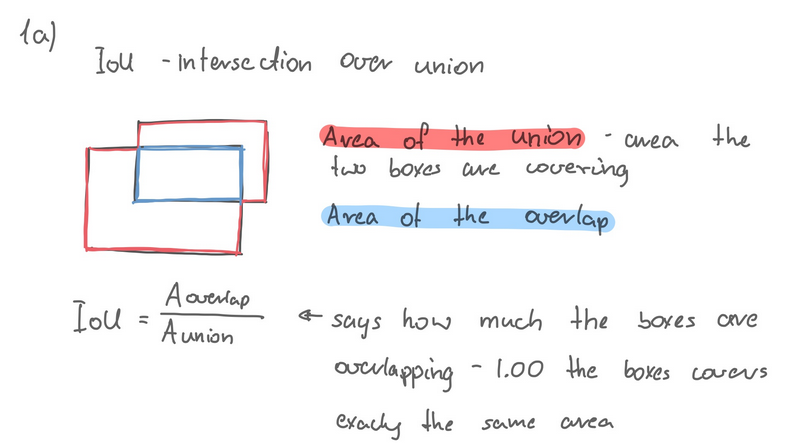
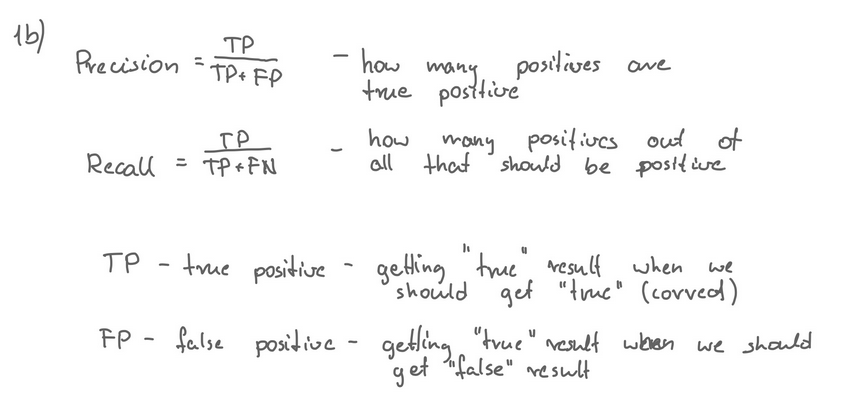
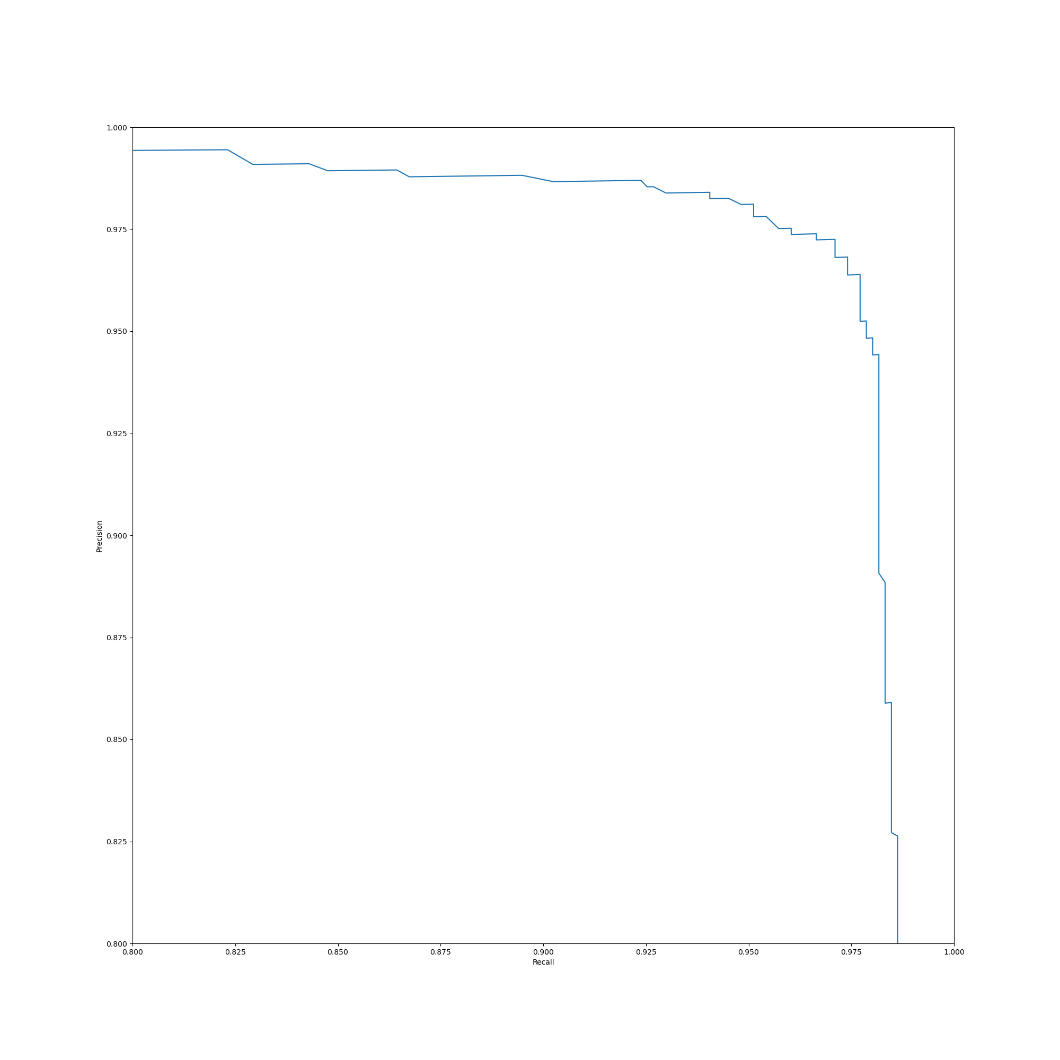
Assignment 4 - group 98

# Task 1



# 

# Task 2

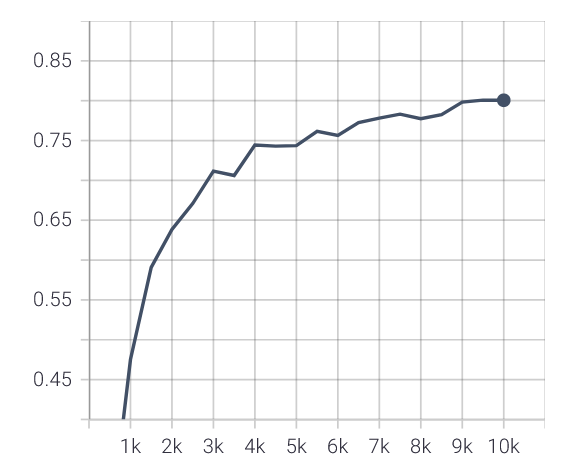


# Task 3

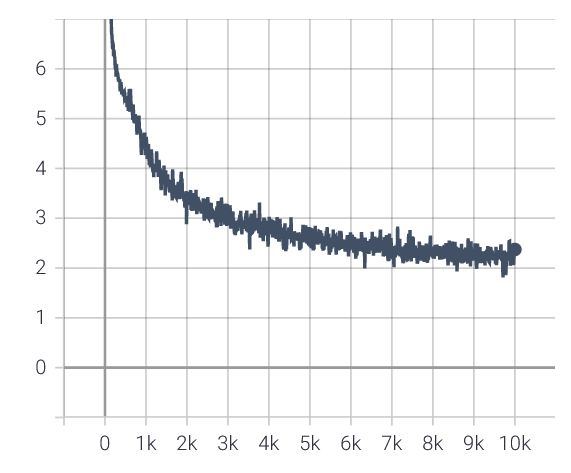
1. To filter out overlapping boxes SSD uses “non-maximum suppression” (nms)
2. Leftmost layers with the highest resolution are only able to detect small objects. Lower resolution layers, deeper in the network, are responsible for detecting bigger objects. Statement is FALSE.
3. Box with different aspect ratios can be matched to more different real-life object that will naturally have varying shapes - aspect ratios. This way predicted boxes will more likely match the ground truth boxes
4. YOLO uses single scale feature map, while SSD uses multi scale feture maps for detection.
5. w x h x 6 = 7776

# Task 4

## 4b



mAP



Total Loss

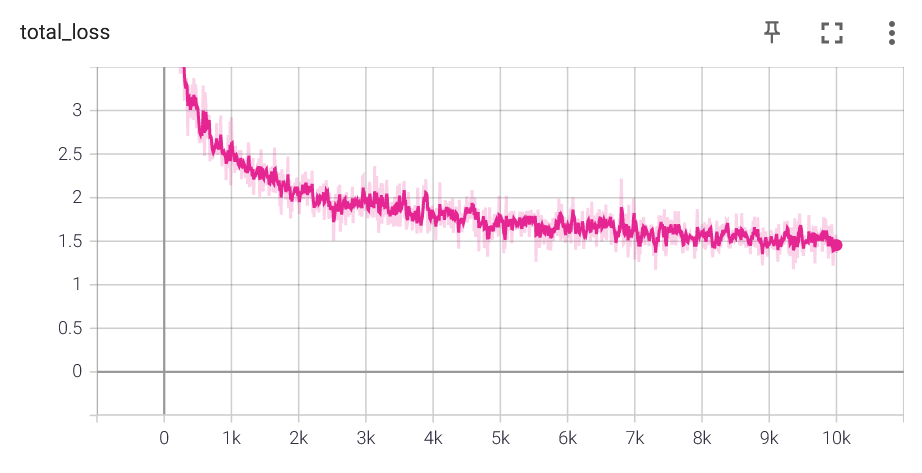
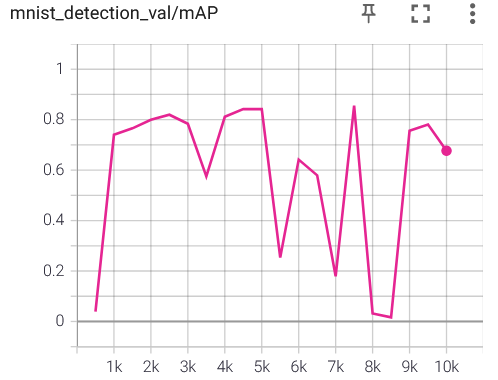
* final mAP (10k steps) = 0.80
* mAP (6k steps) = 0.75

## 4c

The main improvements to the network was batch normalization that resulted in mAP=0.85 after 5k iterations. Other small changes were to change the optimizer to AdamW, reduce learning rate and reduce the batch size. I was however not happy with the results, since the mAP becomes very unstable after 5k iterations. I continued to improve the network and results can be seen in the next task, where the network reached mAP=0.9.

Configs:

* Optimizer: AdamW
  + lr = 1e-3
  + amsgrad = True
* batch size = 10
* OUT\_CHANNELS = [128, 128, 256, 128, 128, 512]



|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Is Output | Layer Type | Number of Filter | Stride | Kernel size |
|  | Conv2d | 32 | 1 | 3 |
|  | MaxPool2d |  |  |  |
|  | ReLU |  |  |  |
|  | Batchnorm2d |  |  |  |
|  | Conv2d | 64 | 1 | 3 |
|  | MaxPool2d |  |  |  |
|  | ReLU |  |  |  |
|  | Batchnorm2d |  |  |  |
|  | Conv2d | 64 | 1 | 3 |
|  | ReLU |  |  |  |
|  | Batchnorm2d |  |  |  |
| Yes – res: 38x38 | Conv2D | Ouput\_channels[0] | 2 | 3 |
|  | ReLU |  |  |  |
|  | Batchnorm2d |  |  |  |
|  | Conv2d | 128 | 1 | 5 |
|  | ReLU |  |  |  |
|  | Batchnorm2d |  |  |  |
|  | Conv2d | 256 | 1 | 5 |
|  | ReLU |  |  |  |
|  | Batchnorm2d |  |  |  |
| Yes – res: 19x19 | Conv2D | Ouput\_channels[1] | 2 | 3 |
|  | ReLU |  |  |  |
|  | Batchnorm2d |  |  |  |
|  | Conv2d | 256 | 1 | 3 |
|  | ReLU |  |  |  |
|  | Batchnorm2d |  |  |  |
|  | Conv2d | 512 | 1 | 3 |
|  | ReLU |  |  |  |
|  | Batchnorm2d |  |  |  |
| Yes – res: 9x9 | Conv2D | Ouput\_channels[2] | 2 | 3 |
|  | ReLU |  |  |  |
|  | Batchnorm2d |  |  |  |
|  | Conv2d | 128 | 1 | 3 |
|  | ReLU |  |  |  |
|  | Batchnorm2d |  |  |  |
| Yes – res: 5x5 | Conv2D | Ouput\_channels[3] | 2 | 3 |
|  | ReLU |  |  |  |
|  | Batchnorm2d |  |  |  |
|  | Conv2d | 128 | 1 | 3 |
|  | ReLU |  |  |  |
|  | Batchnorm2d |  |  |  |
| Yes – res: 3x3 | Conv2D | Ouput\_channels[4] | 2 | 3 |
|  | ReLU |  |  |  |
|  | Batchnorm2d |  |  |  |
|  | Conv2d | 128 | 1 | 3 |
|  | ReLU |  |  |  |
| Yes – res: 1x1 | Conv2D | Ouput\_channels[5] | 1 | 3 |

## 4d

To achieve 0.90 mAP the network from previous task has been redesigned a bit. Main inspiration here was the original VGG19 network. The main difference is the it has been added more convolutional layers at the begging, at the resolutions 150x150 and 75x75. This should help to detected small scaled numbers. The complete network is described in a table on the next page. I addition I am now using decreasing learning rate, which had improved end result and some learning stability. After suggestion from a classmate I have also changed PIXEL \_MEAN and PIXEL\_STD, which also improved the result slightly. No data augmentation was used. Under the training process the mAP is varying a bit. This could be reduced by using dropout and lower learning rate, which I might try later, but I didn’t want to worsen the end result. The final mAP was 0.9013. This is this network that was used for the demo in the next task.

Configs:

* Optimizer: Adam
  + lr = 1e-3
  + amsgrad = True
* batch size = 5
* Input:
  + IMAGE\_SIZE: [300, 300]
  + PIXEL\_MEAN: [0.485,0.456,0.406]

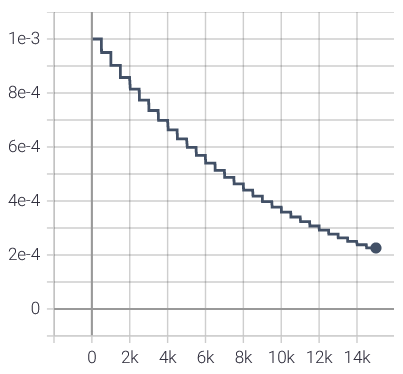
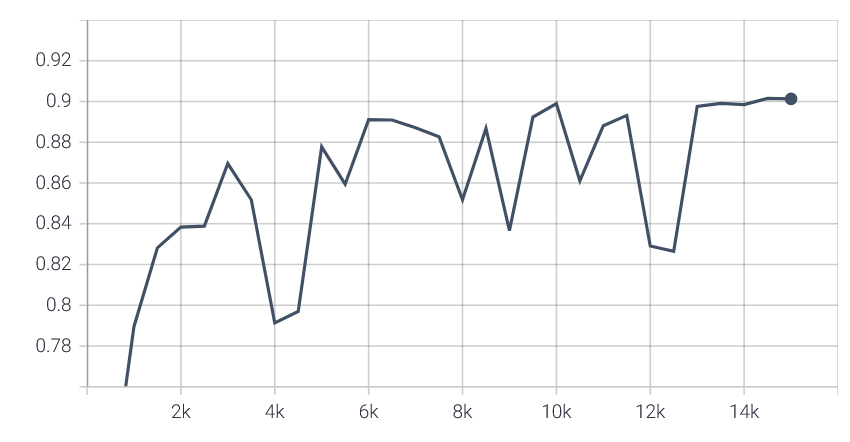
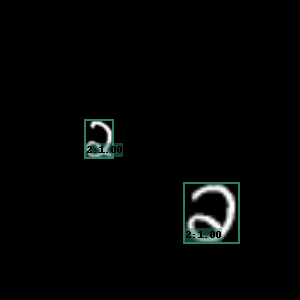
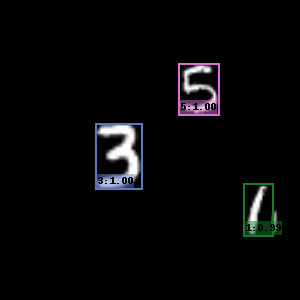
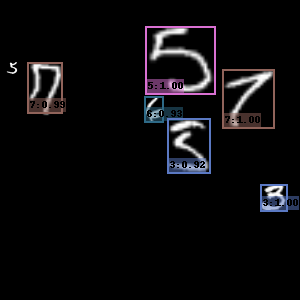
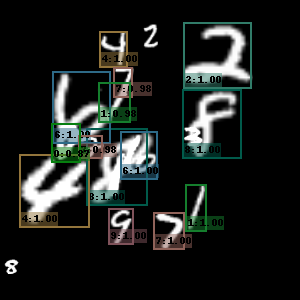
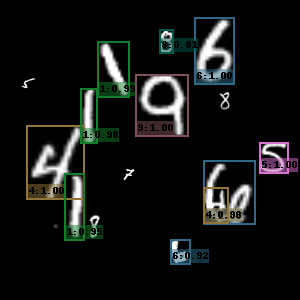
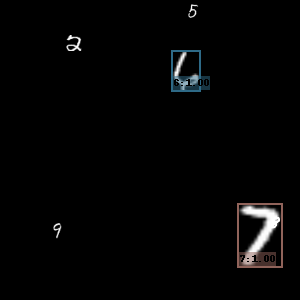
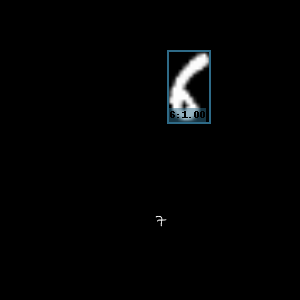


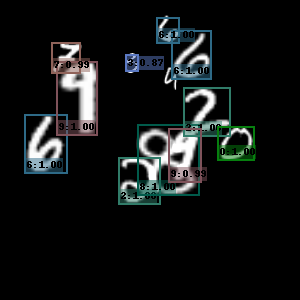
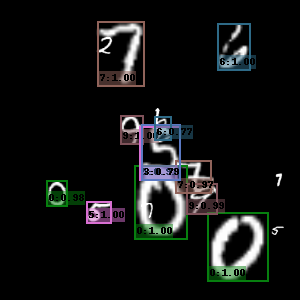
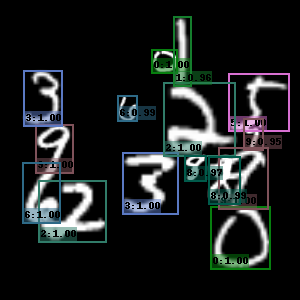
Figure mAP

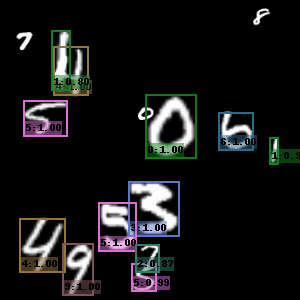
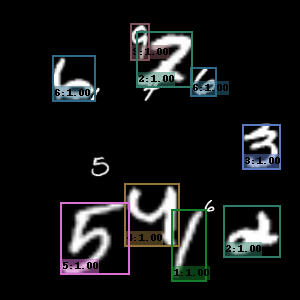
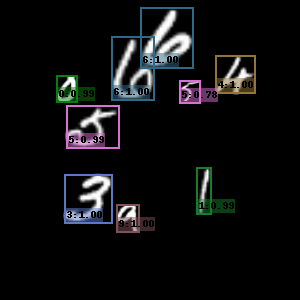
Figure learning rate

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
| Is Output | Layer Type | Number of Filter | Stride | Kernel size |
|  | Conv2d | 32 | 1 | 3 |
|  | ReLU |  |  |  |
|  | Batchnorm2d |  |  |  |
|  | Conv2d | 32 | 1 | 3 |
|  | ReLU |  |  |  |
|  | Batchnorm2d |  |  |  |
|  | Conv2d | 32 | 1 | 3 |
|  | MaxPool2d |  |  |  |
|  | ReLU |  |  |  |
| 150x150 | Batchnorm2d |  |  |  |
|  | Conv2d | 64 | 1 | 3 |
|  | ReLU |  |  |  |
|  | Batchnorm2d |  |  |  |
|  | Conv2d | 64 | 1 | 3 |
|  | ReLU |  |  |  |
|  | Batchnorm2d |  |  |  |
|  | Conv2d | 64 | 1 | 3 |
|  | MaxPool2d |  |  |  |
|  | ReLU |  |  |  |
| 75x75 | Batchnorm2d |  |  |  |
|  | Conv2d | 128 | 1 | 3 |
|  | ReLU |  |  |  |
|  | Batchnorm2d |  |  |  |
|  | Conv2d | 128 | 1 | 3 |
|  | ReLU |  |  |  |
|  | Batchnorm2d |  |  |  |
|  | Conv2d | 128 | 2 | 3 |
|  | ReLU |  |  |  |
| Yes – res: 38x38 | Batchnorm2d |  |  |  |
|  | Conv2d | 256 | 1 | 3 |
|  | ReLU |  |  |  |
|  | Batchnorm2d |  |  |  |
|  | Conv2d | 256 | 1 | 3 |
|  | ReLU |  |  |  |
|  | Batchnorm2d |  |  |  |
|  | Conv2d | 256 | 2 | 3 |
|  | ReLU |  |  |  |
| Yes – res: 19x19 | Batchnorm2d |  |  |  |
|  | Conv2d | 256 | 2 | 3 |
|  | ReLU |  |  |  |
| Yes – res: 10x10 | Batchnorm2d |  |  |  |
|  | Conv2d | 256 | 1 | 3 |
|  | ReLU |  |  |  |
|  | Batchnorm2d |  |  |  |
|  | Conv2d | 256 | 2 | 3 |
|  | ReLU |  |  |  |
| Yes – res: 5x5 | Batchnorm2d |  |  |  |
|  | Conv2d | 512 | 1 | 3 |
|  | ReLU |  |  |  |
|  | Batchnorm2d |  |  |  |
|  | Conv2d | 512 | 2 | 3 |
|  | ReLU |  |  |  |
| Yes – res: 3x3 | Batchnorm2d |  |  |  |
|  | Conv2d | 512 | 1 | 3 |
|  | ReLU |  |  |  |
|  | Batchnorm2d |  |  |  |
| Yes – res: 1x1 | Conv2D | 512 | 1 | 3 |

4e







We can see that model is strugling with small numbers, but those doesn’t seem to be detected by ssd at all. Most digits are however catagorized correctly.

## 4f

Final mAP of the network was 0.2172. It wasn’t able to detect many objects in the provided images either. The only image that got recognized was the one of a cat.



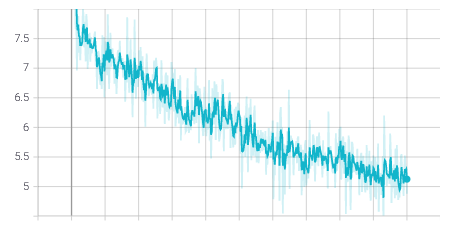


Figure Total loss