Deep Learning with OpenCV

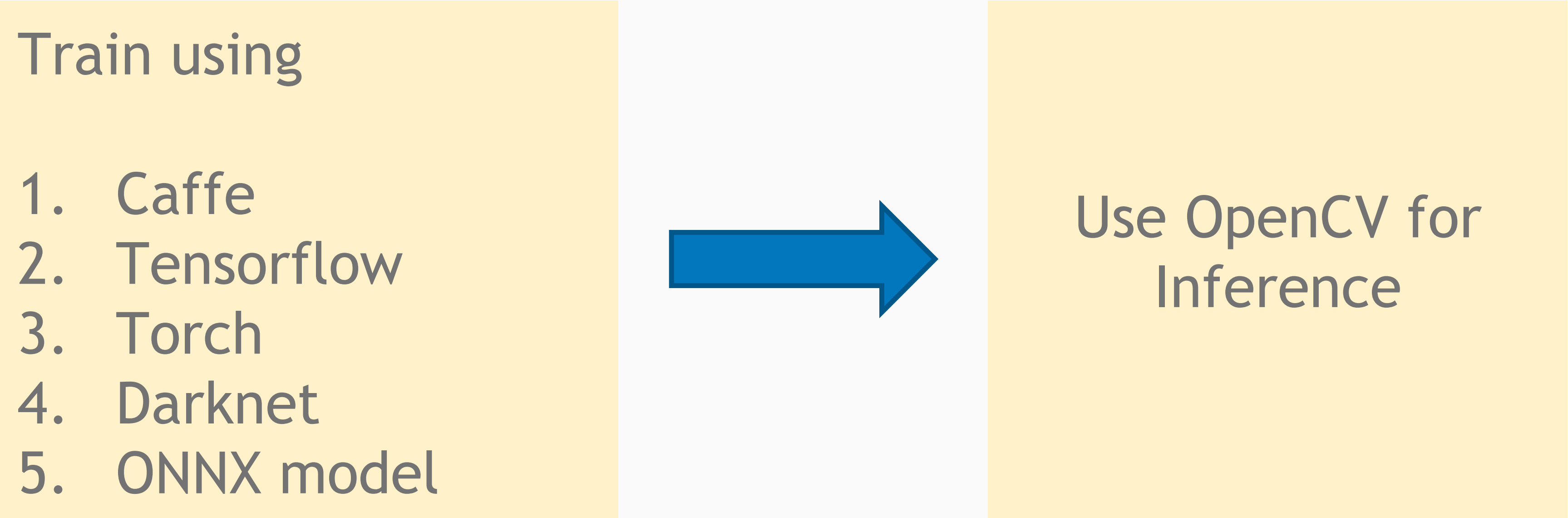
Satya Mallick, Ph.D.

Interim CEO	OpenCV.org	Jan 2019 - Present
Owner	Big Vision LLC	Feb 2014 - Present
Author	LearnOpenCV.com	Jan 2015 - Present
Co-Founder / CTO	Sight Commerce Inc.	2017 - 2018

Deep Neural Network Module

Fastest CPU implementation across many tasks

OpenCV DNN Module : Inference Engine



Train using

1. Caffe
2. Tensorflow
3. Torch
4. Darknet
5. ONNX model

Use OpenCV for Inference

OpenCV Model Zoo : Classification

- AlexNet
- GoogleNet
- CaffeNet
- RCNN_ILSVRC13
- ZFNet512
- VGG16, VGG16_bn
- ResNet-18v1, ResNet-50v1
- CNN Mnist
- MobileNetv2
- LResNet100E-IR
- Emotion FERPlus
- Squeezenet
- DenseNet121
- Inception v1, v2
- Shufflenet

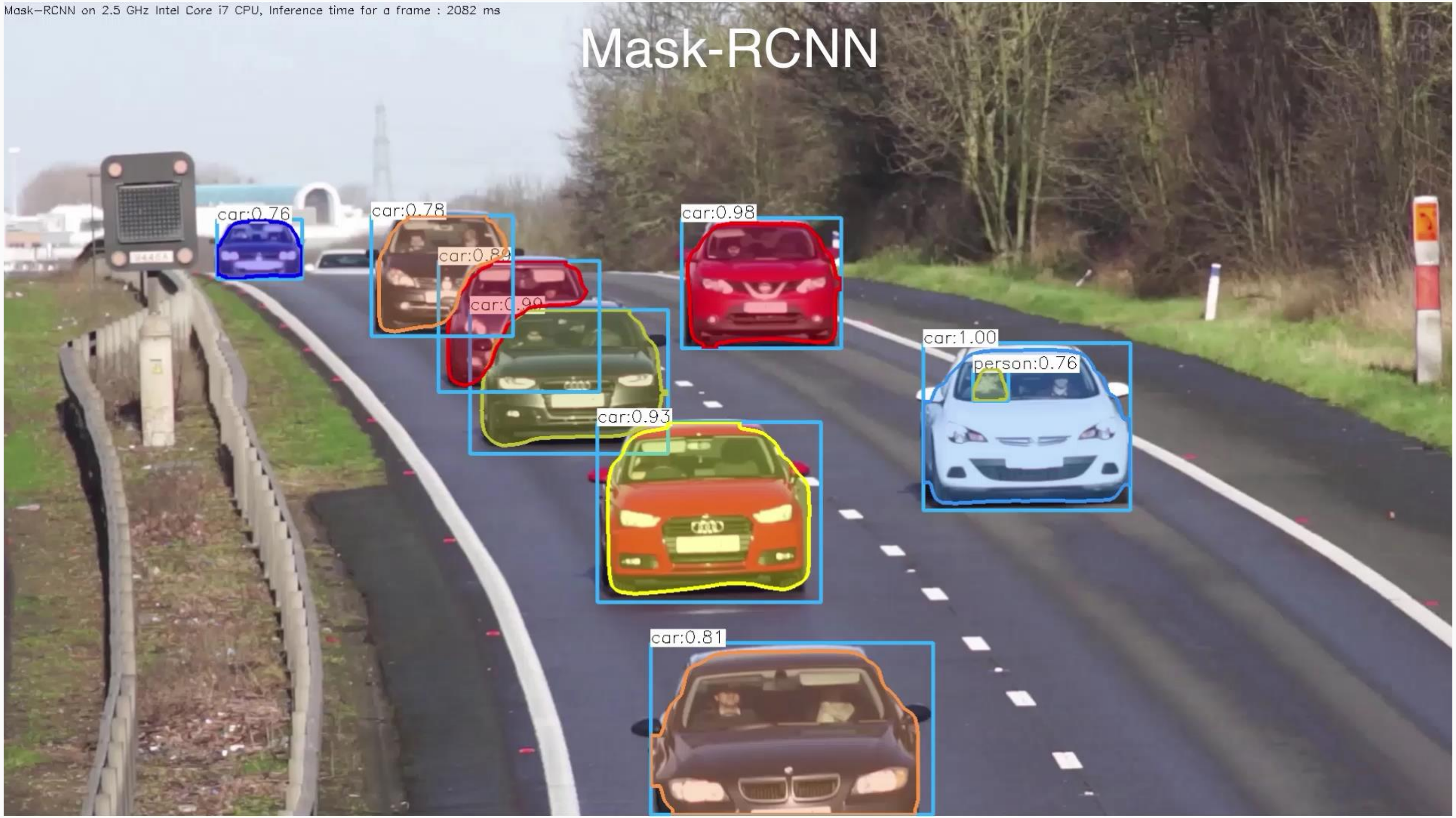
Object Detection

- YOLOv3
- SSD VGG
- MobileNet-SSD
- Faster-RCNN
- R-FCN
- OpenCV face detector
- TinyYolov2

Semantic Segmentation

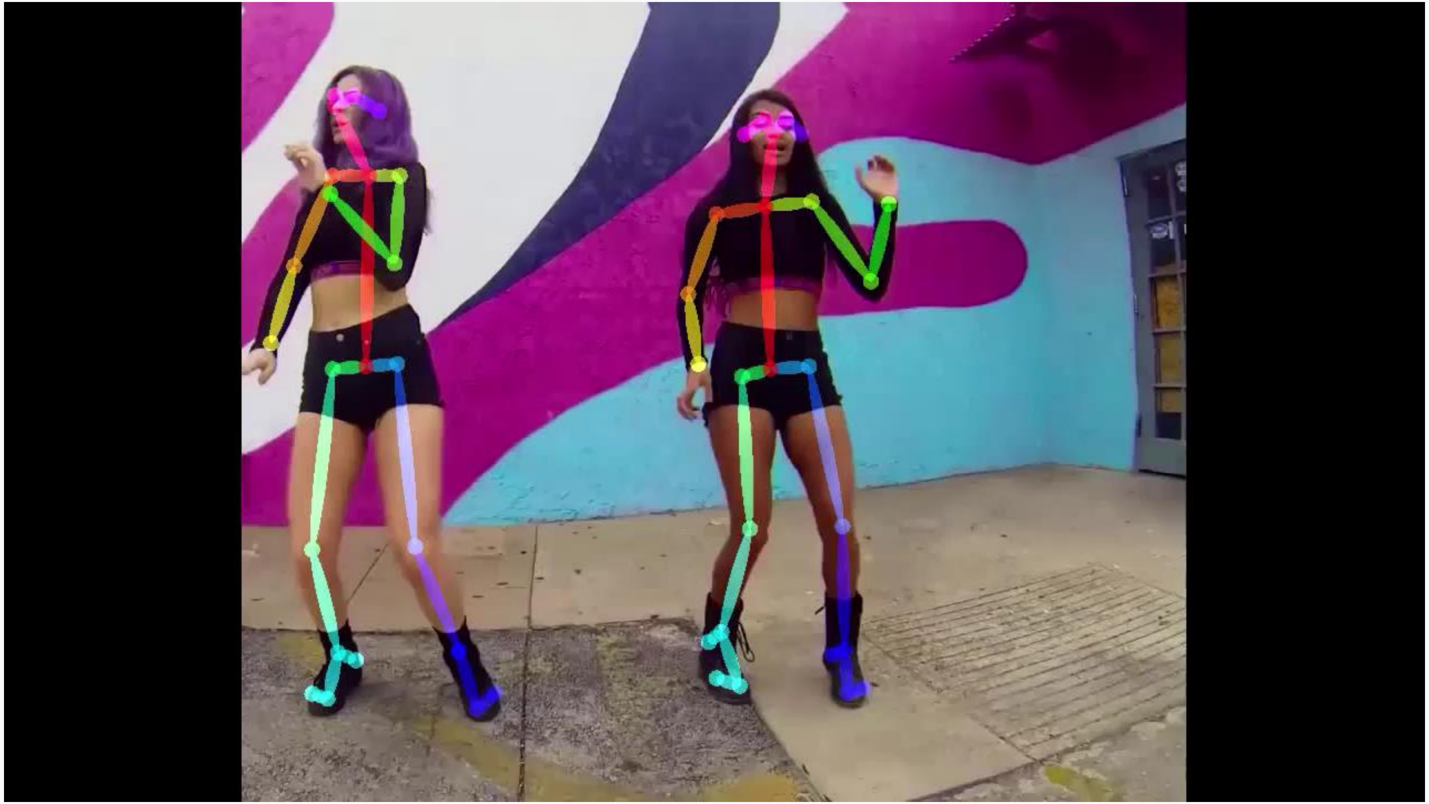
- FCN
- ENet
- ResNet101_DUC_HDC

Mask R-CNN with OpenCV



Mask–RCNN on 2.5 GHz Intel Core i7 CPU, Inference time for a frame : 2082 ms

OpenPose with OpenCV



Compile with opencv_contrib

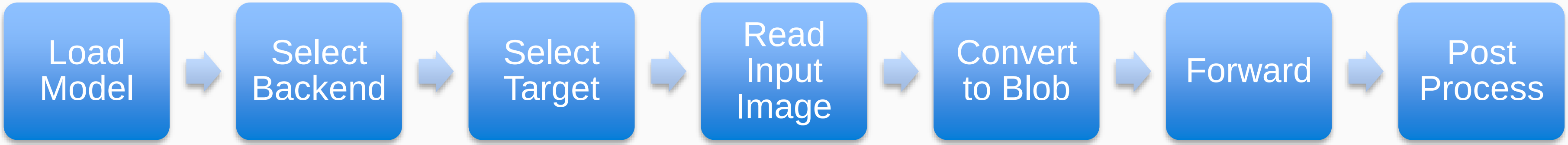
Python

```
pip install opencv-contrib-python
```

CMAKE

```
cmake -D CMAKE_BUILD_TYPE=RELEASE \
      -D CMAKE_INSTALL_PREFIX=/usr/local \
      -D INSTALL_C_EXAMPLES=ON \
      -D INSTALL_PYTHON_EXAMPLES=ON \
      -D WITH_TBB=ON \
      -D WITH_V4L=ON \
      -D WITH_QT=ON \
      -D WITH_OPENGL=ON \
      -D OPENCV_EXTRA_MODULES_PATH=../../opencv_contrib/modules \
      -D BUILD_EXAMPLES=ON ..
```

Import a DNN model in OpenCV



Load Model

Select Backend

Select Target

Read Input Image

Convert to Blob

Forward

Post Process

Model Loaders

```
cv::dnn::readNet
cv::dnn::readNetFromCaffe
cv::dnn::readNetFromDarknet
cv::dnn::readNetFromONNX
cv::dnn::readNetFromTensorflow
cv::dnn::readNetFromTorch
cv::dnn::readNetFromModelOptimizer
```

Load Model : Architecture / Graph + Weights

Python

```python
# Architecture and weight files for the model
textGraph = "./mask_rcnn_inception_v2_coco_2018_01_28.pbtxt"
modelWeights = "./frozen_inference_graph.pb"

# Load the network
net = cv.dnn.readNetFromTensorflow(modelWeights, textGraph);
```

Load Model : Architecture / Graph + Weights

C++

```
// Architecture and weight files for the model
String textGraph = "./mask_rcnn_inception_v2_coco_2018_01_28.pbtxt";
String modelWeights = "./frozen_inference_graph.pb";

// Load the network
Net net = readNetFromTensorflow(modelWeights, textGraph);
```

Set Backend & Target

Backend refers to the implementation

1. DNN_BACKEND_OPENCV
2. DNN_BACKEND_HALIDE
3. DNN_BACKEND_INFERENCE_ENGINE

Target refers to the processor

1. DNN_TARGET_CPU
2. DNN_TARGET_OPENCL
3. DNN_TARGET_OPENCL_FP16
4. DNN_TARGET_MYRIAD
5. DNN_TARGET_FPGA

Set Backend & Target

Python

```python
net.setPreferableBackend(cv.dnn.DNN_BACKEND_OPENCV)
net.setPreferableTarget(cv.dnn.DNN_TARGET_CPU)
```

C++

```cpp
net.setPreferableBackend(DNN_BACKEND_OPENCV);
net.setPreferableTarget(DNN_TARGET_CPU);
```

Read Input Image & Convert to Blob

Python

```python
# Get frame from the video
hasFrame, frame = cap.read()

# Create a 4D blob from a frame.
blob = cv.dnn.blobFromImage(frame, swapRB=True, crop=False)

# Set the input to the network
net.setInput(blob)
```

C++

```cpp
// Get frame from the video
cap >> frame;

// Create a 4D blob from a frame.
blobFromImage(frame, blob, 1.0, Size(frame.cols, frame.rows), Scalar(), true, false);

// Sets the input to the network
net.setInput(blob);
```

Forward Pass

Python

```python
# Run the forward pass to get output from the output layers
boxes, masks = net.forward(['detection_out_final','detection_masks'])
```

C++

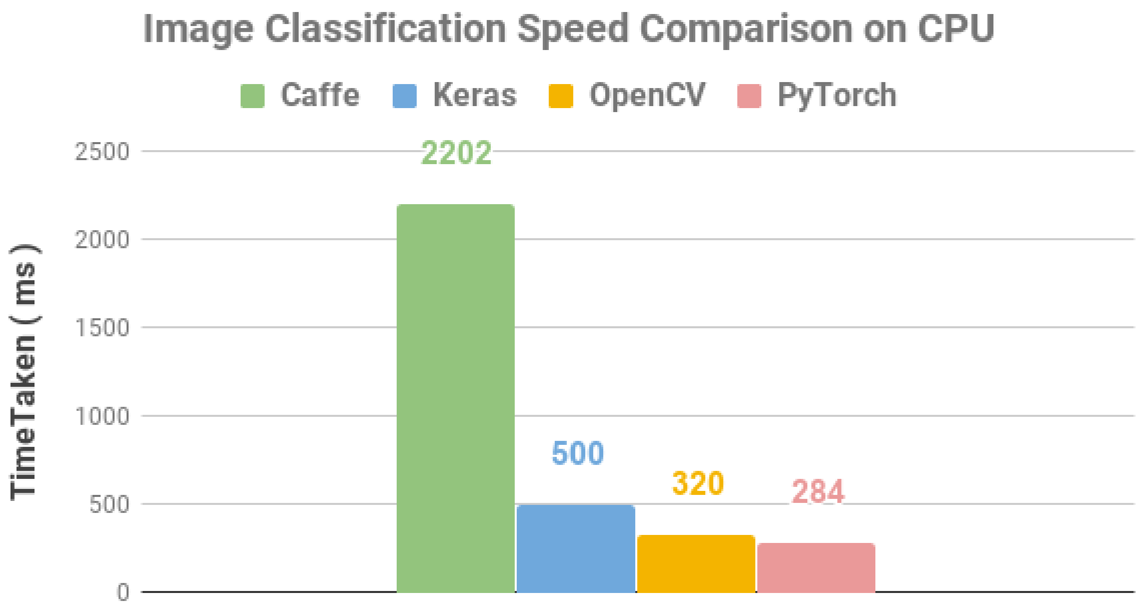
```cpp
// Runs the forward pass to get output from the output layers
std::vector<String> outNames(2);
outNames[0] = "detection_out_final";
outNames[1] = "detection_masks";

vector<Mat> outs;
net.forward(outs, outNames);
```

Check out tutorial and code

https://www.learnopencv.com/tag/mask-rcnn/

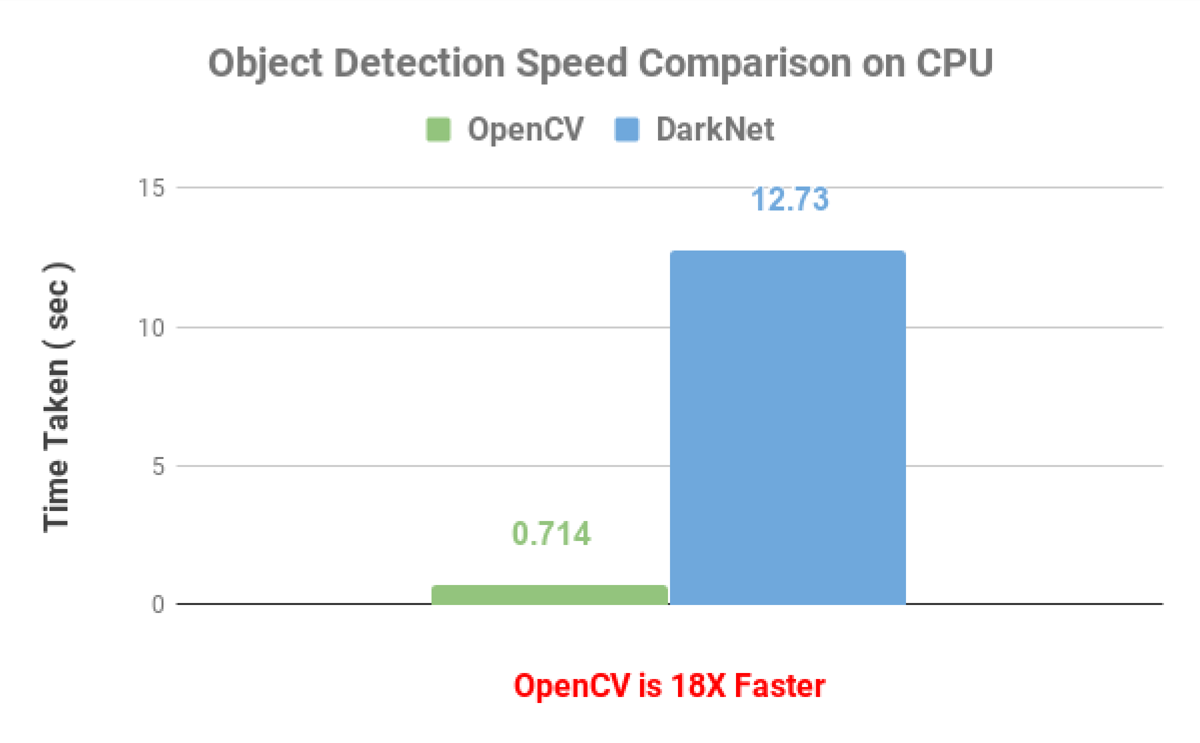
OpenCV DNN on CPU is fast

Image Classification

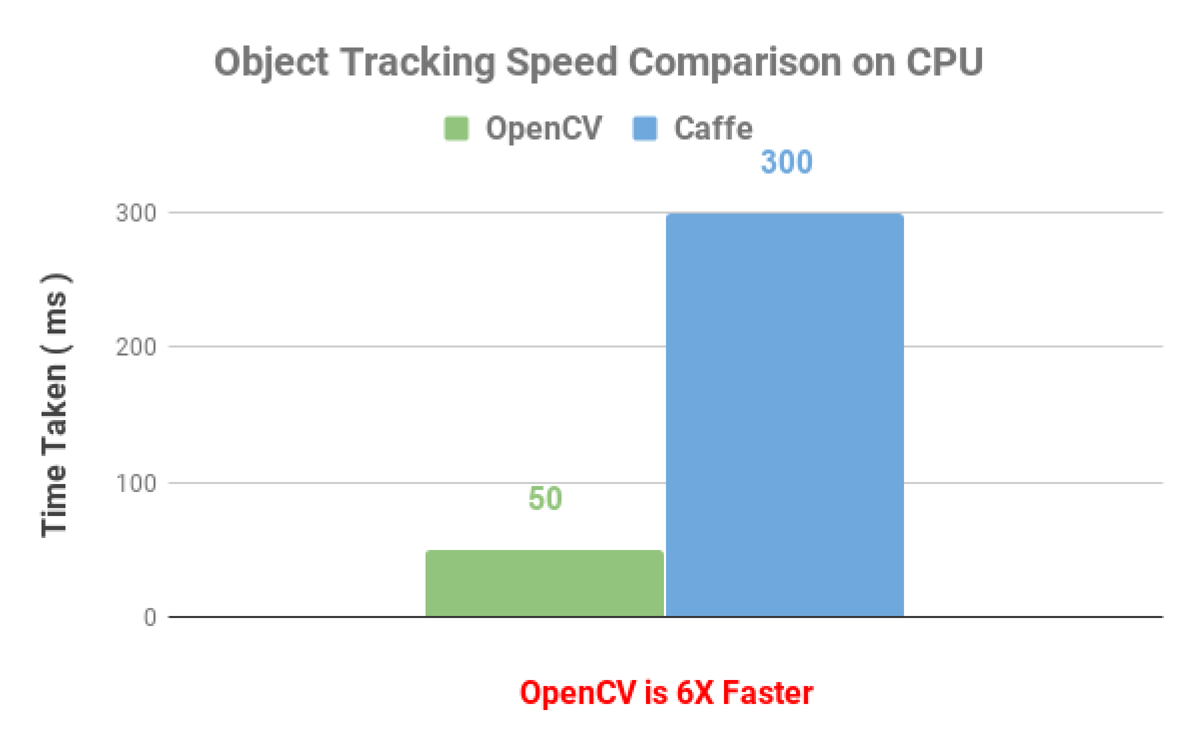


OpenCV is faster by 7X (Caffe) and 1.5X (Keras)

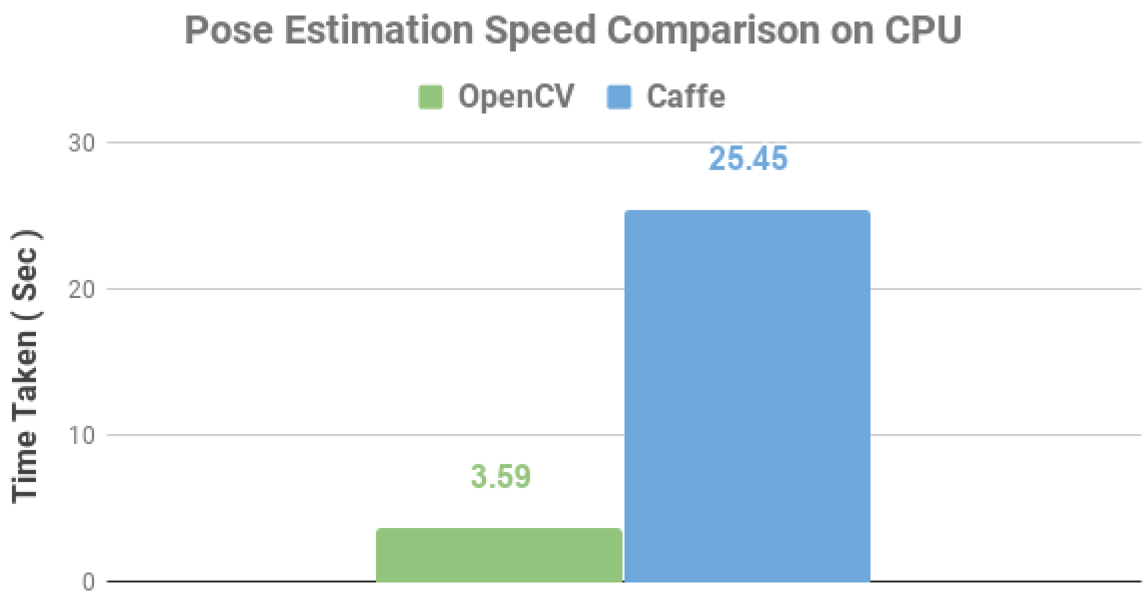
Object Detection (YOLOv3)



Object Tracking (GOTURN)



Pose Estimation

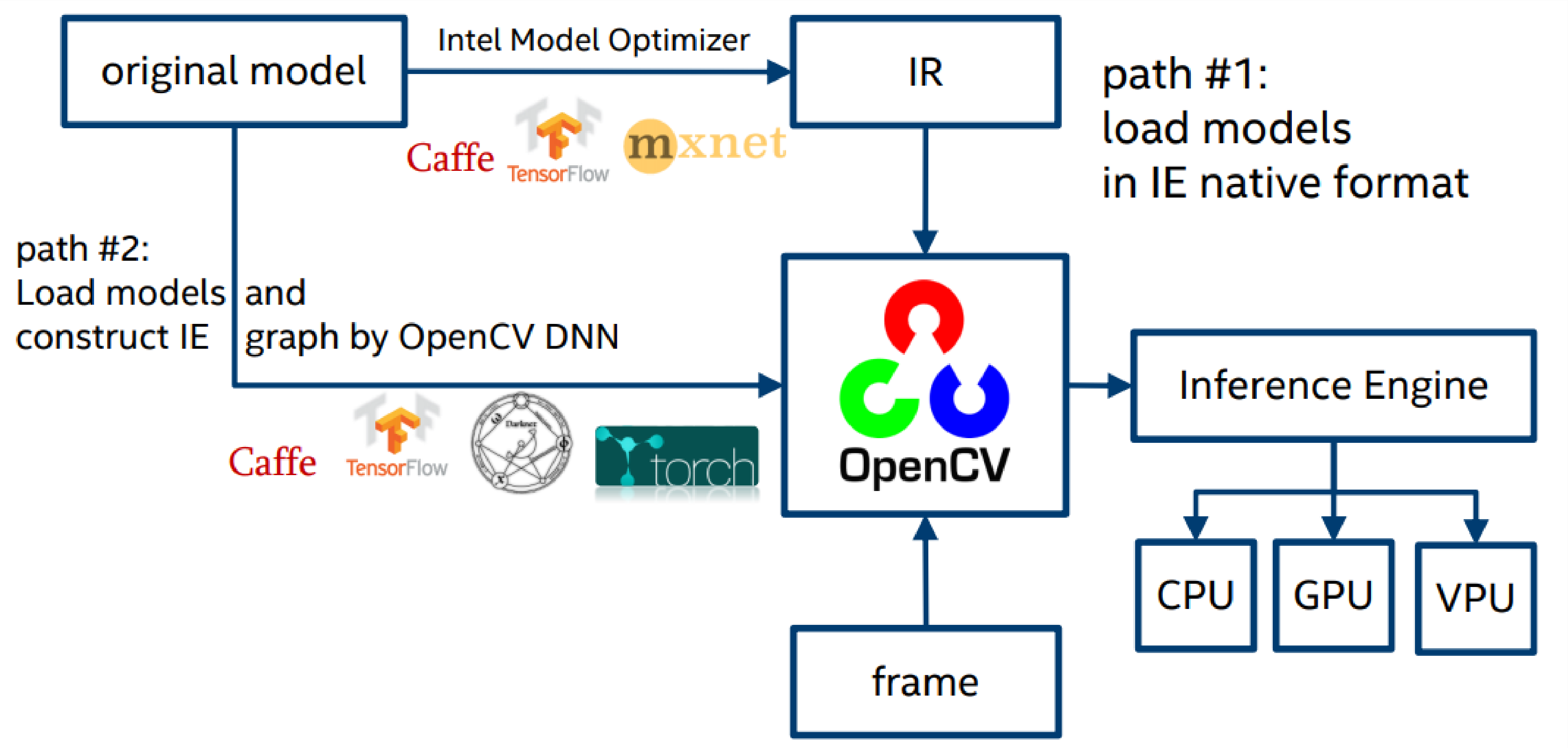


OpenCV is 7X Faster

OpenCV + OpenVINO

Even faster speeds with Intel’s Inference Engine.

OpenCV with Intel’s Inference Engine



Check out tutorial and code at

Using OpenVINO Toolkit to increase OpenCV Performance

https://www.learnopencv.com/tag/openvino

OpenCV NVIDIA GPU support is coming soon!

Part of GSOC 2019

Thank You!