AI DEVCON 2018

intel AI DEVCON 2018

INFERENCE WITH INTEL:

HANDS ON WORKSHOP + FIRESIDE CHAT

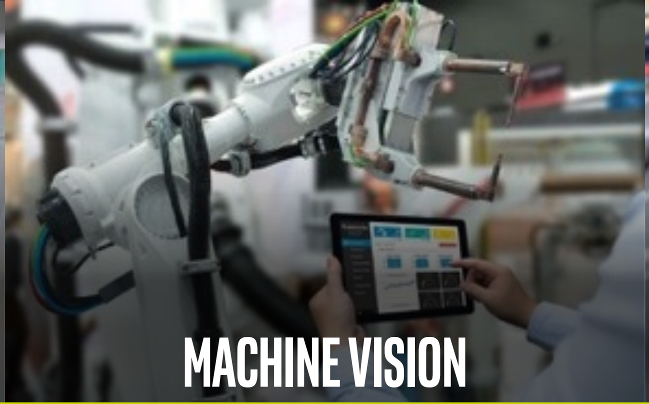
Yi Ge
Technical Consulting Engineer

Monique Jones
Technical Consulting Engineer

INTRODUCTION


EMERGENCY RESPONSE


FINANCIAL SERVICES


MACHINE VISION


CITIES/TRANSPORTATION

VIDEO: THE "EYE OF IOT"

USE OF VIDEO, COMPUTER VISION, AND DEEP LEARNING IS GROWING RAPIDLY


AUTONOMOUS VEHICLES


RESPONSIVE RETAIL


MANUFACTURING

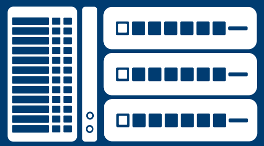

PUBLIC SECTOR

INTEL® IOT VIDEO PORTFOLIO

SMART CAMERAS

ATOM x5 · ATOM x7 · ATOM x5

VIDEO GATEWAYS / NVRs



ATOM x · ATOM x · CORE i3 · CORE i7 · CORE i5 · XEON

DATA CENTER / CLOUD

XEON · XEON PHI · nervana

FPGA SOLUTIONS FROM INTEL

+

Movidius
an Intel company

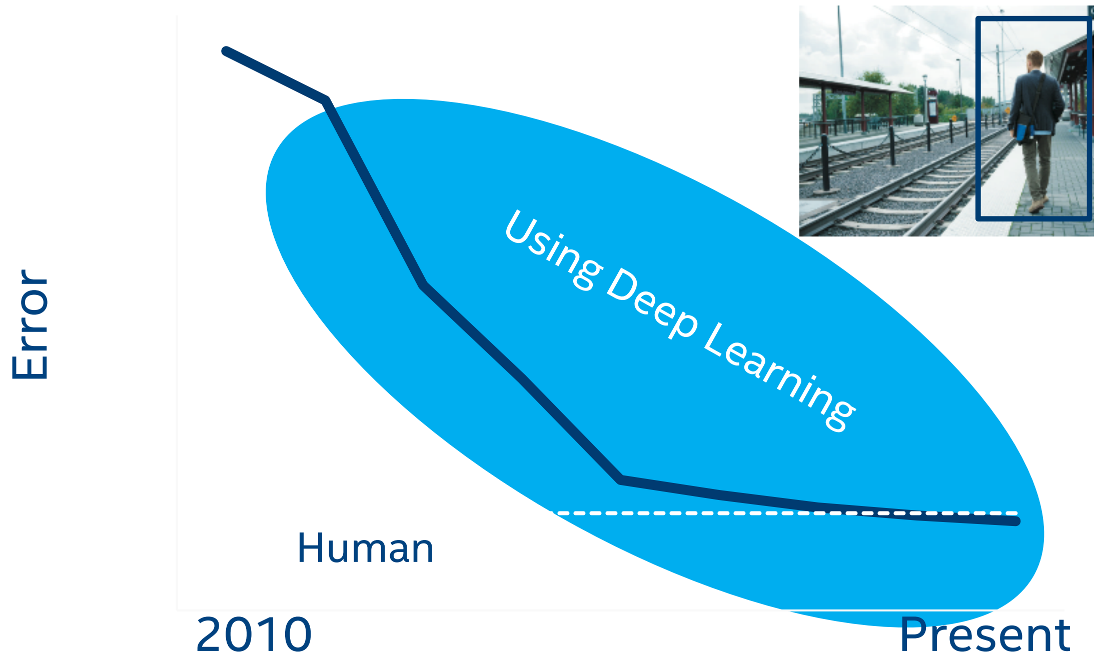
INTEL® MEDIA SDK, INTEL OPENVINO™ TOOLKIT

INDUSTRY'S BROADEST MEDIA, COMPUTER VISION, AND DEEP LEARNING PORTFOLIO

DEEP LEARNING USAGE IS INCREASING

Deep Learning Revenue Is Estimated to Grow from $655M in 2016 to **$35B** by 2025[1]

IMAGE RECOGNITION

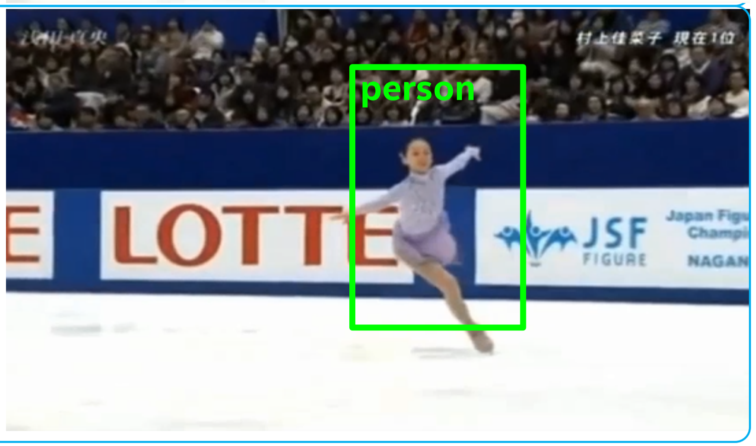


Traditional Computer Vision

Object Detection

Deep Learning Computer Vision

Person Recognition



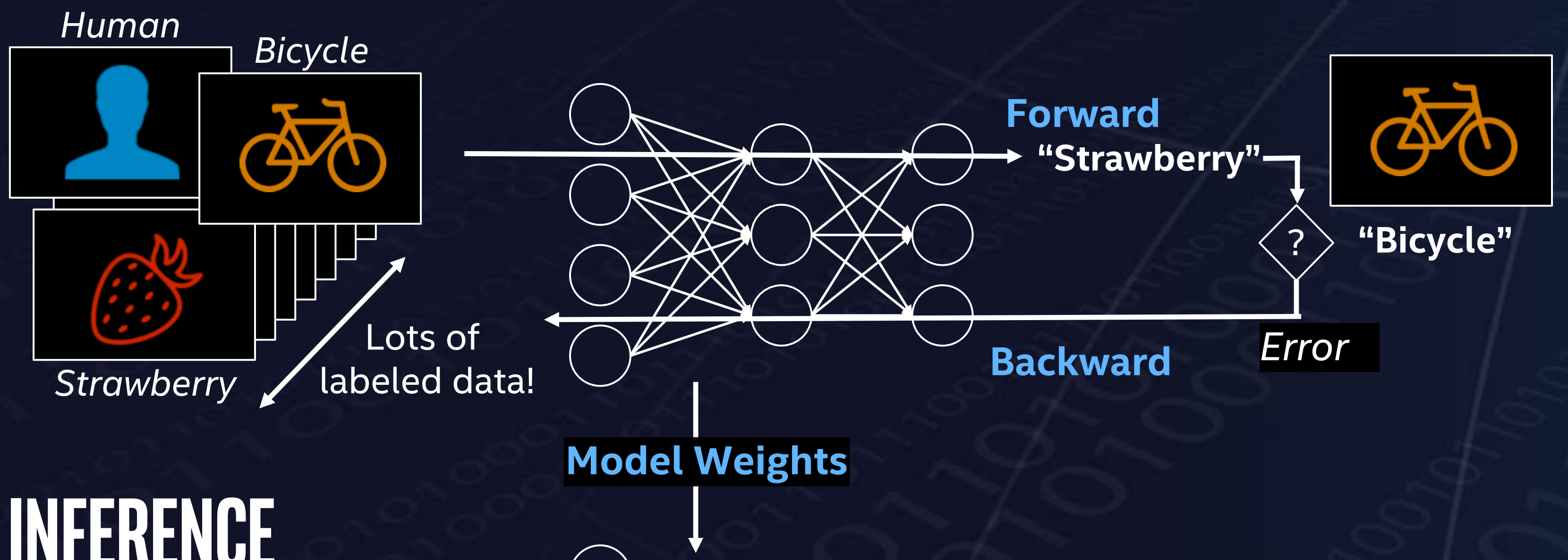
Market Opportunities + Advanced Technologies Have Accelerated Deep Learning Adoption

[1]Tractica 2Q 2017

intel AI

DEEP LEARNING: TRAINING VS. INFERENCE

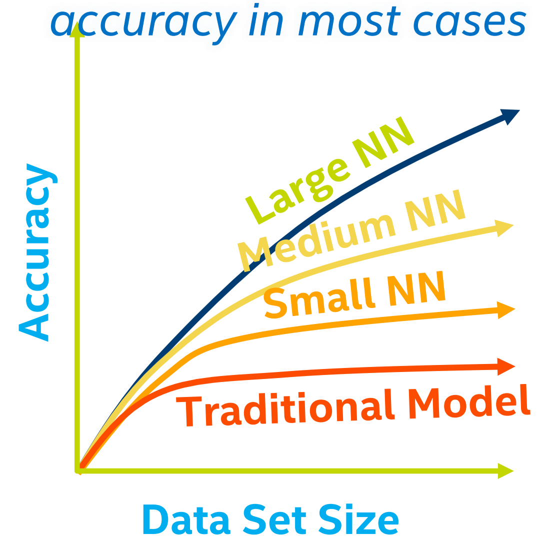
TRAINING



INFERENCE

NOTE

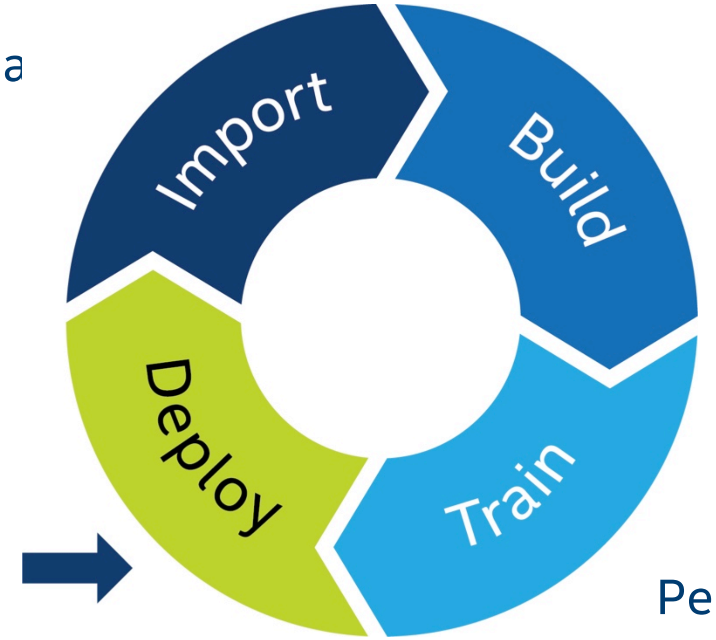
Training requires a very large data set and deep neural networks (NN) (i.e. many layers) to achieve the highest accuracy in most cases



ARTIFICIAL INTELLIGENCE DEVELOPMENT CYCLE

Dataset Aquisition a
Organization

Create Models



Today's focus:

Integrate Trained Models with Application Code

Adjust Models to Meet Performance and Accuracy Objectives

Intel® Deep Learning Deployment Toolkit: Deploy Optimized Inference from Edge to Cloud

WHAT'S INSIDE

TensorFlow
Caffe
mxnet

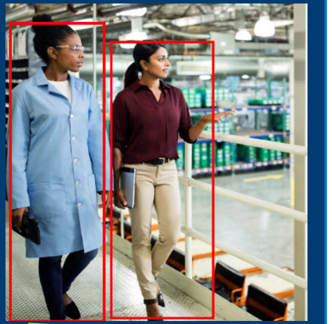
Trained Models

INTEL® DEEP LEARNING DEPLOYMENT TOOLKIT

Model Optimizer
Convert & Optimize

IR

Inference Engine
Optimized Inference



CPU | GPU | FPGA | VPU

COMPONENT TOOLS

Traditional Computer Vision – All SDK versions
Optimized Computer Vision Libraries

- OpenCV*
- OpenVX*

Increase Processor Graphics Performance – Linux* Only

- Intel® Media SDK (Open-Source Version)
- OpenCL™ Intel® Integrated Graphics Drivers & Runtimes

Linux for FPGA only

- FPGA RunTime Environment (RTE) (from Intel FPGA SDK for OpenCL™)
- Bitstreams

GPU = CPU with Intel® Integrated Graphics Processing Unit
VPU = Movidius™ Vision Processing Unit

INTEL® OPENVINO™ TOOLKIT PERFORMANCE

Public Models	Batch Size	OpenCV* Optimized (non-Intel)	Intel OpenVINO™ on CPU	Intel OpenVINO with Floating Point 16 (FP16)[1]	Intel OpenVINO on Intel Arria® 10 - 1150GX FPGA
		Optimize It	Use Intel® Tools	Or offload to Intel Iris™ Pro Graphics	Or offload to Intel FPGA
Squeezenet* 1.1	1	431%	425%	564%	1,623%
Vgg16*	1	174%	549%	295%	435%
GoogLeNet* v1	1	330%	577%	448%	1,619%
SSD 300*	1	185%	448%	248%	819%
Squeezenet* 1.1	32	466%	759%	663%	2,016%
Vgg16*	32	188%	434%	321%	791%
GoogLeNet* v1	32	385%	715%	497%	1,895%

Baseline Caffe* Framework - Out of Box = 100%

Intel OpenVINO Toolkit Accelerates Performance of Deep Learning Models Running on Intel Hardware

Get Faster Results with Less Work

Benchmark results were obtained prior to implementation of recent software patches and firmware updates intended to address exploits referred to as "Spectre" and "Meltdown". Implementation of these updates may make these results inapplicable to your device or system. For more complete information about performance and benchmark results, visit www.intel.com/benchmarks.

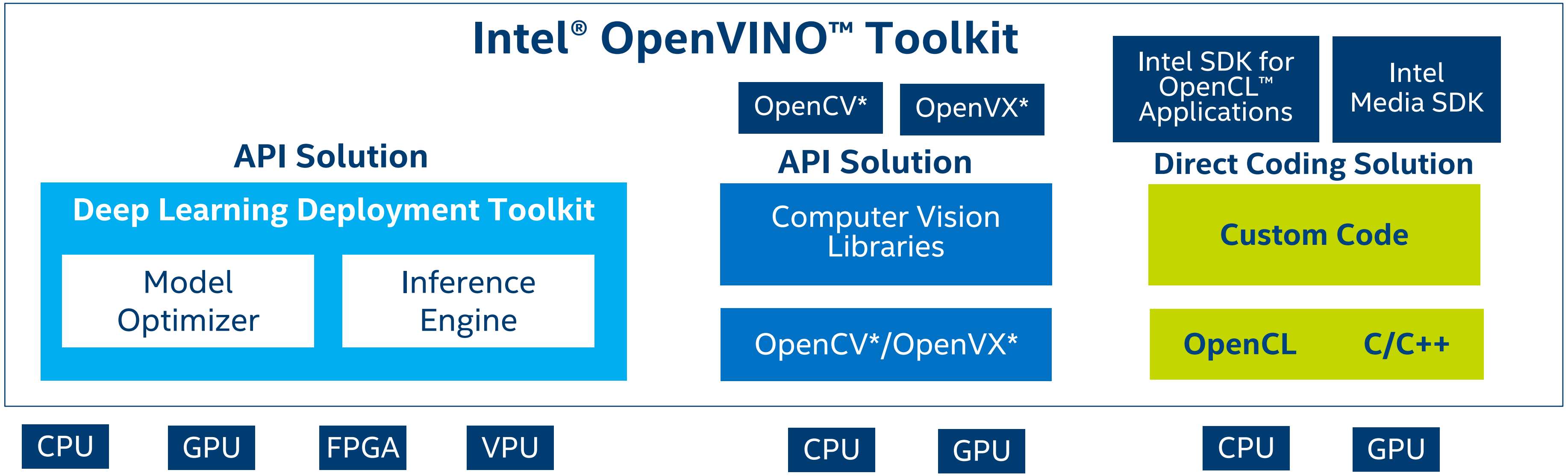
Configuration: Intel® Core™ i7-6700K CPU @ 2.90GHz fixed, GPU GT2 @ 1.00GHz fixed Internal ONLY testing, performed 4/10/2018 Test v312.30, Ubuntu* 16.04, Intel® OpenVINO toolkit 2018 RC4. Tests were based on various parameters such as model used (these are public), batch size, and other factors. Different models can be accelerated with different Intel hardware solutions, yet use the same Intel software tools! Benchmark Source Intel Corporation.

DEEP LEARNING VS. TRADITIONAL COMPUTER VISION

INTEL® OPENVINO™ TOOLKIT HAS TOOLS FOR AN END-TO-END VISION PIPELINE

TensorFlow mxnet Caffe

Pre-Trained Optimized Deep Learning Models



Intel Hardware Abstraction Layer

Deep Learning Computer Vision

- Based on application of a large number of filters to an image to extract features.
- Features in the object(s) are analyzed with the goal of associating each input image with an output node for each type of object.
- Values are assigned to output node representing the probability that the image is the object associated with the output node.

Traditional Computer Vision

- Based on selection and connections of computational filters to abstract key features and correlating them to an object
- Works well with well-defined objects and controlled scenes
- Difficult to predict critical features in larger number of objects or varying scenes

MODEL OPTIMIZER

Train

Train a deep learning model (out of our scope)

Currently supporting:
- Caffe*
- MXNet*
- TensorFlow*

Prepare Optimizer

Model Optimizer
- Converting
- Optimizing
- Preparing to inference

(device agnostic, generic optimization)

Inference

Inference-Engine
a lightweight application programming interface (API) to use in your application for inference

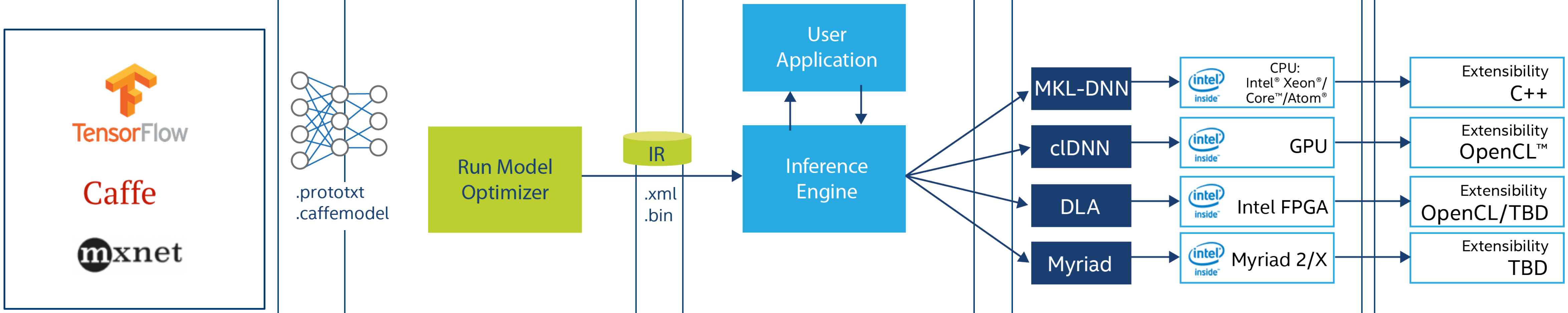
Optimize/ Hetero

Inference-Engine
Supports multiple devices for heterogeneous flows

Device-level optimization

Extend

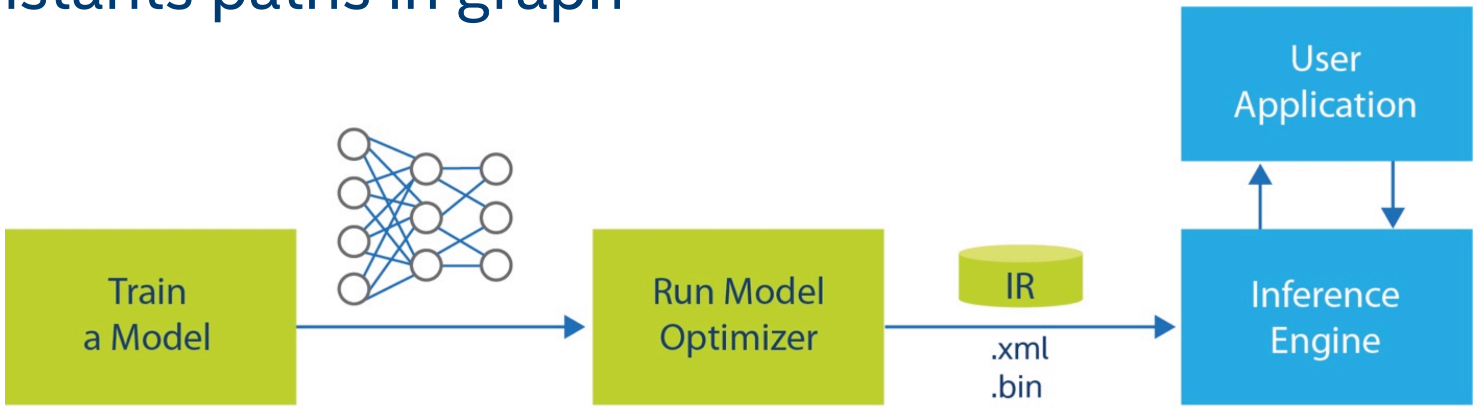
Inference-Engine
Supports extensibility and allows custom kernels for various devices



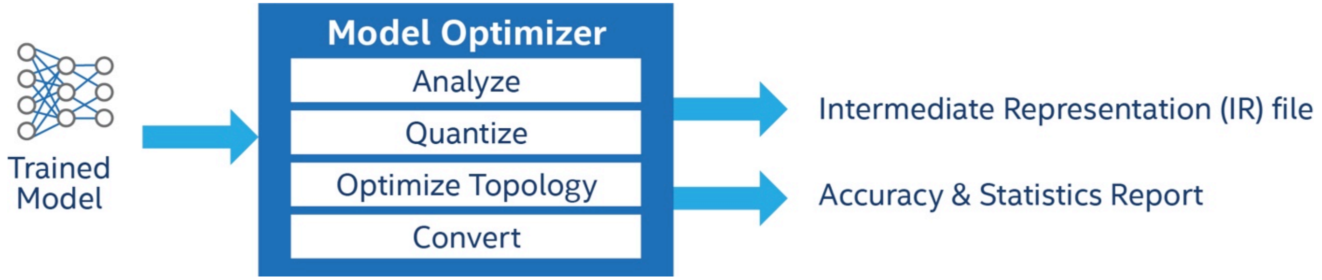
OpenCL and the OpenCL logo are trademarks of Apple Inc. used by permission by Khronos.

MODEL OPTIMIZER

- Convert models from various frameworks (i.e. Caffe*, TensorFlow*, MXNet*)
- Converts to a unified model (i.e. intermediate representation (IR), later n-graph)
- Optimizes topologies (i.e. node merging, batch normalization elimination, performing horizontal fusion)
- Folds constants paths in graph



IMPROVE PERFORMANCE WITH MODEL OPTIMIZER



- Easy to use, Python*-based workflow does not require rebuilding frameworks.
- Import Models from various frameworks (Caffe*, TensorFlow*, MXNet*, more are planned...)
- More than 100 models for Caffe, TensorFlow, and MXNet validated.
- IR files for models using standard layers or user-provided custom layers do not require Caffe
- Fallback to original framework is possible in cases of unsupported layers, but requires original framework

Device	FP32	FP16
CPU	Supported and preferred	Not Supported
GPU	Supported	Supported and Preferred
FPGA	Supported	Supported
MYRIAD	Not Supported	Supported

Intel® Deep Learning Deployment Toolkit supports a wide range of deep learning topologies:

•Classification models:
- AlexNet;
- VGG-16, VGG-19;
- SqueezeNet v1.0/v1.1;
- ResNet-50/101/152;
- Inception v1/v2/v3/v4;
- CaffeNet;
- MobileNet;

•Object detection models:
- SSD300/500-VGG16;
- Faster-RCNN;
- SSD-MobileNet v1, SSD-Inception v2
- Yolo Full v1/Tiny v1
- ResidiualNet-50/101/152, v1/v2
- DenseNet 121/161/169/201

•Face detection models:
- VGG Face;

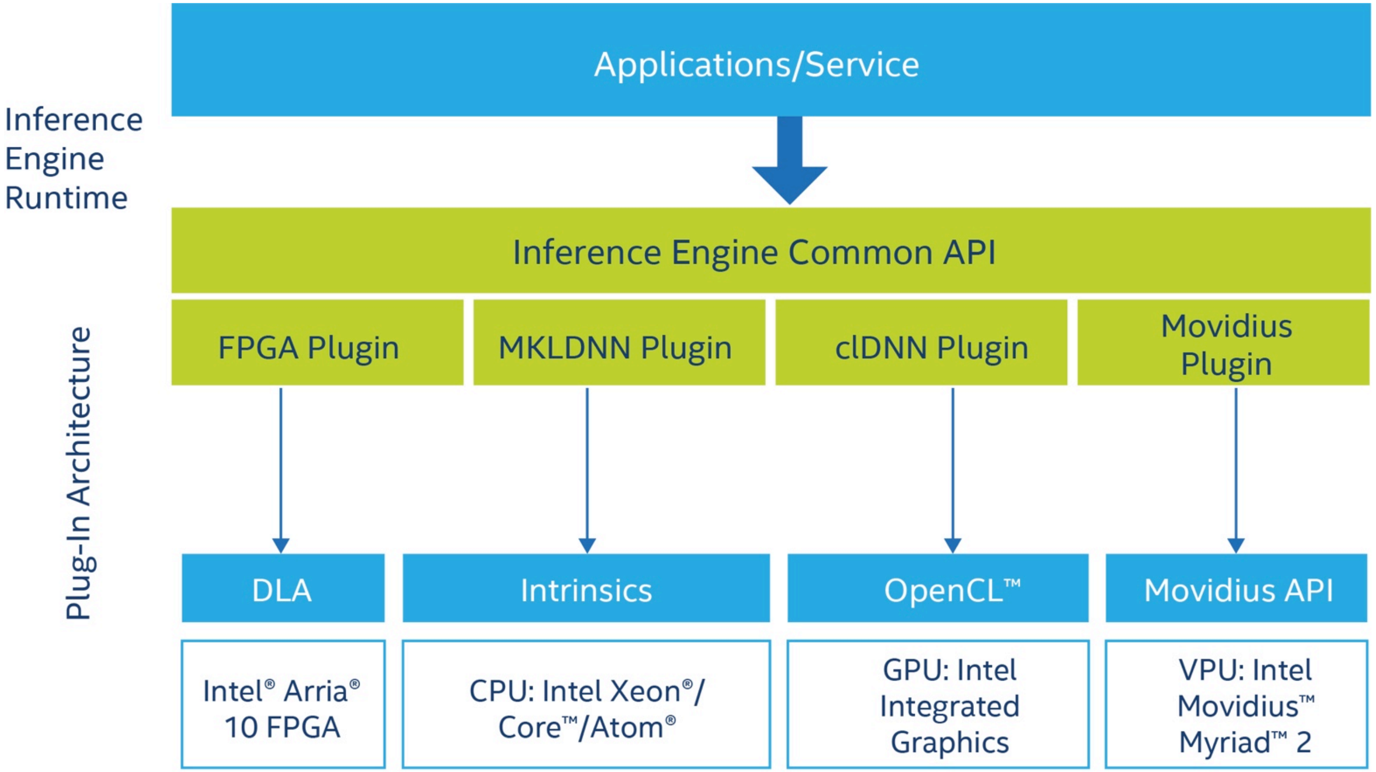
•Semantic segmentation models:
- FCN8;

LAB: OPTIMIZE A DEEP-LEARNING MODEL USING THE MODEL OPTIMIZER (MO)

INFERENCE ENGINE

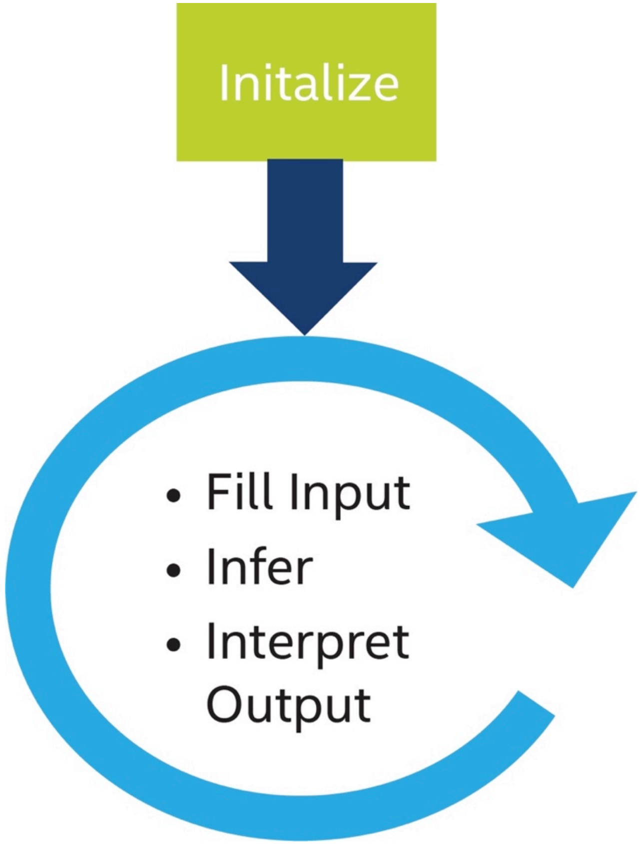
INFERENCE ENGINE

- Simple and unified API for Inference across all Intel® architecture
- Optimized inference on large Intel architecture hardware targets (CPU/GEN/FPGA)
- Heterogeneity support allows execution of layers across hardware types
- Asynchronous execution improves performance
- Futureproof/scale your development for future Intel processors



OpenCL and the OpenCL logo are trademarks of Apple Inc. used by permission by Khronos.

WORKFLOW



Initialization

- Load model and weights
- Set batch size (if needed)
- Load inference plugin (CPU, GPU, FPGA)
- Load network to plugin
- Allocate input, output buffers

Main loop

- Fill input buffer with data
- Run inference
- Interpret output results

LOAD MODEL

```
//-----------------------------------------------------------------------------
// Read network information from XML file
//-----------------------------------------------------------------------------

InferenceEngine::CNNNetReader network;
network.ReadNetwork(FLAGS_m);

//-----------------------------------------------------------------------------
// Read network parameters from BIN file
//-----------------------------------------------------------------------------

std::string binFileName = fileNameNoExt(FLAGS_m) + ".bin";
network.ReadWeights(binFileName.c_str());
```

LOAD PLUGIN

```
//-----------------------------------------------------------------------------
// Set batch size
//-----------------------------------------------------------------------------

network.getNetwork().setBatchSize(FLAGS_batch);

//-----------------------------------------------------------------------------
// Load plugin
//-----------------------------------------------------------------------------

PluginDispatcher dispatcher({""});
InferenceEnginePluginPtr plugin = dispatcher.getSuitablePlugin(TargetDevice::eCPU);
```

SET UP INPUT BLOBS

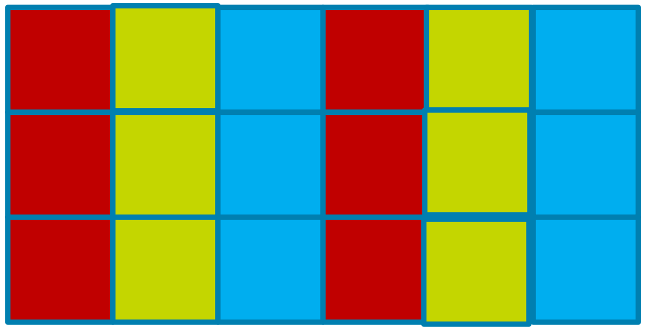
```
//-----------------------------------------------------------------------------
// Setting up input
//-----------------------------------------------------------------------------

InputsDataMap inputs = network.getNetwork().getInputsInfo();

InputInfo::Ptr inputInfo = inputs.begin()->second;
SizeVector inputDims = inputInfo->getDims();
DataPtr imageData = inputs.begin()->second->getInputData();

//-----------------------------------------------------------------------------
// Allocate input blobs
//-----------------------------------------------------------------------------

InferenceEngine::BlobMap inputBlobs;
InferenceEngine::TBlob<float>::Ptr input =
        InferenceEngine::make_shared_blob <float, const SizeVector>(Precision::FP32, inputDims);
input->allocate();

inputBlobs[inputs.begin()->first] = input;
```

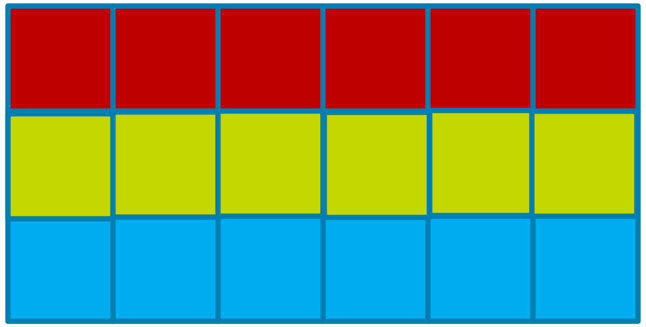
SET UP OUTPUT BLOBS

```
//----------------------------------------------------------------------------
//  Inference engine output setup
//----------------------------------------------------------------------------

OutputsDataMap outputInfo = network.getNetwork().getOutputsInfo();
BlobMap outputBlobs;
SizeVector outputDims = outputInfo.begin()->second->getDims();

TBlob<float>::Ptr output = make_shared_blob<float, const SizeVector>(Precision::FP32, outputDims);
output->allocate();

outputBlobs[outputInfo.begin()->first] = output;

size_t outputSize = outputBlobs.cbegin()->second->size() / batchSize;
```

PRE-PROCESSING

- Most image formats are interleaved (RGB, BGR, BGRA, etc.)
- Models usually expect RGB planar format:
  - R-plane
  - G-plane
  - B-plane





PREPARE INPUT DATA

```
//--------------------------------------------------------------------------
// PREPROCESS STAGE:
// Convert image to format expected by inference engine
// IE expects planar: R plain, G plain, B plain, convert from packed RGBRGB...
//--------------------------------------------------------------------------
// imgIdx: image pixel counter
// channelSize: size of a channel, computed as image width * image height
// inputPtr: a pointer to pre-allocated input buffer
for (size_t i = 0, imgIdx = 0, idx = 0; i < channelSize; i++, idx++) {
    for (size_t ch = 0; ch < inputChannels; ch++, imgIdx++) {
        inputPtr[idx + ch * channelSize] = resized[mb].data[imgIdx];
    }
}
```


intel AI

INFER

```
//---------------------------------------------------------------------------
// Run inference
//---------------------------------------------------------------------------
sts = plugin->Infer(inputBlobs, outputBlobs, &dsc);
```

POST-PROCESSING

Developer responsible to parse inference output.

Many output formats. Some examples:

- Simple classification (alexnet): an array of float confidence scores, # of elements=# of classes in the model
- SSD: many "boxes" with a confidence score, label #, xmin,ymin, xmax,ymax

AUTOMATIC FALLBACK WITH HETERO PLUGIN

$ object_detection_sample_ssd -d HETERO:GPU,CPU -l lib/libicv_extension.so -m ssd.xml -i snake.bmp

- The "priorities" define search order
  - Keeps all layers that can be executed on the device (FPGA)
  - Carefully respecting the topological and other limitations
  - Then follows priorities when searching (e.g. CPU)

LAB: BUILD AND RUN AN OBJECT DETECTION APPLICATION

OPTIMIZATION

FULL PIPELINE OPTIMIZATION

Stage	Decode/ Pre-Process	Infer	Post-Process Encode, Stream Out
Hardware Acceleration	CPU, GPU	CPU, GPU, FPGA, VPU	CPU, GPU

Higher Resolution → Better Accuracy, Faster Detection → More Computing Power, Higher NW Bandwidth, More Storage

COMBINE ADVANTAGES OF INTEL'S PORTFOLIO

Computer Vision	Deep Learning	Media
		

SDKs

- OpenCV — Optimized Computer Vision Capabilities
- Intel® Deep Learning Deployment Toolkit

Intel OpenVino™

- Intel Media SDK

Tools

Compiler, Analyzers, Debuggers

Intel C++ Compiler · VTune Performance Analyzer · PARALLEL STUDIO XE · SYSTEM STUDIO · OpenCL

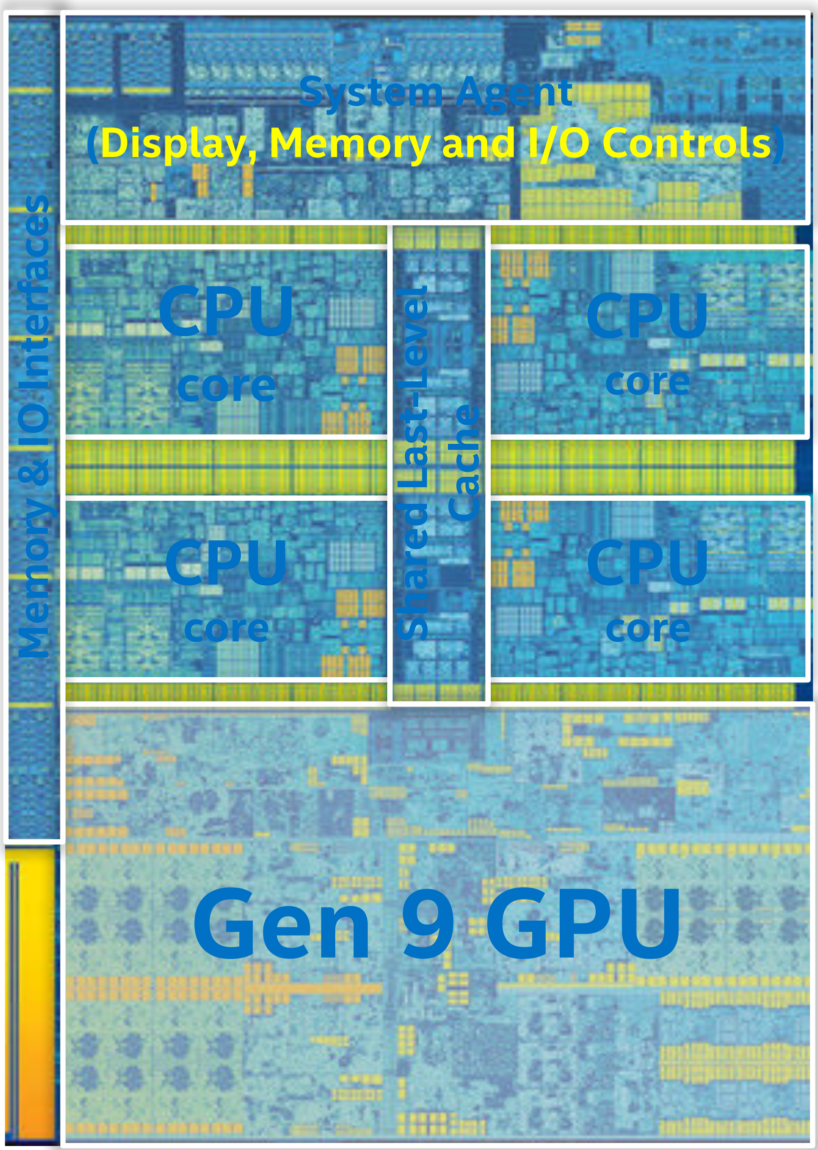
Libraries

IPP · TBB · OpenCL · MKL-DNN · CL-DNN · Intel® MKL DAAL · python Intel Dist

OpenCL and the OpenCL logo are trademarks of Apple Inc. used by permission by Khronos.

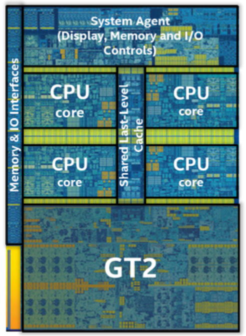
INTEL® INTEGRATED GRAPHICS

- **Gen** is the internal name for Intel's on-die GPU solution. It's a hardware ingredient with various configurations
- Intel Core™ processors include Gen hardware
- Gen GPU can be used for graphics, and also as a general compute resource
- Libraries contained in Intel® OpenVINO™ (and many others) support Gen offload using OpenCL™



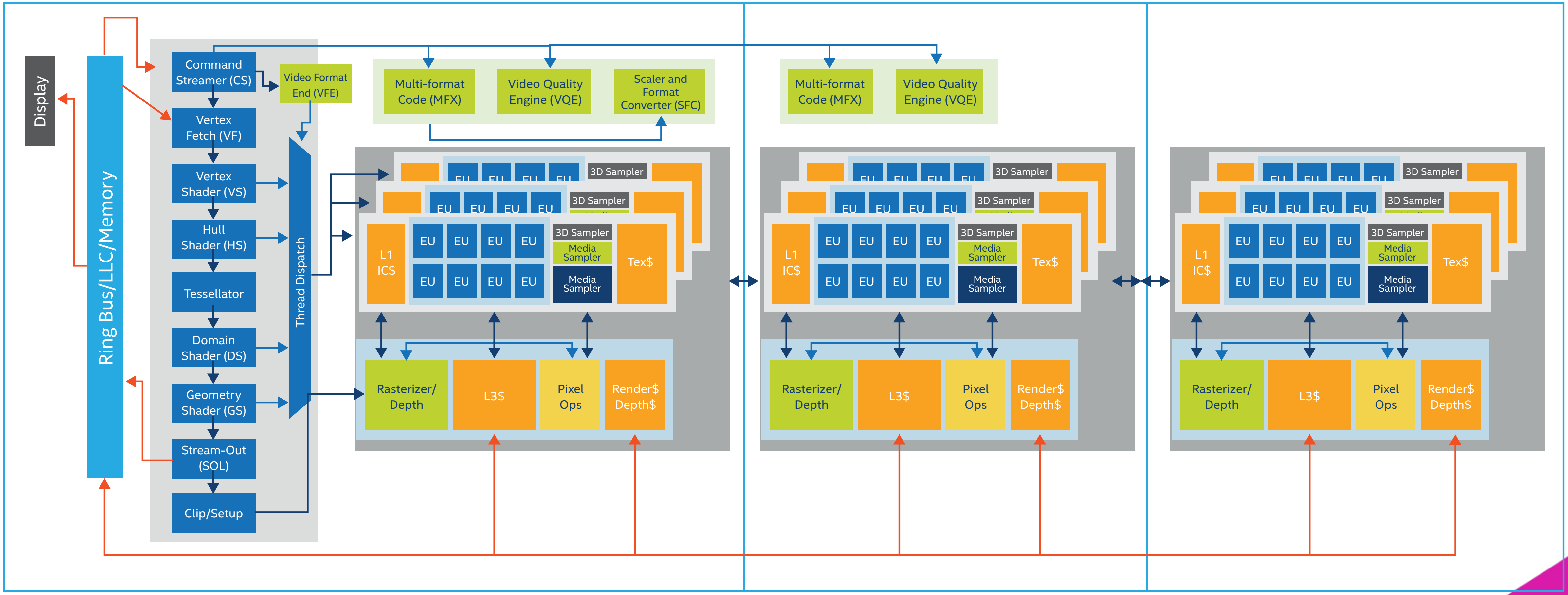
Sixth-Generation Core™ i7 (Skylake) Processor

OpenCL and the OpenCL logo are trademarks of Apple Inc. used by permission by Khronos.



INTEL® GPU SCALABLE ARCHITECTURE

GT2	GT3	GT4
Intel® HD Graphics	**Intel Iris™ Graphics**	**Intel Iris Pro Graphics**
24 EUs, 1 MFX	48 EUs, 2 MFX	72 EUs, 2 MFX

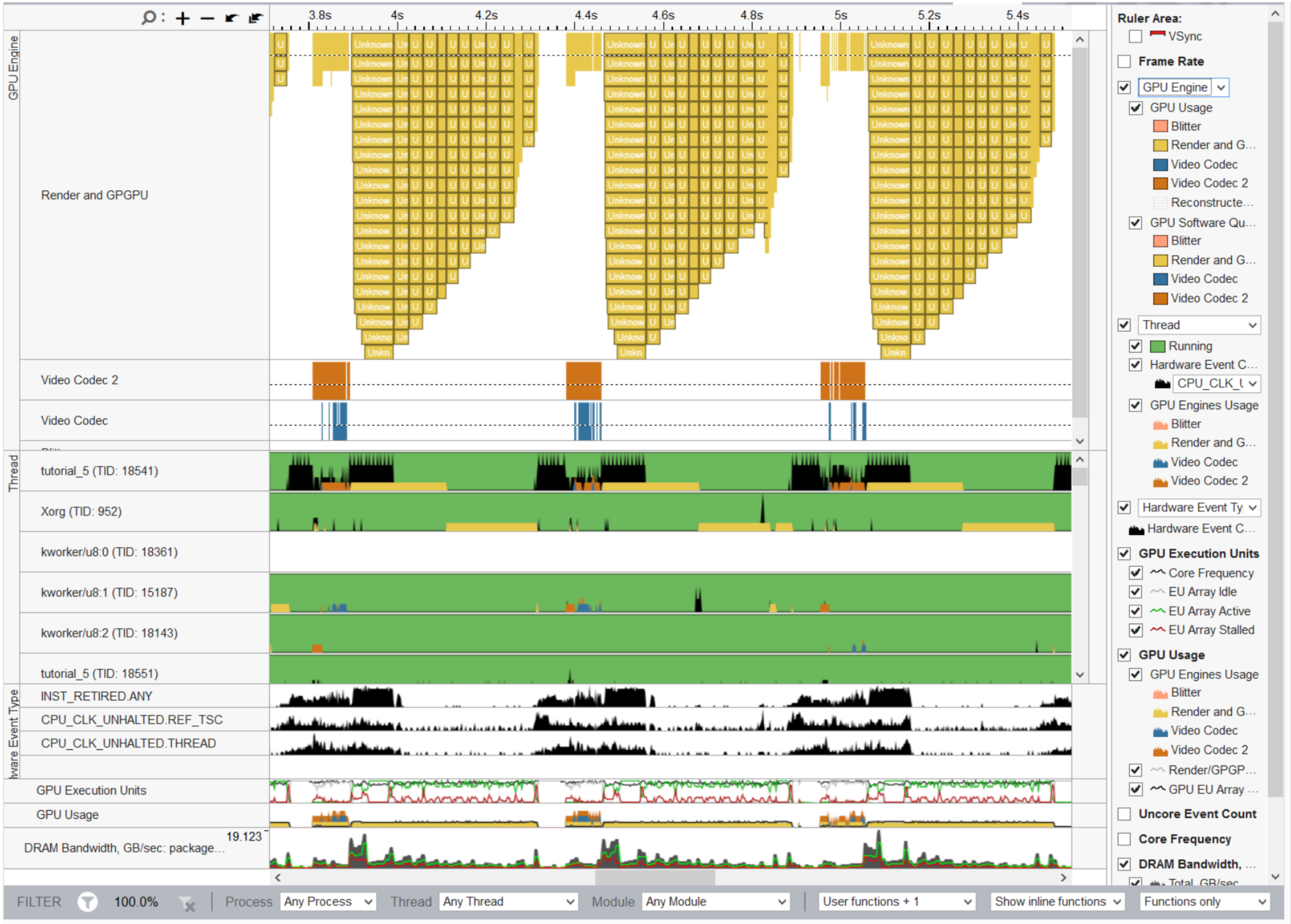


intel AI

INTEL® VTUNE™ AMPLIFIER HETEROGENEOUS CAPABILITIES:

Full platform visualization

- CPU threads
- GPU
- EUs (render)
- Fixed function / unslice



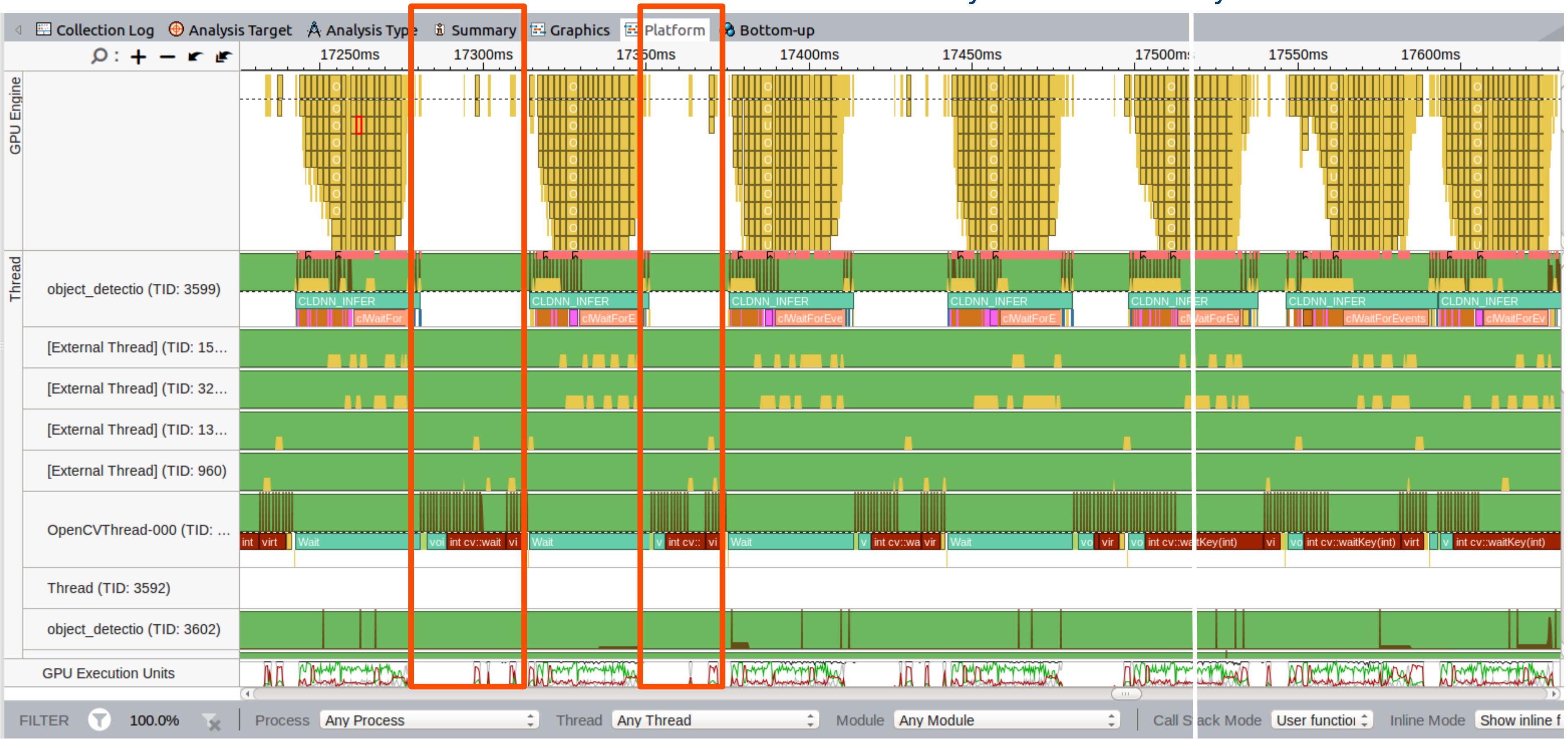
VISUALIZING INFERENCE ENGINE PERFORMANCE

large gaps between inferences
serialized video capture + render

infer concurrency == reduced gaps

Non-Async Mode

Async Mode



LAB: OPTIMIZATION TECHNIQUES

CONCLUSION/NEXT STEPS

FOR MORE INFORMATION

ACCELERATE VIDEO PROCESSING

Intel® Media SDK

Free Download >
software.intel.com/media-sdk

INTEGRATE VISUAL UNDERSTANDING

Just released as Gold!

Intel OpenVINO™

Free Download >
software.intel.com/computer-vision-sdk

CUSTOMIZE WITH OPENCL

Intel® SDK for OpenCL Applications

Free Download >
software.intel.com/intel-opencl

Specialized Hardware Acceleration

General CPU + System Optimization

software.intel.com/system-studio

OpenCL and the OpenCL logo are trademarks of Apple Inc. used by permission by Khronos.

OPTIMIZATION NOTICE

- Intel's compilers may or may not optimize to the same degree for non-Intel microprocessors for optimizations that are not unique to Intel microprocessors. These optimizations include SSE2, SSE3, and SSSE3 instruction sets and other optimizations. Intel does not guarantee the availability, functionality, or effectiveness or any optimization on microprocessors not manufactured by Intel. Microprocessor-dependent optimizations in this product are intended for use with Intel microprocessors. Certain optimizations not specific to Intel microarchitecture are reserved for Intel microprocessors. Please refer to the applicable product User and Reference Guides for more information regarding the specific instruction sets covered by this notice. Notice Revision #20110804

LEGAL NOTICES AND DISCLAIMERS

- Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Performance varies depending on system configuration. No computer system can be absolutely secure. Check with your system manufacturer or retailer or learn more at www.intel.com.
- Performance estimates were obtained prior to implementation of recent software patches and firmware updates intended to address exploits referred to as "Spectre" and "Meltdown." Implementation of these updates may make these results inapplicable to your device or system.
- Cost reduction scenarios described are intended as examples of how a given Intel-based product, in the specified circumstances and configurations, may affect future costs and provide cost savings. Circumstances will vary. Intel does not guarantee any costs or cost reduction.
- This document contains information on products, services and/or processes in development. All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest forecast, schedule, specifications and roadmaps.
- Any forecasts of goods and services needed for Intel's operations are provided for discussion purposes only. Intel will have no liability to make any purchase in connection with forecasts published in this document.
- ARDUINO 101 and the ARDUINO infinity logo are trademarks or registered trademarks of Arduino, LLC.
- Altera, Arria, the Arria logo, Intel, the Intel logo, Intel Atom, Intel Core, Intel Nervana, Intel Xeon Phi, Movidius, Saffron and Xeon are trademarks of Intel Corporation or its subsidiaries in the U.S. and/or other countries.
- *Other names and brands may be claimed as the property of others.

LEGAL NOTICES AND DISCLAIMERS

- This document contains information on products, services and/or processes in development. All information provided here is subject to change without notice. Contact your Intel representative to obtain the latest forecast, schedule, specifications and roadmaps.
- Intel technologies' features and benefits depend on system configuration and may require enabled hardware, software or service activation. Learn more at intel.com, or from the OEM or retailer. No computer system can be absolutely secure.
- Tests document performance of components on a particular test, in specific systems. Differences in hardware, software, or configuration will affect actual performance. Consult other sources of information to evaluate performance as you consider your purchase. For more complete information about performance and benchmark results, visit http://www.intel.com/performance.
- Cost reduction scenarios described are intended as examples of how a given Intel-based product, in the specified circumstances and configurations, may affect future costs and provide cost savings. Circumstances will vary. Intel does not guarantee any costs or cost reduction.
- Statements in this document that refer to Intel's plans and expectations for the quarter, the year, and the future, are forward-looking statements that involve a number of risks and uncertainties. A detailed discussion of the factors that could affect Intel's results and plans is included in Intel's SEC filings, including the annual report on Form 10-K.
- The products described may contain design defects or errors known as errata which may cause the product to deviate from published specifications. Current characterized errata are available on request.
- Performance estimates were obtained prior to implementation of recent software patches and firmware updates intended to address exploits referred to as "Spectre" and "Meltdown." Implementation of these updates may make these results inapplicable to your device or system.
- No license (express or implied, by estoppel or otherwise) to any intellectual property rights is granted by this document.
- Intel does not control or audit third-party benchmark data or the web sites referenced in this document. You should visit the referenced web site and confirm whether referenced data are accurate.
- Intel, the Intel logo, Pentium, Celeron, Atom, Core, Xeon, Movidius, Saffron and others are trademarks of Intel Corporation in the U.S. and/or other countries. *Other names and brands may be claimed as the property of others.
-

intel® AI