

AssignmentReport-Group1

March 23, 2021

1 Assignment 1 Report

This is an outline for your report to ease the amount of work required to create your report. Jupyter notebook supports markdown, and I recommend you to check out this [cheat sheet](#). If you are not familiar with markdown.

Before delivery, **remember to convert this file to PDF**. You can do it in two ways: 1. Print the webpage (ctrl+P or cmd+P) 2. Export with latex. This is somewhat more difficult, but you'll get somewhat of a "prettier" PDF. Go to File -> Download as -> PDF via LaTeX. You might have to install nbconvert and pandoc through conda; `conda install nbconvert pandoc`.

2 Task 1

2.1 task 1a)

1. a)

fig 1




fig 2


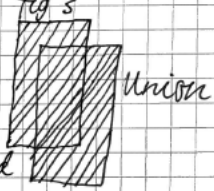


fig 3



The intersection over union is a measure for how much a predicted boundary overlaps with the truth. It is defined as

$$IoU = \frac{\text{area of overlap}}{\text{area of union}}$$

Where the overlap is shown in fig 2, the area where both boxes cover, and the union is shown in fig 3, where any box covers.

2.2 task 1b)

$$1. b) \text{ precision} = \frac{\text{true positives}}{\text{true positives} + \text{false positives}} = \frac{\text{true positives}}{\text{all positives}}$$

$$\text{recall} = \frac{\text{true positives}}{\text{true positives} + \text{false negatives}} = \frac{\text{true positives}}{\text{all cases}}$$

True positive is a positive prediction on a positive case.

False positive is a positive prediction on a negative case.

False negative is a negative prediction on a positive case.

2.3 task 1c)

$$1. c) \mu(r) = \max_{\hat{r} \geq r} \mu(\hat{r})$$

$\mu_1(0,0) = 1,0$	$\mu_2(0,0) = 1,0$	$\mu_1(0,7) = 0,5$	$\mu_2(0,7) = 0,5$
$\mu_1(0,1) = 1,0$	$\mu_2(0,1) = 1,0$	$\mu_1(0,8) = 0,20$	$\mu_2(0,8) = 0,20$
$\mu_1(0,2) = 1,0$	$\mu_2(0,2) = 1,0$	$\mu_1(0,9) = 0,20$	$\mu_2(0,9) = 0,20$
$\mu_1(0,3) = 1,0$	$\mu_2(0,3) = 1,0$	$\mu_1(1,0) = 0,20$	$\mu_2(1,0) = 0,20$
$\mu_1(0,4) = 1,0$	$\mu_2(0,4) = 0,80$		
$\mu_1(0,5) = 0,5$	$\mu_2(0,5) = 0,60$		
$\mu_1(0,6) = 0,5$	$\mu_2(0,6) = 0,5$		

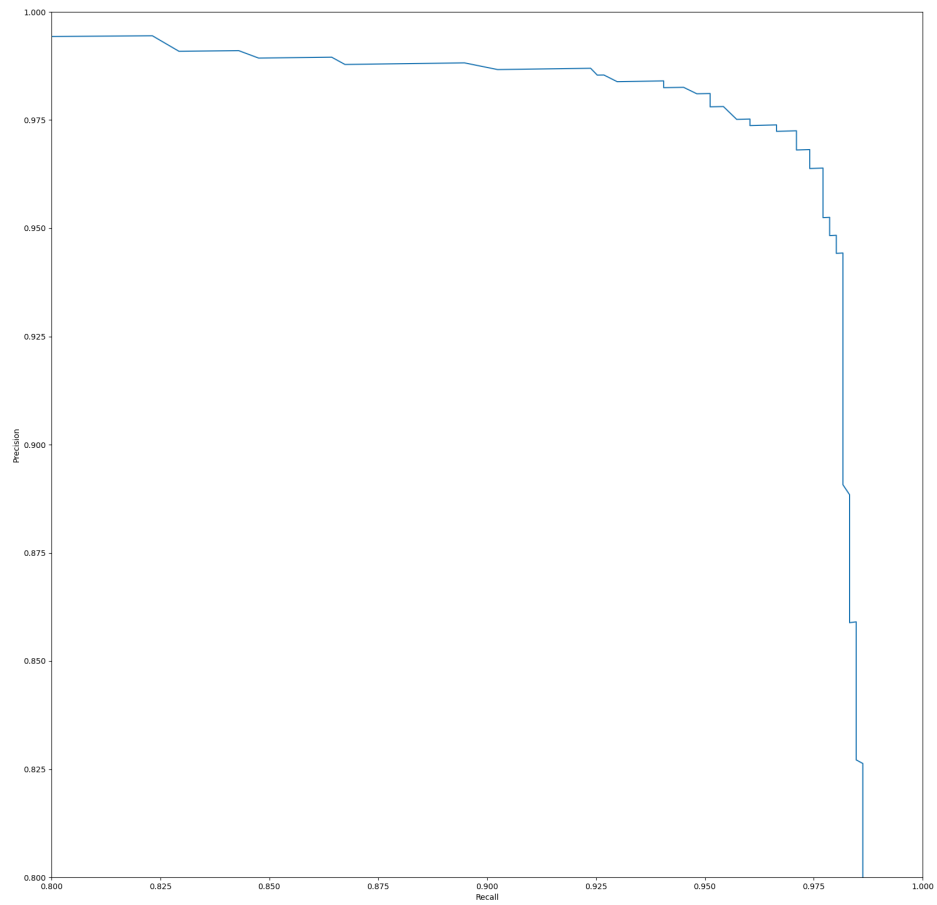
$$mAP_1 = \frac{1}{11} (5 \cdot 1 + 3 \cdot 0,5 + 3 \cdot 0,2) = 0,645$$

$$mAP_2 = \frac{1}{11} (4 \cdot 1 + 0,8 + 0,6 + 2 \cdot 0,5 + 3 \cdot 0,2) = 0,636$$

$$mAP = \frac{1}{2} (mAP_1 + mAP_2) = \underline{\underline{0,641}}$$

3 Task 2

3.0.1 Task 2f)



4 Task 3

4.0.1 Task 3a)

3. a) This operation is called non-maximum suppression and removes boxes with an ~~overlap~~^{IoU} higher than a set threshold for another box with higher confidence score.

4.0.2 Task 3b)

3. b) False, the deeper layers have lower resolutions and are used to detect larger objects. The smaller objects are detected by higher resolution feature maps earlier in SSD.

4.0.3 Task 3c)

3. c) They use different bounding box aspect ratios at the same spatial location to cover a lot of different object types at the location. For example a car and a person have different aspect ratios and using different aspect ratios allow one bounding box to approach the true boundary for the object no matter what shape the object has.

4.0.4 Task 3d)

3. d) The main difference between SSD and YOLO is the use of multi-scale feature maps and convolutional predictors. SSD uses feature maps of different sizes to detect objects of varying size, as well as convolutional filters which produce category scores or shape offsets. YOLO uses a single scale feature map and a fully connected layer for predictions.

4.0.5 Task 3e)

3. e) For this feature map we have

$$H \cdot W \cdot C = 38 \cdot 38 \cdot 6 = \underline{\underline{8664}}$$

anchor boxes.

4.0.6 Task 3f)

3. f) In total we have

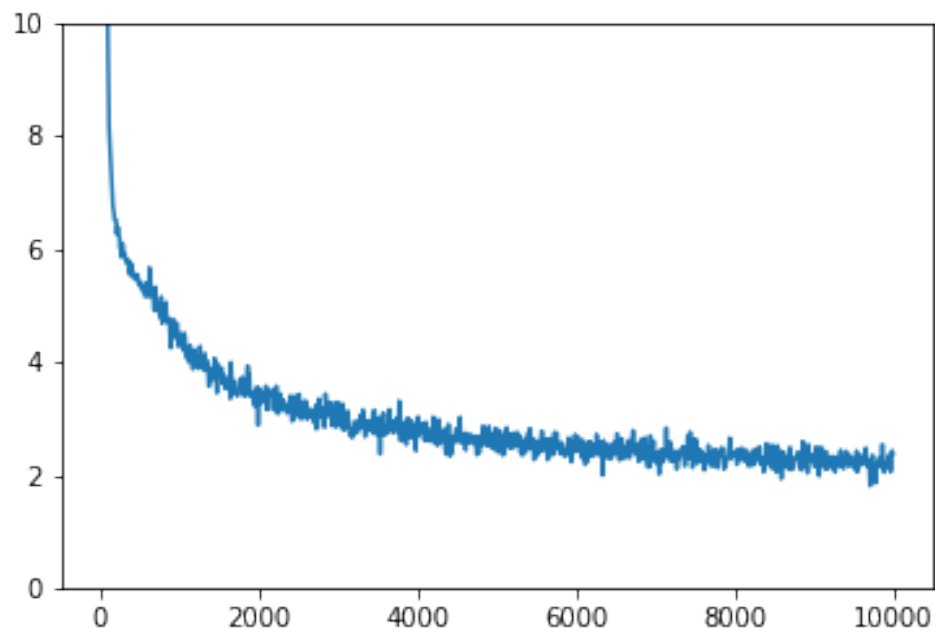
$$6 \cdot \left(\sum_{i=1}^6 H_i \cdot W_i \right) = 6(38^2 + 19^2 + 10^2 + 5^2 + 3^2 + 1^2) = \underline{\underline{11640}}$$

anchor boxes for the entire network.

5 Task 4

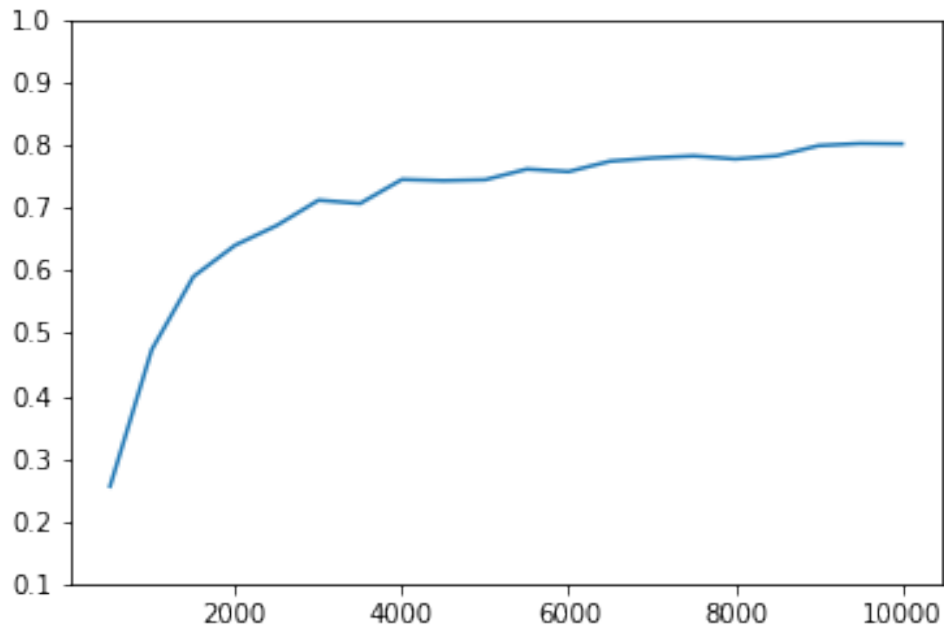
5.1 Task 4b)

Total loss over 10 000 iterations:



The final mAP after 10 000 iterations was 0.8011.

mAP over 10 000 iterations:



Class distributed mAP:

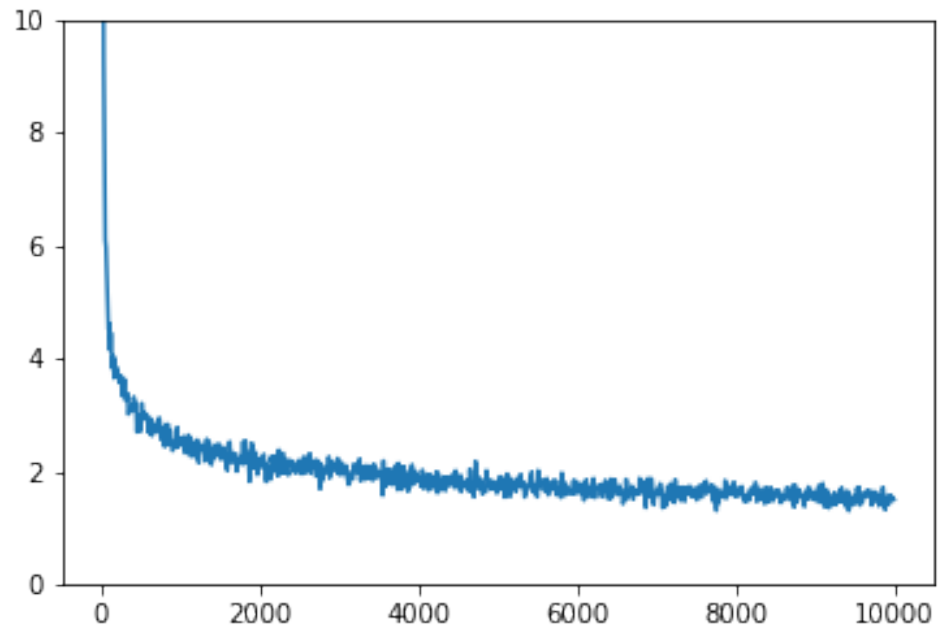
Class	mAP
0	0.8511
1	0.7240
2	0.7841
3	0.8185
4	0.8255
5	0.8061
6	0.8109
7	0.8036
8	0.8087
9	0.7788

5.2 Task 4c)

The model used in 4c and 4d was implemented in improved.py in the backbone folder.

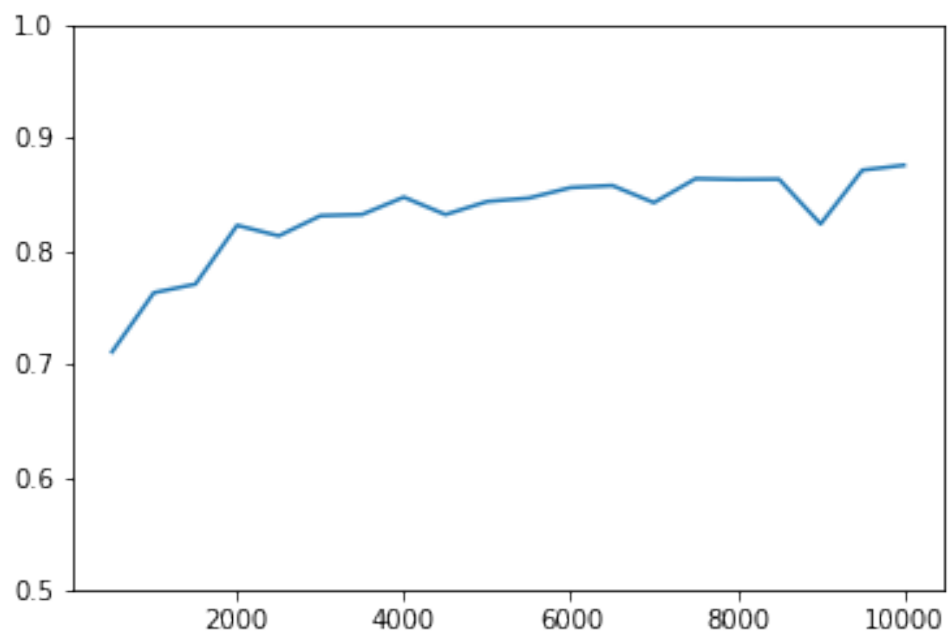
To reach 85% I first added batch normalization which sped up the convergence of the training but the mAP flattened very early. Next I doubled the number of filters in all the layers, which gave me around 84% mAP. Then I doubled the number of filters again but this did not increase the mAP so I tried adding dropout of 10% after each convolutional layer to counteract overfitting and this pushed the mAP to 87,5%.

Total loss over 10 000 iterations:



Final mAP: 0.8753

mAP over 10 000 iterations:



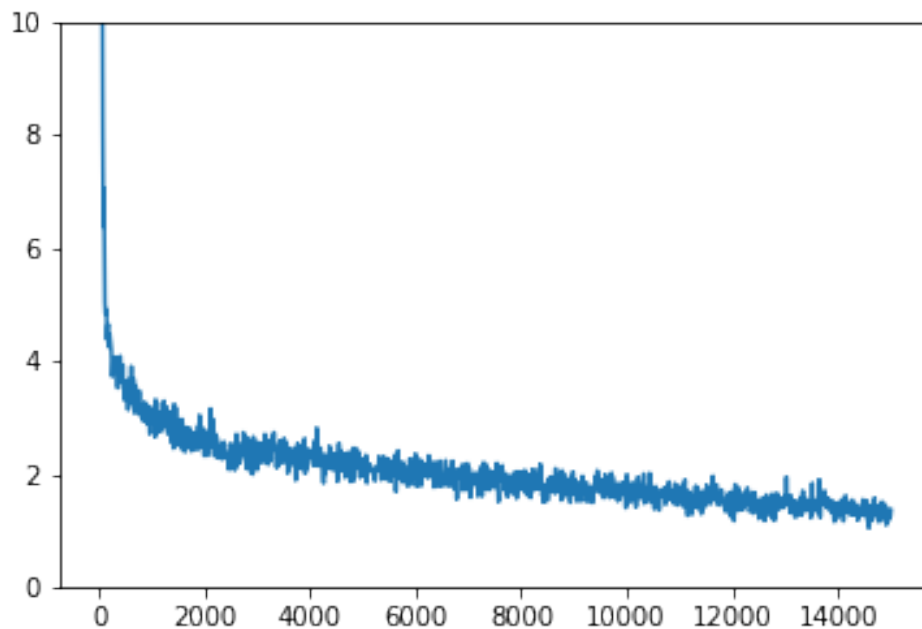
Class distributed mAP:

Class	mAP
0	0.8940
1	0.8096
2	0.8760
3	0.8971
4	0.8807
5	0.8830
6	0.8859
7	0.8569
8	0.8900
9	0.8794

5.3 Task 4d)

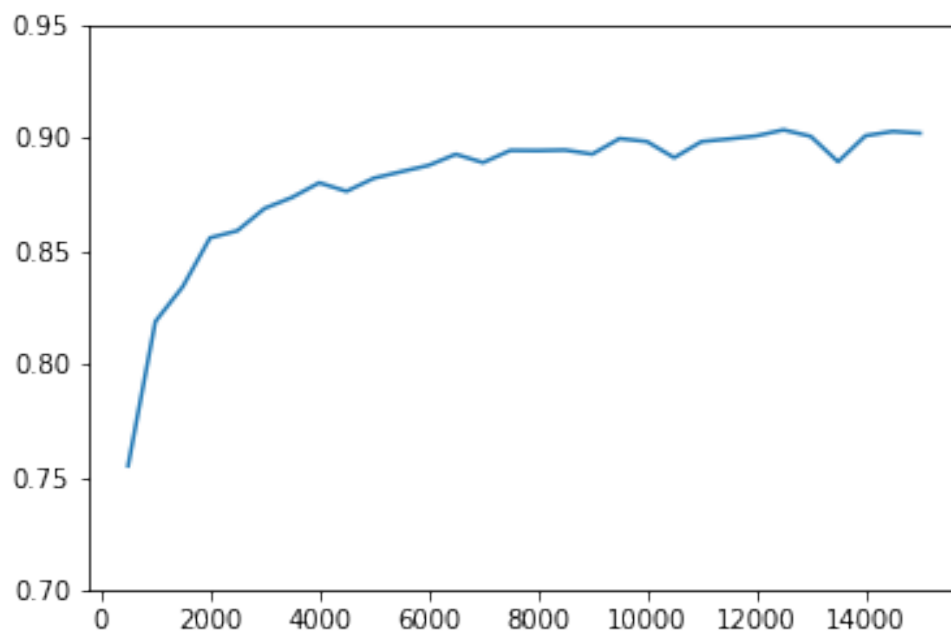
The model had a hard time detecting class 1 and i suspected this might be a small object. To improve on this i reduced the smallest minimum size in MIN_SIZES to [20, 20] in the configuration. This made the learning unstable so I also had to reduce learn rate to 10^{-3} instead of $2 * 10^{-3}$. This change pushed the mAP to around 89.5%, and to get the final boost I increased the IoU threshold to 0.7 and reduced the smallest minimum size further to [10, 10]. Again this caused unstable learning which I fixed by reducing learn rate to $5 * 10^{-4}$. These changes pushed the mAP to 90%.

Total loss over 15 000 iterations:



The final mAP after 15000 iterations was 0.9018.

mAP over 15 000 iterations:

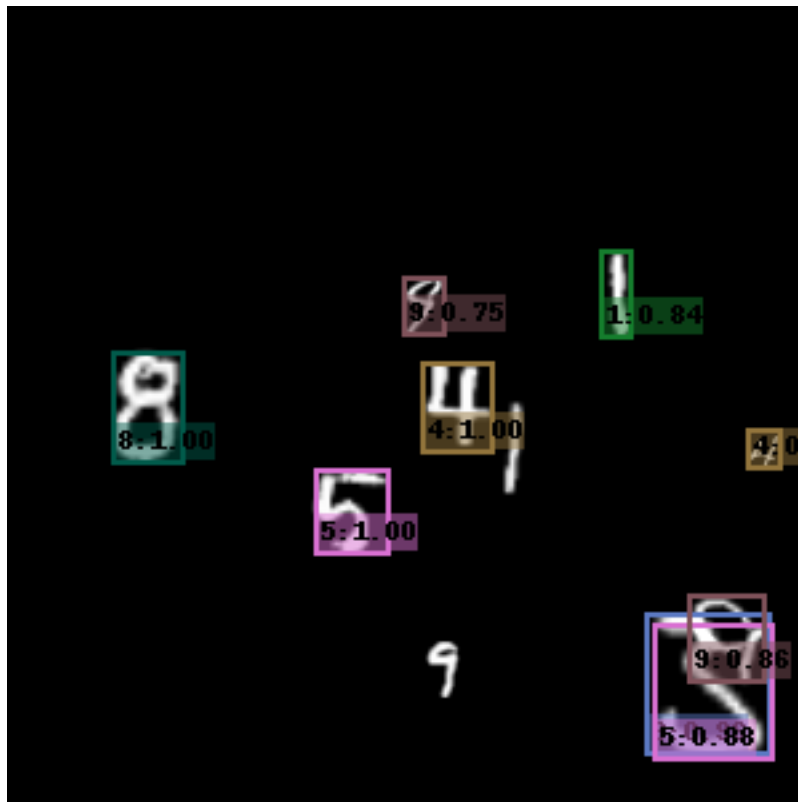
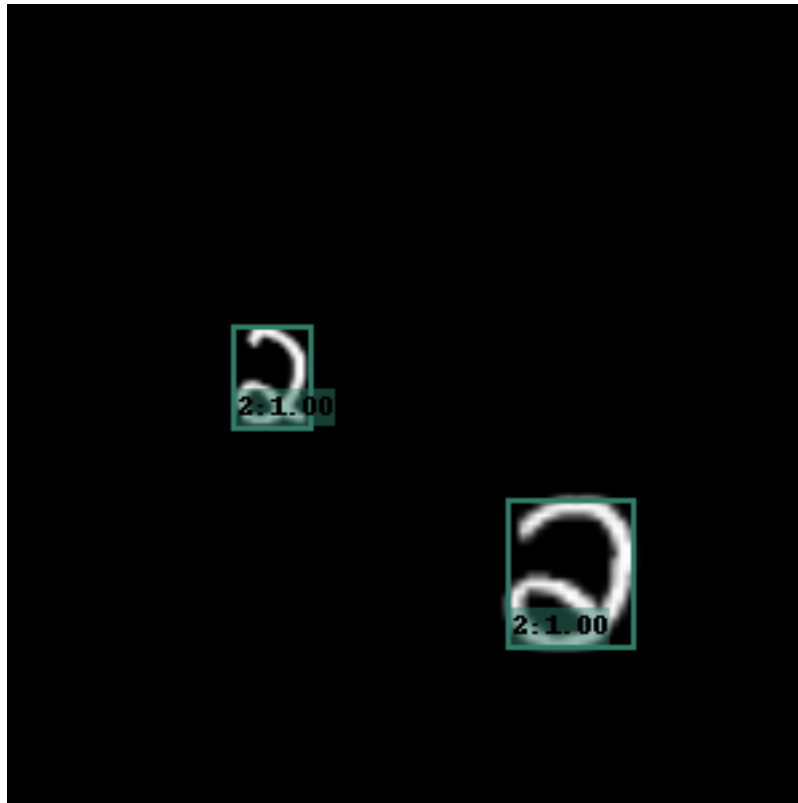


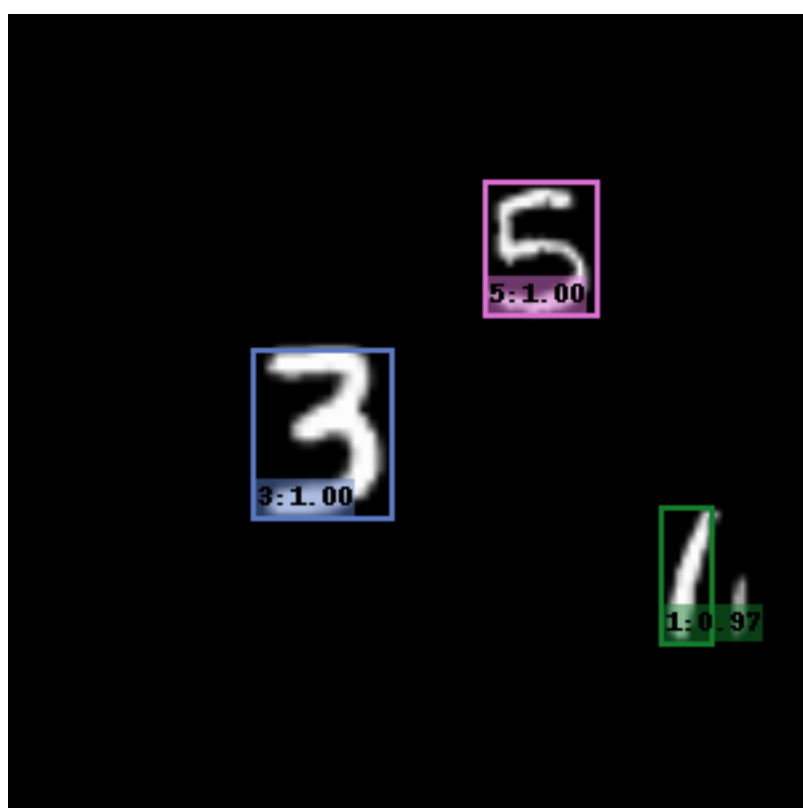
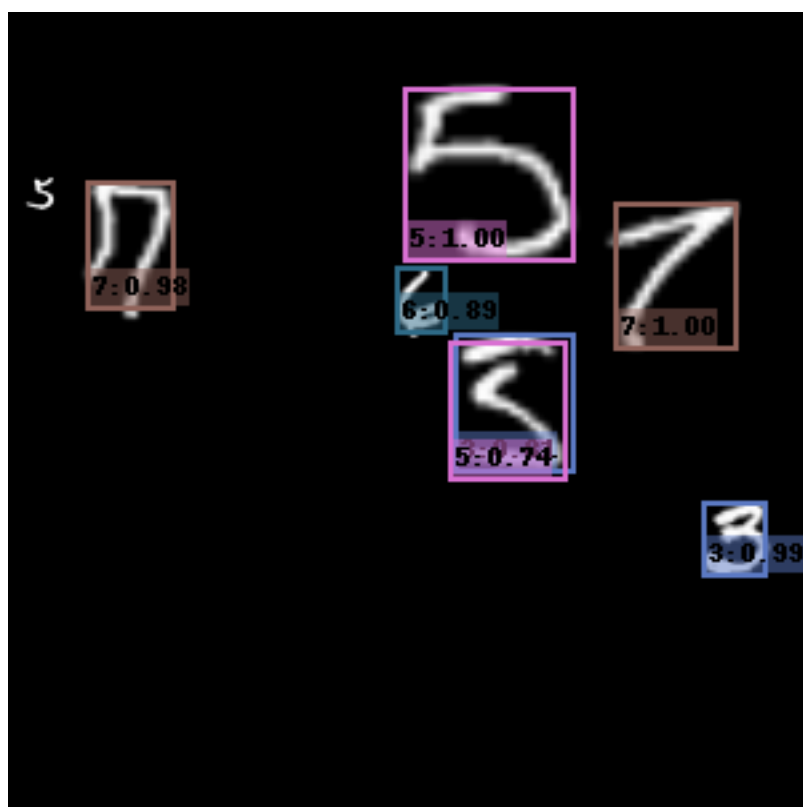
Class distributed mAP:

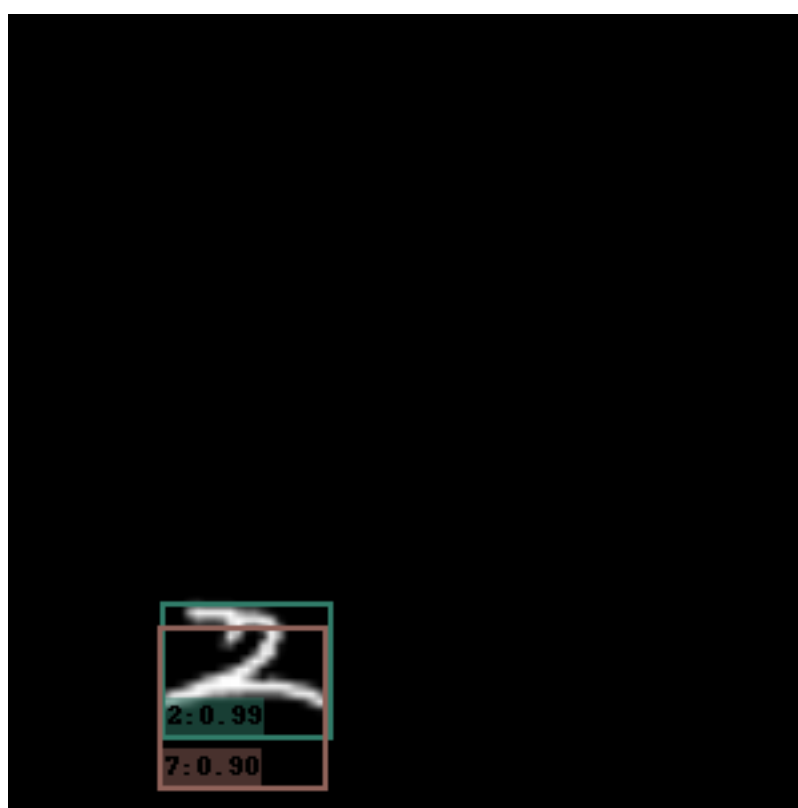
Class	mAP
0	0.9078
1	0.8646
2	0.9052
3	0.9064
4	0.9060
5	0.9075
6	0.9061
7	0.9026
8	0.9083
9	0.9036

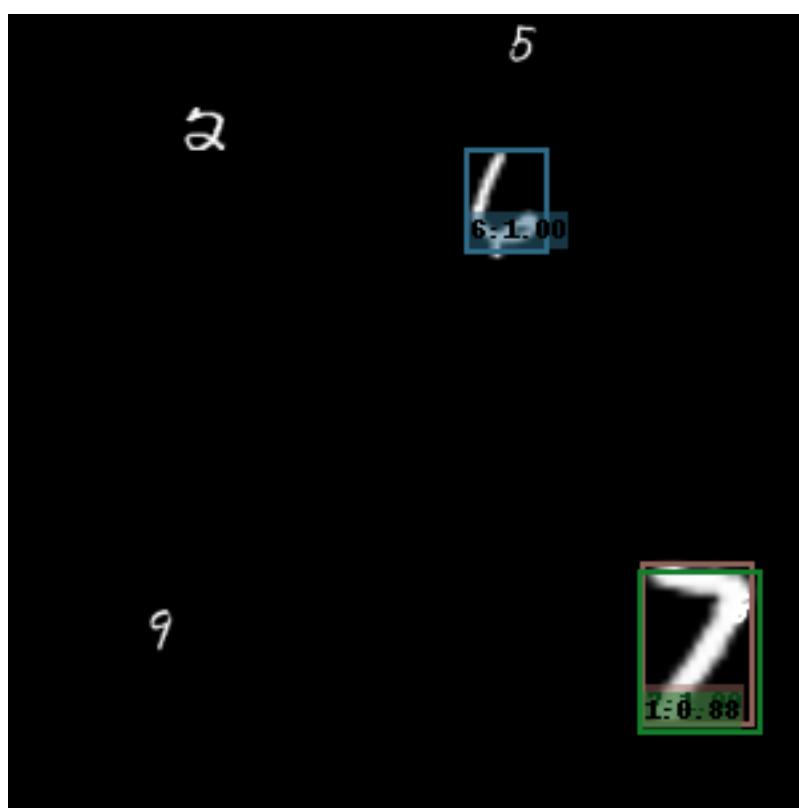
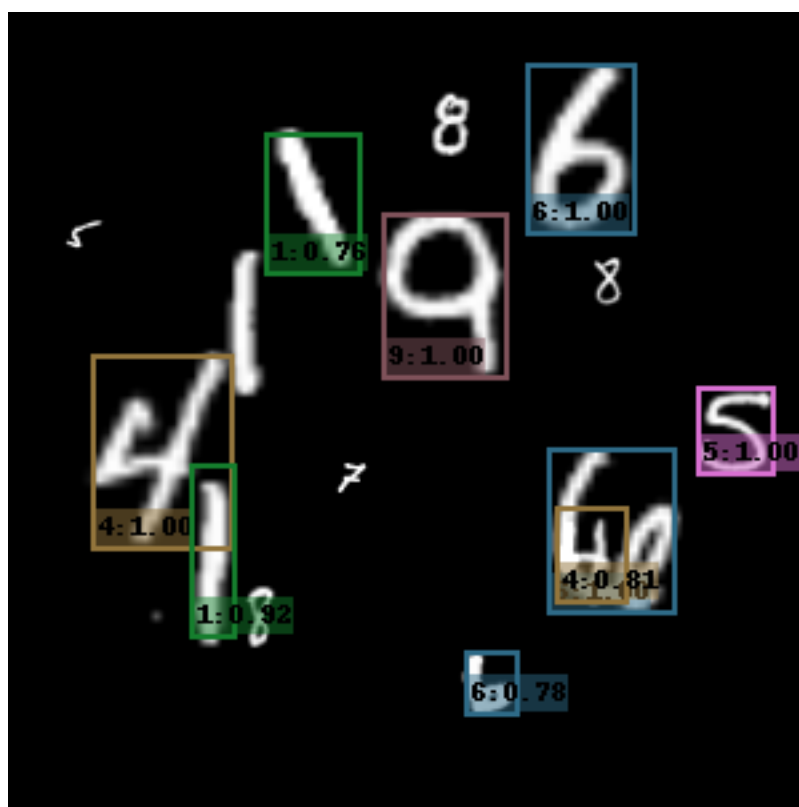
5.4 Task 4e)

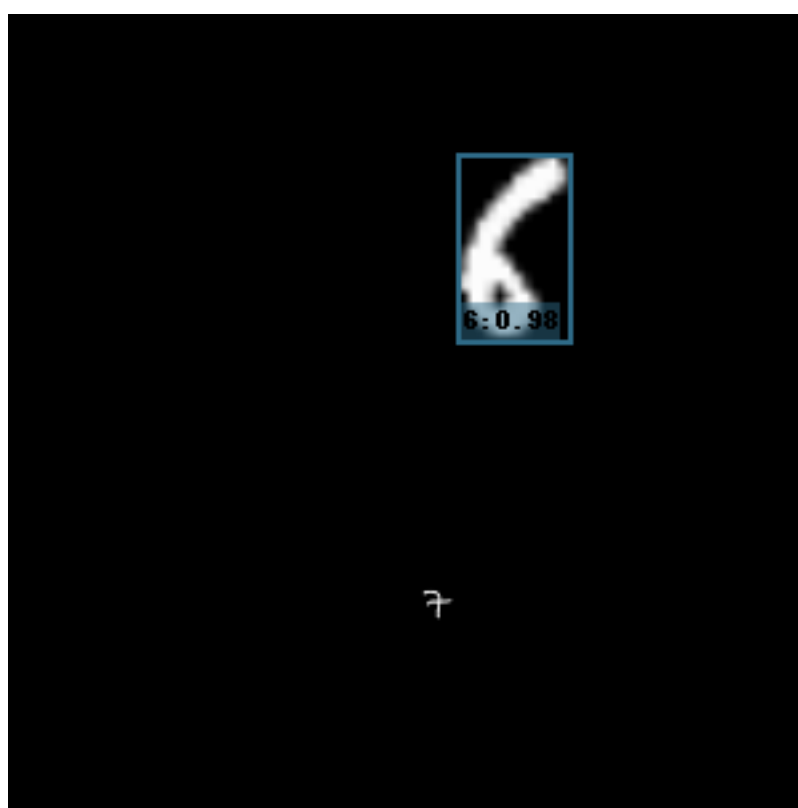
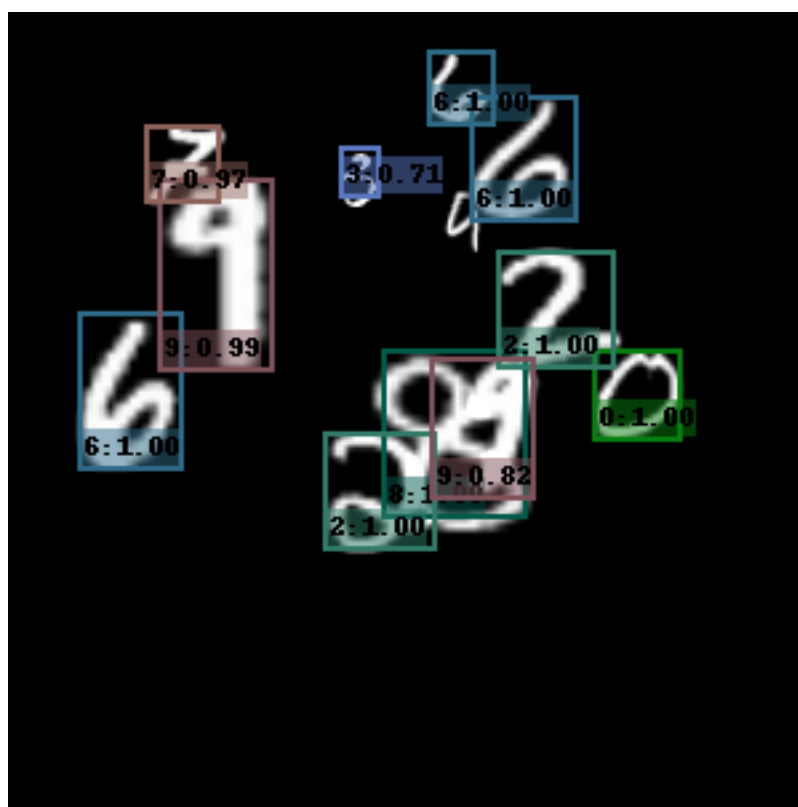
Below are classified images from the demo folder, classified with the model which reached 87% accuracy. I chose to use this model as it has a lower IoU threshold resulting in more digits being detected than when using the model with 90% accuracy. From the images we can see that the model struggles detecting digits that are small in size. This is a common problem when using SSD, it often struggles detecting small objects. Increasing the resolution of the input images could help reduce this problem at the cost of longer computing time for both training and inference.

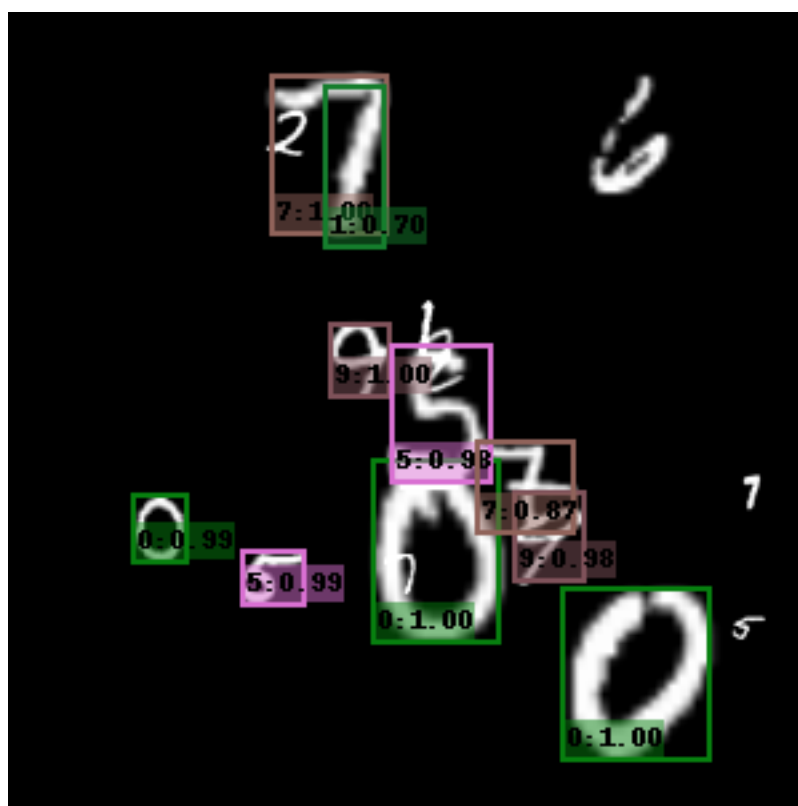
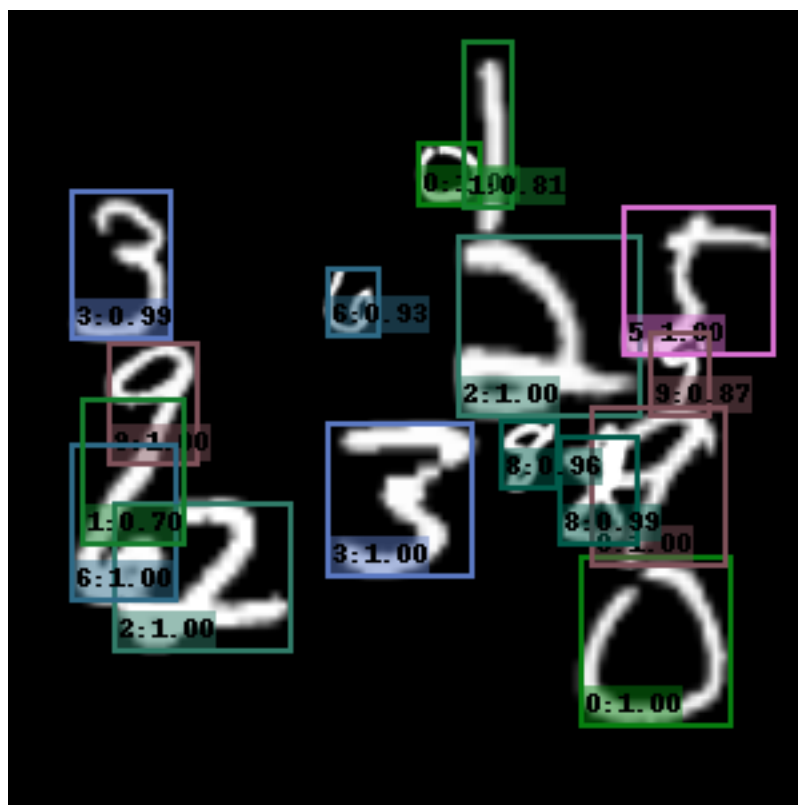




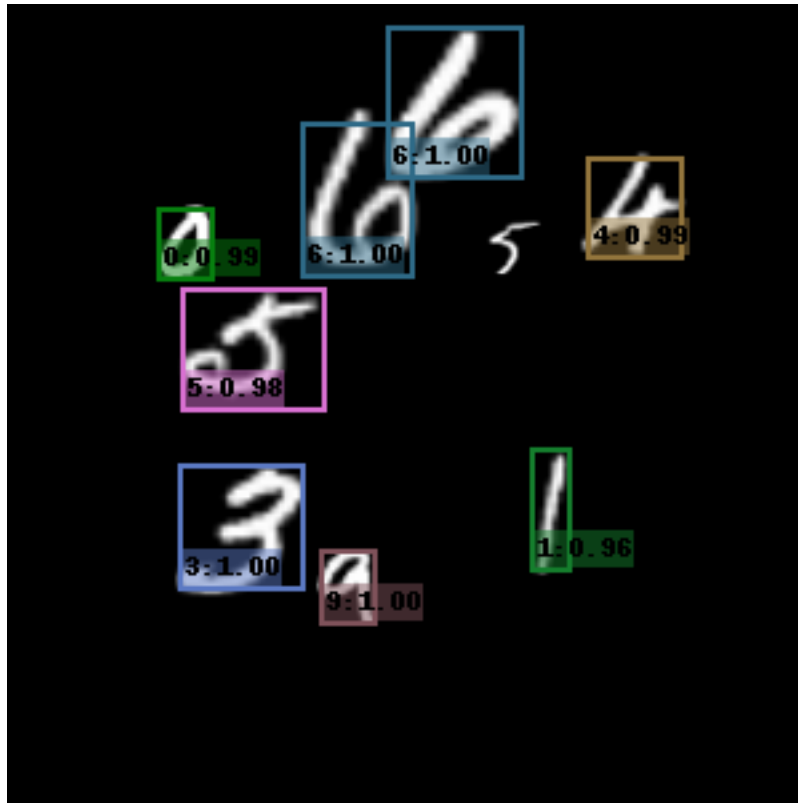






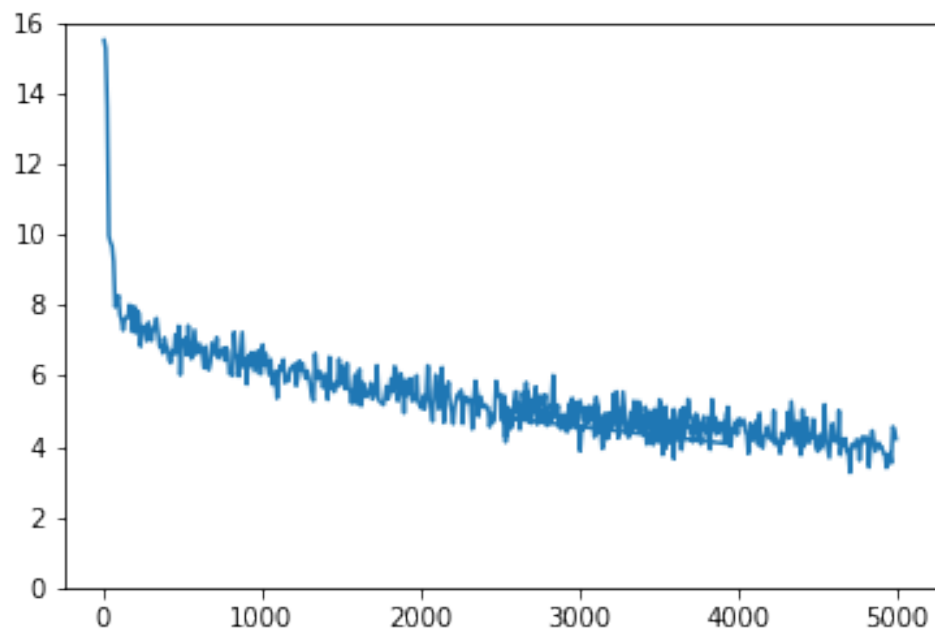






5.5 Task 4f)

Total loss over 5000 iterations:



The final mAP was 0.4700

The images tested are shown below, only the photo of the cat was detected.







