AssignmentReport-Group1

March 24, 2022

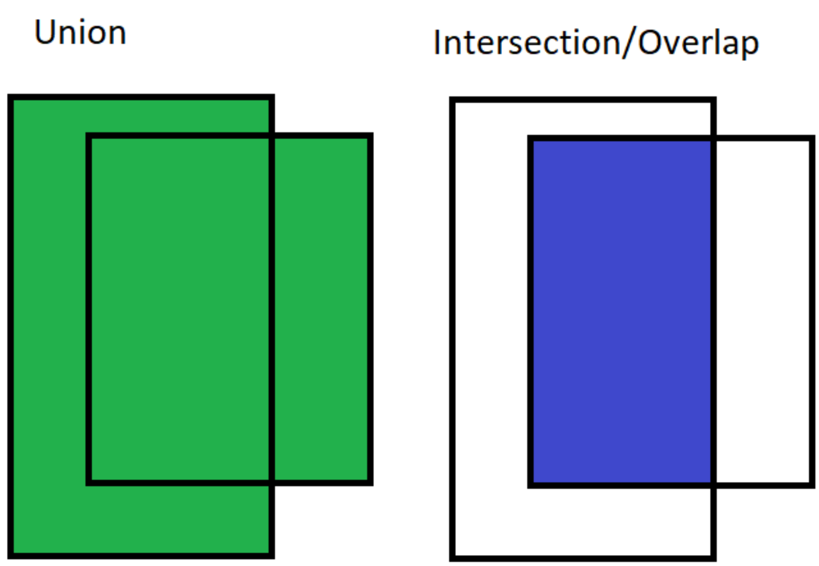
1 Assignment 4 Report - Sigmund S. Mestad

2 Task 1

2.1 task 1a)

Intersection over Union (IoU) is a measure of how much our predicted boundry overlaps with the ground truth. IoU is defined as

$$IoU = \frac{Area\,of\,overlap}{Area\,of\,union}$$



2.2 task 1b)

$$Precision = \frac{True\,Positive}{All\,Positive} = \frac{True\,Positive}{True\,Positive + False\,Positive}$$

$$Recall = \frac{True\,Positive}{All\,Cases} = \frac{True\,Positive}{True\,Positive + False\,Negative}$$

In terms of object detection a true positive is when the model correctly detects and classifies an object. A false positive, means the model detects and classifies an object which is not correct.

2.3 task 1c)

Class 1:

Recall	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
Precision	1.0	1.0	1.0	1.0	1.0	1.0	1.0	0.5	0.5	0.5	0.2

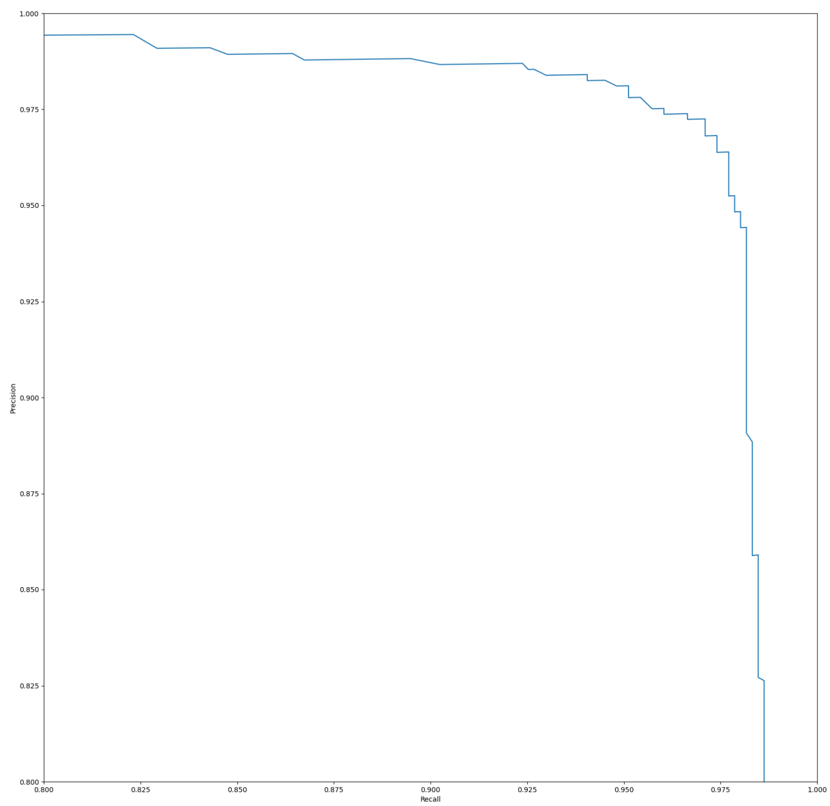
$AP_1 = 1/11 * (7 * 1 + 3 * 0.5 + 1 * 0.2) = 0.79$
Class 2:

Recall	0.0	0.1	0.2	0.3	0.4	0.5	0.6	0.7	0.8	0.9	1.0
Precision	1.0	1.0	1.0	1.0	0.8	0.6	0.6	0.5	0.5	0.5	0.2

$AP_2 = 1/11 * (4 * 1 + 1 * 0.8 + 2 * 0.6 + 3 * 0.5 + 1 * 0.2) = 0.70$

3 Task 2

3.0.1 Task 2f)



4 Task 3

4.0.1 Task 3a)

The filtering operation is called **non-maximum supression** (nms).

4.0.2 Task 3b)

The statement is **false**. The deeper layers has a lower resulution and is responsible for detecting the larger objects.

4.0.3 Task 3c)

SSD use different bounding box aspect ratios at the same spatial location because we want our model to detect objects of different shapes. E.g. a tall bounding box would match well with a tree or a pedestrian, but not with a car or a dog. Using different default boxes at the same location gives more predictions, but encourage each prediction to predict shapes closer to the corresponding default box. This leads to more diverse and more stable predictions during training.

4.0.4 Task 3d)

The main difference between SSD and YOLO is that SSD adds several feature layers which decrease in size progressively and allow predictions of detections at multiple scales. YOLO on the other hand, operates on a single scale feature map.

4.0.5 Task 3e)

6 boxes at 38x38 locations gives in total $6 * 38 * 38 = 8664$ boxes.

4.0.6 Task 3f)

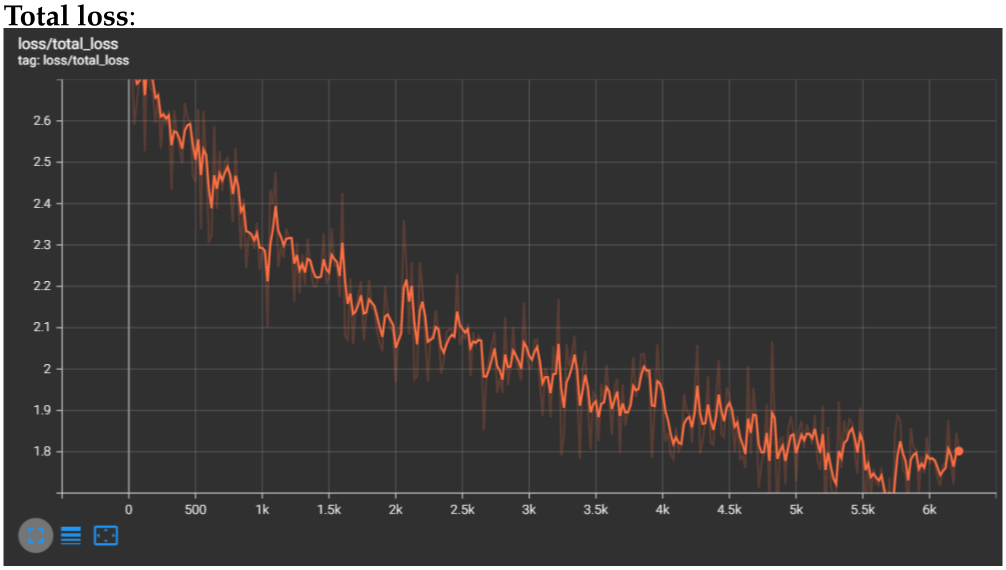
Total number of anchor boxes:

$$6 * 38 * 38 + 6 * 19 * 19 + 6 * 10 * 10 + 6 * 5 * 5 + 6 * 3 * 3 + 6 * 1 * 1 = 11640$$

5 Task 4

5.1 Task 4b)

Total loss:



My final mean Average Precision was 0.76067

5.2 Task 4c)

Changes/Improvements:
- Optimizer: SGD -> Adam
- Learning rate: 5e-3 -> 2.6-e4
- Weight decay: 0.0005 -> 0
- Batch size: 32 -> 64
- Data augmentation: RandomHorizontalFlip, RandomSampleCrop, Resize

- Weight initialization: Added gain=nn.init.calculate_gain('relu') (=1.414)
- Added 'BatchNorm2d' after each 'Conv2d' layer
- Added a few dropout layers in the first feature extraction layer.
- Added a few more layers in the first feature extraction layer.
- Total number of parameters: 3563634

Final mean Average Precision: 0.85524

```
Epoch 41: 100%|                                                                    | 156/156 [01:42<00:00,  1.53it/s]
Evaluating on dataset: 100%|                                                       | 16/16 [00:03<00:00,  4.12it/s]
Loading and preparing results...
Converting ndarray to lists...
(9748, 7)
0/9748
DONE (t=0.02s)
creating index...
index created!
Running per image evaluation...
Evaluate annotation type *bbox*
DONE (t=2.13s).
Accumulating evaluation results...
DONE (t=0.34s).
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.64983
 Average Precision  (AP) @[ IoU=0.50      | area=   all | maxDets=100 ] = 0.85524
 Average Precision  (AP) @[ IoU=0.75      | area=   all | maxDets=100 ] = 0.78674
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.50281
 Average Precision  (AP) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.76110
 Average Precision  (AP) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = -1.00000
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=  1 ] = 0.44129
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets= 10 ] = 0.68362
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=   all | maxDets=100 ] = 0.68362
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= small | maxDets=100 ] = 0.52849
 Average Recall     (AR) @[ IoU=0.50:0.95 | area=medium | maxDets=100 ] = 0.79904
 Average Recall     (AR) @[ IoU=0.50:0.95 | area= large | maxDets=100 ] = -1.00000
```

5.3 Task 4d)

- Pixel values of the 25 center points for the anchor boxes with feature map with resolution 5x5 and stride 64:
  - Assumin the feature map is centered over the picture, the center points in the middle will be in the middle of the picture (pixel 150). Adding or substracting the stride(64 gives the other values).

(22, 22)	(22,86)	(22, 150	(22,214)	(22,278)
(86,22)	(86,86)	(86, 150)	(86,214)	(86,278)
(150,22)	(150,86)	(150,150)	(150,214)	(150,278)
(214,22)	(214,86)	(214, 150)	(214,214)	(214,278)
(278,22)	(278,86)	(278, 150)	(278,214)	(278,278)

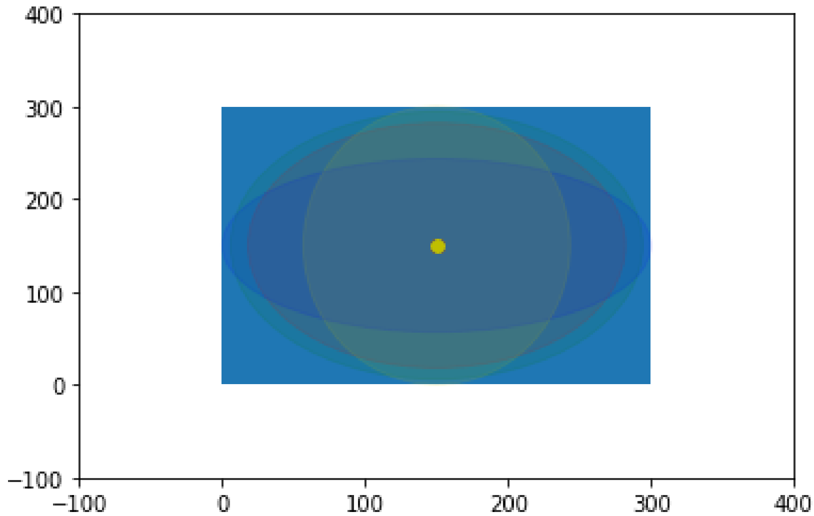
- Sizes:

[20]:
```python
from math import sqrt
min_size = 213
next_min_size=264
print('Size for squared box 1: 213x213')
s = sqrt(min_size*next_min_size)
print('Size for squared box 2: {:.0f}x{:.0f}'.format(s,s))
for aspect_ratio in [2, 3]:
  print('Sizes for aspect ration = {}:'.format(aspect_ratio))
  print('\t - {:.0f}x{:.0f}'.format(min(300, min_size*sqrt(aspect_ratio)),␣
 ↪min(300,min_size/sqrt(aspect_ratio))))
```

```
print('\t - {:.0f}x{:.0f}'.format(min(300, min_size/sqrt(aspect_ratio)),
min(300,min_size*sqrt(aspect_ratio))))
```

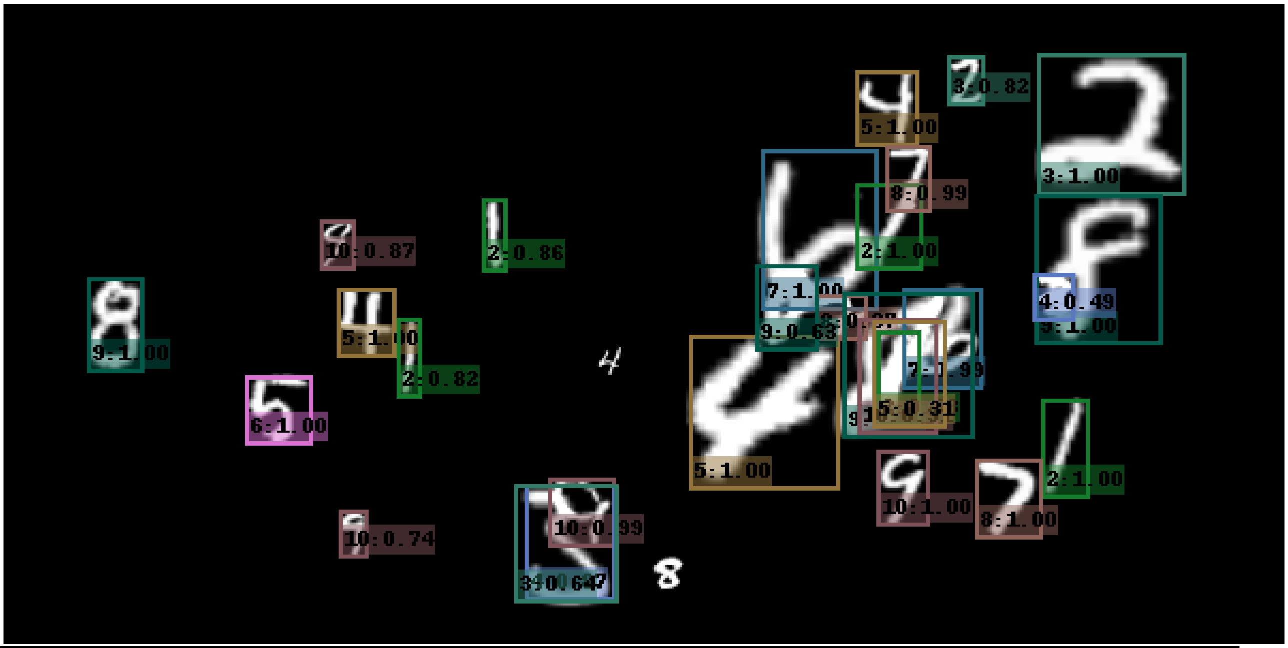
```
Size for squared box 1: 213x213
Size for squared box 2: 237x237
Sizes for aspect ration = 2:
         - 300x151
         - 151x300
Sizes for aspect ration = 3:
         - 300x123
         - 123x300
```

Here is the plot from visualize_priors. To be hones I'm don't understand what it's trying to visualize.



5.4 Task 4e)

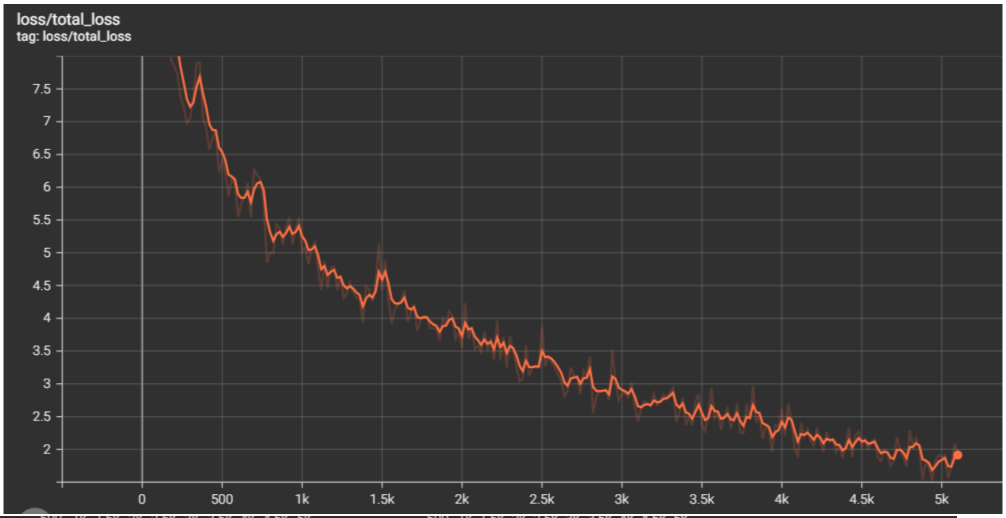
The model seemed to have problem detecting some of the smaller digits.

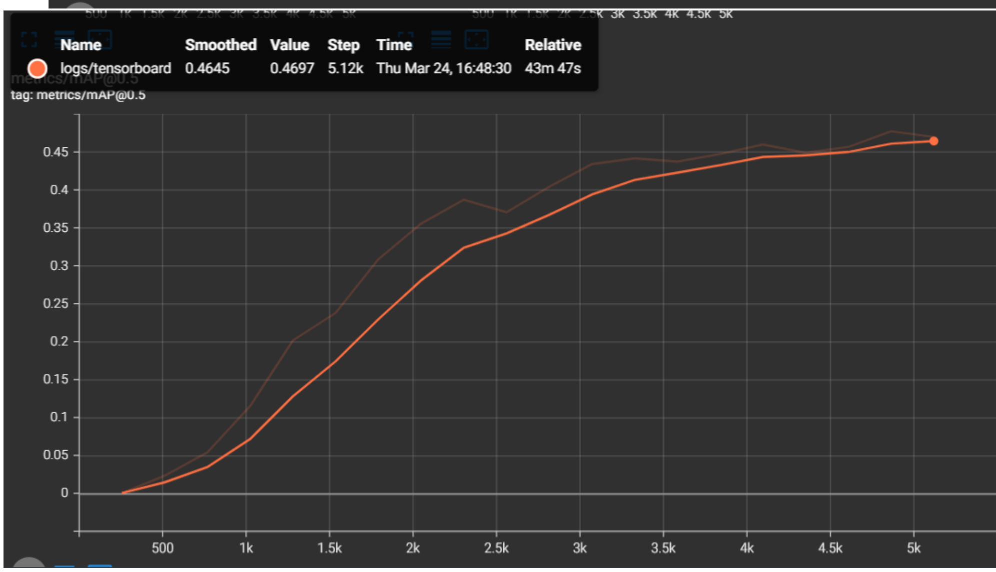


Note: The classes doesn't correspond to the digits, so class 1 is '0', class 2 is '1' etc.

5.5 Task 4f)

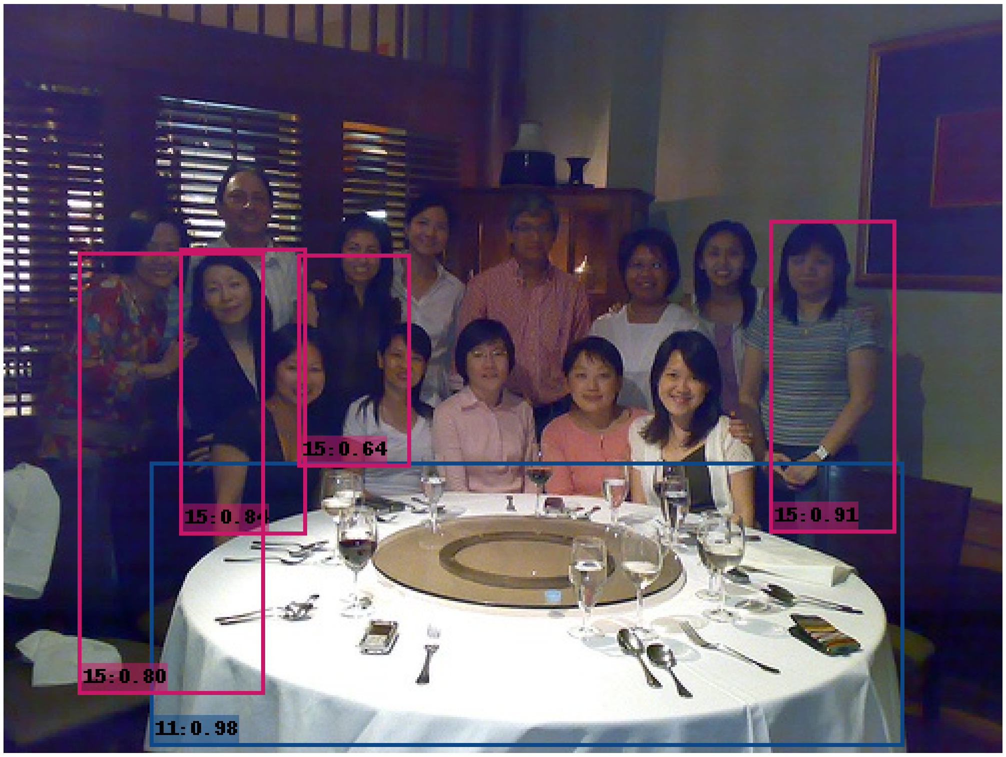
Total loss and mAP (0.497)

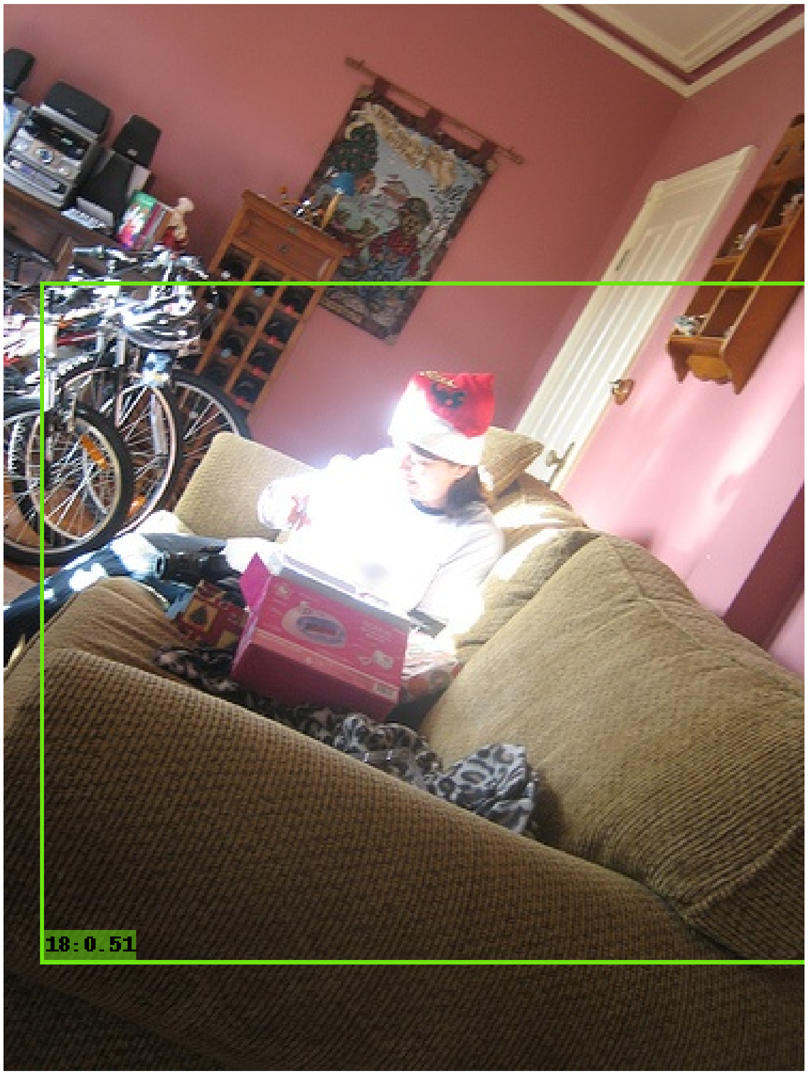

loss/total_loss
tag: loss/total_loss
7.5
7
6.5
6
5.5
5
4.5
4
3.5
3
2.5
2
0
500
1k
1.5k
2k
2.5k
3k
3.5k
4k
4.5k
5k


Name Smoothed Value Step Time Relative
logs/tensorboard 0.4645 0.4697 5.12k Thu Mar 24, 16:48:30 43m 47s
tag: metrics/mAP@0.5
0.45
0.4
0.35
0.3
0.25
0.2
0.15
0.1
0.05
0
500
1k
1.5k
2k
2.5k
3k
3.5k
4k
4.5k
5k


AGR
15: 0.93
20: 0.35


:1.00


15:0.64
15:0.84
15:0.91
15:0.80
11:0.98


18: 0.51


5:0.55