# Machine Learning for the Real World: What is Acceptable Accuracy, and How Can You Achieve It?

Tim Hartley

Director Product & Marketing

arm

2020 embedded VISION summit®

# Arm Insight Platform

# ML on the Edge or in the Cloud?

## At or near the Edge

- Reduced round-trip latency
- Better privacy
- Reduced bandwidth costs
- Reduced cloud compute costs

## In the Cloud

- Plenty of compute power
- Scalable
- Easy to deploy

# Creating Distributed ML Applications

## In an ideal world

- We could run ML where it makes best sense to run it
  - Moving workloads from edge to cloud and back at will
  - And from edge platform to edge platform with no stickiness
  - With a wide range of edge devices created with ML acceleration capabilities

## In reality

- Edge compute power varies hugely
  - ML workloads will not run across all platforms
- Varying software APIs and libraries
  - ML & CV functionality often proprietary
  - Performance-portable applications difficult to write

# Creating Distributed ML Applications

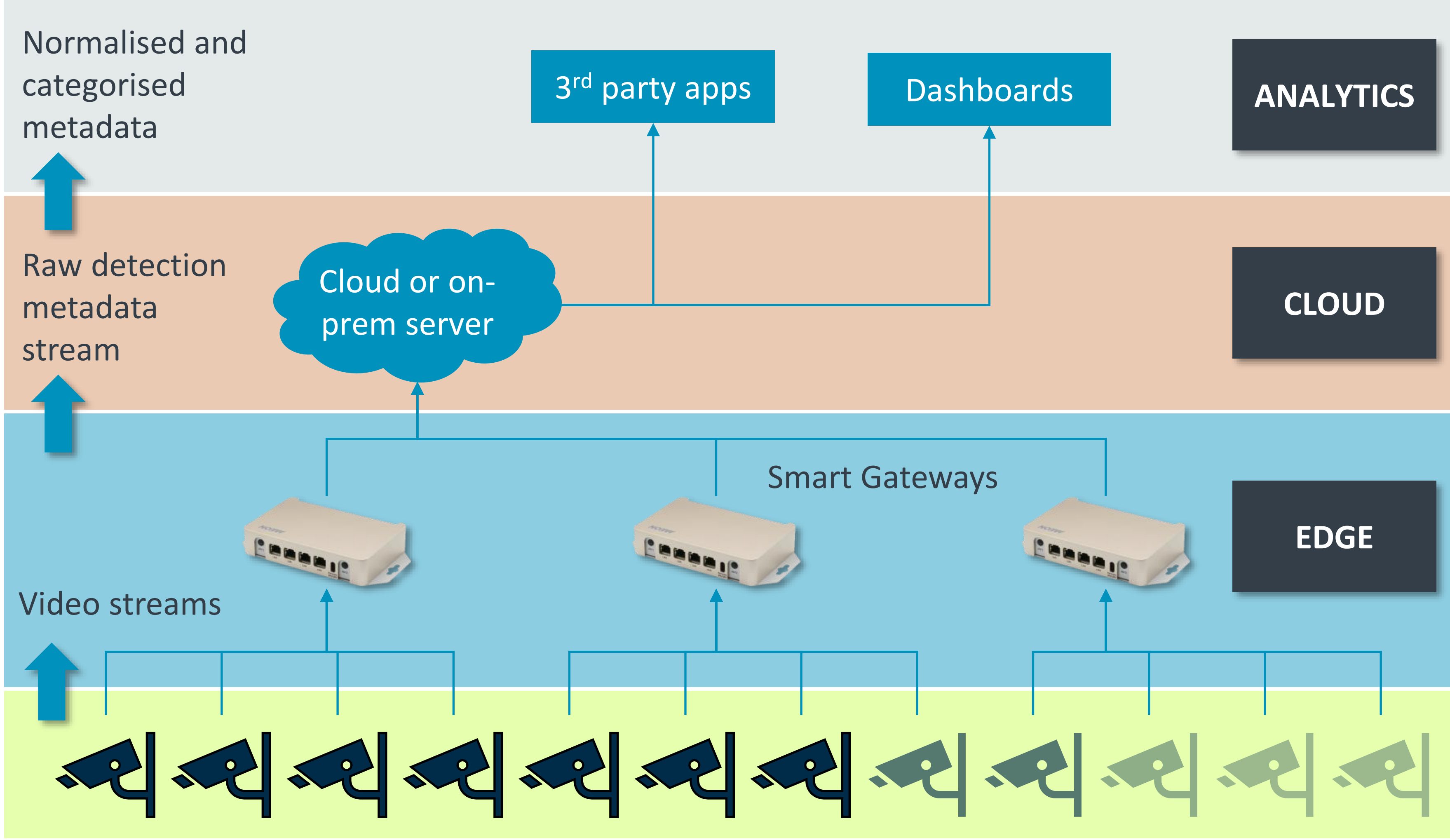
## In an ideal world

- We could move and store media easily between edge and cloud
  - Applying ML compute on media freely captured and distributed
  - With guarantees of security for personal information

## In reality

- GDPR and other privacy concerns
  - What can be captured, stored and distributed is under ever-increasing scrutiny
  - Anything that can personally identify you is coming under strict control
  - Difficult to prove system-wide security, particularly with the cloud

# Compute Hierarchy: Edge to Gateway to Cloud




- Analytics on processed metadata
- Provide live and historical analysis
- Deliver notifications
- Aggregate metadata from smart gateways
- Determine and deliver notifications
- Smart gateways process video streams from one or more cameras
- Run ML algorithms to determine location of people and other objects
- IP cameras and other sensors

# A Real World Case Study

People / object detection for determining space occupancy in real time

# A Real World Case Study: Real-Time Space Occupancy

## Overview

- People and object detection (<u>not</u> face recognition)
- To determine real-time space occupancy

Requirements opposite to determine

- Machine learning detection model
- And a suitable hardware platform

## Key requirements

- Off-the-shelf IP cameras
- No personally identifiable information to be sent to the cloud
- Independent of viewing angle
- ~80ft detections, HD @ 50mm lens
- Low false negatives and false positives
- 1 frame every 15s minimum

# Step 1

## Find a Suitable Model

# Datasets and Models

## Dataset candidates

Many to choose from:

- COCO, Kitti, Open Images, etc.

Selected COCO for now

- Good range of classes and models
- And a high image count in the classes we cared most about

## Model candidates

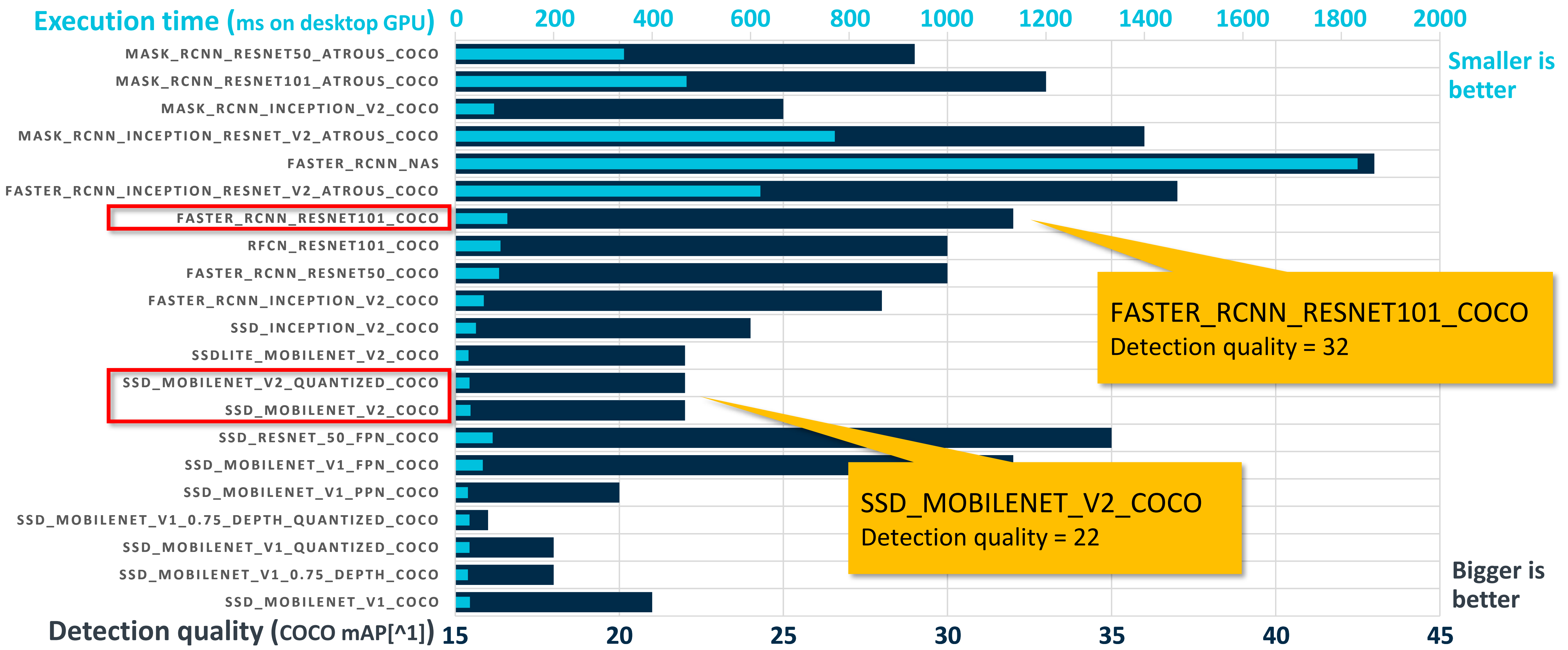
A good selection of COCO-trained models

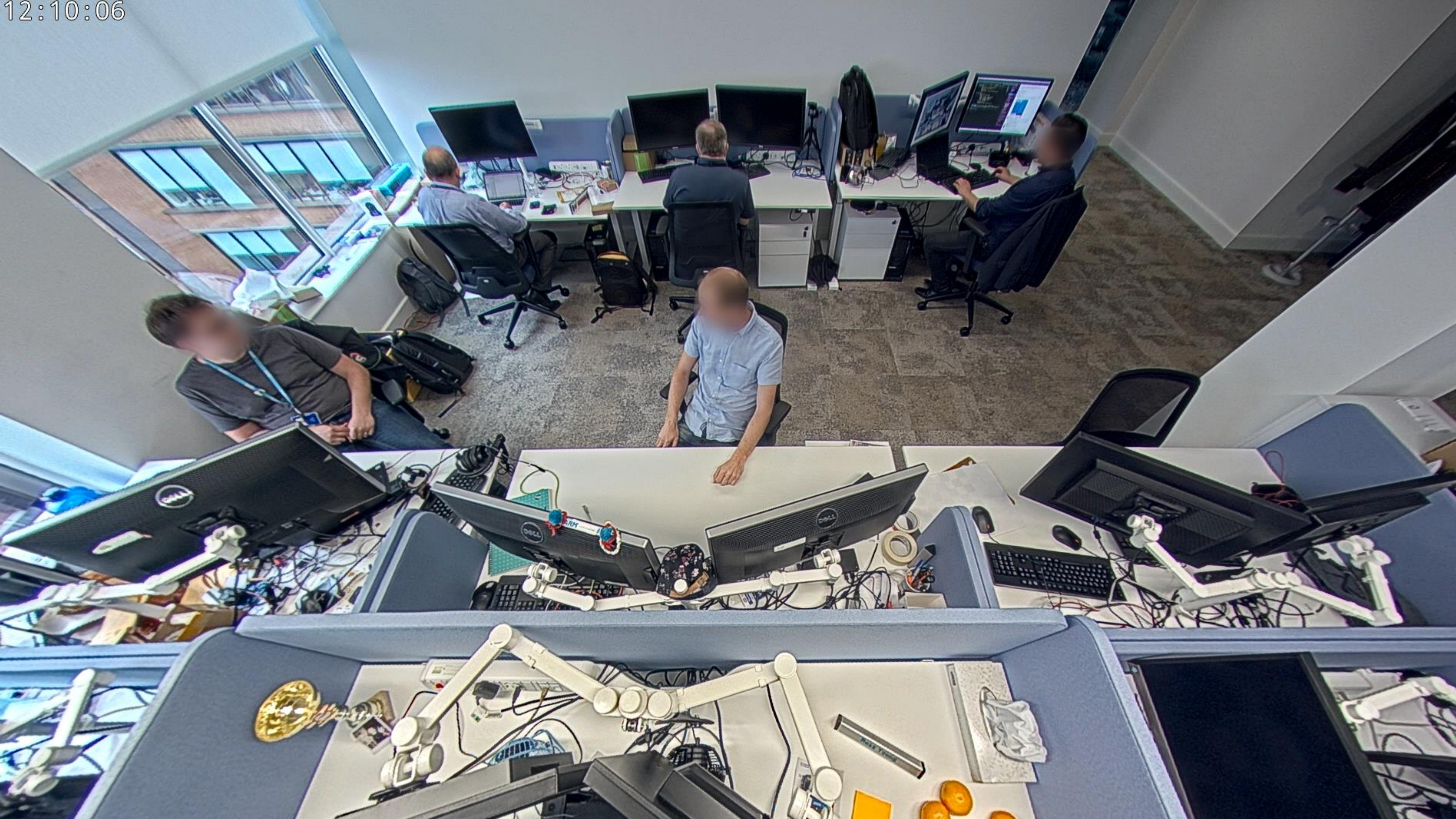
- ranked using a “COCO mAP[^1]” metric
- and by execution time

These are excellent starting points

- Downloadable and usable quickly

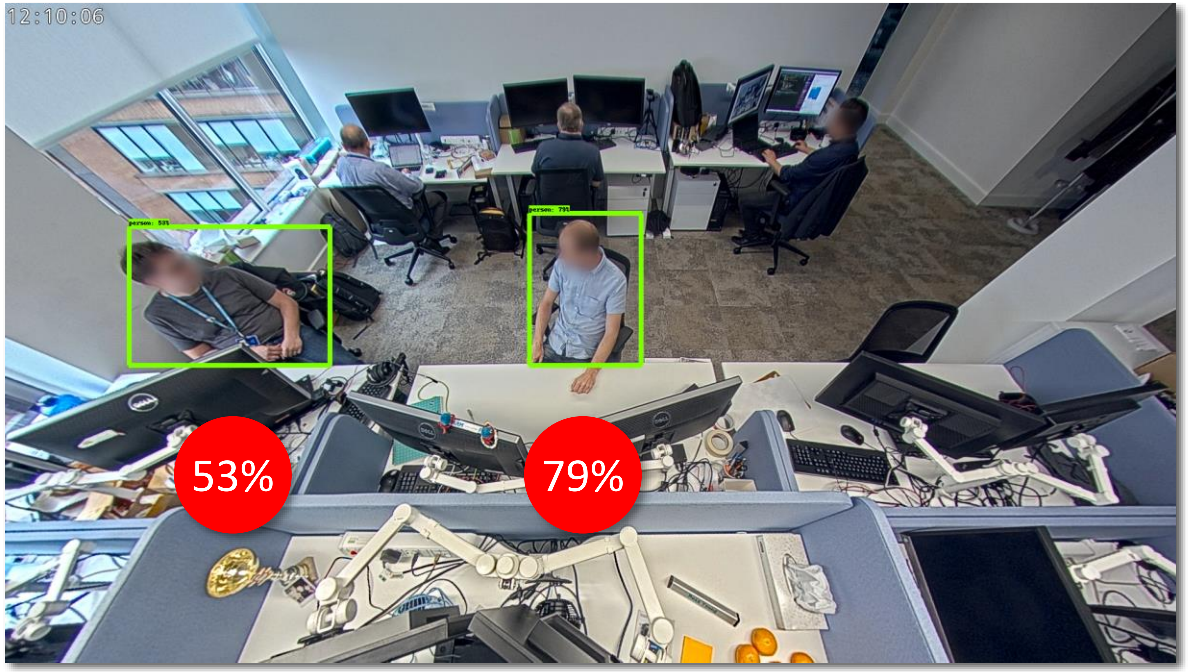
# Comparison of COCO-Trained Models



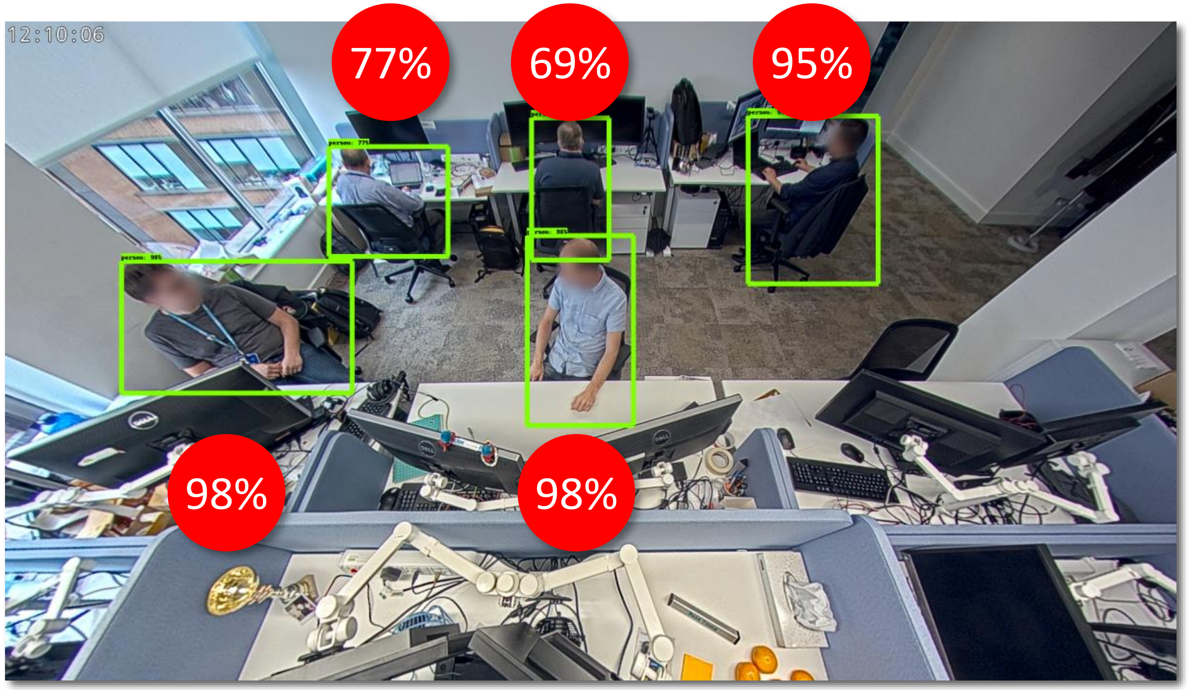

12:10:06

# Comparison of Detection Performance

## SSD_MOBILENET_V2_COCO



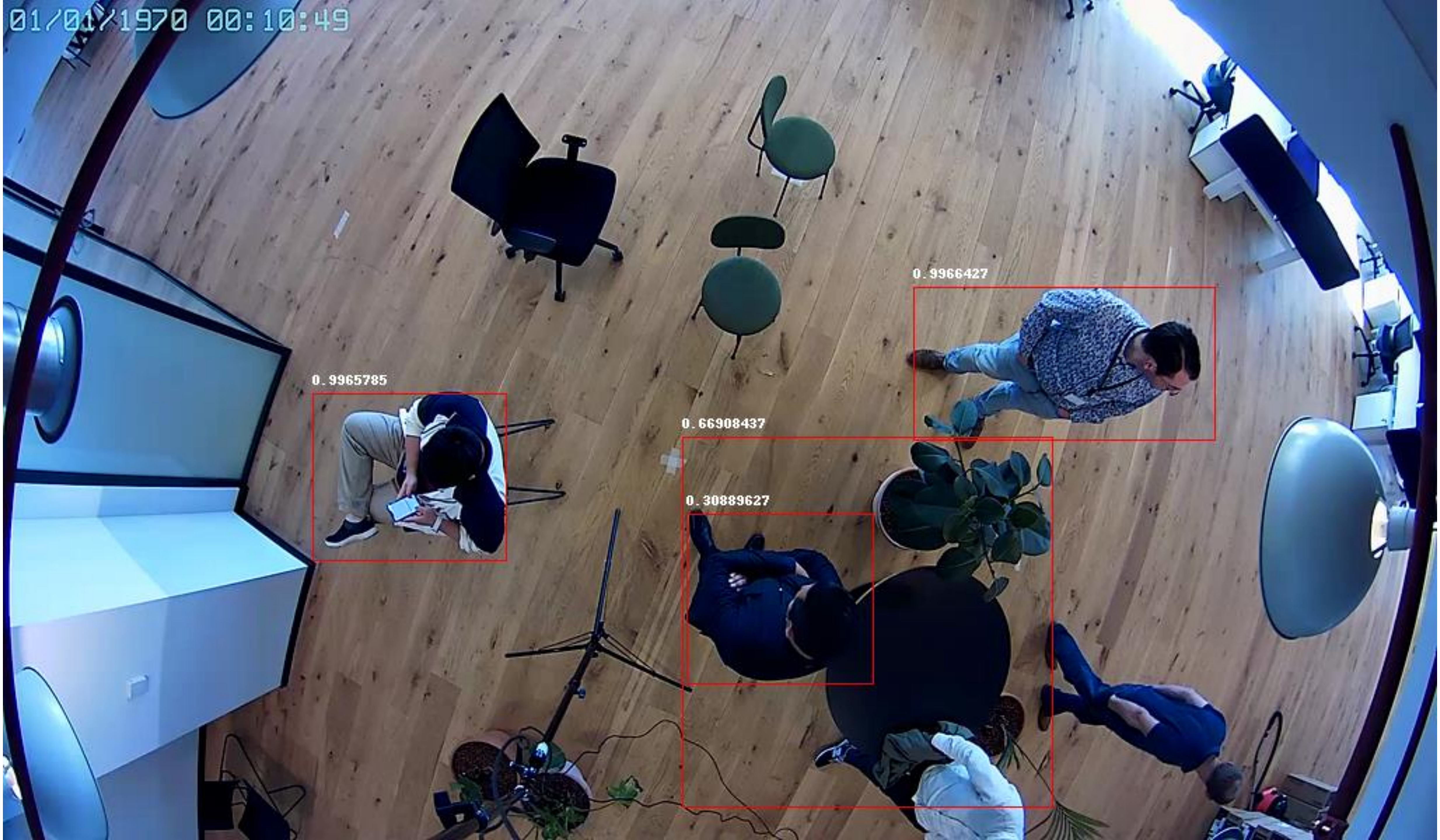

## FASTER_RCNN_RESNET101_COCO




01/01/1970 00:10:24
0.9988193


01/01/1970 00:10:36
0.99630237
0.9888771


01/01/1970 00:10:46
0.99680835
0.9964174
0.420948

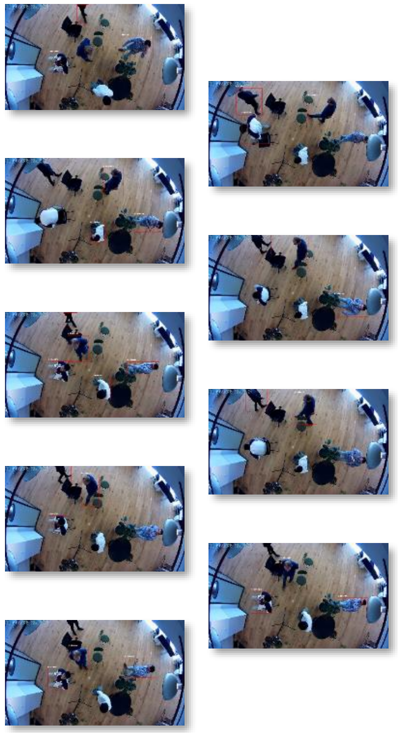

01/01/1970 00:10:49
0.9966427
0.9965785
0.66908437
0.30889627


01/01/1970 00:10:24
0.9988193
01/01/1970 00:10:36
0.99630237
0.9888771
01/01/1970 00:10:46
0.99680835
0.9964174
0.420948
01/01/1970 00:10:49
0.9966427
0.9965785
0.66908437
0.30889627

# Step 2

## Tuning the Model

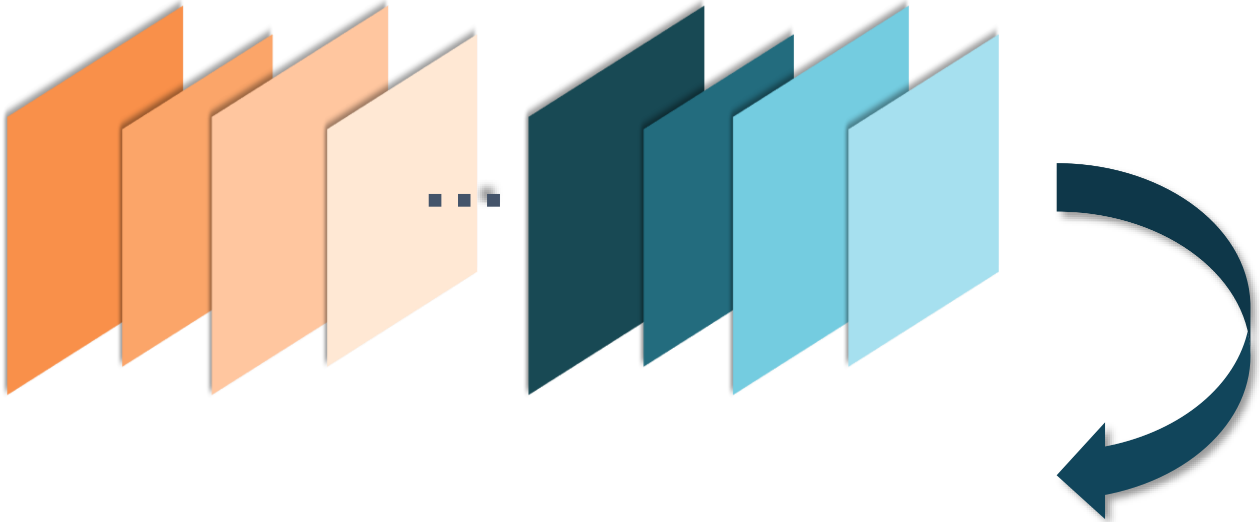
# Tuning the Model Using Transfer Learning



Gather and label new set of sample images (~250)

Crop & create rotated, brightness, quality variations

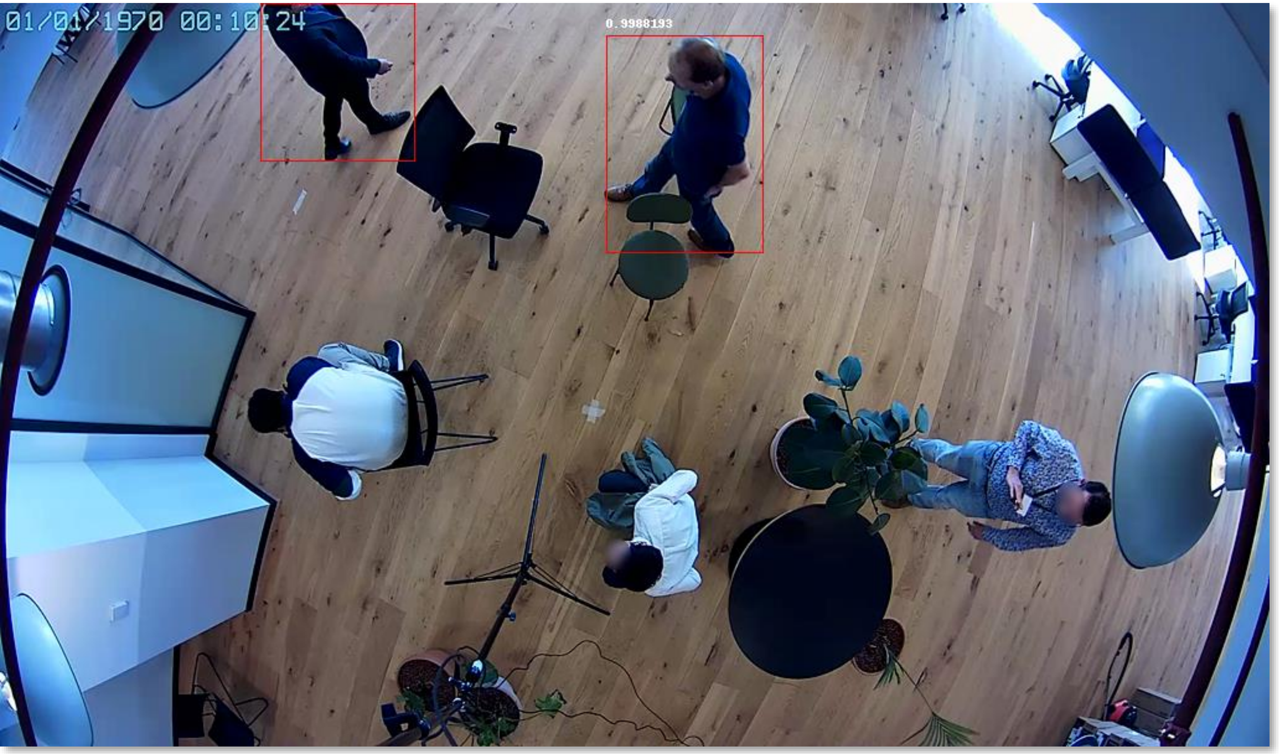
Pre-trained model based on FasterRCNN & ResNet101



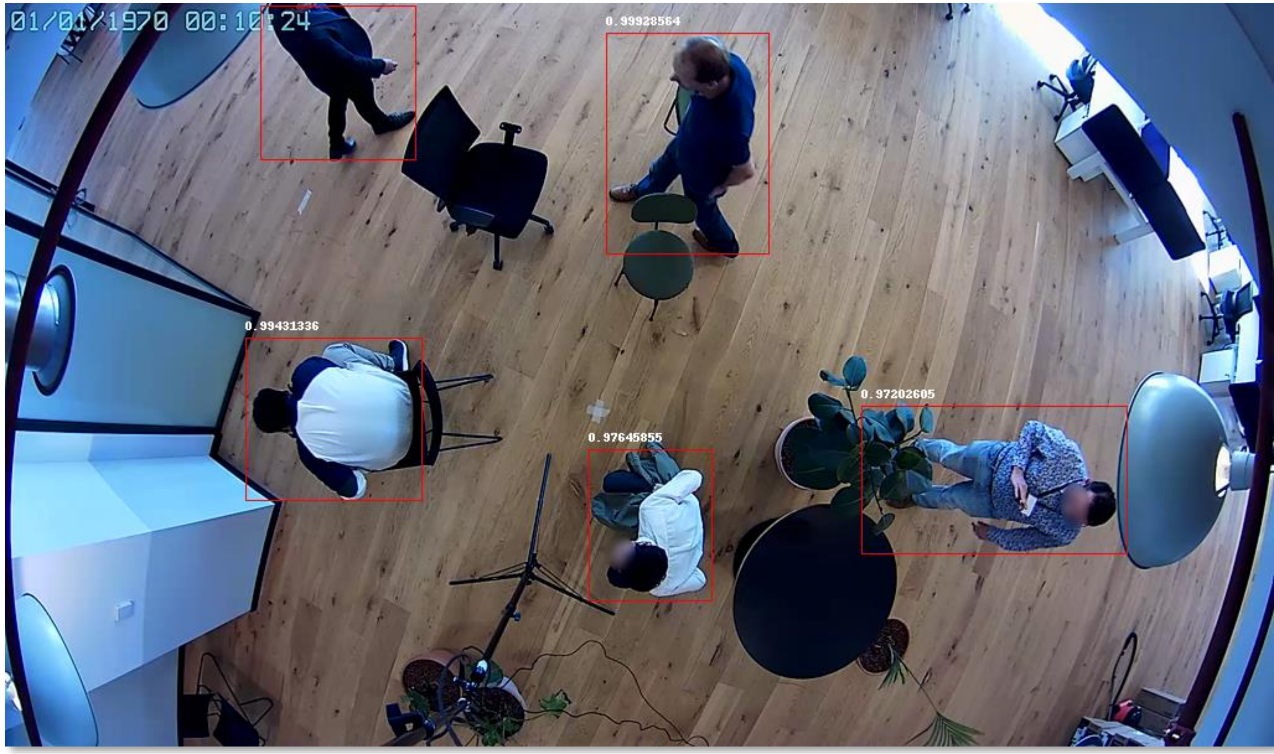
Feed through training pipeline using existing network

Repeat for each image

# After Tuning (1 of 4)

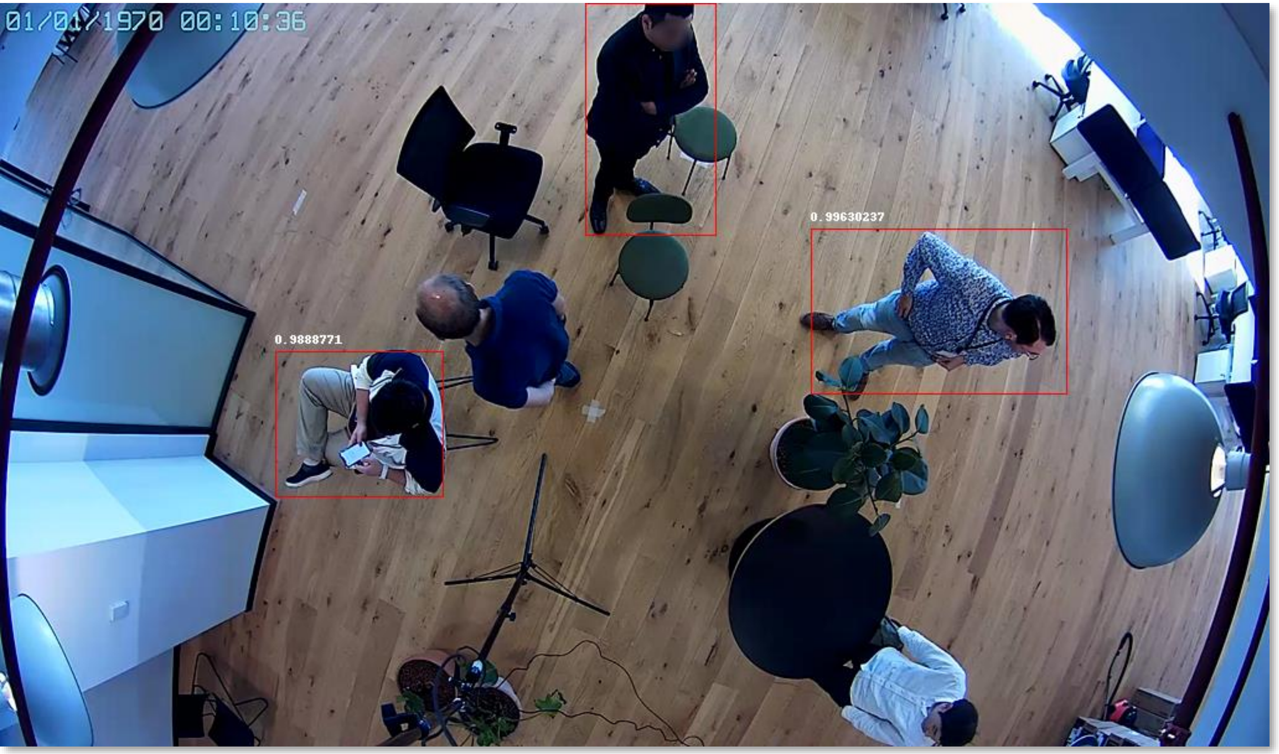



Before

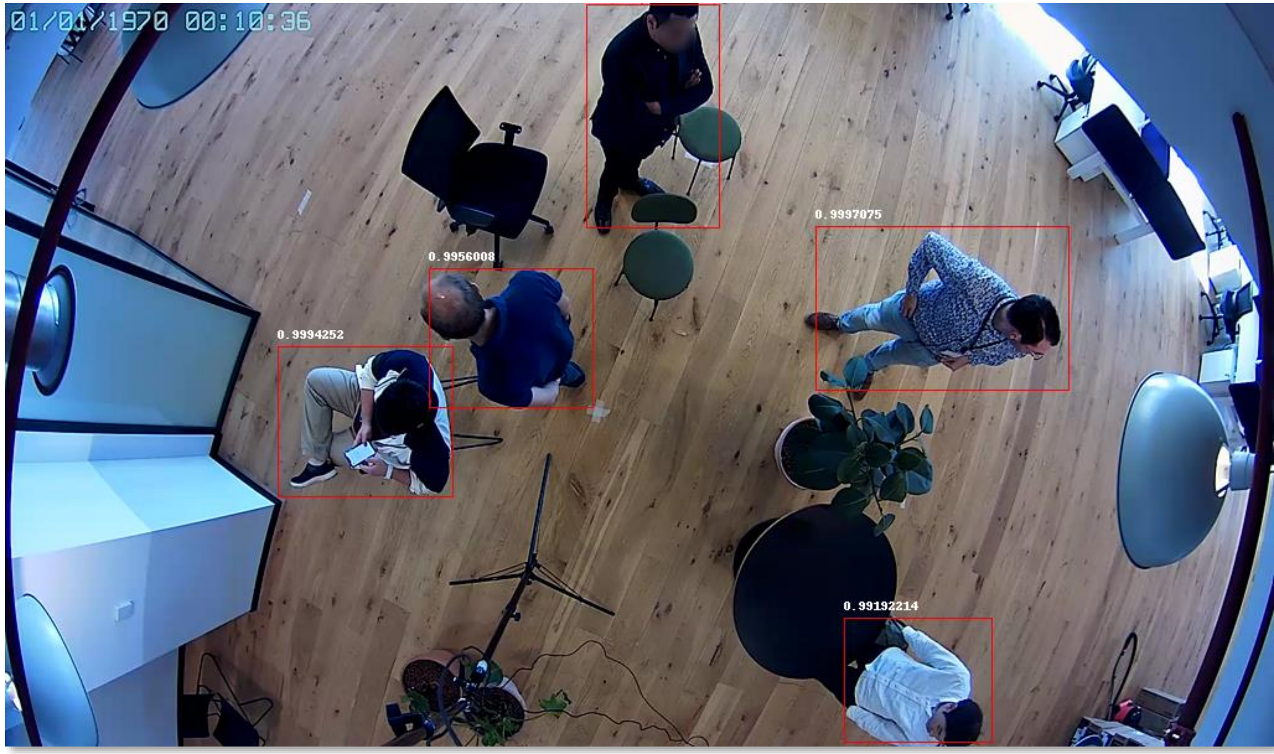



After

# After Tuning (2 of 4)




Before



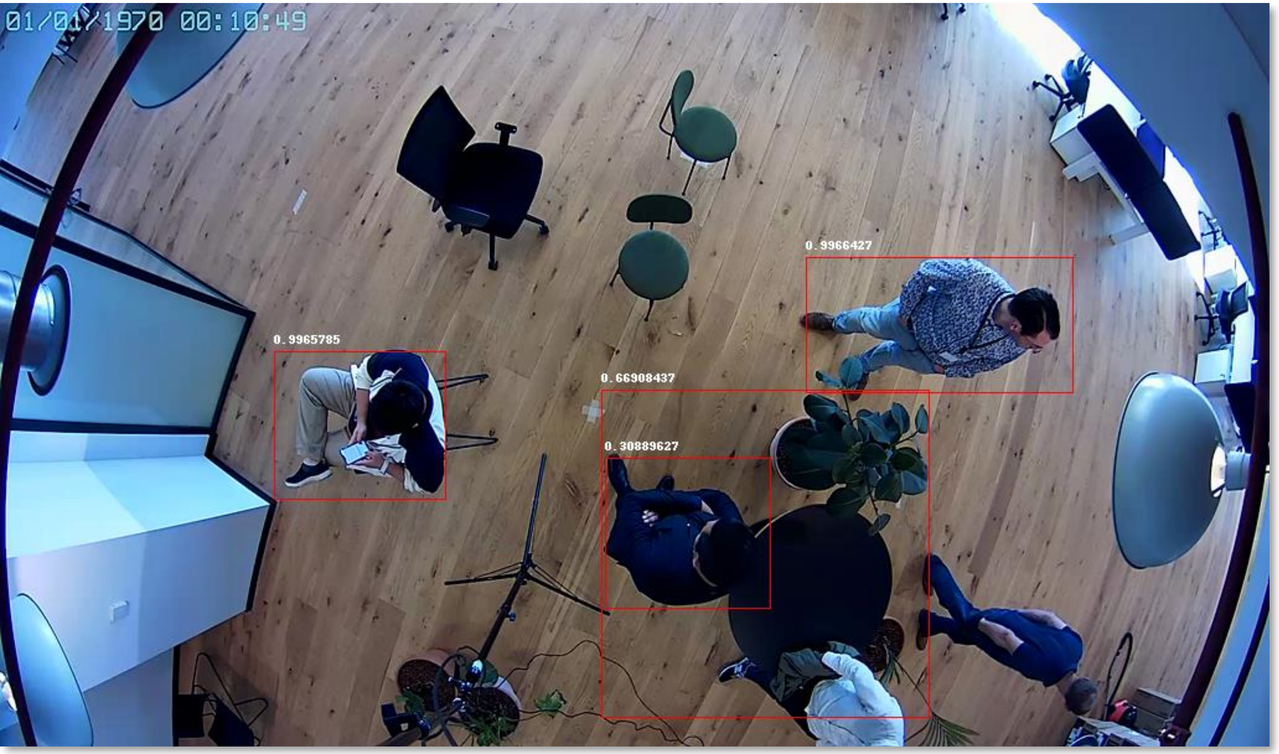

After

# After Tuning (3 of 4)

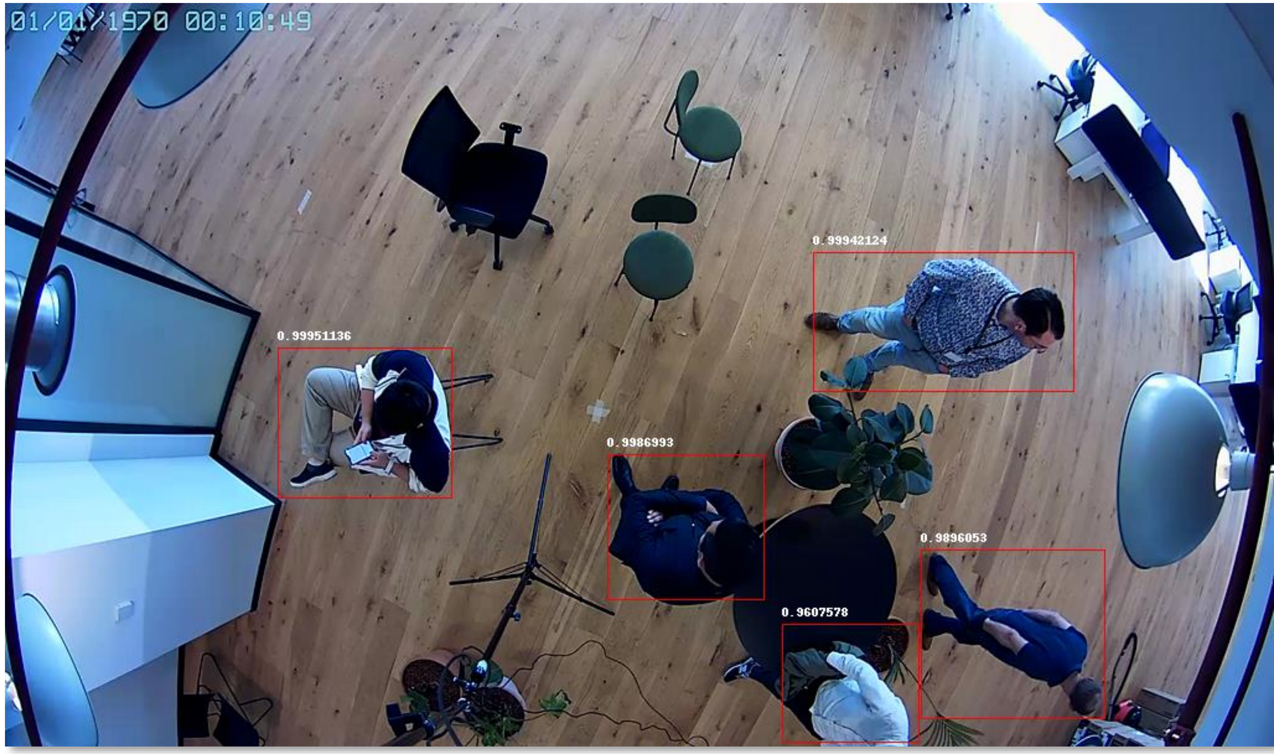

Before


After

# After Tuning (4 of 4)

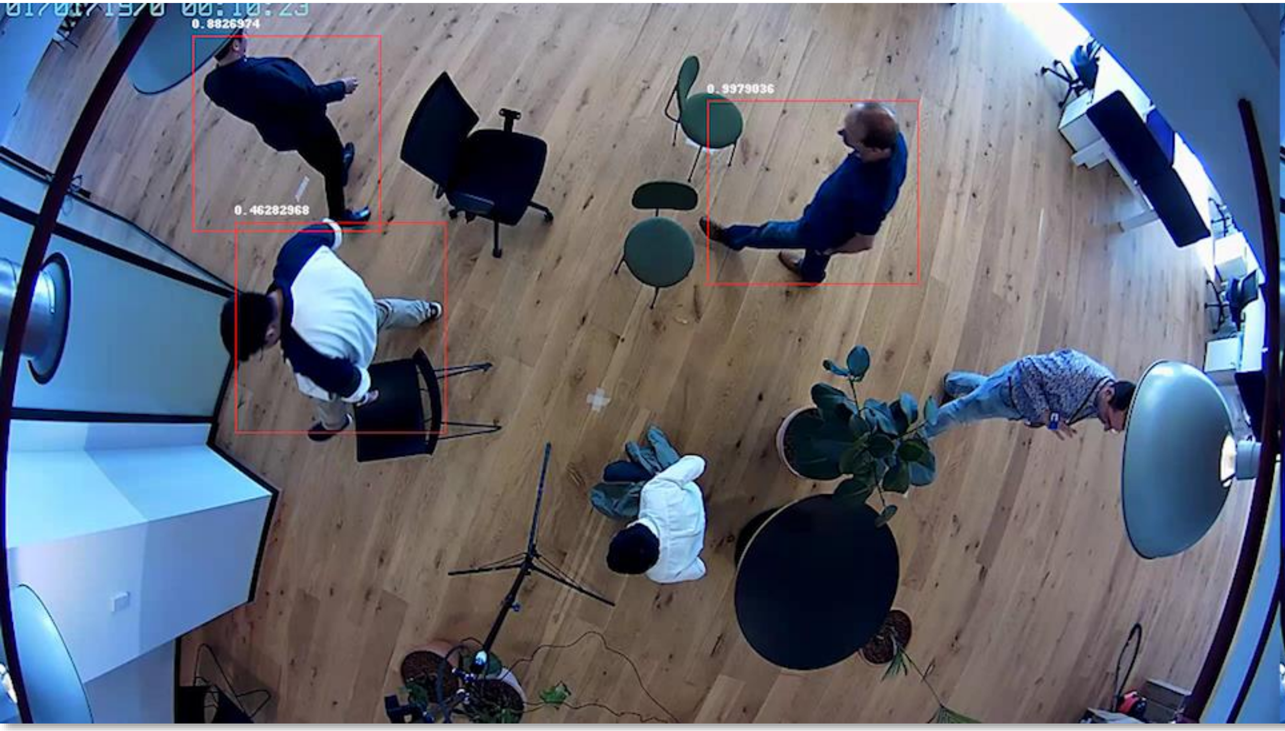



Before

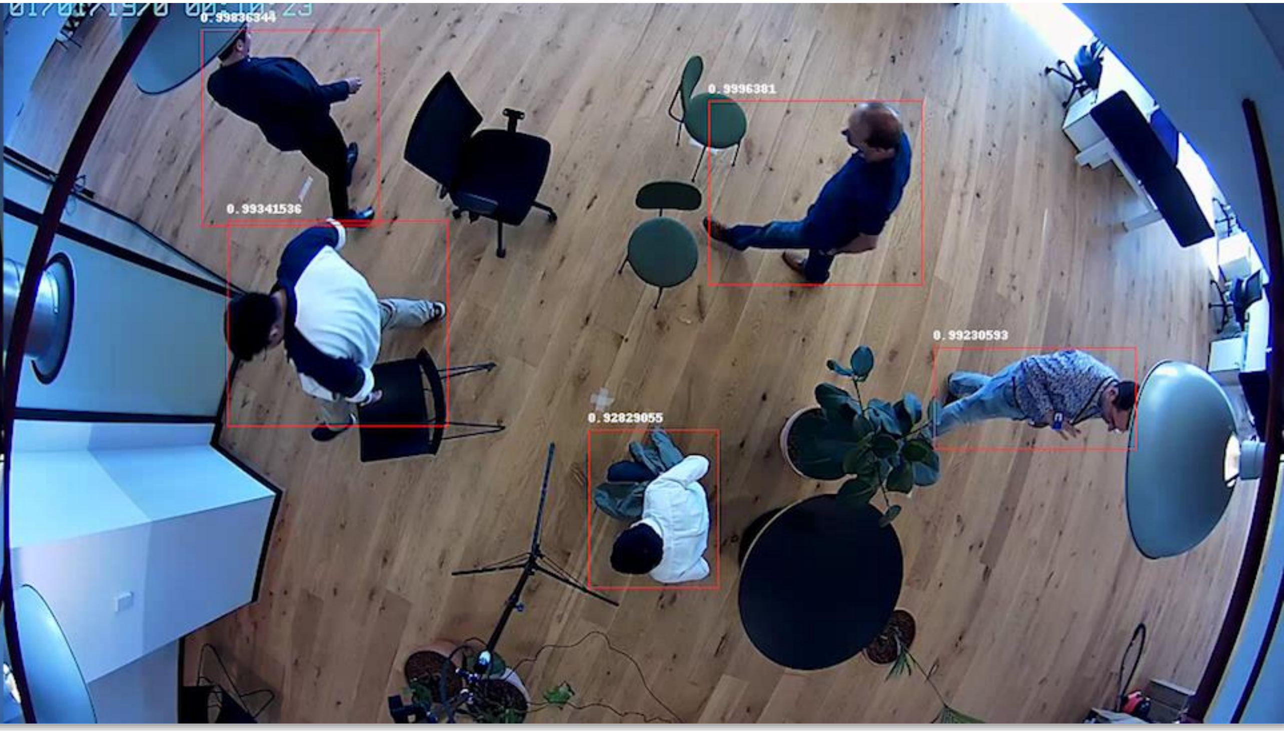



After

# Transfer Learning: Reducing the Corner Cases

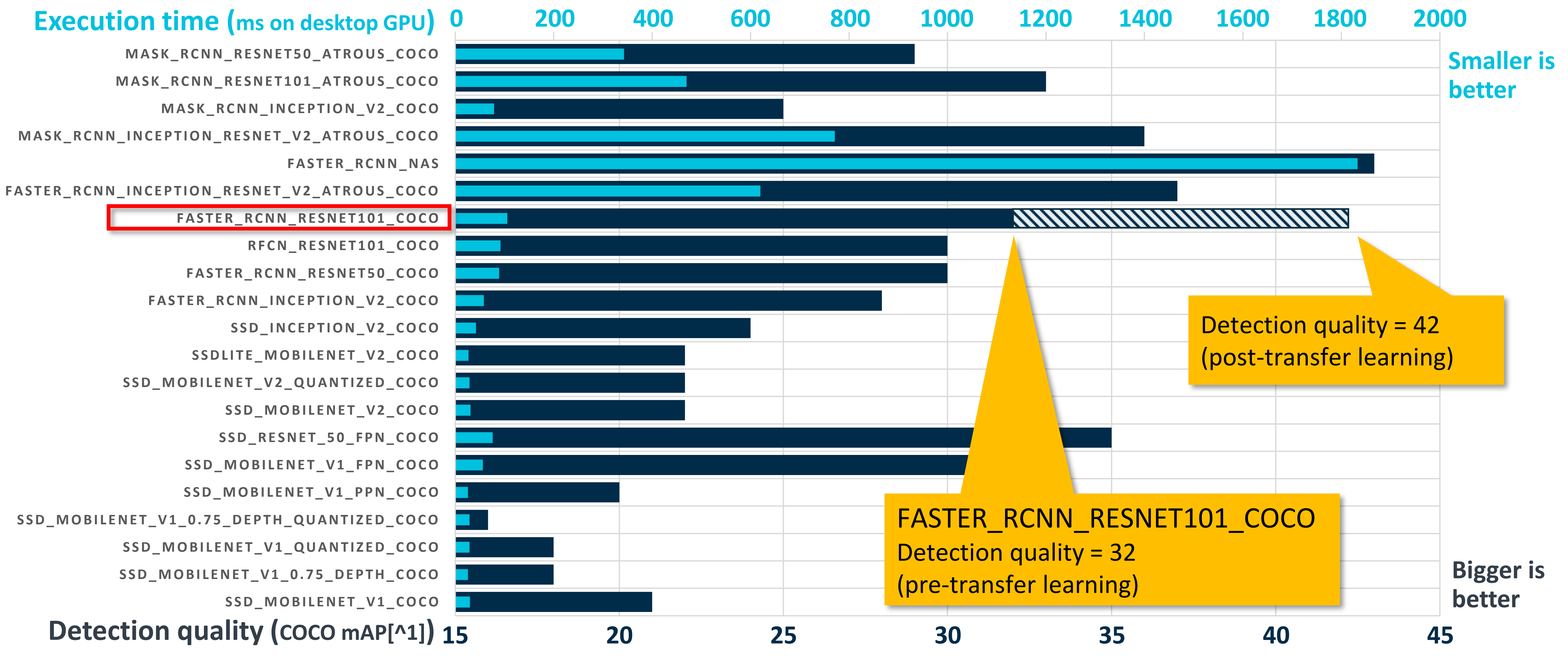


Before



After

# Impact of Transfer Learning



# Product Implications of Tuned Networks

## The retraining process

- Additional training takes ~2 hours
  - Using around 250 images
  - On a portable training rig
- Pipeline allows for tuning to support…
  - corner cases
  - lower quality cameras
  - difficult lighting conditions

## Product considerations

- Significantly reduce false +ve & – ve
- Allows for tuning to specific venues
  - e.g. customising for specific desk and chair types
  - Training pipeline relatively painless compared to retraining the entire model
  - Implies more skilled installation labor

# Step 3

## Applying to the Real World

arm

# Applying to the Real-World

## Working with cameras

### “Everything matters…”

- Model choice key to quality
- Quality of lens and sensor, resolution
- Sharpness, noise, compression
- Lighting & lighting variability

## Real-world product implications

### Installation / configuration

- Complex and skilled, unless…
  - Can we mitigate low quality devices by training with degraded images?
  - Use AI to automate camera setup?
- Camera cost vs features
  - e.g. HDR expensive but can help with variable lighting

# Summary



| | |
|---|---|
| **Choice of model fundamental to successful real-world vision-based use cases** | • As is everything else ☺<br>• Model tuning can provide significant benefits |
| **Useful models often need significant compute** | • Today's off-the-shelf cameras unlikely to have sufficient capability<br>• Aggregating multiple devices into gateways can work well, and this creates real opportunities today |
| **Compute requirement will grow, enabling new use cases** | • More sophisticated models as ML continues to evolve<br>• Edge-based complex action recognition is particularly challenging |

# Useful Resources

## Useful Links

COCO dataset

http://cocodataset.org/

COCO-trained models

https://github.com/tensorflow/models/blob/master/research/object_detection/g3doc/tf2_detection_zoo.md

Arm Insight Platform

https://www.arm.com/products/arm-insight-platform

## 2020 Embedded Vision Summit

*Machine Learning for the Real World: What is Acceptable Accuracy, and How Can You Achieve It?*

Tim Hartley

# Thank You!