FOMO: Real-Time Object Detection on Microcontrollers

Jan Jongboom
CTO
Edge Impulse
Leading development platform for machine learning on edge devices

Launched two years ago... Now 240 new projects daily (!)

40% of these are vision projects

Wait... What? Edge Impulse?
Object Detection

Image classification vs. object detection
Significantly Different Performance (Rpi4)

Image classification vs. object detection

60 fps

2 fps

Face

© 2022 Edge Impulse Inc.
But Why?

"The tiny version of YOLO only uses 516 MB of GPU memory"

Source: https://pjreddie.com/darknet/yolov1/
Object Detection (MobileNet SSD, YOLO)
Replace with single per-region class probability map
Single Per-Region Class Probability What?
Convolutional Image Classification Model

- 64x64 input
- 16x16 after conv
- 4x4 after conv
- 1x1 output

High locality to Zero locality
Loss function that forces the top-left tile here to map to 16x16 top-left tile in the input.
Output = Heatmap
Postprocessing in Software

Not constrained to just bounding boxes!
Heatmap Ratio is Configurable
Flexibility in Input / Output Resolution

320 x 320 input => 80 x 80 output
Flexibility in Input / Output Resolution

96 x 96 input => 12 x 12 output
Centroids

Bounding boxes are an implementation detail

Most of the time you just want to know **where** and **how many** objects there are

FOMO trains on the centroid of an object, w/ loss function allowing some error

Convolutional network, so will still look around the object, but lot less error prone against background
"Equal to an object detector, but bounding boxes are fixed and always square"

Constrained object detection for constrained problems

Works best with similar sized objects, at similar distances

One activation per heat map cell, so objects overlapping each other might be fused together

Few classes? Fine. 100s of objects across 60 classes? Use YOLOv5.
Overlapping Objects?
Fixed and Always Square

But... merging leads to issue here

Potential issue, merge cells?
Centroids Should Have Meaning
Demo (Rpi4 on CPU, 160x160 input, 60 fps)
Demo (Cortex-M7, 96x96 input, 28 fps)
Demo (M1, 160x160, 1000 fps)
Other Cool Features of FOMO

Add-on to any convolutional image network (incl. transfer learning models), so highly configurable

Fully convolutional, just the ratio is set

Can count objects

Can be a segmentation model

Performs much better on small objects than YOLOv5 or MobileNet SSD
Cortex-M7 @ 480MHz: **30 fps** (96x96 MobileNetV2 a=0.1)
Raspberry Pi 4: **60 fps** (160x160 MobileNetV2 a=0.35)
Himax DSP @ 400MHz: **14 fps** (96x96 MobileNetV2 a=0.35)
Cortex-M4F @ 156MHz: **5 fps** (96x96 MobileNetV2 a=0.05)

My Macbook: **1000 fps** :-)

(Can be bolted on other CNNs, e.g. MobileNetV1)
Getting Started

• FOMO is available today for free: https://edgeimpulse.com/fomo

• Very wide range of dev boards, from Cortex-M4F to Jetson Nano.

• Deploy to any device that has a C++ compiler

• Or use your phone!

• Be one of today's 194 new projects!
Vision is such a cool sense

Classification is cool, locality and count matters

Want a 30x increase in performance? Use FOMO!

edgeimpulse.com
Questions?

Full docs:
https://docs.edgeimpulse.com

FOMO
https://docs.edgeimpulse.com/docs/fomo-object-detection-for-constrained-devices

We're hiring!
https://edgeimpulse.com/careers

More questions:
forum.edgeimpulse.com / jan@edgeimpulse.com