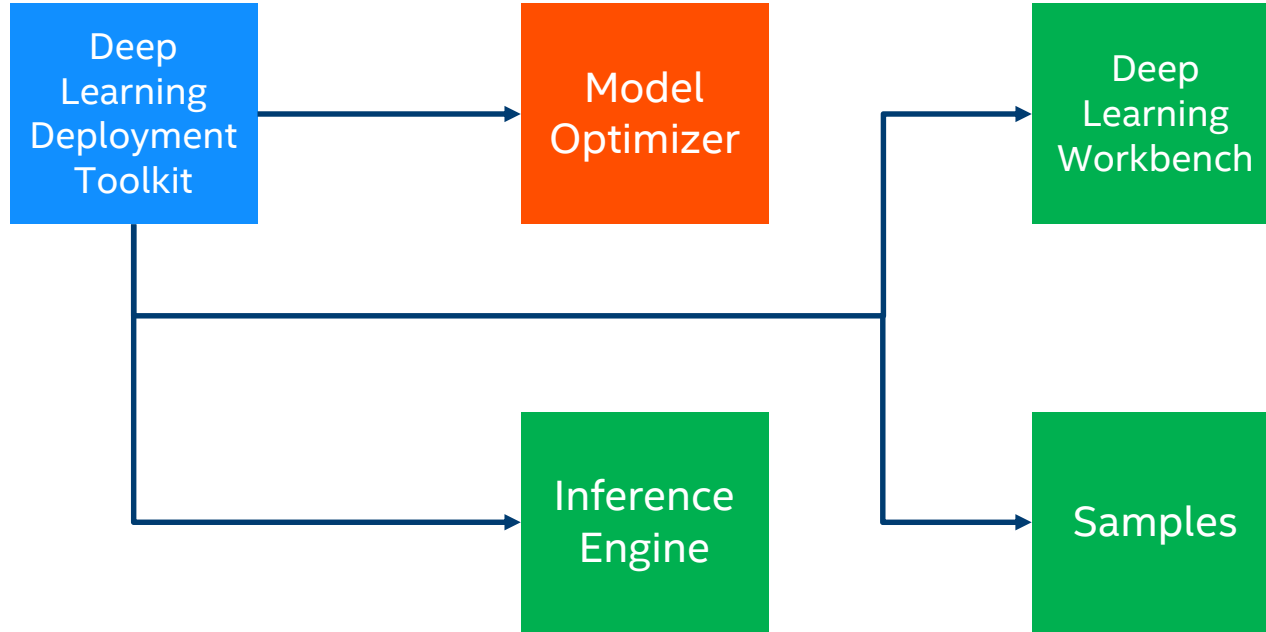
OpenVINO™



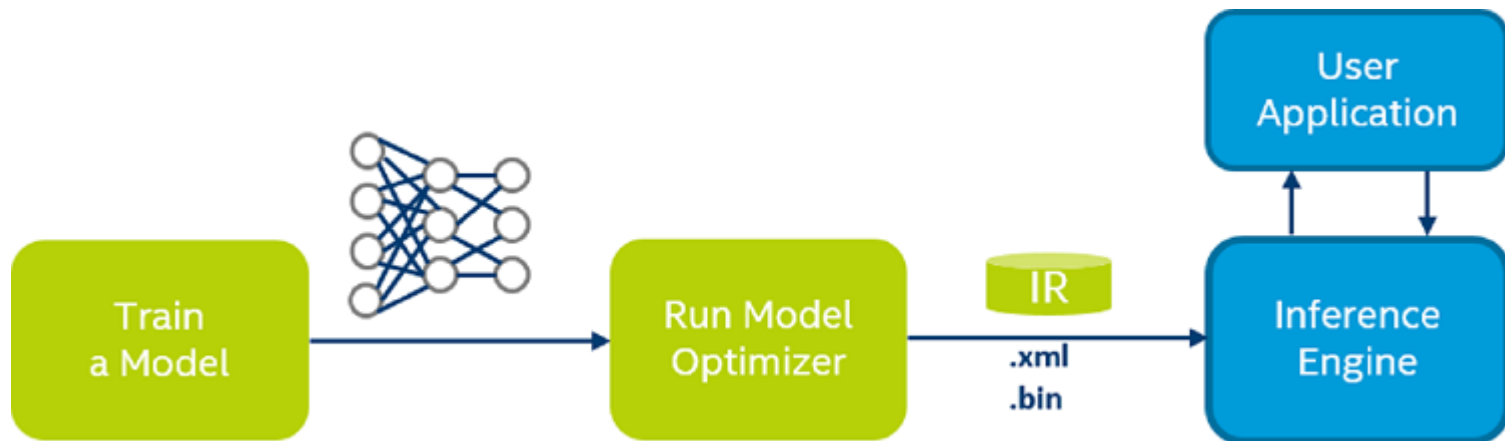
OpenVINO™



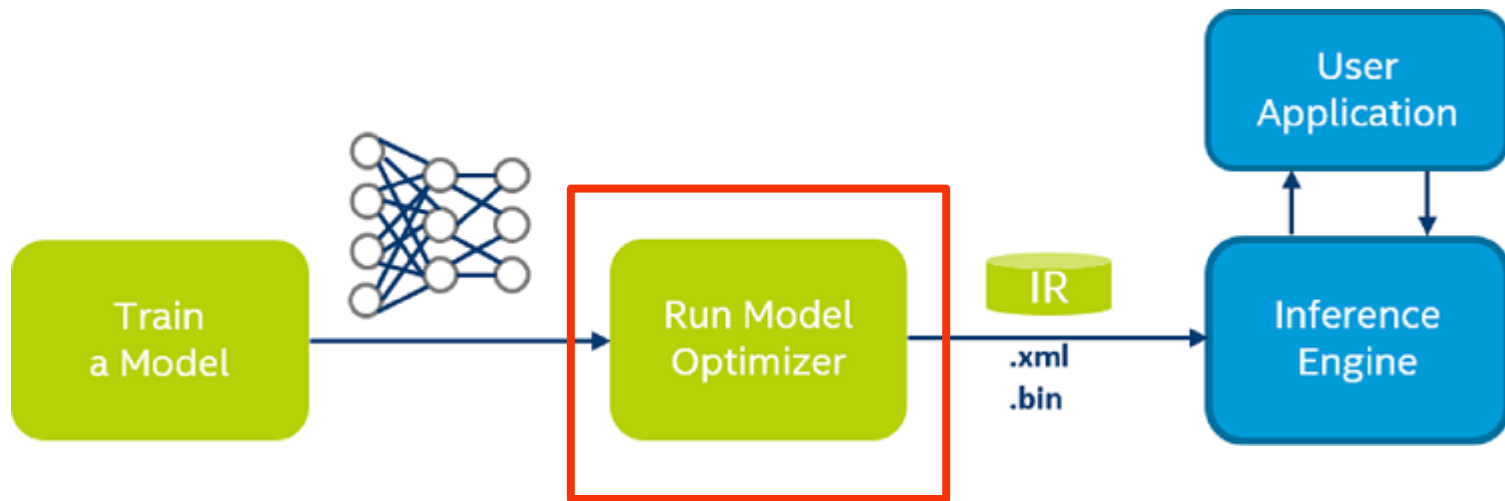
OpenVINO™



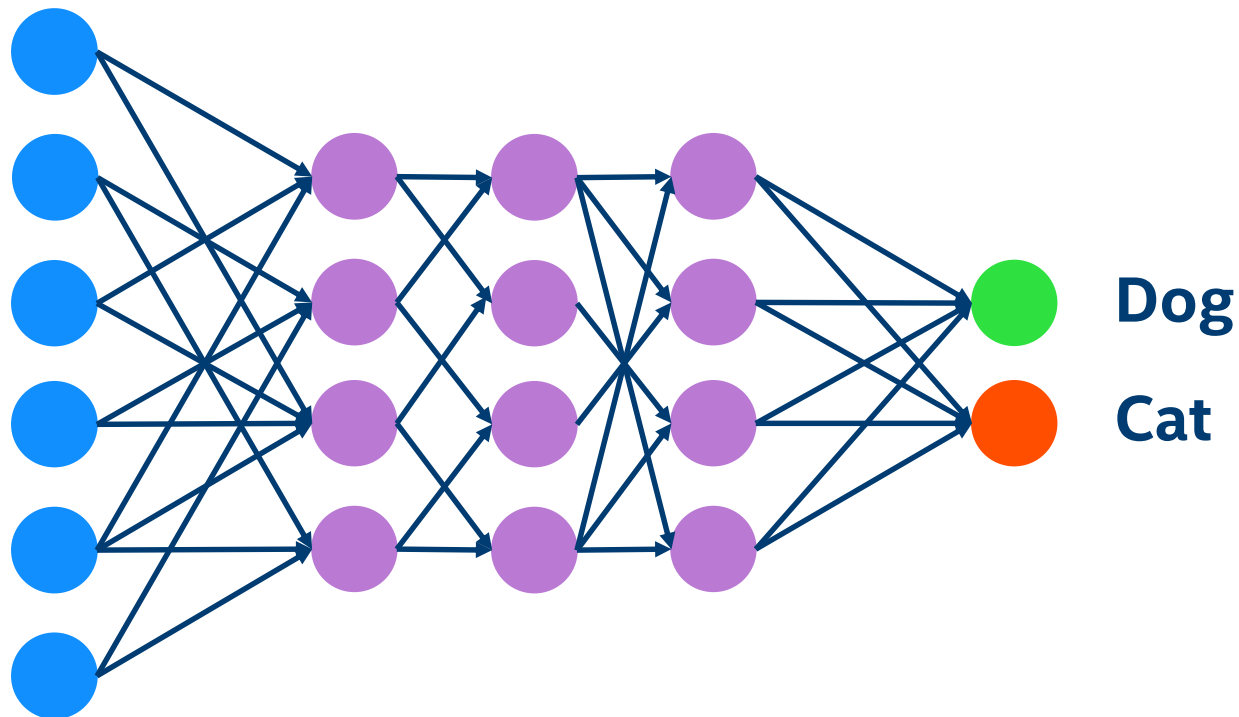
Model Optimizer



Model Optimizer

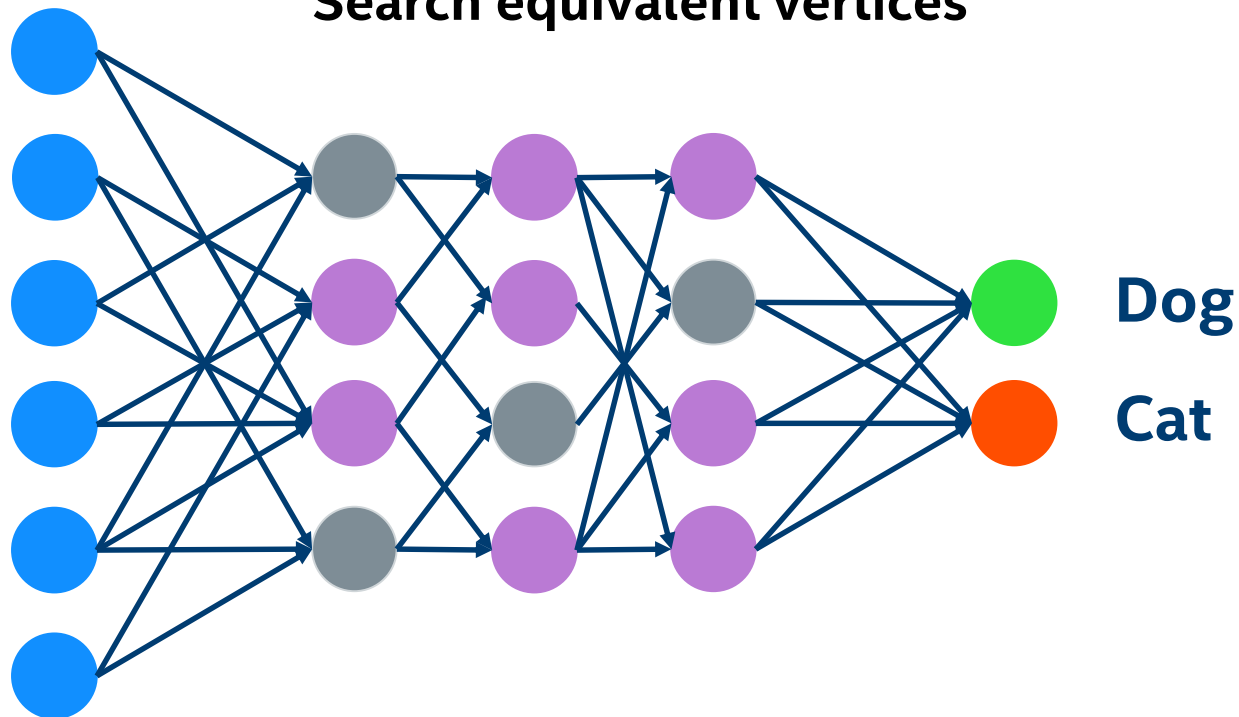


Model
Optimizer



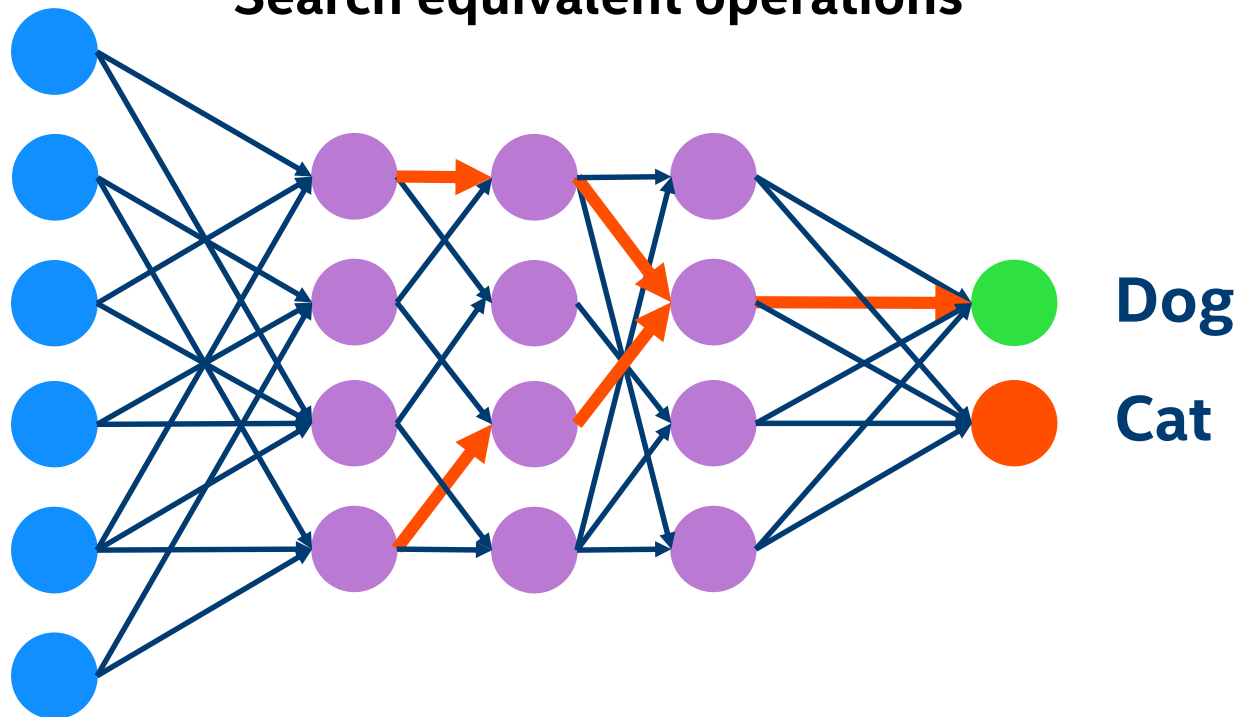
Model
Optimizer

Search equivalent vertices



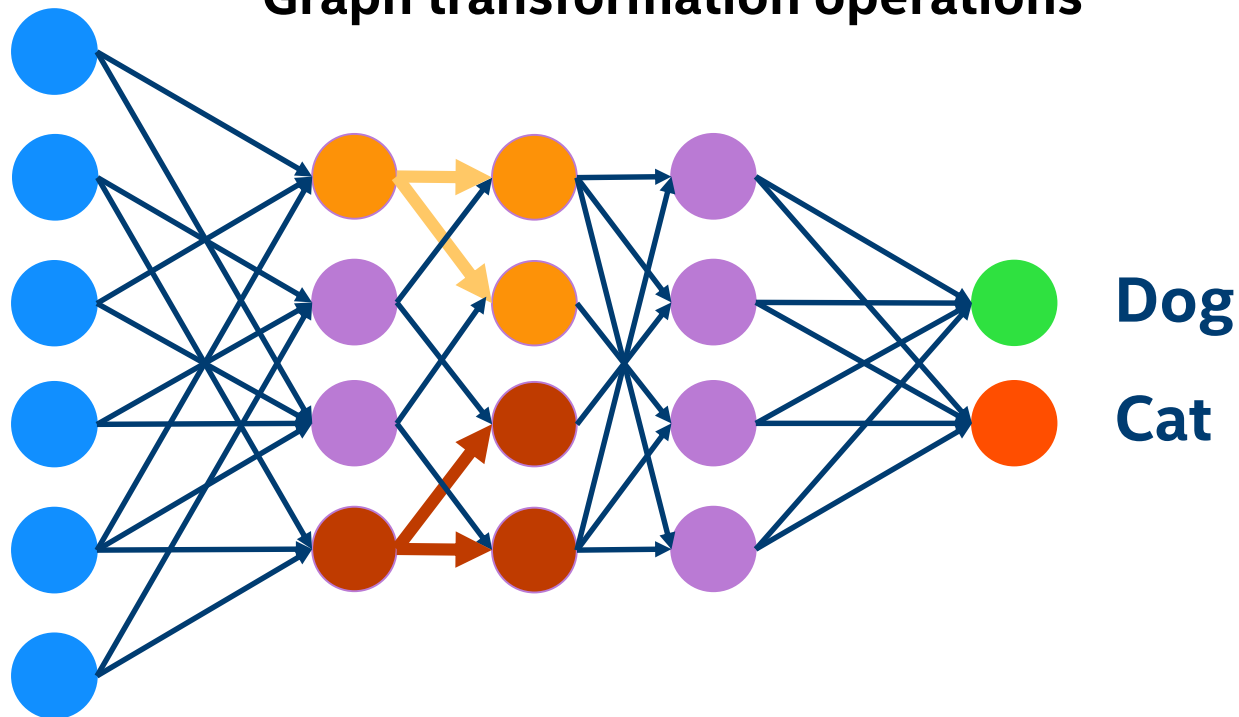
Model
Optimizer

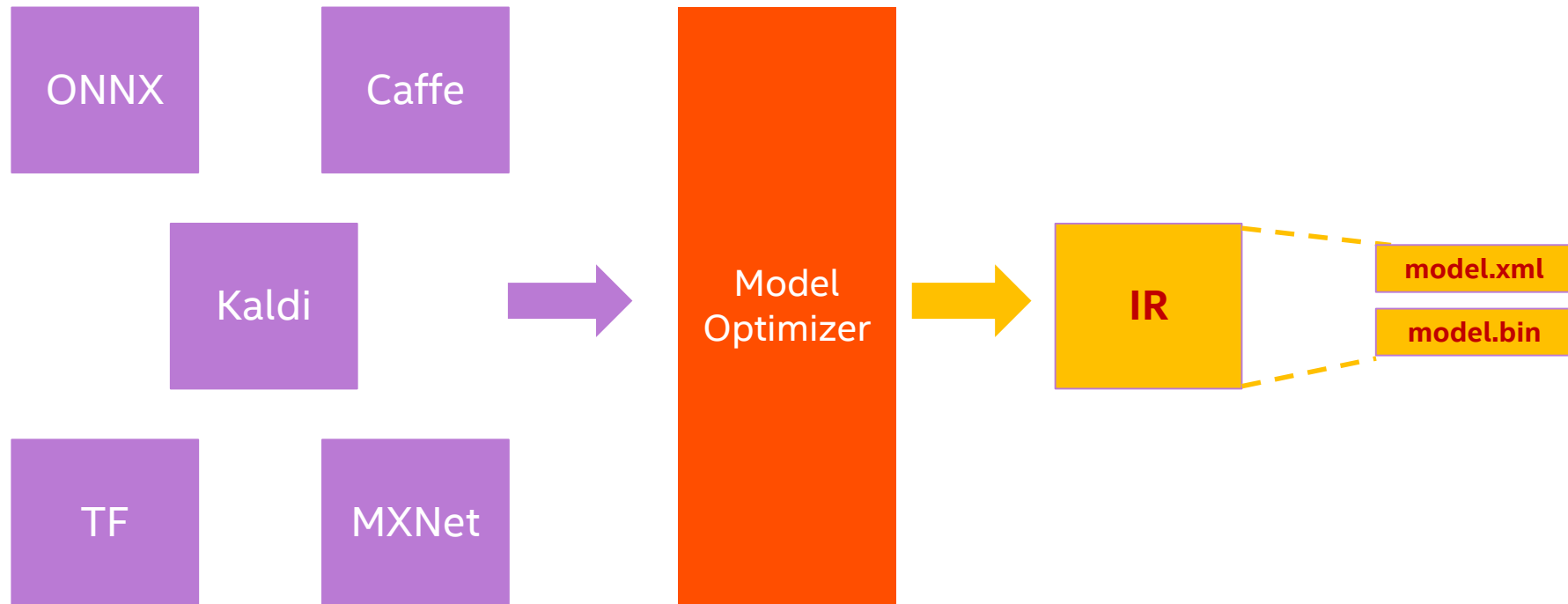
Search equivalent operations



Model
Optimizer

Graph transformation operations



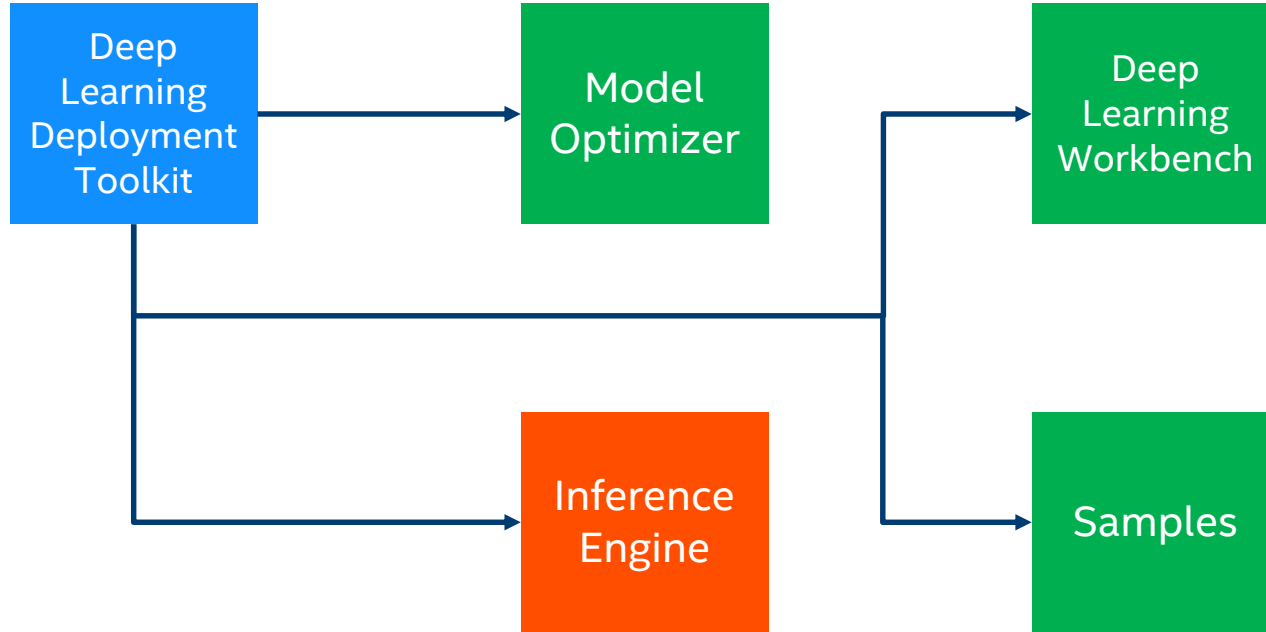



```
<INSTALL_DIR>/deployment_tools/model_optimizer
```

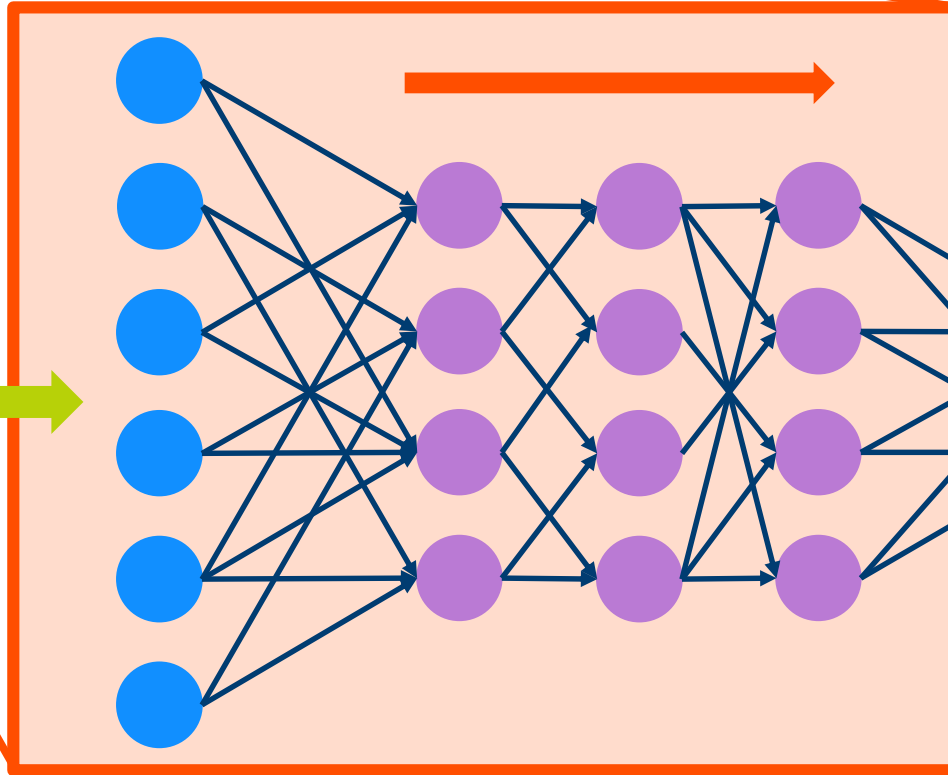
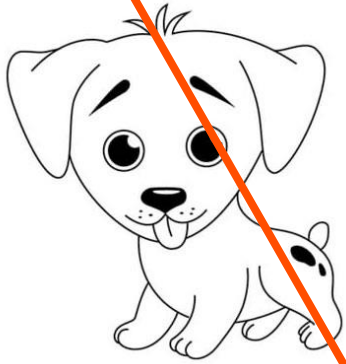
```
python3 mo.py --input_model INPUT_MODEL
```

```
python3 mo.py --framework tf --input_model /user/models/model.pb
```

OpenVINO™



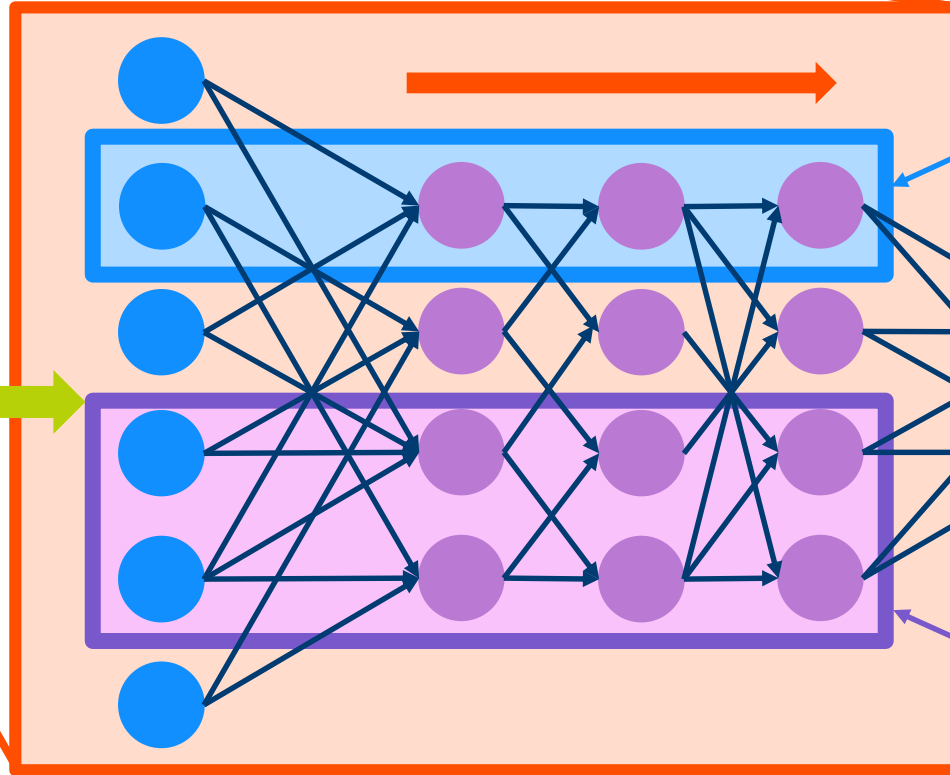
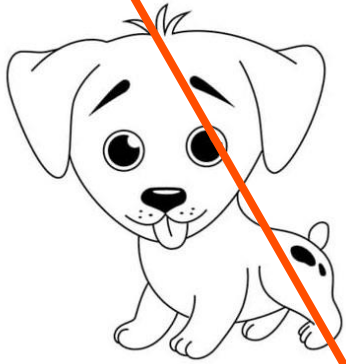
Inference
Engine
(Default)



Dog

Cat

Inference
Engine
(Hetero)



CPU

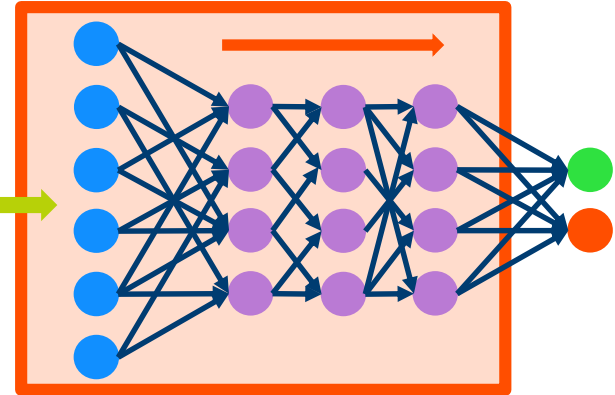
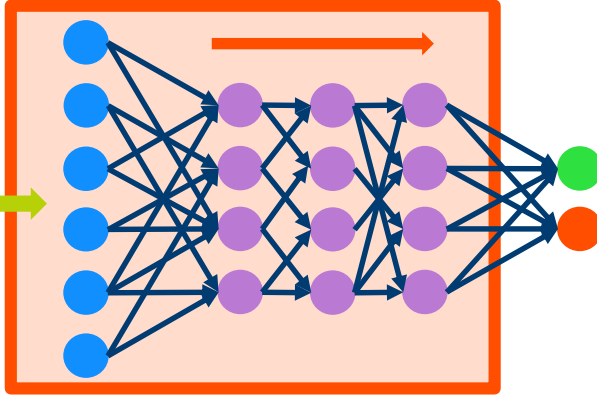
Dog

Cat

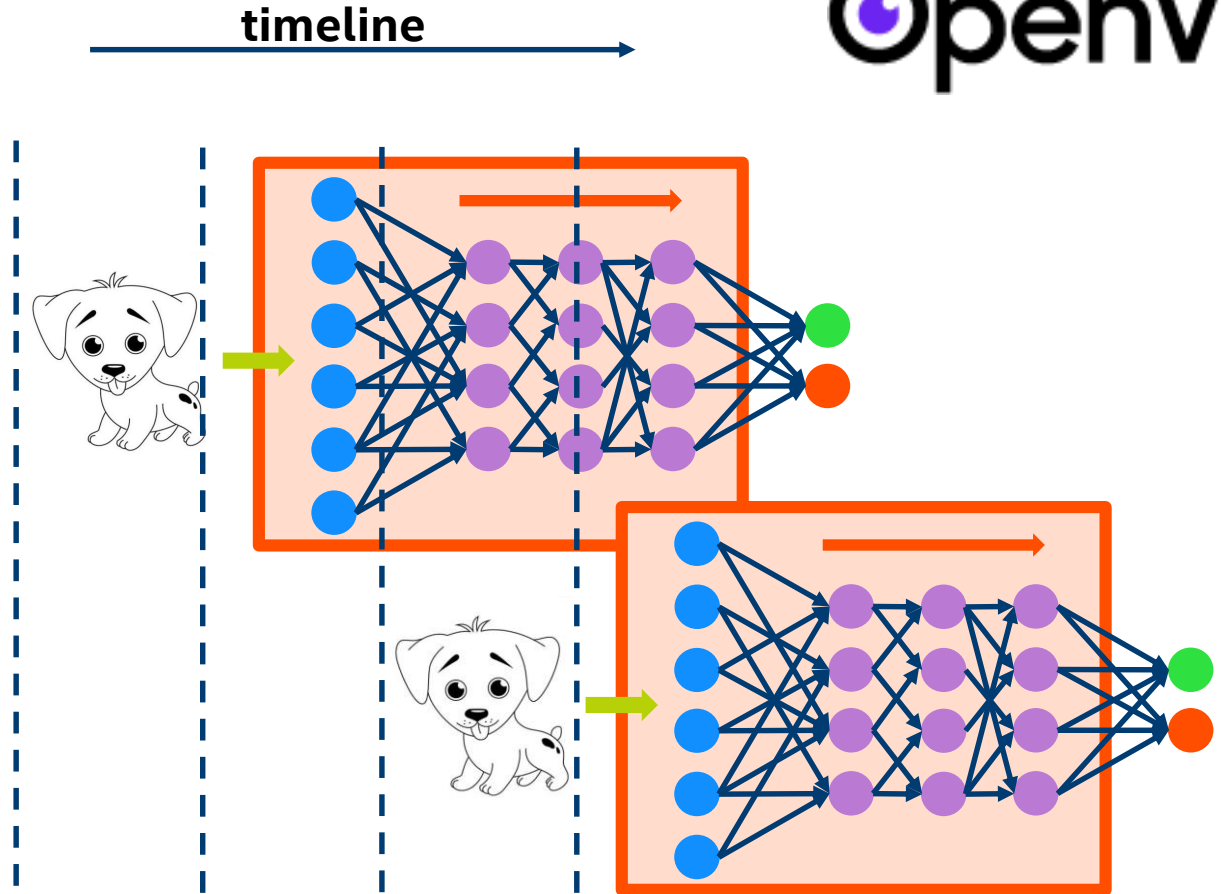
VPU

Inference
Engine
(Sync)

timeline



Inference
Engine
(Async)



```
#include <inference_engine.hpp>
using namespace InferenceEngine;

Core ie;
CNNNetwork network = ie.ReadNetwork(input_model, input_weights);

InputInfo::Ptr input_info = network.getInputsInfo().begin()->second;
std::string input_name = network.getInputsInfo().begin()->first;

input_info->getPreProcess().setResizeAlgorithm(RESIZE_BILINEAR);
input_info->setLayout(Layout::NHWC);
input_info->setPrecision(Precision::U8);
```

```
DataPtr output_info = network.getOutputsInfo().begin()->second;
std::string output_name = network.getOutputsInfo().begin()->first;
output_info->setPrecision(Precision::FP32);

ExecutableNetwork executable_network = ie.LoadNetwork(network, device_name);
InferRequest infer_request = executable_network.CreateInferRequest();
cv::Mat image = imread(input_image_path);
Blob::Ptr imgBlob = wrapMat2Blob(image);
infer_request.SetBlob(input_name, imgBlob);

infer_request.Infer();
Blob::Ptr output = infer_request.GetBlob(output_name);
```



```
from opencvino.inference_engine import IENetwork, IECore

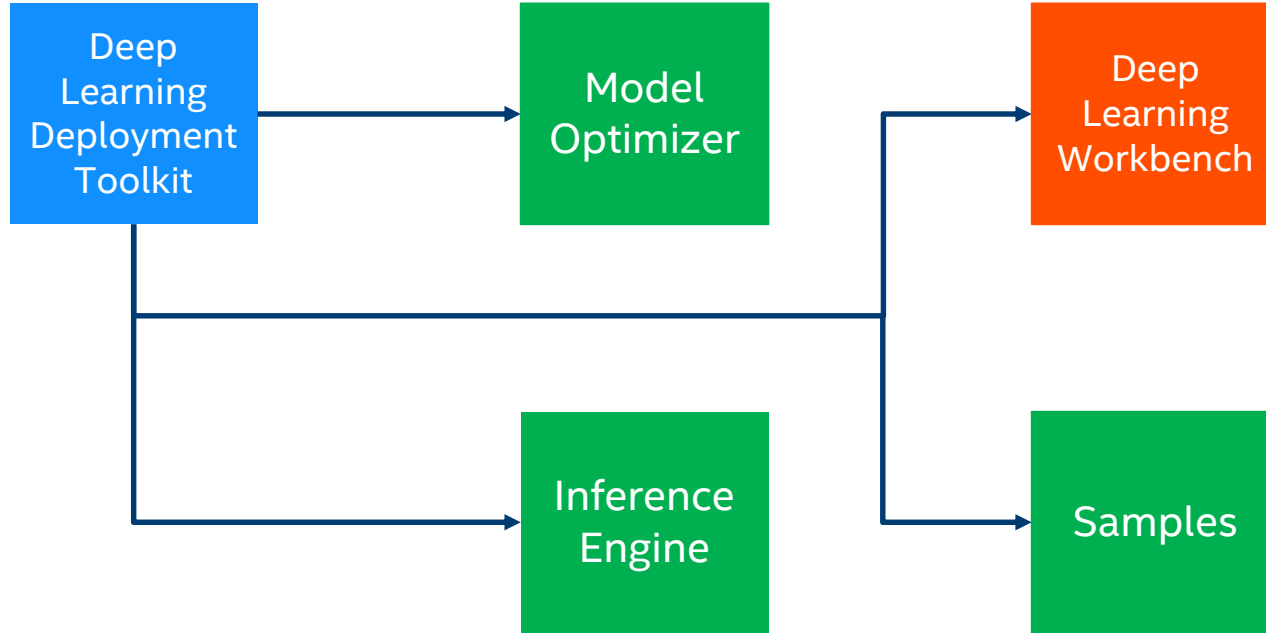
configPath = `path_to_model_config.xml`
weightsPath = `path_to_model_weights.bin`

ie = IECore()
net = IENetwork(model = configPath, weights = weightsPath)
exec_net = ie.load_network(network = net, device_name = 'CPU')

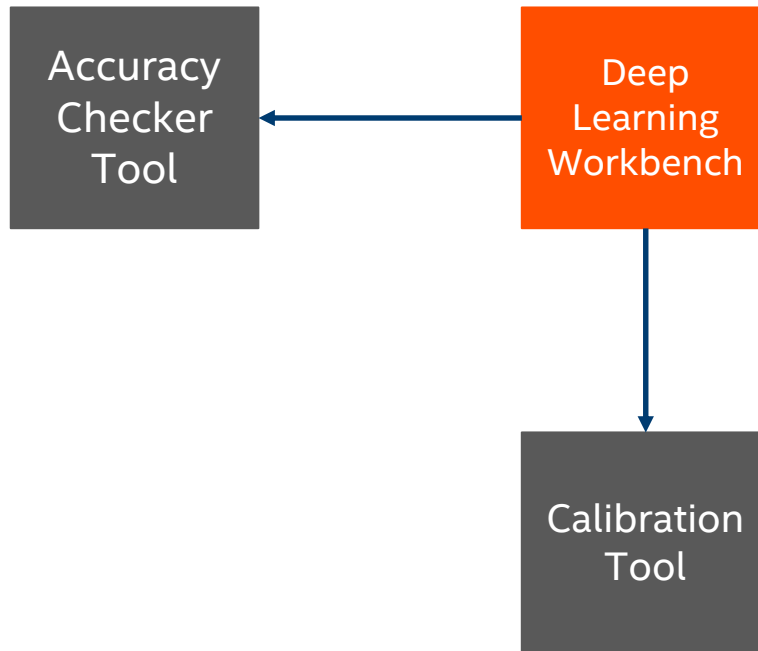
input_blob = next(iter(load_net.inputs))
output_blob = next(iter(load_net.outputs))

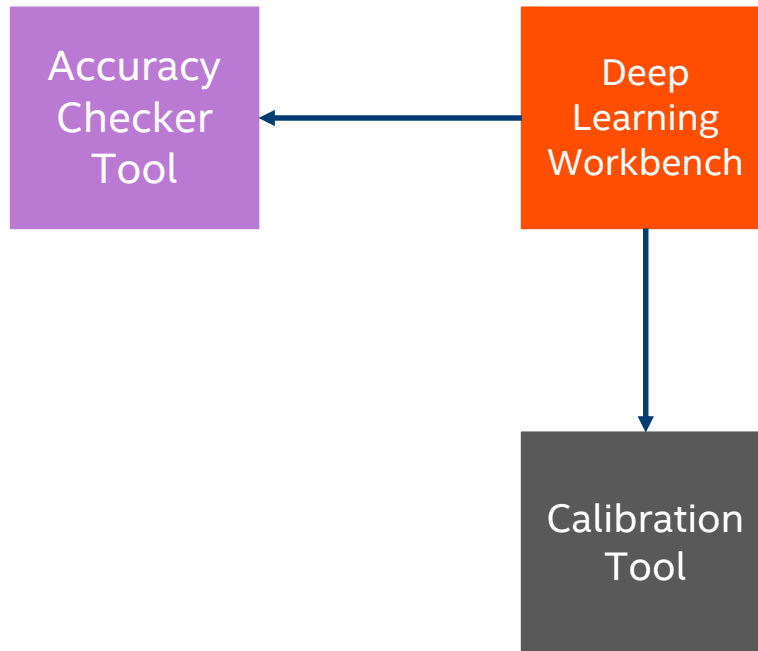
res = exec_net.infer(inputs={input_blob: images})
out = res[output_blob]
```

OpenVINO™

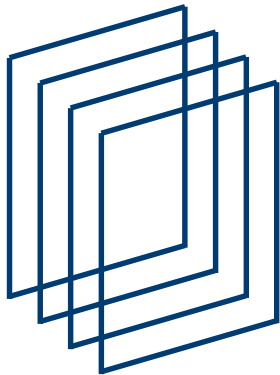


OpenVINO™

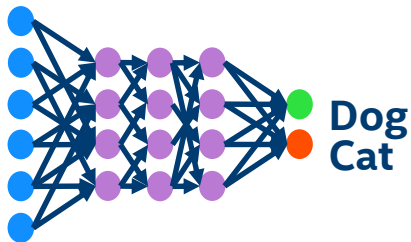




Dataset



Network



Accuracy
Checker
Tool

%

?

=

%

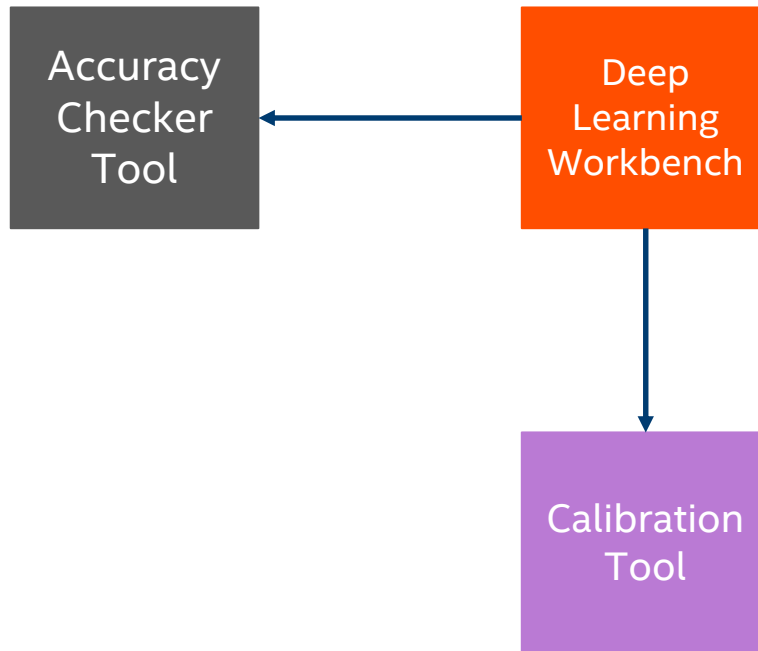
Article

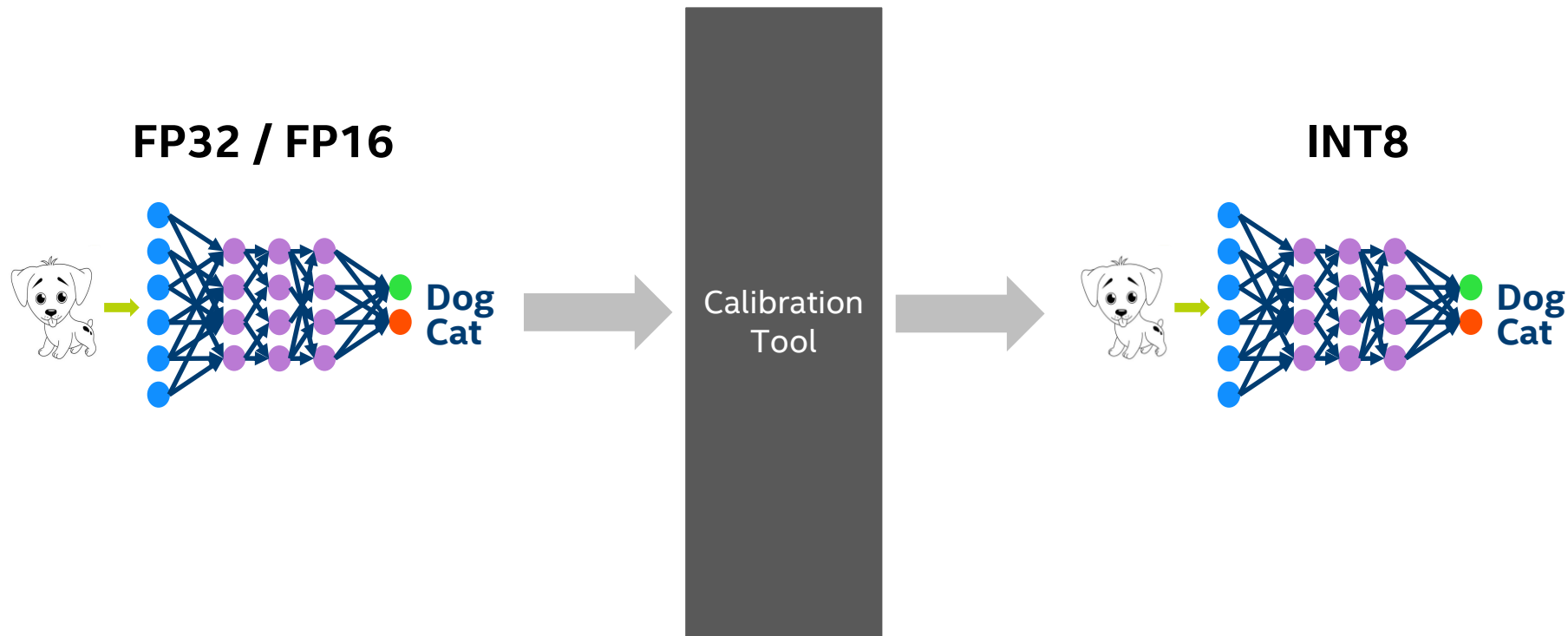




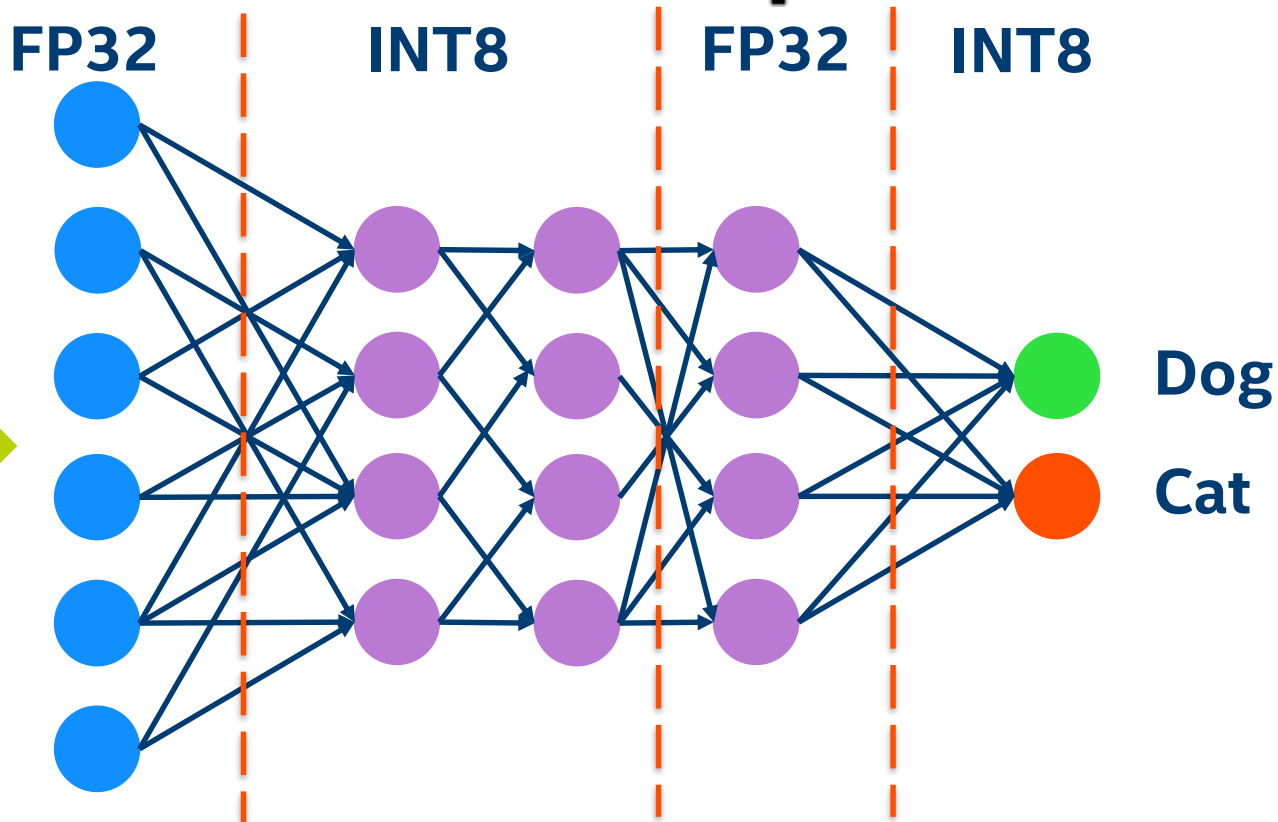
```
accuracy_check -c path/to/configuration_file \  
               -m /path/to/models           \  
               -s /path/to/source/data       \  
               -a /path/to/annotation
```

```
models:  
- name: model_name  
  launchers:  
    - framework: caffe  
      model: public/alexnet/caffe/bvlc_alexnet.prototxt  
      weights: public/alexnet/caffe/bvlc_alexnet.caffemodel  
      adapter: classification  
      batch: 128  
  datasets:  
    - name: dataset_name
```

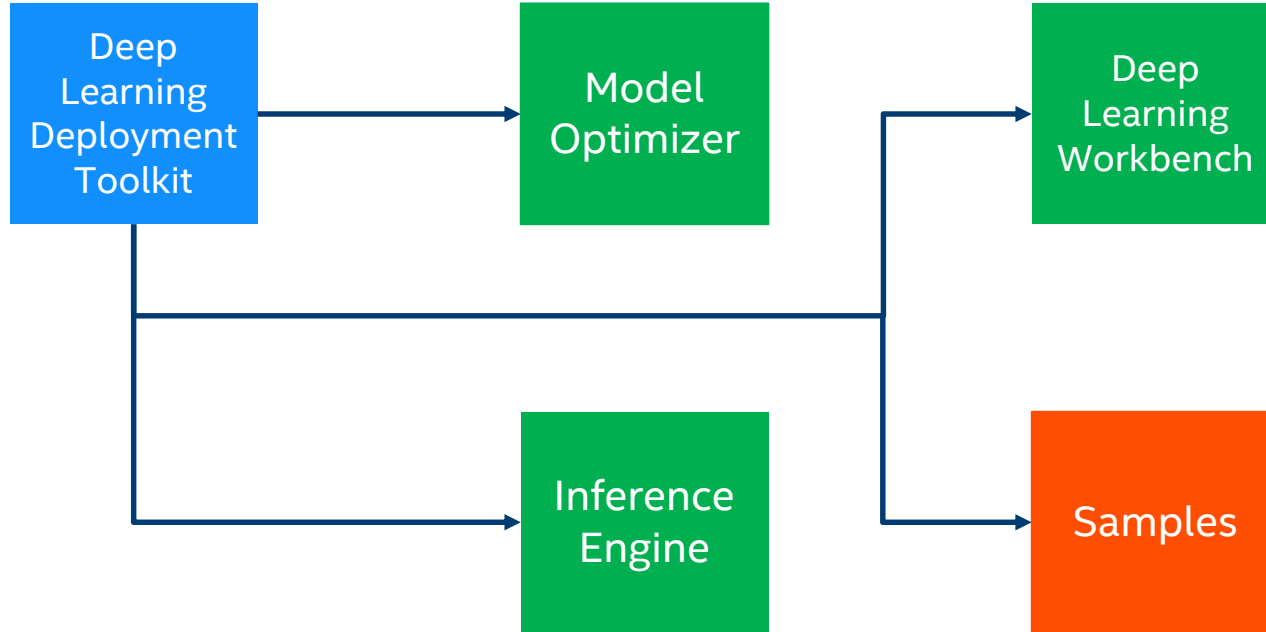




Calibration Tool

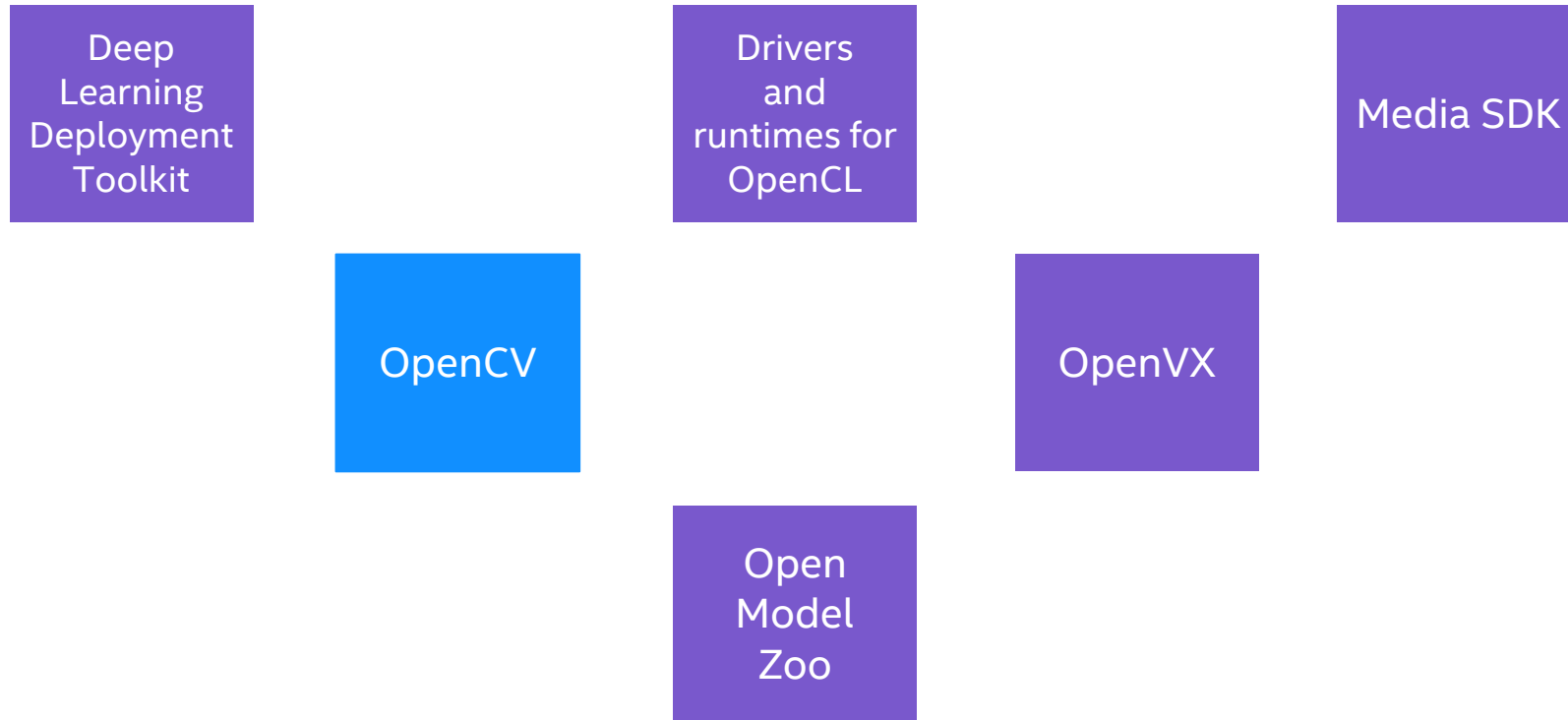


OpenVINO™



<https://docs.openvinotoolkit.org>

OpenVINO™





C/C++

Python

Java

Matlab

JavaScript

core

imgproc

imgcodecs

videoio

highgui

video

calib3d

features2d

objdetect

dnn

ml

flann

photo

stitching

gapi

tracking



```
import cv2
import numpy as np

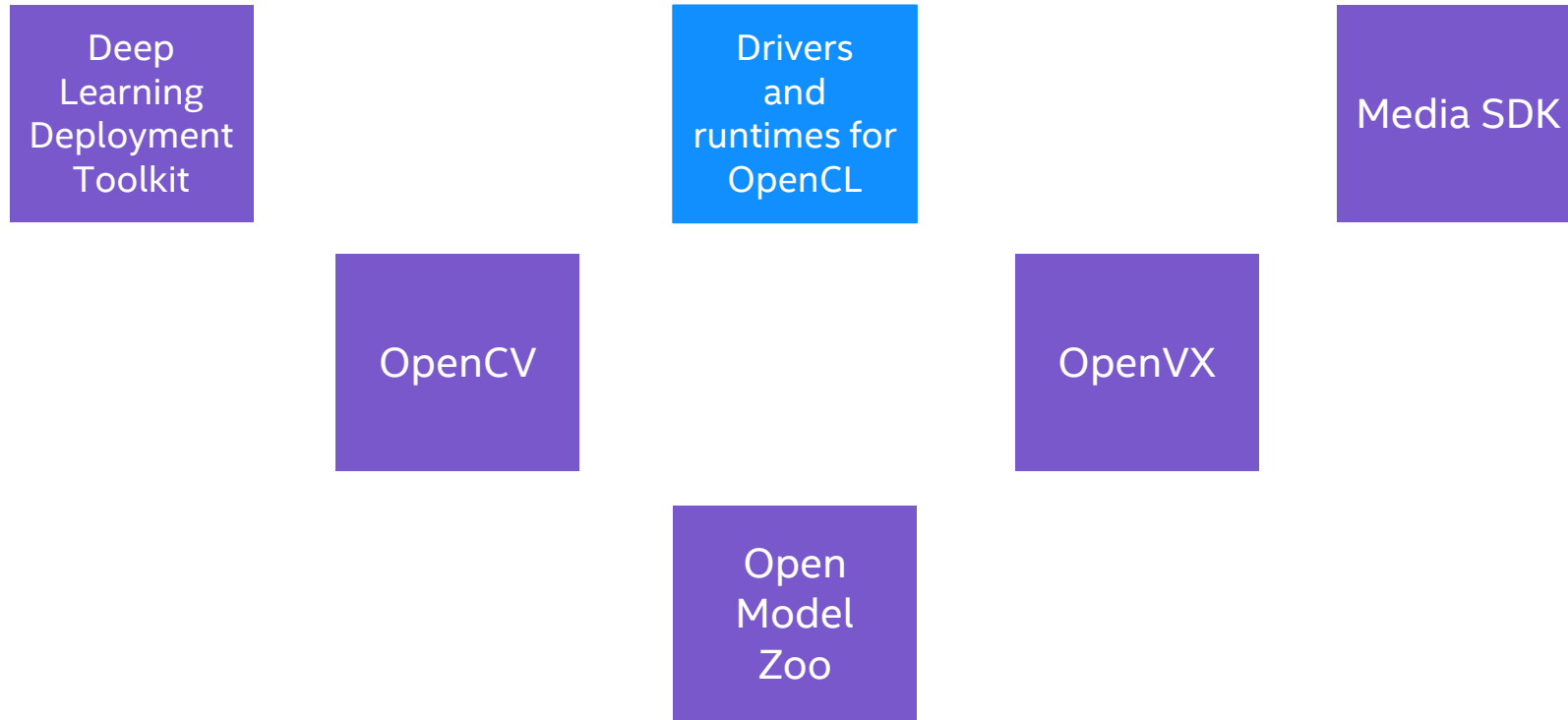
img = cv2.imread('watch.jpg',cv2.IMREAD_GRAYSCALE)
cv2.imwrite('watchgray.png',img)
```

```
#include <opencv2/opencv.hpp>
#include <iostream>
```

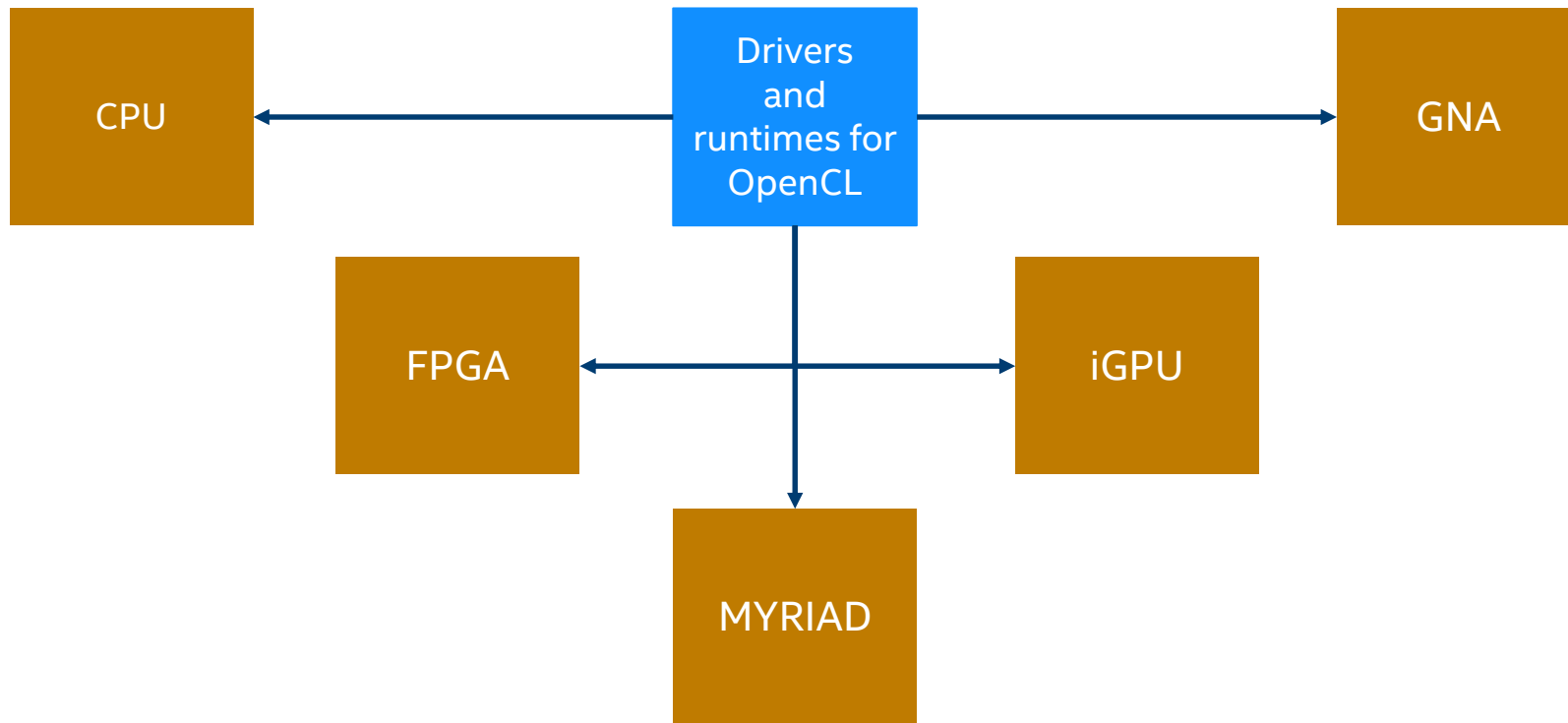
```
using namespace cv;
using namespace std;
```

```
int main(int argc, char** argv)
{
    Mat image = imread("D:/OCV/lo1.jpg");
    imwrite("C:/OCV/lo1.jpg", image)
    return 0;
}
```

OpenVINO™



OpenVINO™



OpenVINO™

Deep
Learning
Deployment
Toolkit

Drivers
and
runtimes for
OpenCL

Media SDK

OpenCV

OpenVX

Open
Model
Zoo

OpenVINO™

Deep
Learning
Deployment
Toolkit

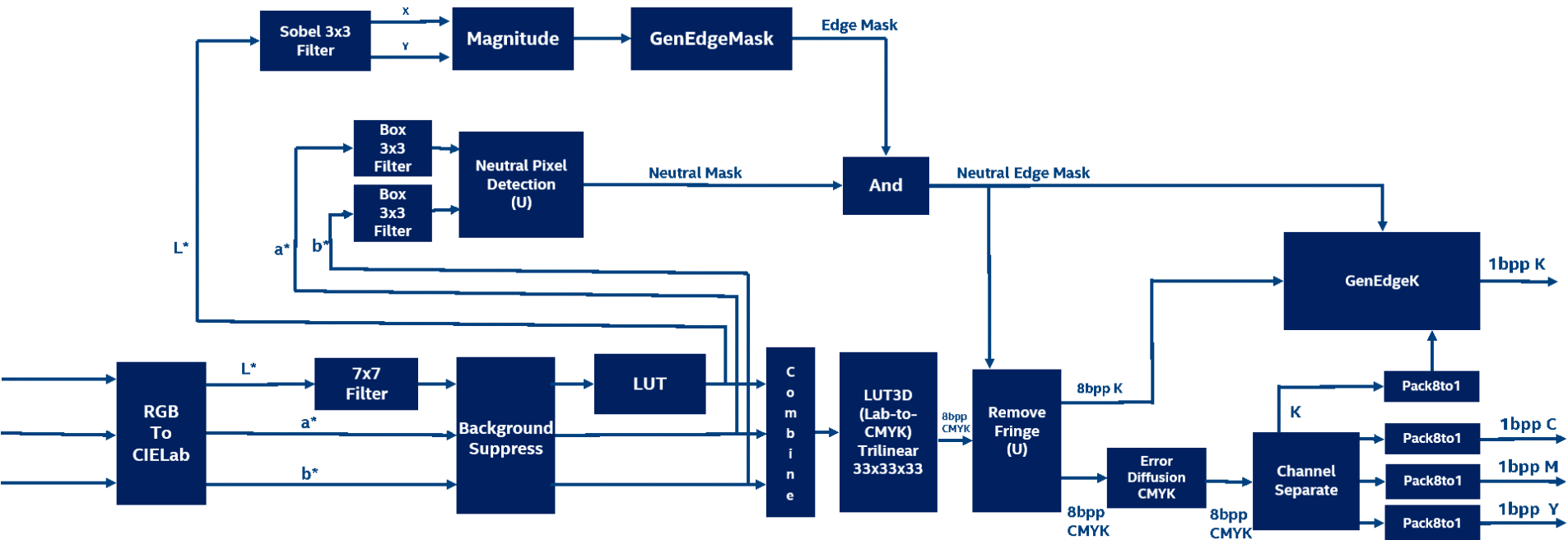
Drivers
and
runtimes for
OpenCL

Media SDK

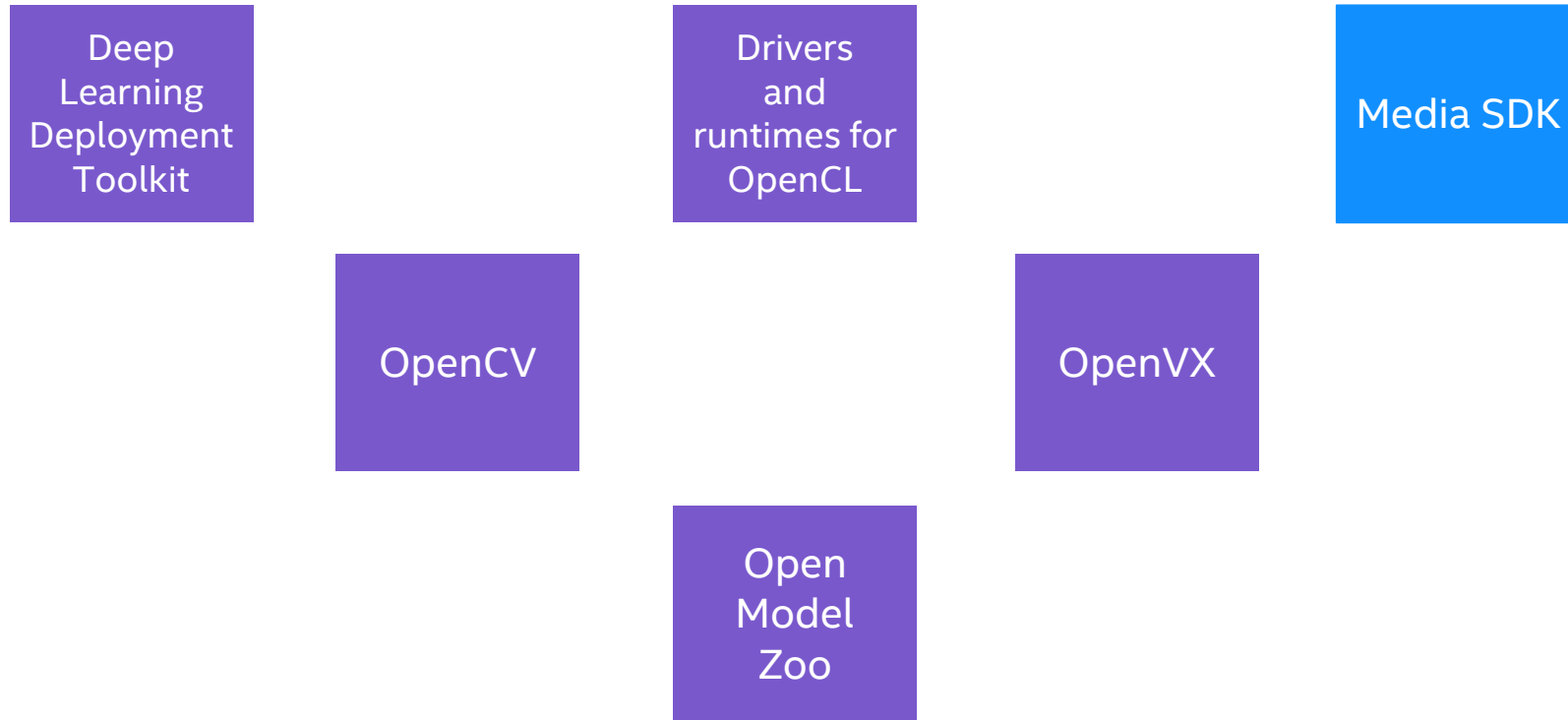
OpenCV
G-API

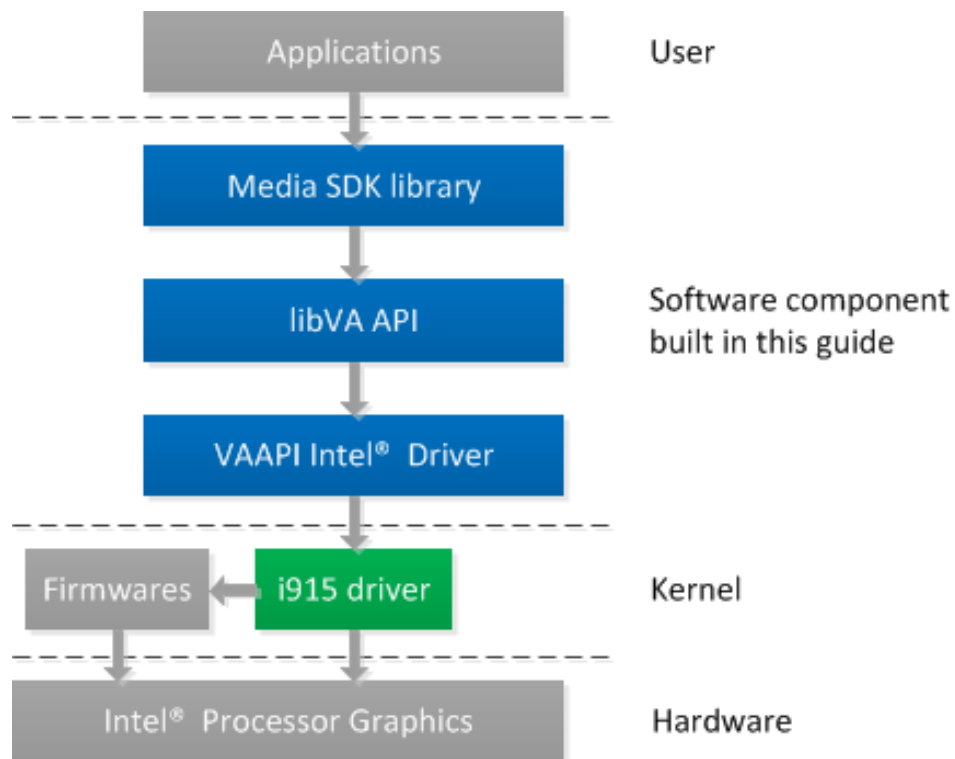
OpenVX

Open
Model
Zoo



OpenVINO™





OpenVINO™

Deep
Learning
Deployment
Toolkit

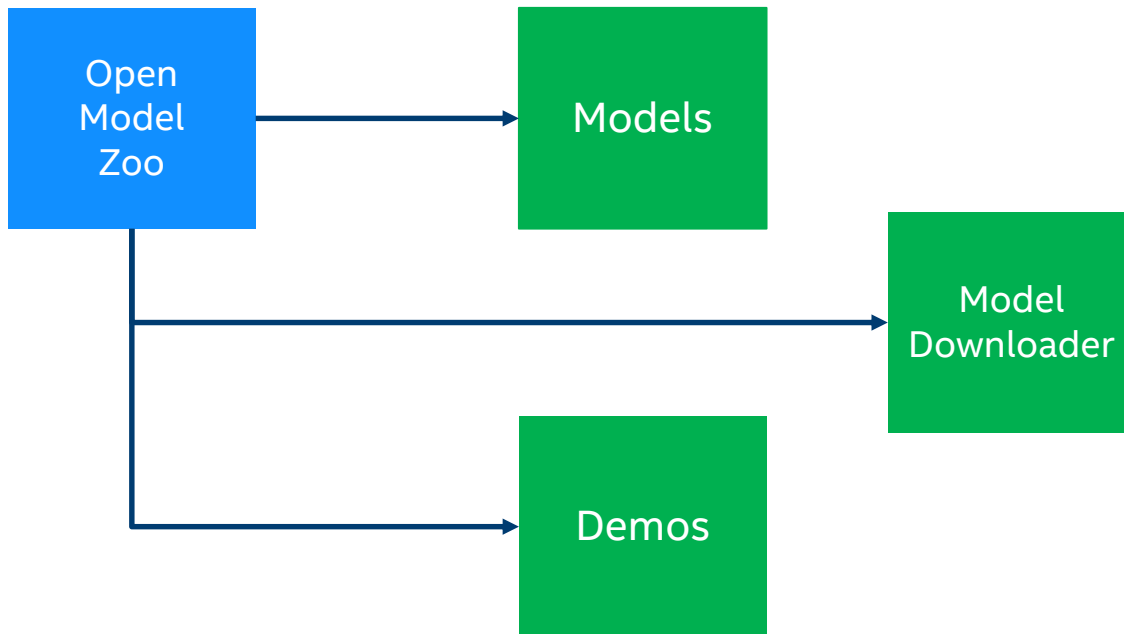
Drivers
and
runtimes for
OpenCL

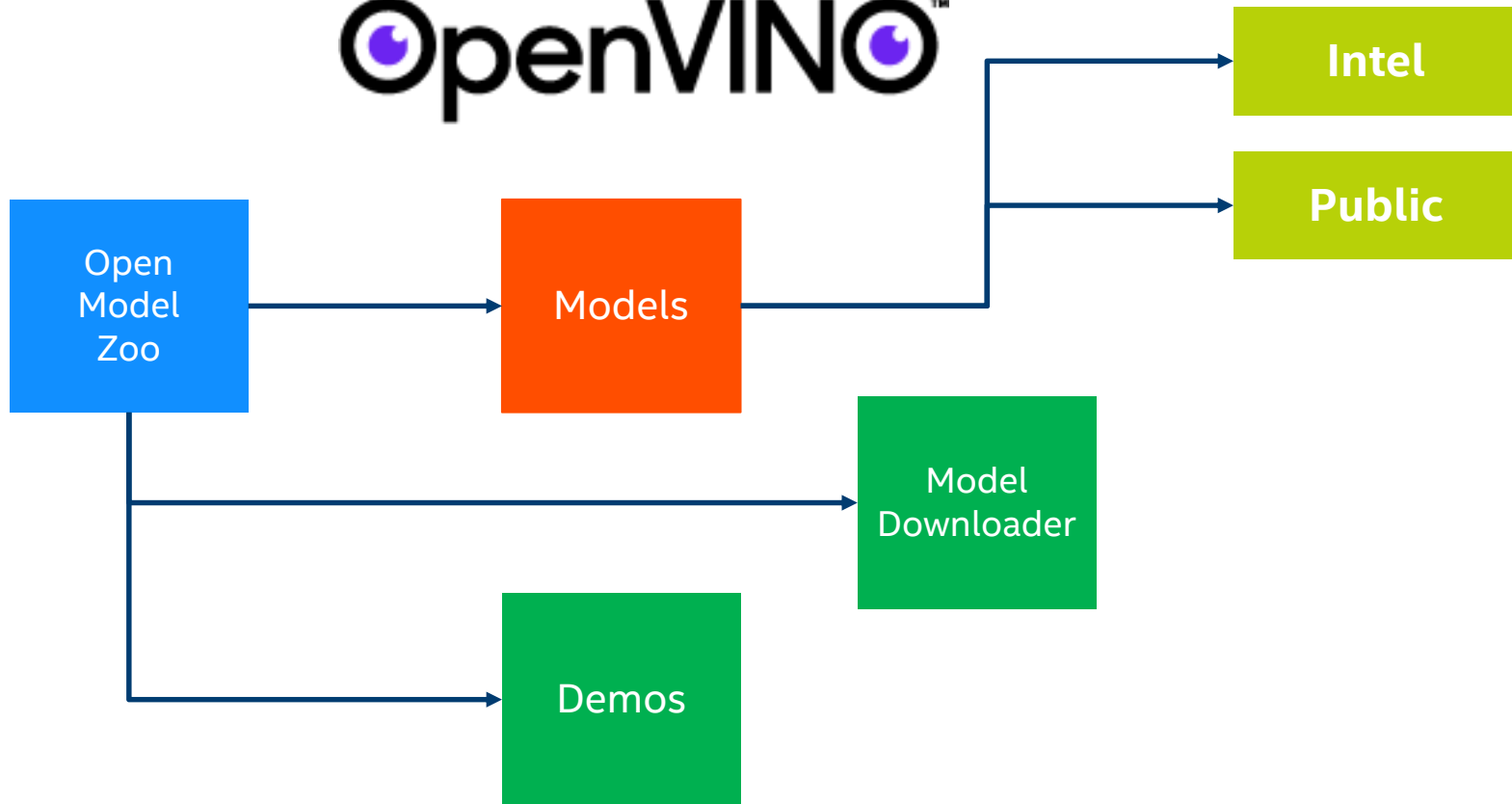
Media SDK

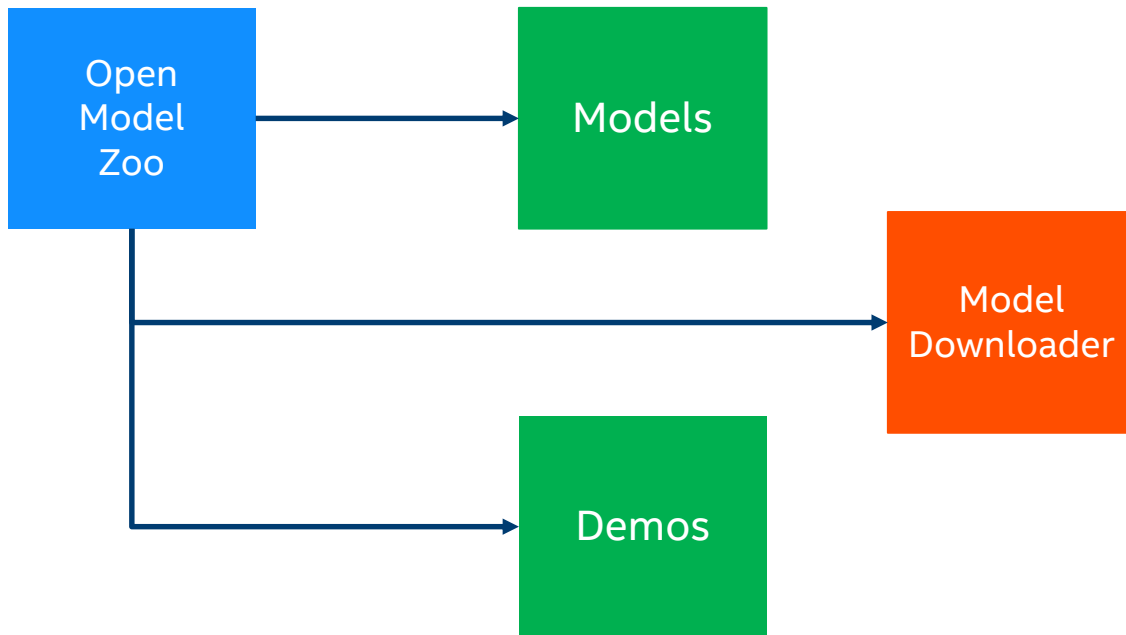
OpenCV

OpenVX

Open
Model
Zoo





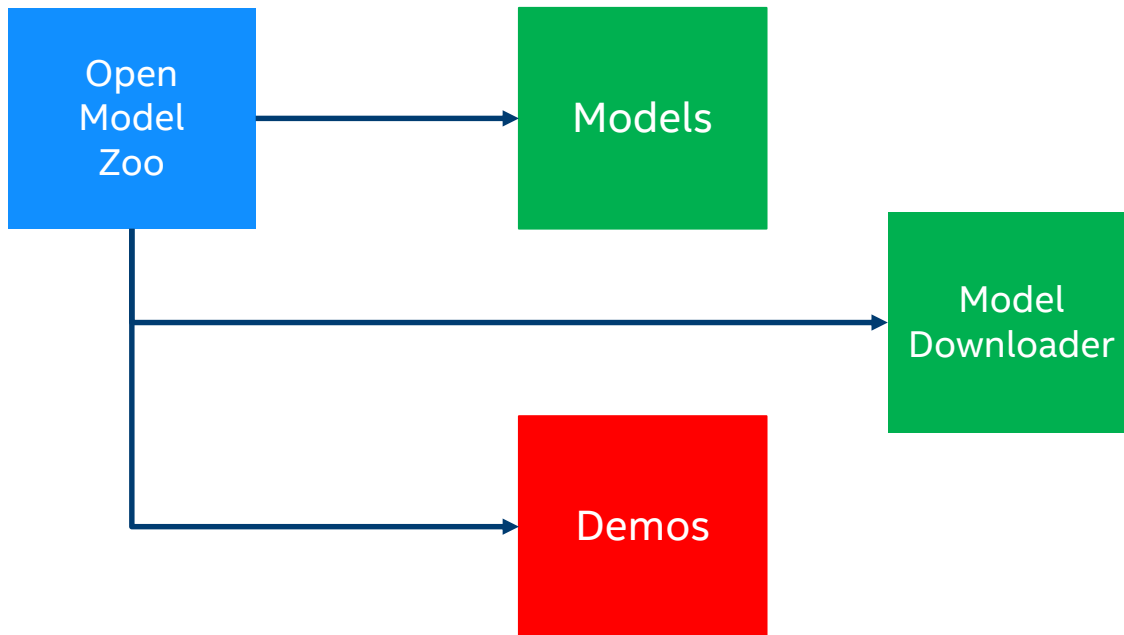




```
./downloader.py --all
```

```
./downloader.py --all --output_dir my/download/directory
```

```
./downloader.py --name face-detection-retail-0004 --precisions FP16,INT8
```



Demos



**Object
recognition**

Segmentation

**Image
processing**

**Action
recognition**

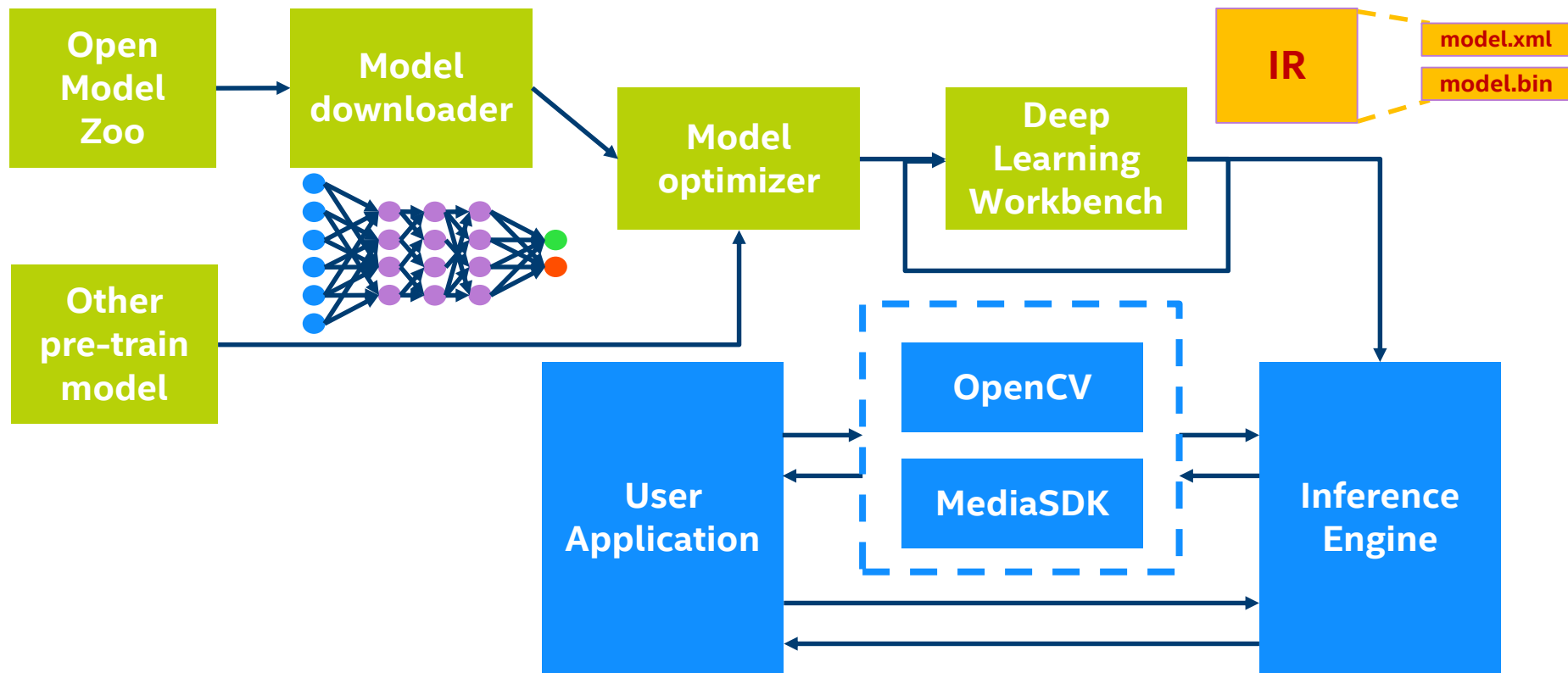
Tracking

**Text
processing**

**Object
detection**

**Audio
processing**

OpenVINO™ pipeline





Questions & Answers

Internet of Things Group