Assignment 4 Report

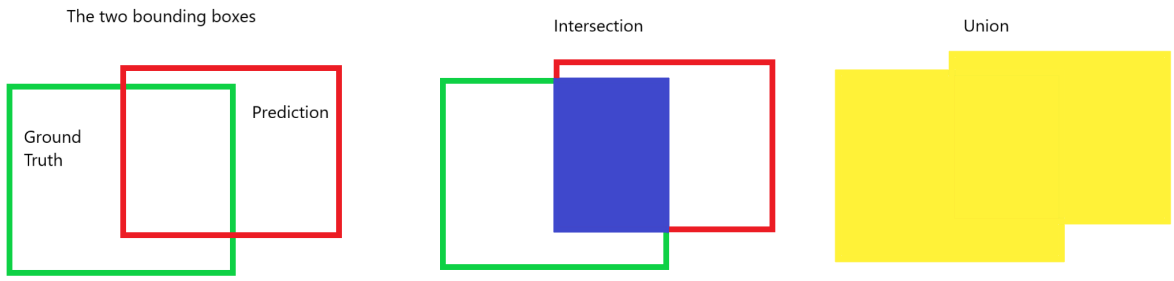
This is an outline for your report to ease the amount of work required to create your report. Jupyter notebook supports markdown, and I recommend you to check out this cheat sheet. If you are not familiar with markdown.

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 somehwat 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` .

Task 1

task 1a)



The drawing illustrates the intersection and union of two bounding boxes. The "Intersection over Union" (IoU) is the ratio of these two areas:

$$\text{IoU} = \frac{\text{Intersection}}{\text{Union}}$$

If we first find the intersection by comparing the two boxes x and y coordinates. The union can be found by summing up the two areas of the two bounding boxes and subtracting the intersection.

The intersection in x can be found by summing up the width of the two rectangles and subtracting the largest distance between two x coordinates:

$$\text{Intersection}_x = \Delta x_{gt} + \Delta x_{pred} - (x_{max} - x_{min})$$

where x_{max} and x_{min} are the maximum and minimum of all the x coordinates.

Similarly the intersection in y can be found using the same formula with the y coordinates.

Multiplying the intersection in x and y gives the intersectional area.

$$\text{Intersection} = (\Delta x_{gt} + \Delta x_{pred} - (x_{max} - x_{min}))(\Delta y_{gt} + \Delta y_{pred} - (y_{max} - y_{min}))$$

If either the x or y intersection is negative then the Intersection is 0.

task 1b)

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

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

True positive (TP) is when a predicted positive is also a ground truth positive, whereas a false positive is a predicted true which did not correspond with the ground truth.

task 1c)

Fill in task 1a image of hand-written notes which are easy to read, or latex equations here

Smoothened:

$mAP_{1,s} = 1 \cdot 0.4 + 0.5 \cdot 0.3 + 0.2 \cdot 0.3 = 0.61$

$mAP_{2,s} = 1 \cdot 0.3 + 0.8 \cdot 0.1 + 0.6 \cdot 0.1 + 0.5 \cdot 0.2 + 0.2 \cdot 0.3 = 0.6$

$mAP_s = \frac{0.61+0.6}{2} = 0.606$

Not smoothened:

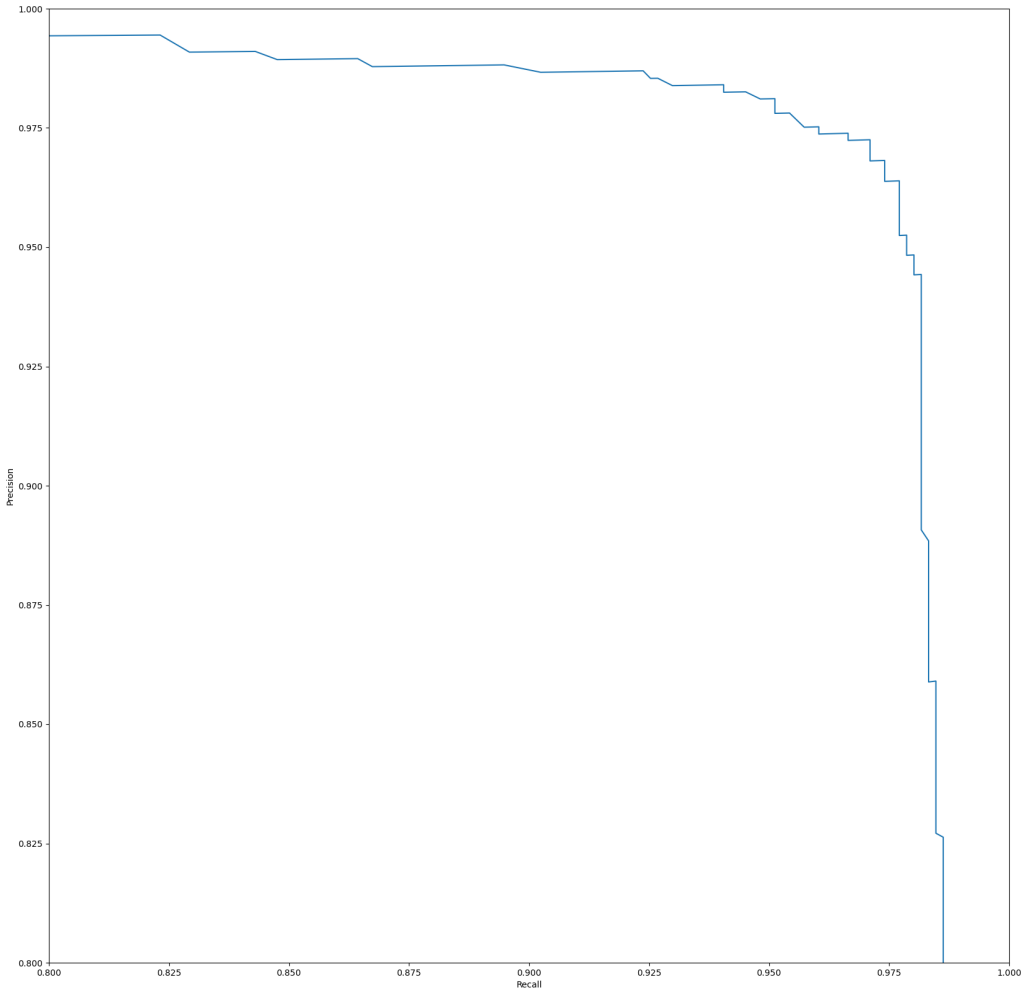
$mAP_1 = mAP_{1,s} + 0.5 \cdot (0.3 \cdot 0.5 + 0.3 \cdot 0.3) = 0.73$

$mAP_2 = mAP2, s + 0.5 \cdot (0.1 \cdot 0.2 + 0.1 \cdot 0.2 + 0.1 \cdot 0.2 + 0.3 \cdot 0.3) = 0.675$

$mAP = \frac{0.73+0.675}{2} = 0.7025$

Task 2

Task 2f)



Task 3

Task 3a)

The filtering operation is called NMS, or Non-Maximum Suppression. It filters out the boxes that has an IoU lower than a given threshold.

Task 3b)

False. Because the images become smaller when we go deeper down the layers, it is harder to detect small objects. The deeper layers are more useful for detecting key features in objects.

Task 3c)

They use different bounding box aspect rations for classification. Different objects will have different ratios and therefore it is useful with different bounding box aspect ratios.

Task 3d)

YOLO uses k-means clustering to determine the boundary boxes whereas SSD uses default boxes that are chosen manually.

Task 3e)

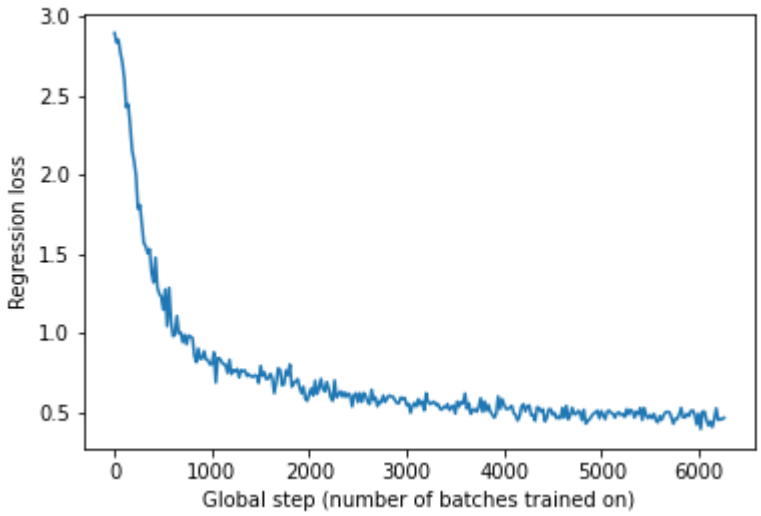
38×38 is 1444 anchor locations on a 38 by 38 grid. Num anchors: $1444 \cdot 6 = 8664$

Task 3f)

$(38 \cdot 38 + 19 \cdot 19 + 10 \cdot 10 + 5 \cdot 5 + 3 \cdot 3 + 1 \cdot 1) \cdot 6 = 11640$

Task 4

Task 4b)



The final mean average precision (mAP) was 76.5\%

Task 4c)

In order to improve the model, we introduced batch normalization and changed the optimizer from SGD to Adam (with slightly reduced learning rate and weight decay). In addition to this, we increased the batch size to 64 and doubled the amount of nodes in the last layer. We also removed the first ReLU in the "blocks", because the model previously had ReLU's twice in a row where the latter one is redundant. This got the training at best to **81.5%** precision, and the final mAP was **79.3%**.

This was the best attempt out of many...

We tried using:

- tuning of the SGD optimizer (lr, momentum, and weight decay)
- drop-out
  - With a low drop-out rate (0.2-0.3) it did not sufficiently improve the model, and beyond that it only made it worse.
- average pooling
  - Similarly as in Assignment 3, it decreased the performance of the model.
- increasing number of output nodes in the different "blocks"
  - With more output nodes it seemed like the model had a harder time training and converged around 70% mAP. However, we probably could experiment more here
- adding a convolutional layer in all the middle of the blocks
  - Similar to increasing the number of output nodes..

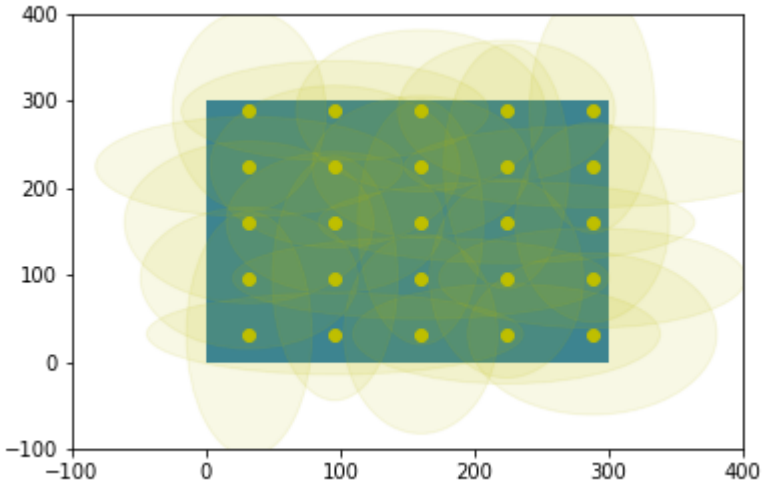
It has been challenging and time consuming to experiment with different architechtures as it takes a long time to see how it performs. Sometimes the model had very bad performance in the beginning, but got a lot better after many steps. Other times it was very good in the beginning, however it did not improve much after. Many times during training, the mAP fluxuated between higher and lower values. This could indicate that the model was overtrained. It also really struggled with class 1 at times...

Task 4d)

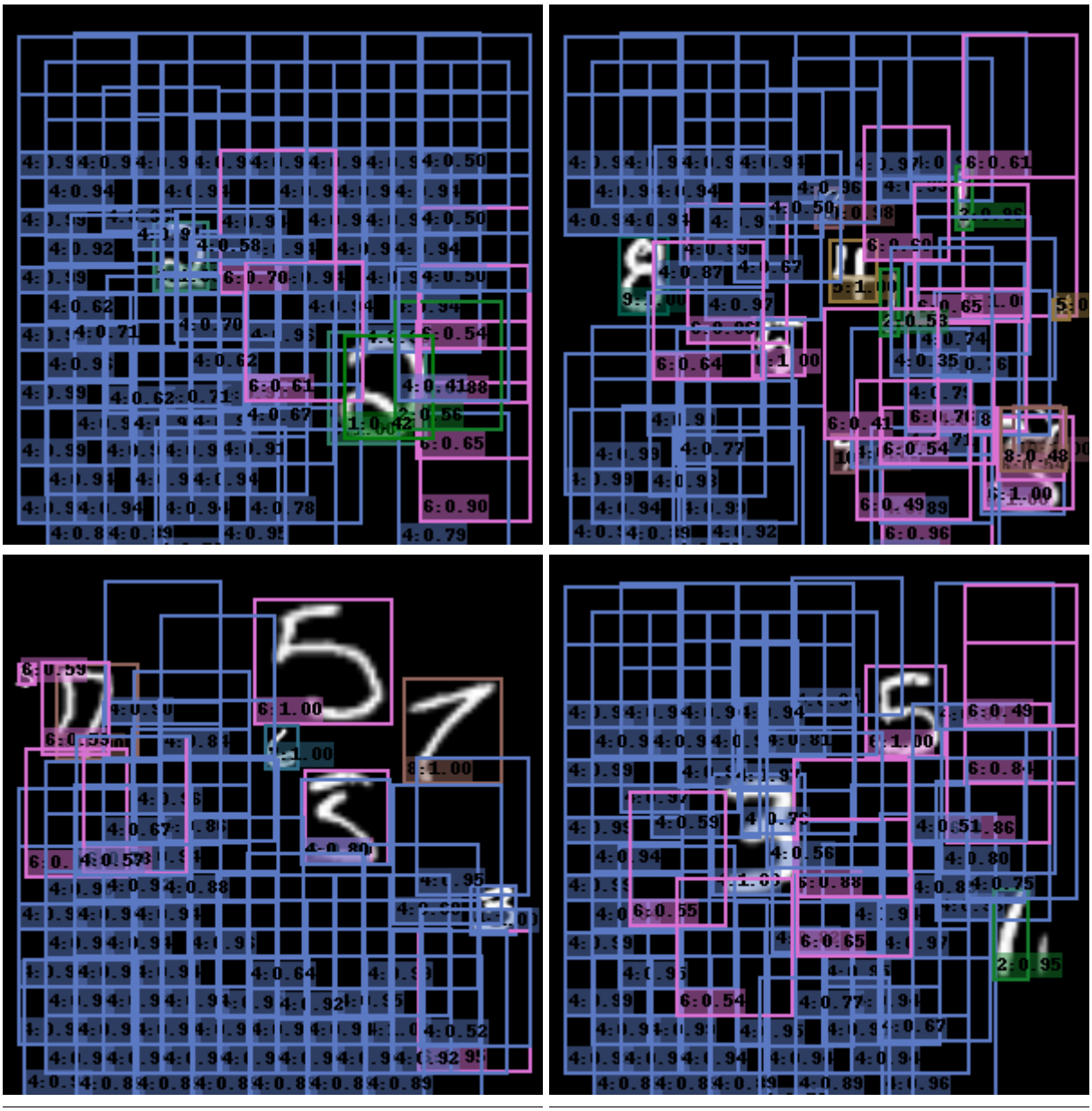
The stride corresponding to the 5×5 feature map is $(64, 64)$. This means that the anchor locations are placed in $(32 + 64 \cdot i, 32 + 64 \cdot j)$ Pixel values of anchor locations:

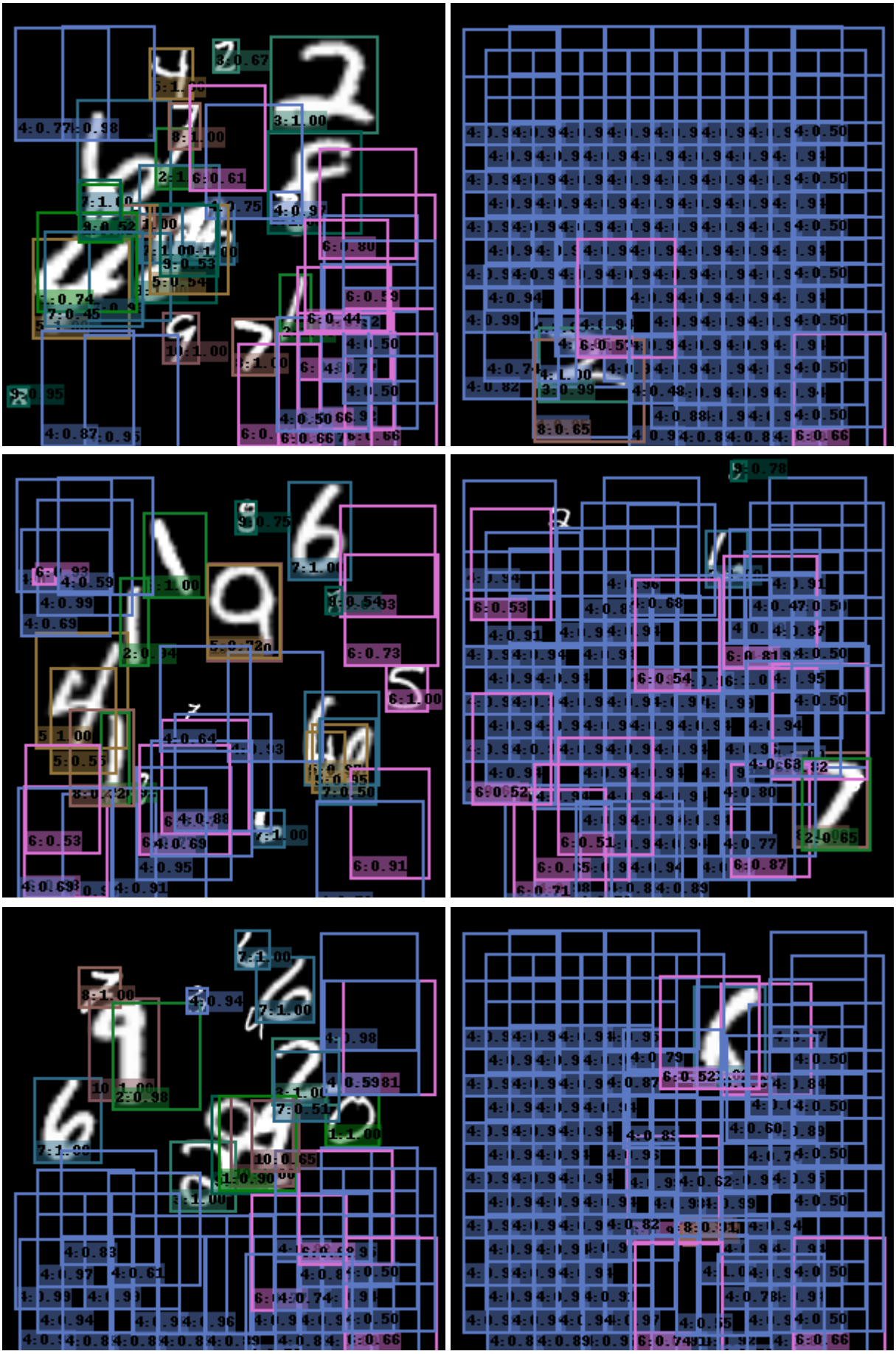
(32, 32), (32, 96), (32, 160), (32, 224), (32, 288), (96, 32), (96, 96), (96, 160), (96, 224), (96, 288), (160, 32), (160, 96), (160, 160), (160, 224), (160, 288), (224, 32), (224, 96), (224, 160), (224, 224), (224, 288), (288, 32), (288, 96), (288, 160), (288, 224), (288, 288)

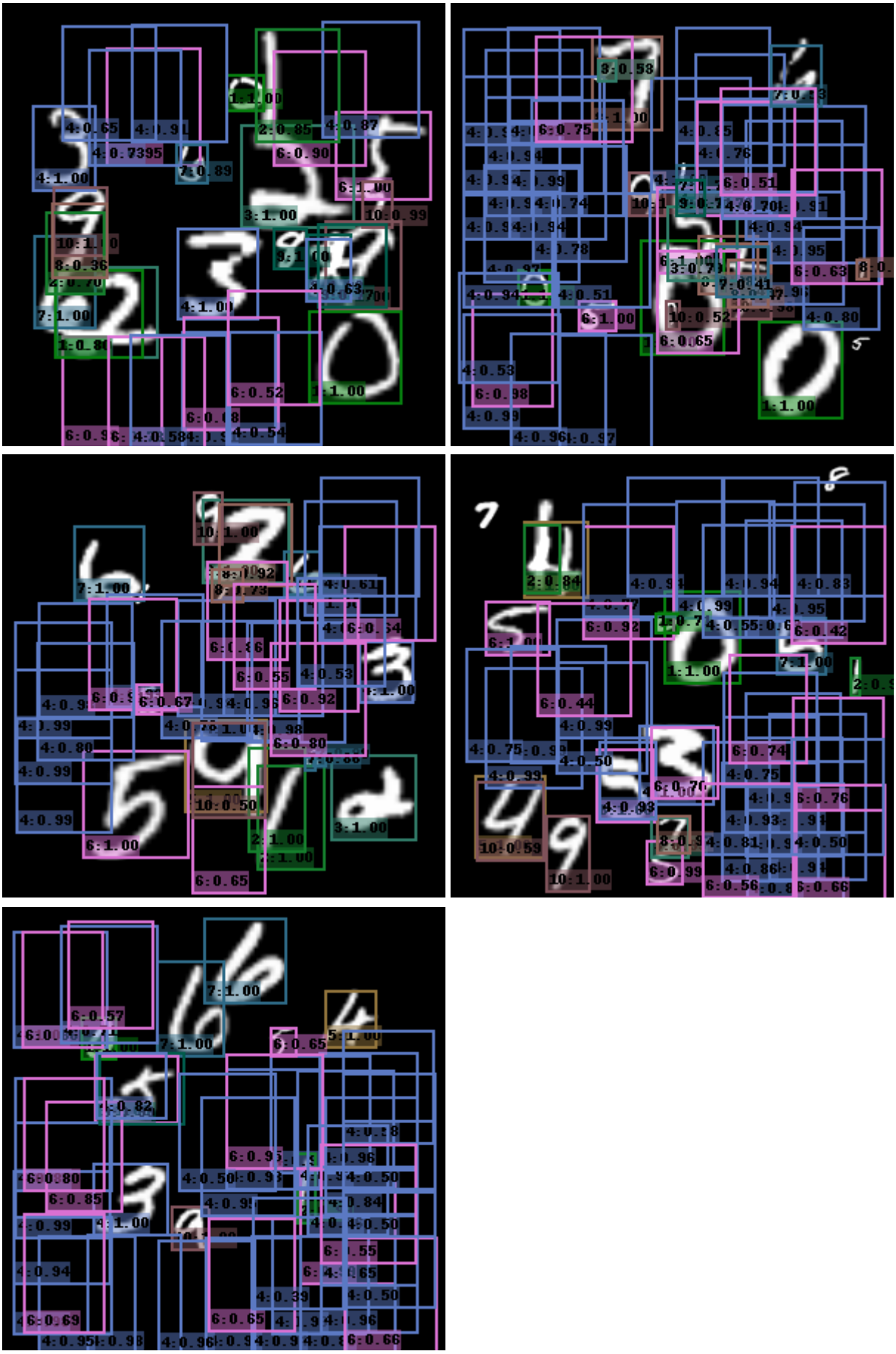
Size of anchor boxes: [162, 162], [185, 185], [114, 229], [229, 114], [93, 280], [280, 93]



Task 4e)





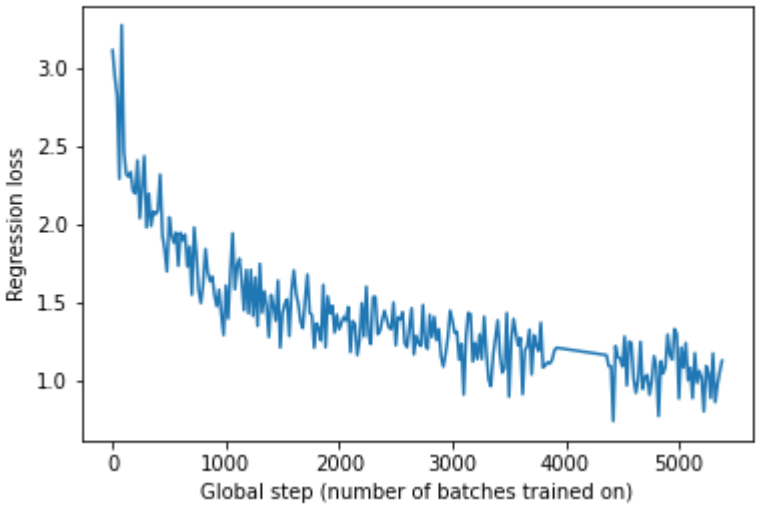


It seems the model was struggling to detect smaller numbers. It also assumes that there are 3's everywhere. Even though average precision seems to be good, it does not seems like it after looking

at these results. Regarding the issue earlier with our model struggling with one specific class, it might be caused by classifying many parts of the picture as that number.

Task 4f)

Plot off training loss for around 500 iterations:

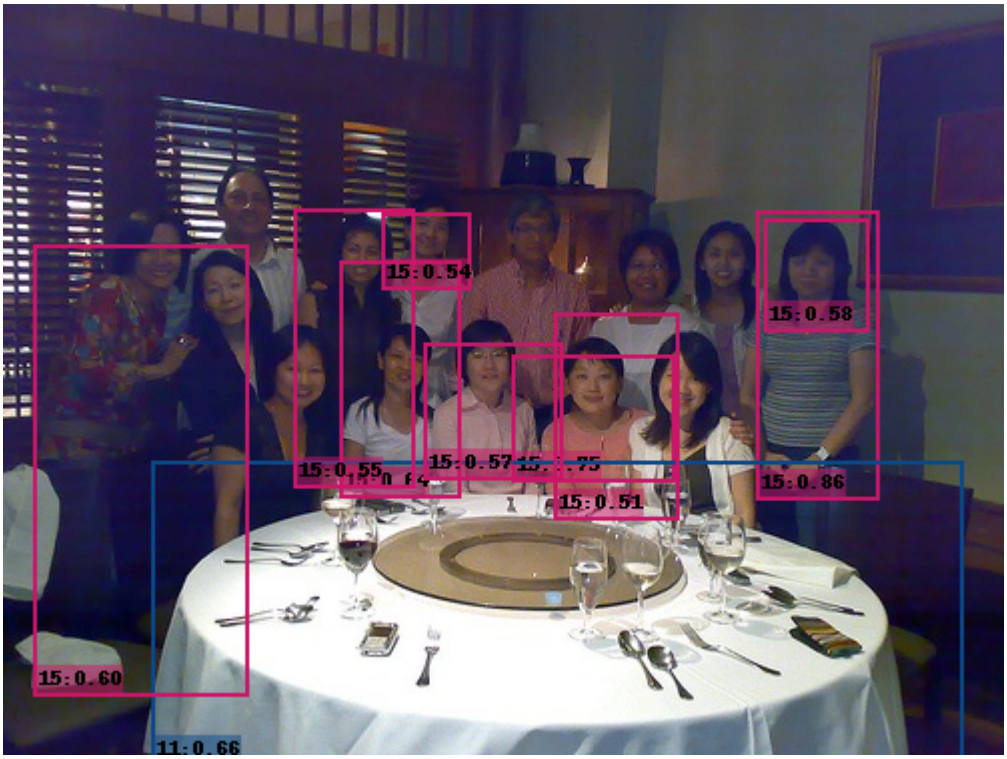


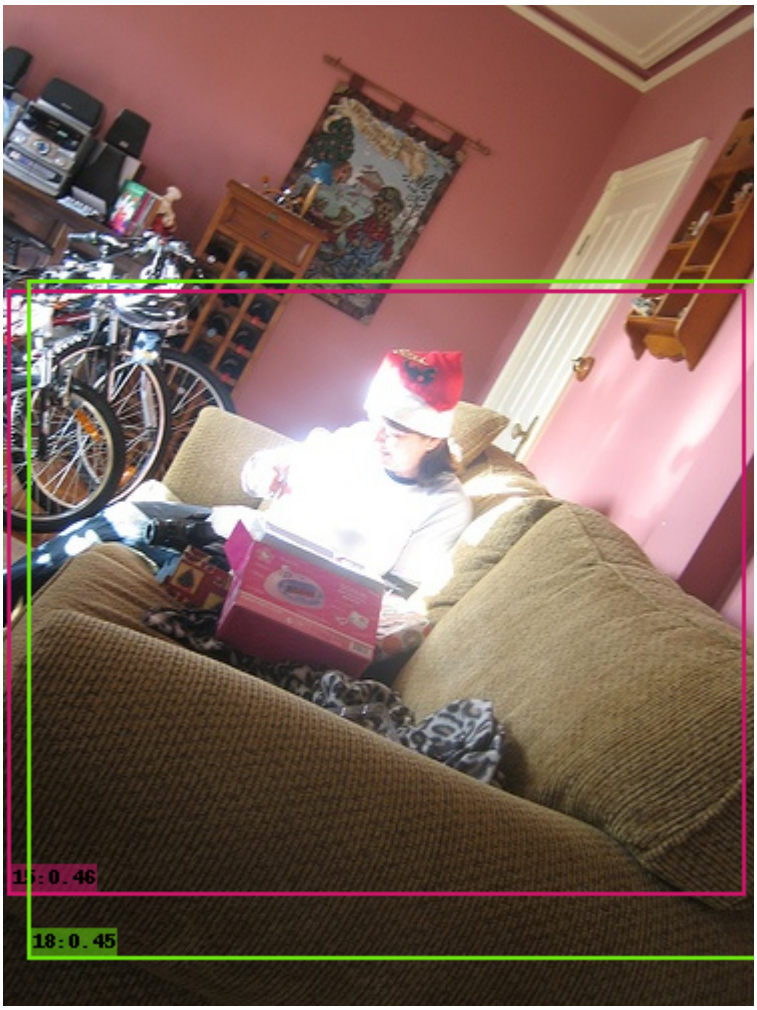
The final mean average precision (mAP) was 39.3\%

Result on the demo images:


AGR
23
15:0.80
20:0.36


8:0.99


15:0.54
15:0.58
15:0.55
15:0.57
15:0.51
15:0.86
15:0.60
11:0.66


15:0.46
18:0.45

It performs well on the demo images. It only struggles with detecting all humans when there are many in the same image close to each other.