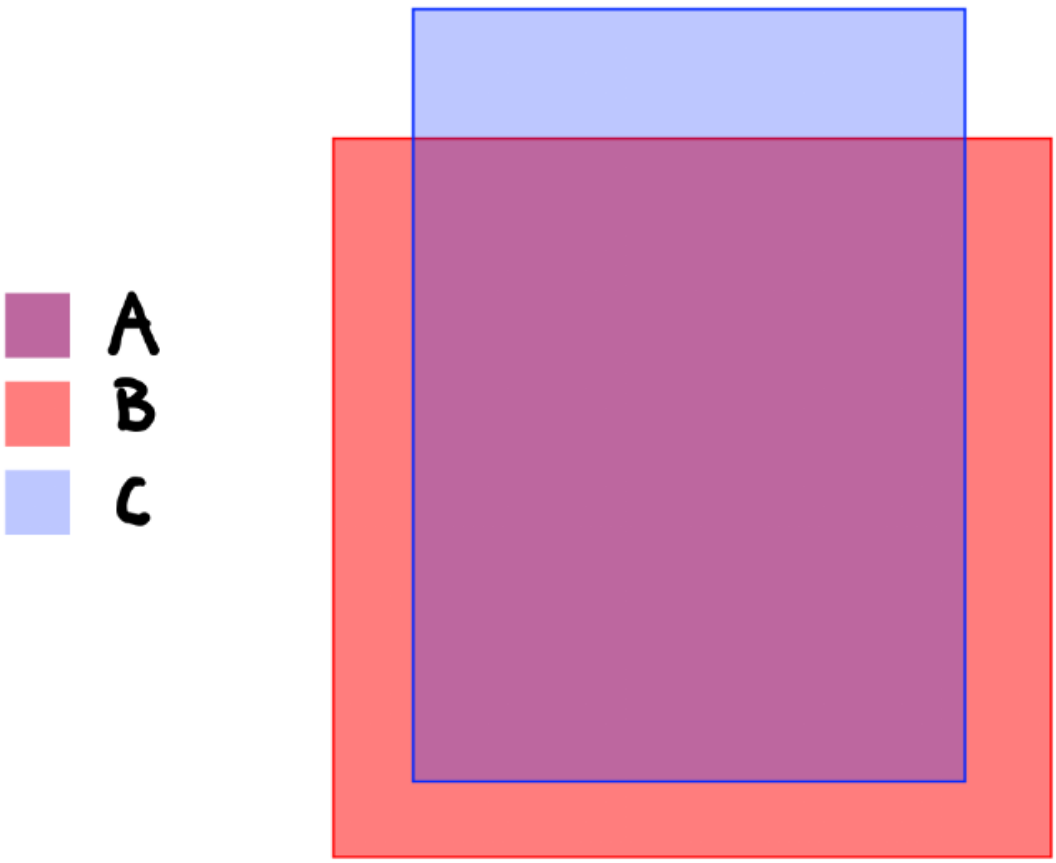
AssignmentReport-Group7

March 21, 2022

1 Task 1

1.1 task 1a)

Intersection over union is used in measuring precision. It is particularly useful when we want to measure how well a neural network is able to separate objects by using bounding boxes.



In intersection over union, we measure how much the predicted bounding box overlaps with the real bounding box. Say we have the red bounding box which is our prediction, and the blue one

is the ground truth. To measure how well the neural network is able to predict this bounding box, we look at the ratio of the intersection of the two boxes (area A), divided by the union of the two boxes (areas A+B+C).

1.2 task 1b)

True positive (TP): We have predicted a result as a positive, and the ground truth is also positive (correct prediction).

False positive (FP): We have predicted the result as a positive, but the ground truth is actually negative (incorrect prediction).

Precision: Out of all the positives we report, how many of them are really positive? i.e. how many of our positive predictions are correct?

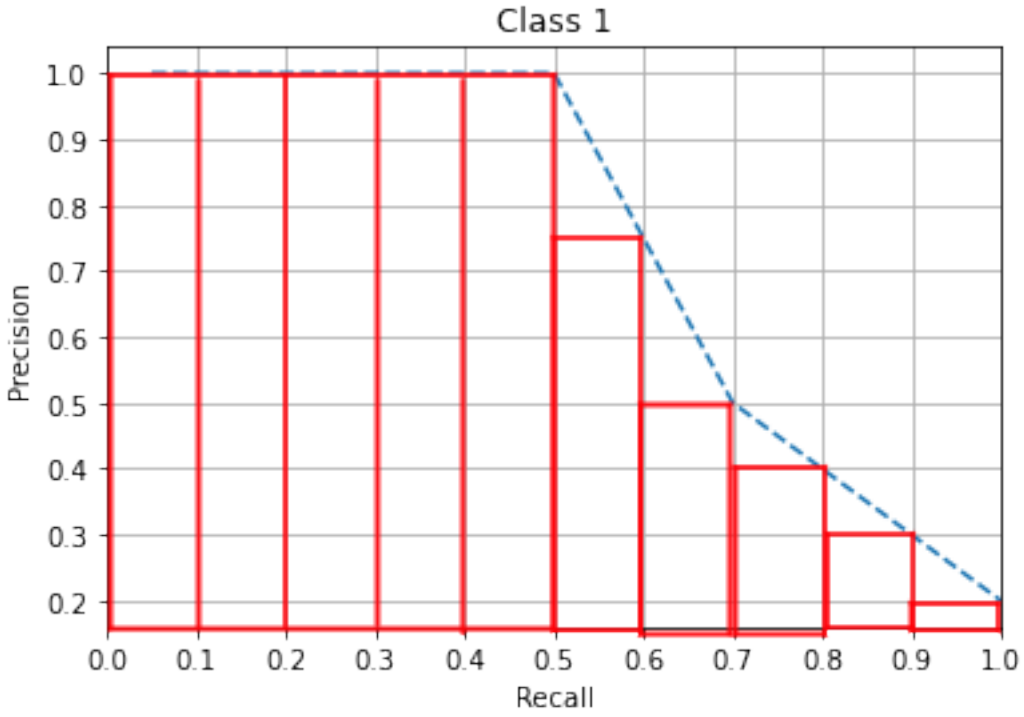
$$Precision = \frac{TP}{TP + FP}$$

Recall: How well are we able to find all the positives? High recall means we have few false negatives (FN), i.e we are able to catch almost all positive cases.

$$Recall = \frac{TP}{TP + FN}$$

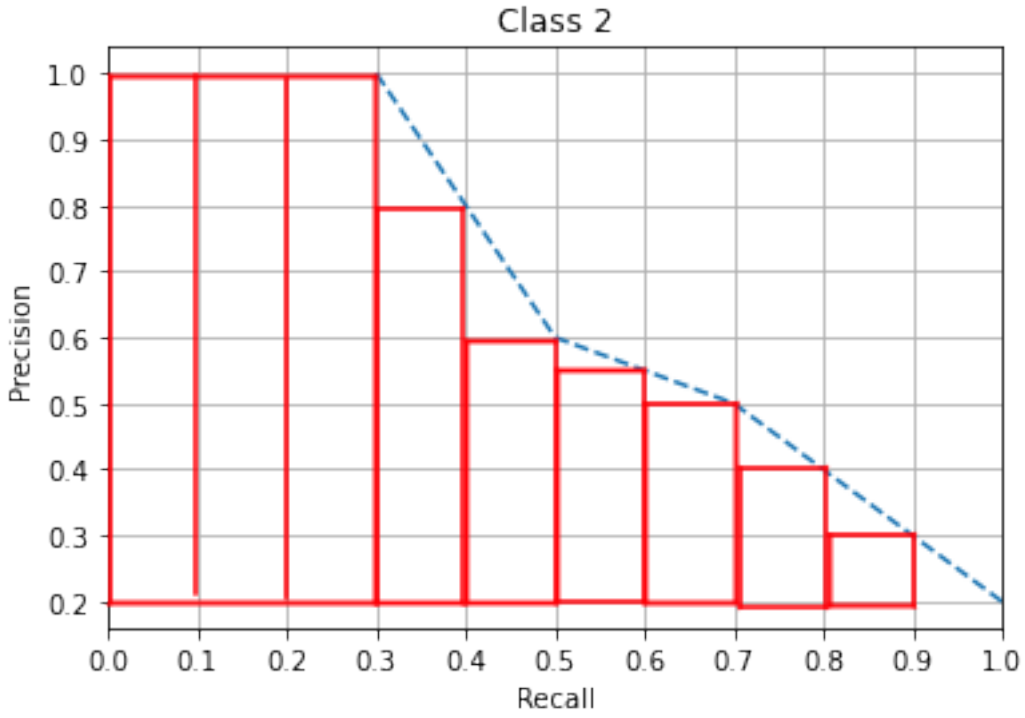
1.3 task 1c)

Here I have plotted the precision recall curves for both classes. The red boxes are the boxes over which we sum AP.



For class 1, we have:

$$AP = \frac{1}{11} \sum_{r \in \{0.0, \cdots 1.0\}} AP_r = \frac{1}{11}(5 \cdot 1.0 + 0.75 + 0.5 + 0.4 + 0.3 + 0.2) = \frac{7.15}{11} = 0.65$$



For class 2:

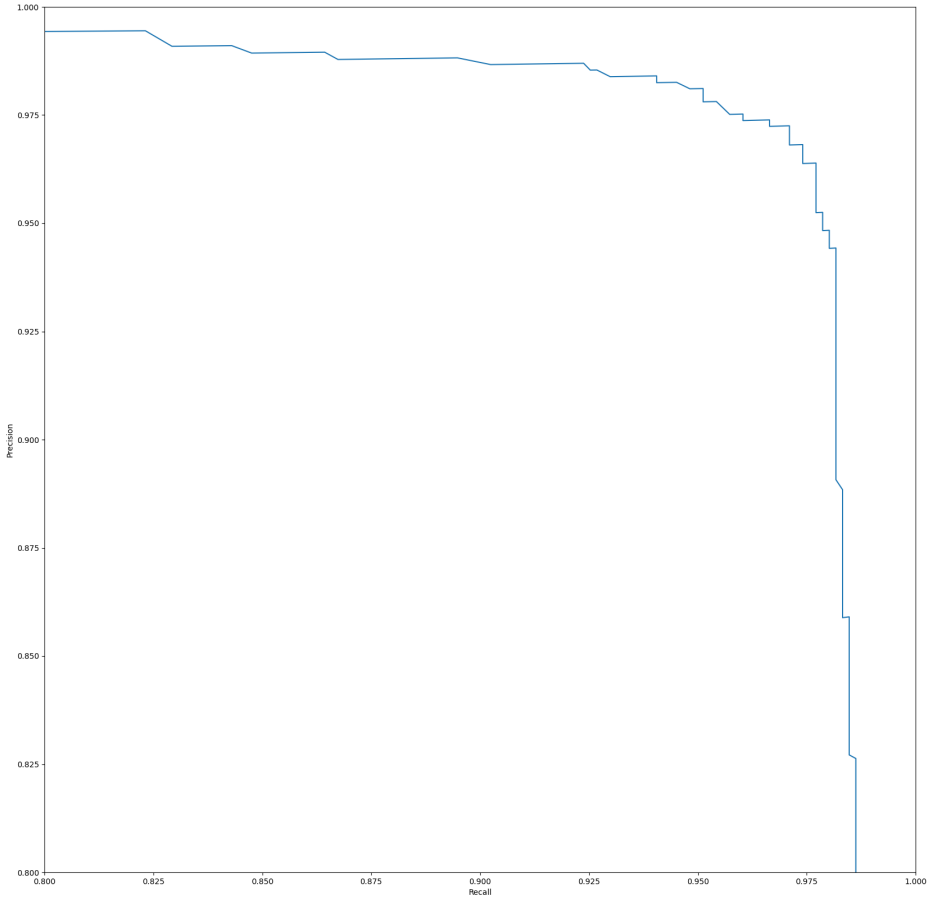
$$AP = \frac{1}{11}(3 \cdot 1.0 + 0.8 + 0.6 + 0.55 + 0.5 + 0.4 + 0.3) = \frac{6.25}{11} = 0.568$$

The mean average precision is the mean of the two APs:

$$mAP = \frac{1}{2}\frac{6.25 + 7.15}{11} = 0.609$$

2 Task 2

2.1 Task 2f)



3 Task 3

3.1 Task 3a)

This is called **non-max suppression**. It removes duplicate predictions that point to the same object by picking only the prediction with the highest confidence score (as long as it has an IoU over a certain threshold).

3.2 Task 3b)

This is false: The earlier layers have a higher resolution, and high-resolution feature maps are better at detecting small objects.

3.3 Task 3c)

It might make sense to use a different set of bounding boxes depending on what class we are trying to identify. For example, a wide bounding box would make sense for a car but not for a pedestrian. By using anchors, we are essentially saying "Since we are looking for either cars or pedestrians, I'm pretty certain that the bounding box will be either flat or tall and skinny".

Using several bounding boxes at the same location also allows us to detect multiple classes in the same location. For example, for a person standing in front of a car, we want to be able to detect both the person and the car. If we had not used bounding boxes of different shapes, the network might not have been able to come to one answer as it would have scored high both for person and car. However, when using bounding boxes of different aspect ratios, the network might say "for the tall and skinny aspect ratio, I'm seeing a human. For the flat aspect ratio, I'm seeing a car".

3.4 Task 3d)

In SSD, additional convolutional feature layers are added after the base layer. These layers decrease in size progressively which allows the network to detect multiple sizes of the same class. YOLO does not have this gradual decrease in feature layer sizes.

3.5 Task 3e)

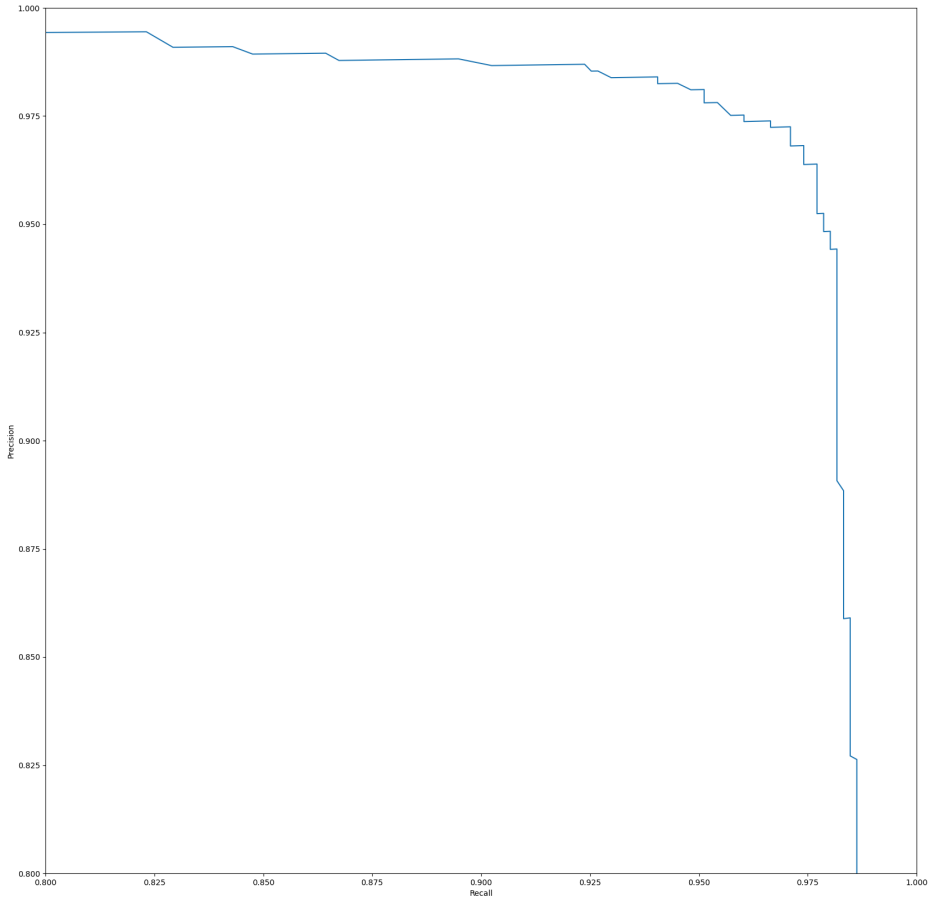
We have 38×38 locations for our anchors. There are 6 anchors per location. So, we have $38 \cdot 38 \cdot 6 = 8664$ anchor boxes in the feature map.

3.6 Task 3f)

Same as last time, just that now we repeat the operation for multiple image sizes. This means we have $6 \cdot (38 \cdot 38 + 19 \cdot 19 + 10 \cdot 10 + 5 \cdot 5 + 3 \cdot 3 + 1 \cdot 1) = 11640$ anchor boxes in total.

4 Task 4

4.1 Task 4b)



Average Precision (AP) @[IoU=0.50 | area= all | maxDets=100] = 0.748

4.2 Task 4c)

I used a lot of the things I learned from last lecture. Mainly batch normalization and network size increasing, did a lot for my network, with some other small changes as well.

The gist of it is: - Implement batch normalization (after conv2d, but before reLU). - Increase the network size: - Change `output_channels=[128, 256, 512, 256, 128, 64]` - Add more layers within the blocks already made (no new blocks were implemented) - I also changed the bounding boxes a little to only use aspect ratios of 2:1 and 1:2, as I can imagine only the number "1" has an aspect ratio of 3:1/1:3. - Change image augmentation: `mean = [0.485, 0.456, 0.406], std = [0.229, 0.224, 0.225]` - Use default Adam optimizer At 5928 iterations, I got a reported mAP of 0.883

4.3 Task 4d)

The stride for the 5×5 feature map is 64×64.

```
[32, 32], [32, 96], [32, 160], [32, 224], [32, 288]
[96, 32], [96, 96], [96, 160], [96, 224], [96, 288],
[160, 32], [160, 96], [160, 160], [160, 224], [160, 288],
[224, 32], [224, 96], [224, 160], [224, 224], [224, 288],
[288, 32], [288, 96], [288, 160], [288, 224], [288, 288],
```

For aspect ratios I will assume that we are still looking at the 5×5 feature map. Then, I end up with:

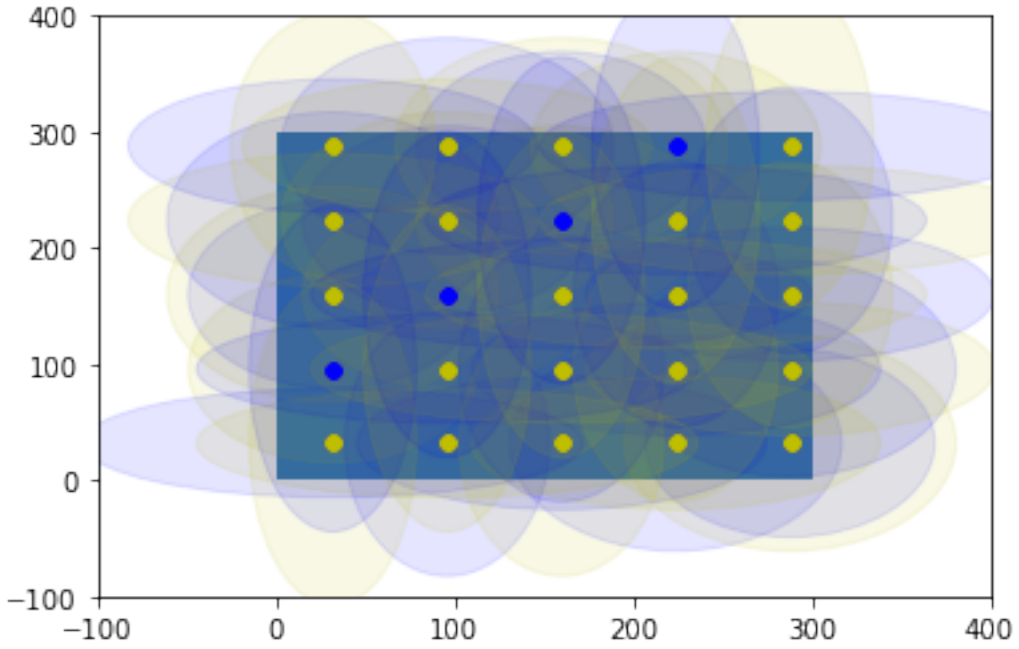
Square of side *min_size*: 162^2

Square of side $\sqrt{minSize \cdot nextMinSize}$: $\sqrt{162 \cdot 162}^2 = 162^2$

$[minSize \cdot \sqrt{aspectRatio}, minSize/\sqrt{aspectRatio}]$: $162 \cdot \sqrt{\frac{1}{2}}, 162/\sqrt{\frac{1}{2}} = [114.55129, 229.10259] = [115, 229]$

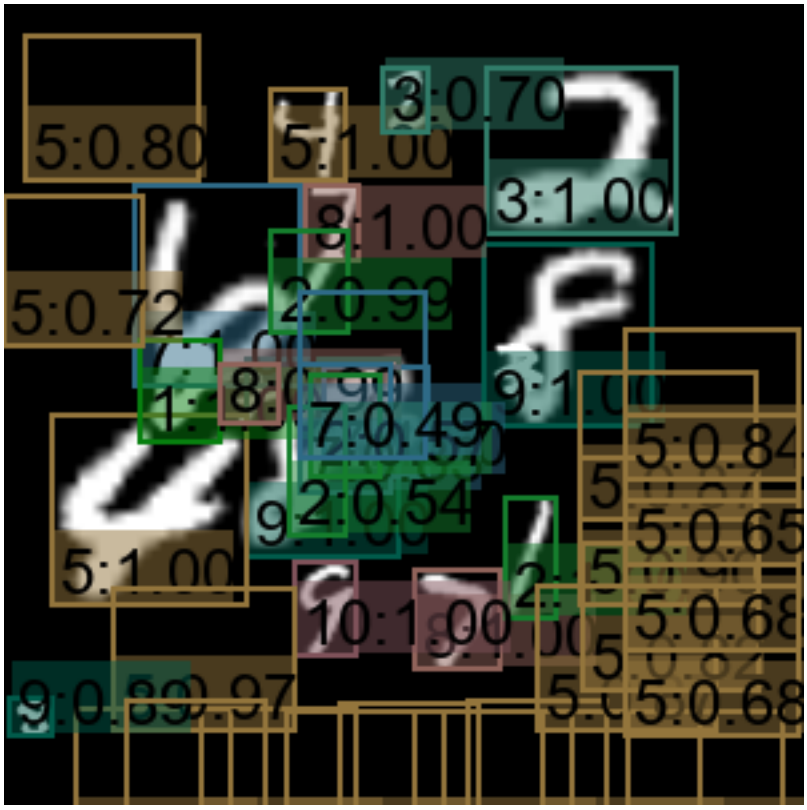
$[minSize/\sqrt{aspectRatio}, minSize \cdot \sqrt{aspectRatio}]$: $162/\sqrt{\frac{1}{2}}, 162 \cdot \sqrt{\frac{1}{2}} = [229.10259, 114.55129] = [229, 115]$

By looking at the anchor box plot, I think my calculations look correct.



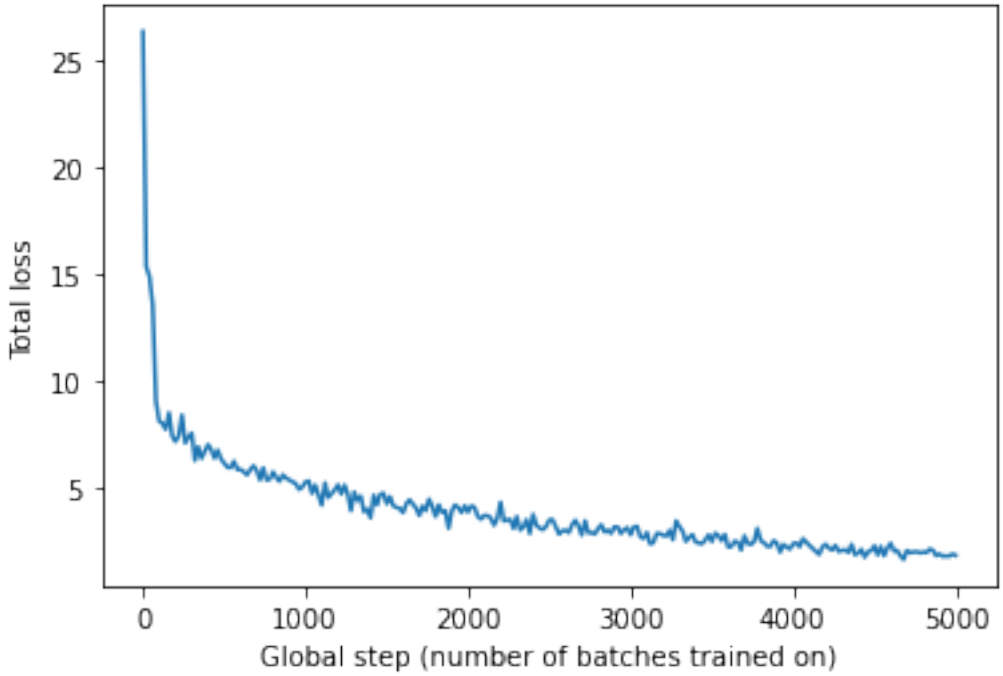
4.4 Task 4e)

I think the network in general has a lot of problems with numbers that overlap, as seen in the image below:



My network, for some reason, also seems to predict image class 5 a whole lot (the number 4). I find this pretty strange and I'm then wondering how I could get such a high mAP. Perhaps I would have the really good model do predictions, I should use a checkpoint other than 9999 but I'm not so sure how to do that.

4.5 Task 4f)


25
20
15
10
5
Total loss
0
1000
2000
3000
4000
5000
Global step (number of batches trained on)