Constrained Object Detection on Microcontrollers with FOMO
Agenda

1. What is Edge Impulse?
2. Object Detection
3. Image Segmentation
4. Constrained Object Detection
5. FOMO Use Cases and Limitations
6. Live Demo
Edge Impulse
Go to market faster with confidence
Object Detection
# Image Classification

<table>
<thead>
<tr>
<th></th>
<th>Background</th>
<th>Capacitor</th>
<th>Diode</th>
<th>LED</th>
<th>Resistor</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>First Image</strong></td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
<td>1.00</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th></th>
<th>Background</th>
<th>Capacitor</th>
<th>Diode</th>
<th>LED</th>
<th>Resistor</th>
</tr>
</thead>
<tbody>
<tr>
<td><strong>Second Image</strong></td>
<td>0.00</td>
<td>1.00</td>
<td>0.00</td>
<td>0.00</td>
<td>0.00</td>
</tr>
</tbody>
</table>
## Image Classification

<table>
<thead>
<tr>
<th>Background</th>
<th>Capacitor</th>
<th>Diode</th>
<th>LED</th>
<th>Resistor</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00</td>
<td>0.04</td>
<td>0.00</td>
<td>0.28</td>
<td>0.68</td>
</tr>
</tbody>
</table>

<table>
<thead>
<tr>
<th>Background</th>
<th>Capacitor</th>
<th>Diode</th>
<th>LED</th>
<th>Resistor</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.00</td>
<td>0.77</td>
<td>0.00</td>
<td>0.00</td>
<td>0.23</td>
</tr>
</tbody>
</table>
Definitions

• **Image Classification**: Predict class of object in an image

• **Object Localization**: Locate presence of object(s) in an image

• **Object Detection**: Locate and classify object(s) in an image
Object Detection
Object Detection

Bounding box

- Cat: 0.98
- Dog: 0.83
48 x 100 ms = 4.8 seconds!
(~0.208 fps)
Object Detection Model

- **Object 1**
  - Class: dog (0.92)
  - Bounding box
    - $(x_1, y_1)$
    - $(w_1, h_1)$

- **Object 2**
  - Class: toy (0.85)
  - Bounding box
    - $(x_2, y_2)$
    - $(w_2, h_2)$

- **Object 3**
  - Class: ball (0.77)
  - Bounding box
    - $(x_3, y_3)$
    - $(w_3, h_3)$
Region-based CNN (R-CNN)

Class predictions:
- background
- ball
- dog
- toy

Regressor
\(d_x, d_y, d_w, d_h\)
Single Shot MultiBox Detector (SSD)

e.g. VGG19, MobileNet

Class predictions + boundary boxes
Image Segmentation
Image Segmentation
Image Segmentation

threshold = 145
seg = (img < threshold).astype(int)
Image Segmentation
Image Segmentation Use Case
Constrained Object Detection
MobileNet V2

240x240

Class predictions for image
Faster Objects, More Objects (FOMO)

- **Height and width are each divided by 8 (default)**
  - Convolutional layer(s)
  - Bottleneck residual block 1
  - Bottleneck residual block 2
  - Bottleneck residual block 3
  - 16 feature maps, each 30x30 cells

- **2D convolution with 1x1 kernel is used**
  - Fully Connected layer
  - Fully Connected layer
  - Fully Connected layer
  - Softmax

- **Class predictions per cell**
- Looks like segmentation of feature maps

- **240x240**
  - Weights in residual blocks are pre-trained from ImageNet
Faster Objects, More Objects (FOMO)

Each cell is given scores:
- $P(\text{background})$
- $P(\text{ball})$
- $P(\text{dog})$
- $P(\text{toy})$
Faster Objects, More Objects (FOMO)
Faster Objects, More Objects (FOMO)

Example: screws

- Grayscale
- Image: 96x96
- Feature maps: 12x12
Faster Objects, More Objects (FOMO)

Example: screws
- Grayscale
- Image: 96x96
- Feature maps: 12x12
Faster Objects, More Objects (FOMO)

Example: screws
- Grayscale
- Image: 96x96
- Feature maps: 12x12

Neighboring cells with same class are removed (leaving highest scores)
FOMO Ground Truth

Example: screws
- Grayscale
- Image: 96x96
- Feature maps: 12x12

User draws bounding boxes, tool picks cell with centroid of bounding box
FOMO Ground Truth

Example: screws
  - Grayscale
  - Image: 96x96
  - Feature maps: 12x12

User draws bounding boxes, tool picks cell with centroid of bounding box

Those cells are now representatives of that class
FOMO

Uses

+ Limitations
Use Cases

Want to know **where** and **how many** objects there are

Recommendations for success:

- Objects are same size
- Objects are square/round
- Objects take up 1 cell

Very fast!

- Cortex-M7 at 480 MHz
- 240x240 image input
- 30 fps
- 245K RAM

https://matpalm.com/blog/counting_bees/
Limitations

- Each cell has its own classifier
- Small objects may be missed
- Neighboring objects may get lumped together
- Ends of oblong objects may be ignored
- Lots of objects/classes: use YOLOv5
Demo

studio.edgeimpulse.com/public/89461/latest
studio.edgeimpulse.com/public/104110/latest
Getting Started

docs.edgeimpulse.com/docs/

- Tutorials > Counting objects using FOMO
- Various supported dev boards