CS783 Assignment3

Harshit Sharma (160283)

Ashish Kumar (160160)

Introduction

Aim of this assignment was object detection using 1 layer and 2 layer Resnet18 Model. Given a test image we need to predict bounding box only if it belongs to one of the three classes(airplane,bottle,chair) and if not then no bounding box should be predicted.

Data Generation and Pre-Processing

We made **two types of datasets**. The first one had lesser number of background images while the **second one contains much greater** number of background images.

First Dataset

- **Approach :** Initially we crop our training and test data into standard ImageNet size i.e (224,224) and then for obtaining background images we used **sliding window** approach and then we take the **IoU(Intersection of Union)** of the patch with the **3 classes** and if it comes out to be less than 0.2 than we're taking it as the background candidate and after obtaining the candidates from all the images we'll be randomly choosing background images from the candidates.
- **Challenges faced :** As background images were **not sufficient** to fully generalize the whole background because background is too much diverse and due to this, cases of recognizing sofas as chairs , windows as chairs, humans as chairs and the cases where humans were sitting on something else but that was still recogninzed as chairs etc, were very prominent and this was decreasing accuracy humongously. So we switched to different to our second method of data generation.

Second Dataset

- **Approach :** Cropping of both training and test into (224,224) was same as before, but this time we're also included the rest of 17 classes as background and now to enhance our training on background which doesn't belong to any of the 20 classes, we took **IoU of background patch with all of the 20 classes and if it's less than 0.2** then we took it as candidate and then did random selection from the candidates.
- **Challenges :** Due to very much background images in training it's possible that so
- **Benefits :** This dataset contains sufficient background images for generalization and thus our earlier wrong predictions of predicting background as some class got eliminated from this approach.

Training

One Layer Network

- **Model :** We're using the same network as given in the python notebook for training and then passing it through the linear Fully connected layer which is then classifying it into 4 classes(background + 3 classes). We're using **SGD** as optimizer and **CrossEntropyloss** as criterion.

- **Training Accuracy :**
  Overall Accuracy : 96%
  Classwise Accuracy : Background : 95% Airplane : 98% Bottle : 94% Chair : 96%

Two Layer Network

- **Model :** In this we took the **last 2 conv layer** of the Resnet18 model with the last layer of 512 size and second last layer of 256 size and then applied **adaptive average pooling** on both conv layers seperately and then concatenated them and after passed it through the **linear fully connected layer** which classifies it into 4 classes. We're using **SGD** as optimizer and **CrossEntropyloss** as criterion.
- **Improvisations :** As we were training it on large amount of data thus we decreased our number of epochs from 18 to 6 so that first training occurs fast and second to avoid overfitting.
- **Training Accuracy :**
  Overall Accuracy : 94%
  Classwise Accuracy : Background : 97% Airplane : 96% Bottle : 92% Chair : 93%.

Predicting Bounding Boxes

- **Approach** We took sliding windows of different aspect ratios and were also downscaling our original image because it is helpful for those cases in which bounding box is not big enough to completely contain the whole object, we slid these windows through our test images to get multiple bounding boxes and then applied **NMS(Non-Maximum Supression)** with threshold 0.3 such that adjacent bounding boxes with same labels get merged to generate a better bounding box.
- **Test Accuracy**
  - One Layer : Overall : 92.1% Background : 93.4% Airplane : 92.6% Bottle : 91.2% Chair : 89.5%
  - Two Layer : Overall : 92.9% Background : 94.7% Airplane : 93.0% Bottle : 91.5% Chair : 90.1%
- **mAP Scores**
  - Threshold = 0.25
    One layer : Overall : 0.324 Airplane : 0.490 Bottle : 0.215 Chair : 0.269
    Two Layer : Overall : 0.334 Airplane : 0.507 Bottle : 0.231 Chair : 0.270
  - Threshold = 0.35
    One layer : Overall : 0.243 Airplane : 0.380 Bottle : 0.151 Chair : 0.120
    Two Layer : Overall : 0.261 Airplane : 0.398 Bottle : 0.147 Chair : 0.125
  - Threshold = 0.50
    One layer : Overall : 0.128 Airplane : 0.173 Bottle : 0.091 Chair : 0.121
    Two Layer : Overall : 0.141 Airplane : 0.187 Bottle : 0.103 Chair : 0.129
  - Final mAP average of above three
    One layer : Overall : 0.232 Airplane : 0.348 Bottle : 0.152 Chair : 0.170
    Two Layer : Overall : 0.245 Airplane : 0.364 Bottle : 0.160 Chair : 0.175

References

- https://www.pyimagesearch.com/2015/03/23/sliding-windows-for-object-detection-with-python-and-opencv

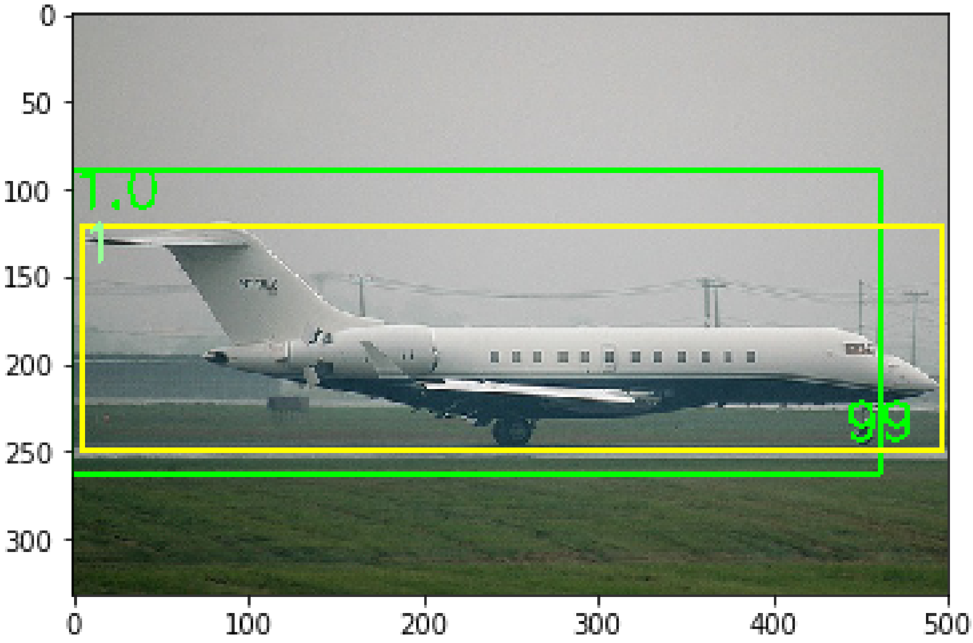
- https://pytorch.org/tutorials/beginner/blitz/cifar10tutorial.html

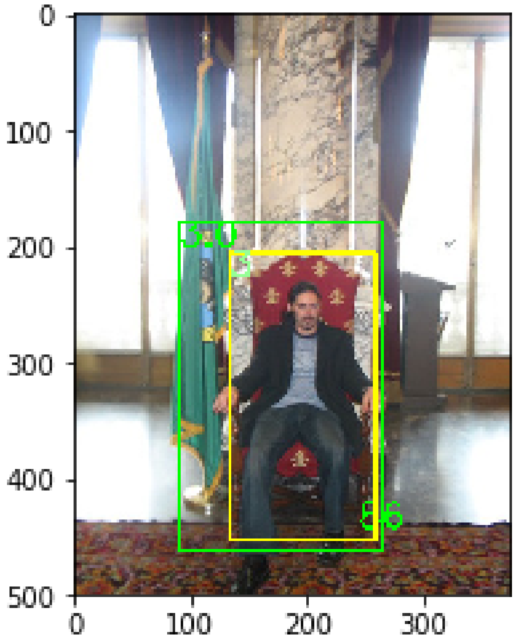
- https://stackoverflow.com/questions/25349178/calculating-percentage-of-bounding-box-overlap-for-image-detector-evaluation
- https://github.com/sgrvinod/a-PyTorch-Tutorial-to-Object-Detection/blob/master/utils.py
- https://github.com/rbgirshick/fast-rcnn/blob/master/lib/utils/nms.py

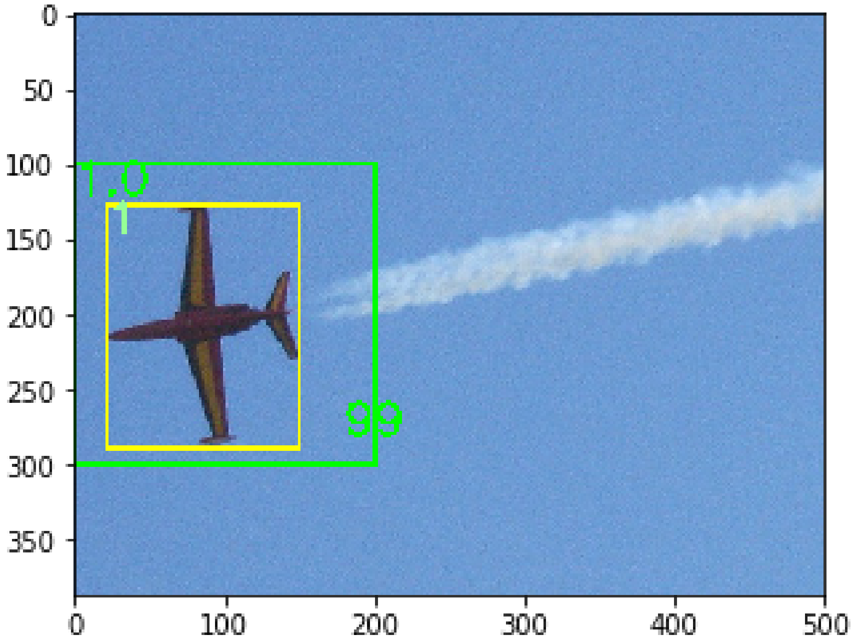
Correct Samples

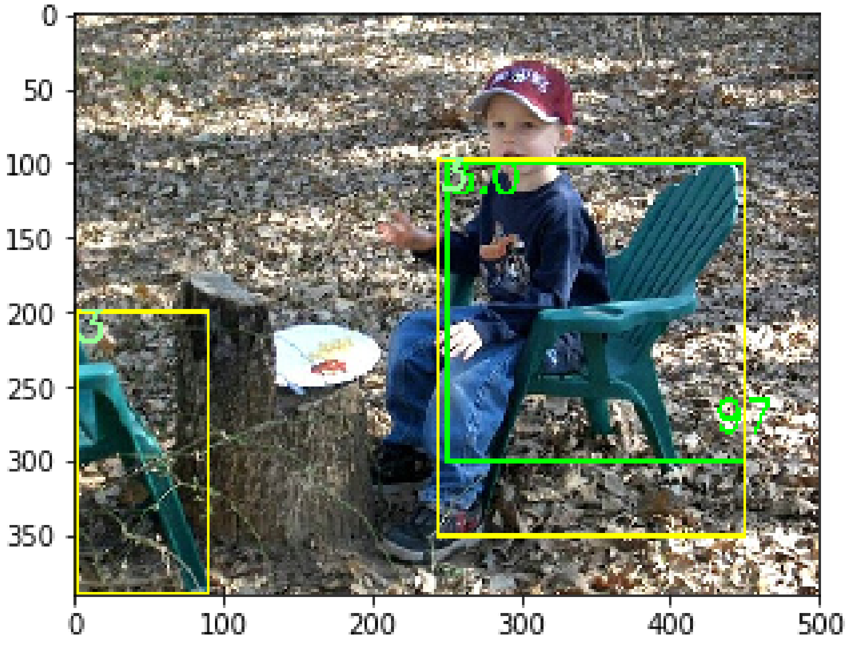
Green bounding box is our predicted box and yellow is the original box.

One Layer



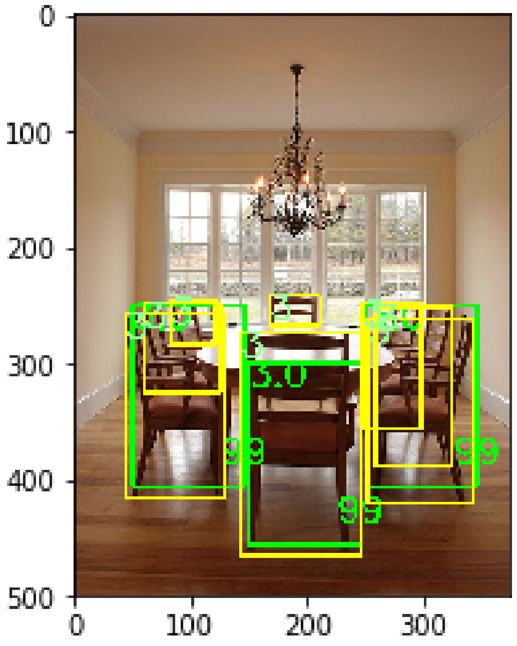

3.0
3
56
0
100
200
300
400
500
0
100
200
300


1.0
1
99
0
50
100
150
200
250
300
350
0
100
200
300
400
500

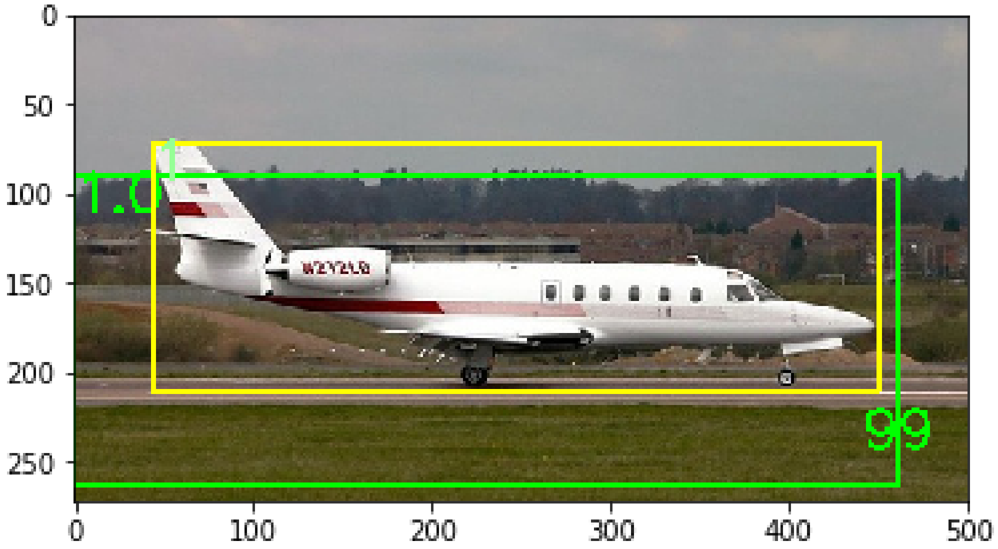

3.0
97
3
0
50
100
150
200
250
300
350
0
100
200
300
400
500

3.0
3
9
0
50
100
150
200
250
300
350
0
100
200
300
400
500

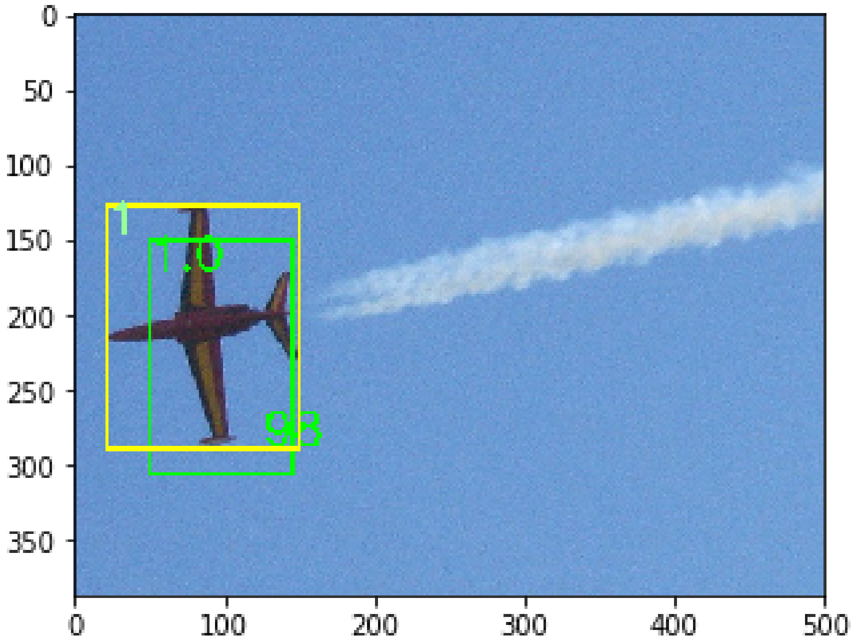
Two Layer



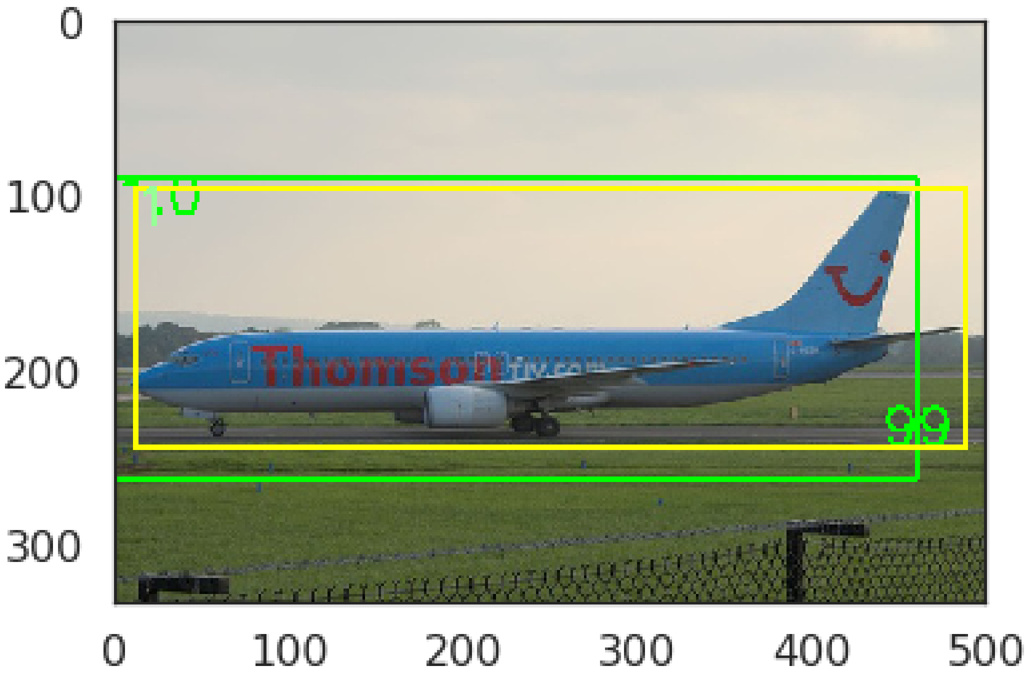
0
100
200
300
400
500
0
100
200
300
3.0
99
99
99



0
50
100
150
200
250
0
100
200
300
400
500
1.0
99

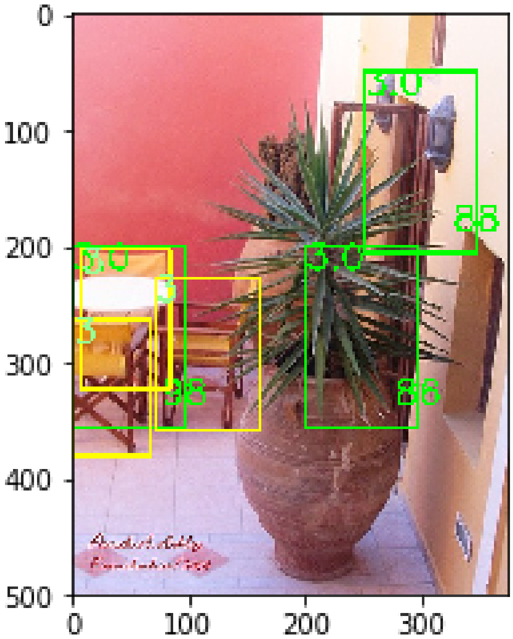

0
50
100
150
200
250
300
350
0
100
200
300
400
500
1
1.0
98

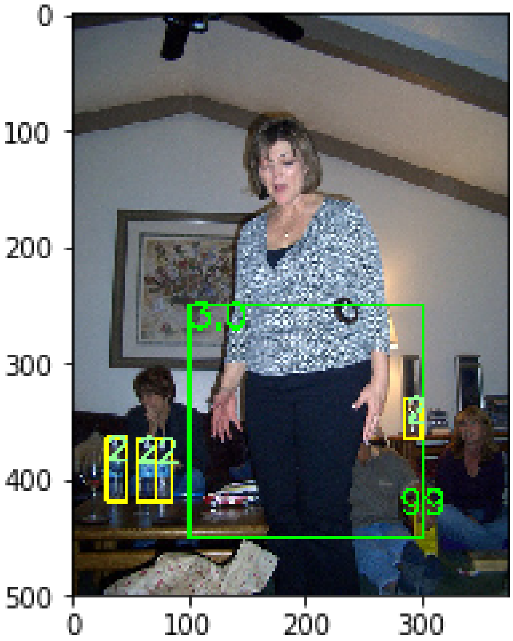
0
100
200
300
0
100
200
300
400
500
1.0
99

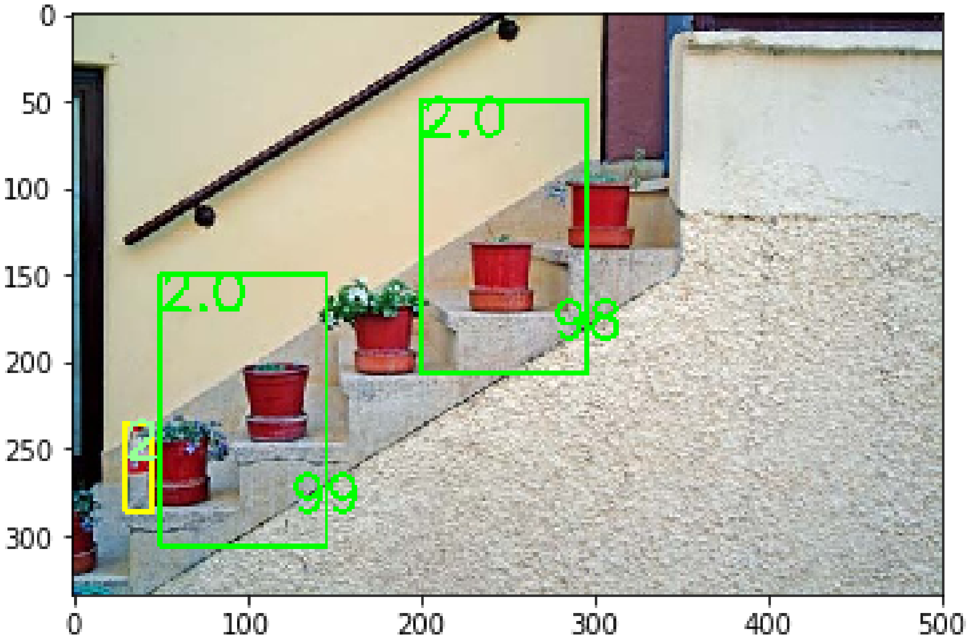

0
100
200
300
0
100
200
300
400
500
1.0
99
Thomson

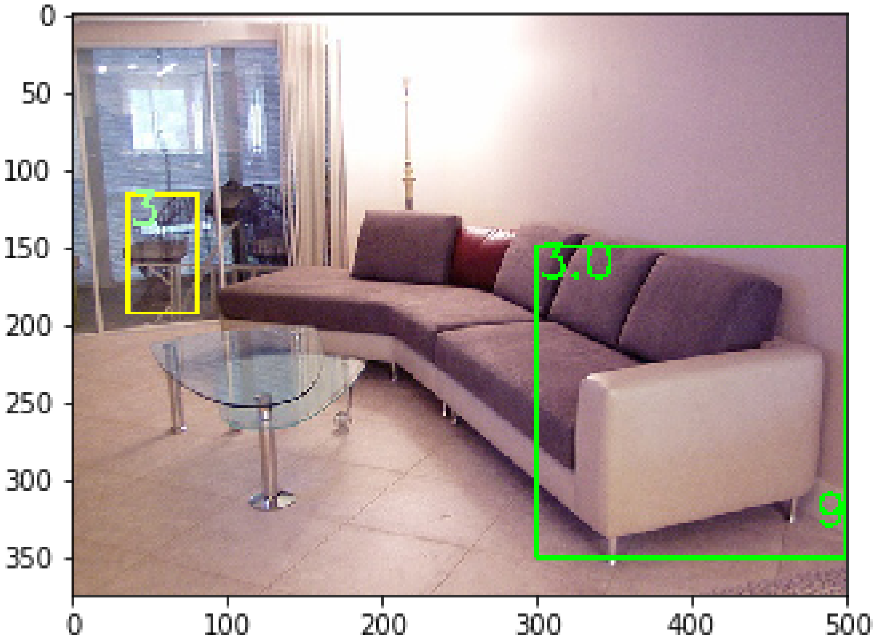
Incorrect Samples

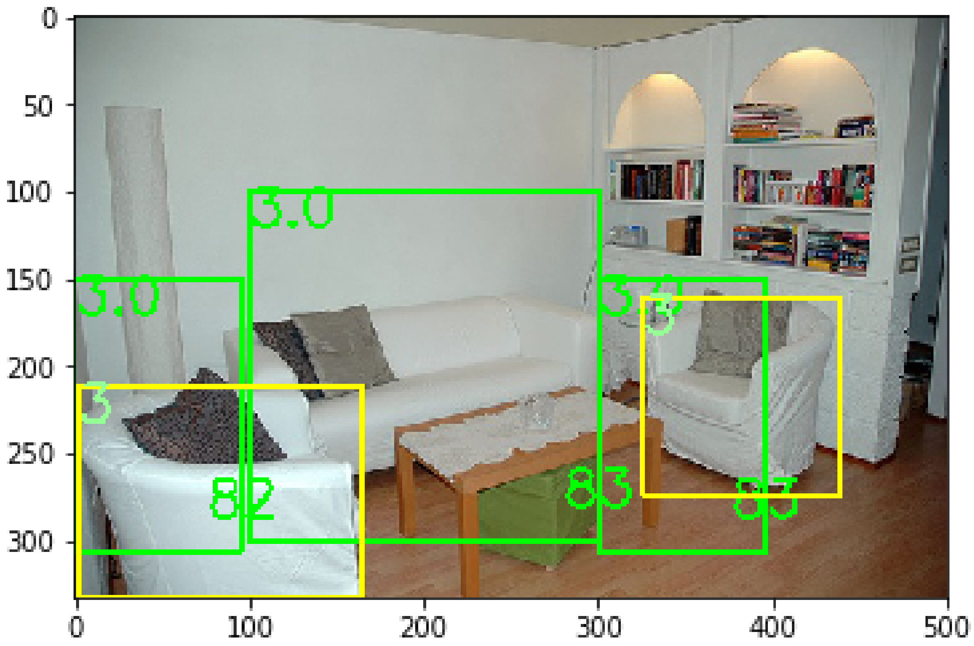
One Layer



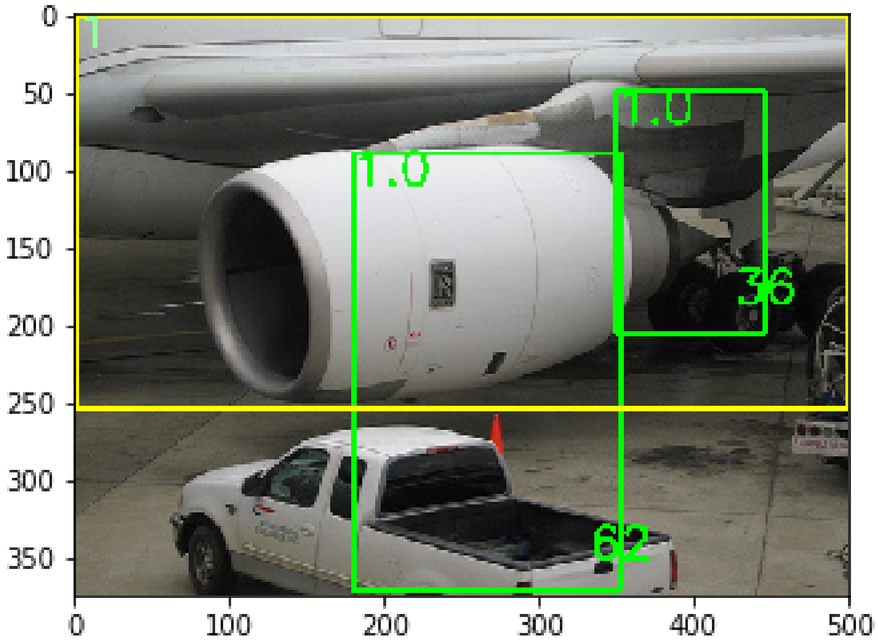



0
50
100
150
200
250
300
2.0
98
2.0
99
2
0
100
200
300
400
500

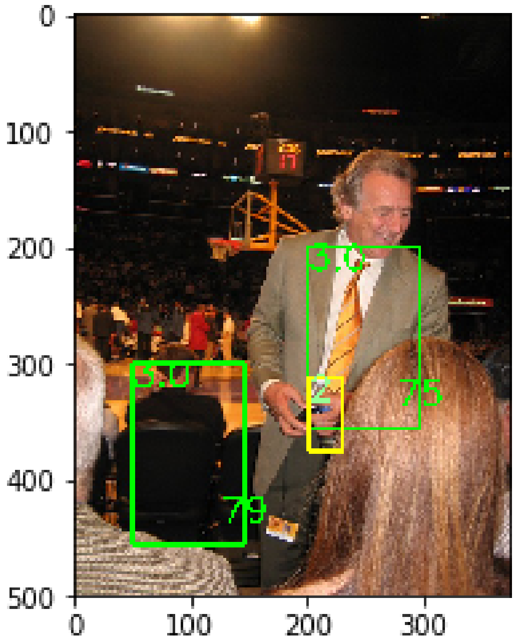

0
50
100
150
200
250
300
350
3
3.0
9
0
100
200
300
400
500


0
50
100
150
200
250
300
0
100
200
300
400
500
3.0
3.0
3.0
3
3
82
83
83

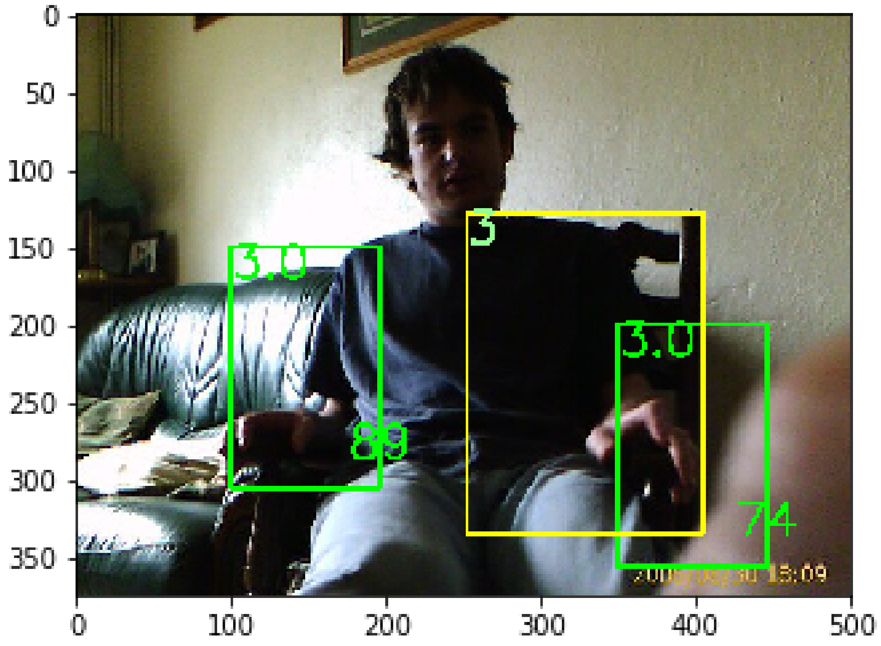
Two Layer

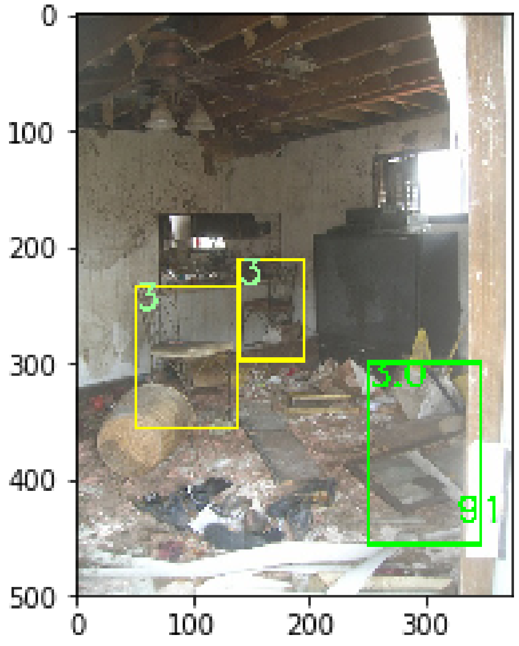


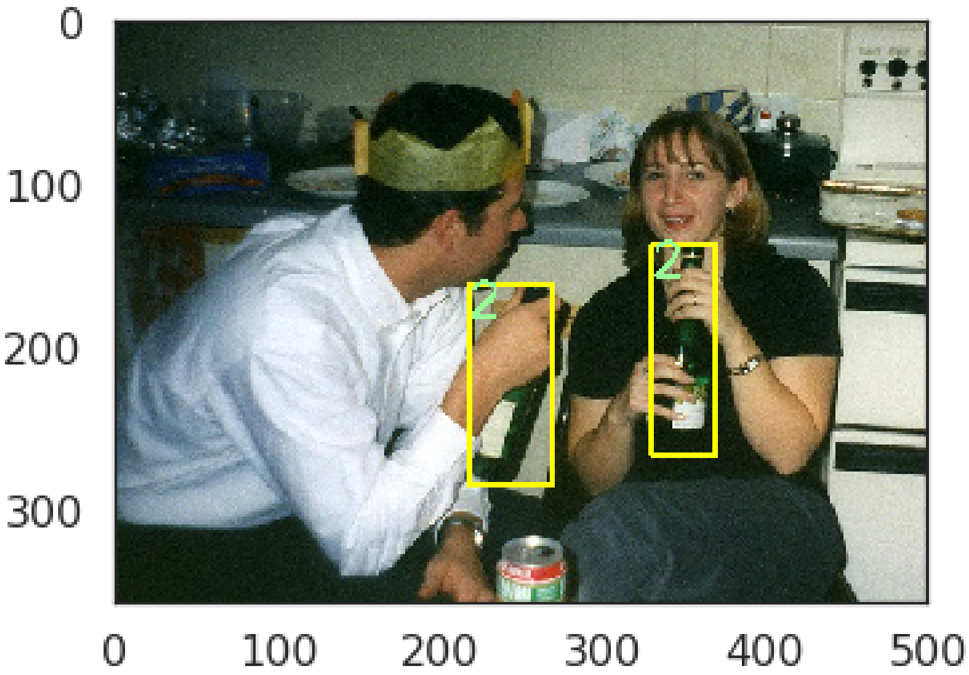
1
1.0
1.0
36
62
0
50
100
150
200
250
300
350
0
100
200
300
400
500



3.0
2
75
3.0
79
0
100
200
300
400
500
0
100
200
300


0
50
100
150
200
250
300
350
3.0
89
3
3.0
74
0
100
200
300
400
500


0
100
200
300
400
500
3
3
3.0
91
0
100
200
300


0
100
200
300
0
100
200
300
400
500
2
2