MaskRCNN Report V2.0 (adaptive bg_thresh)(No special engineering needed in the single object detection!!!)

Visual result on testset:

Setup:

In the inference phase, I divide whole maskrcnn task into two main tasks:

- Single object detection task
- Multimple objects detection task

Colors

- red is the ground truth object bounding box
- Blue is prediction for car
- green is prediction for person
- yellow is the prediction for animals
- light pink is the original proposal box

Training:

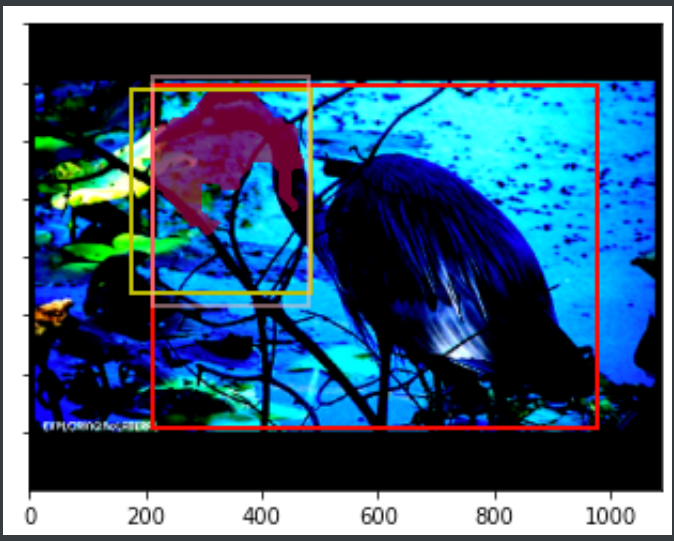
Results are based on 40-th epoch training.

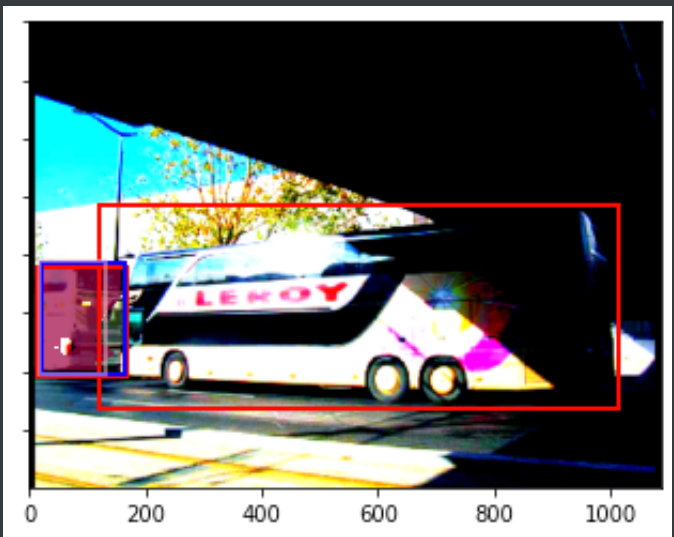
single object detection task(without any special engineering):

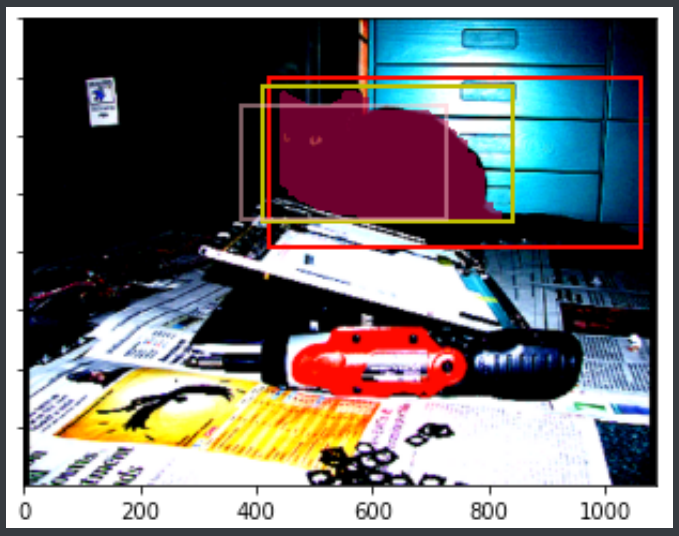
Here, to avoid the bias, I choose the first 20 images from test set without skip as a preview of the perform ence of this network.

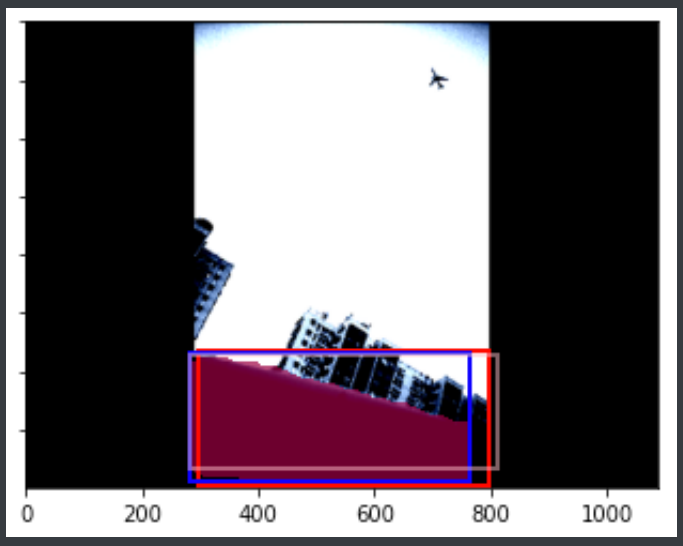
All these detection and mask representations come from only one row of prediction, that means no manully inference is involved.

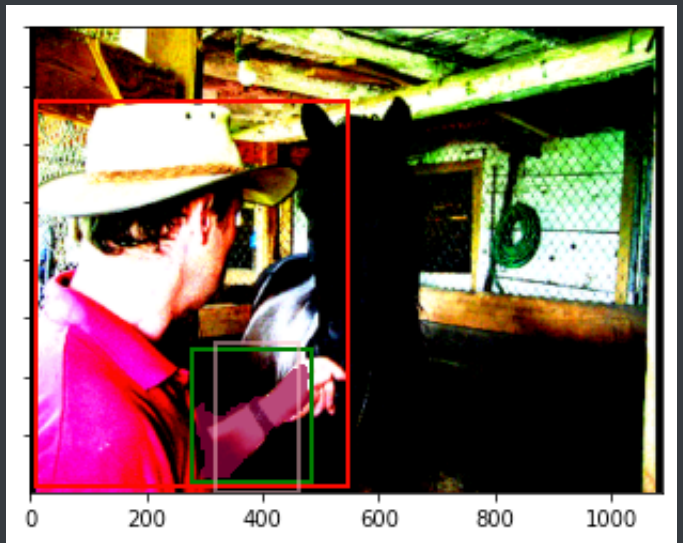
Note: multiple resulting image represent the detections with the same confidence level.



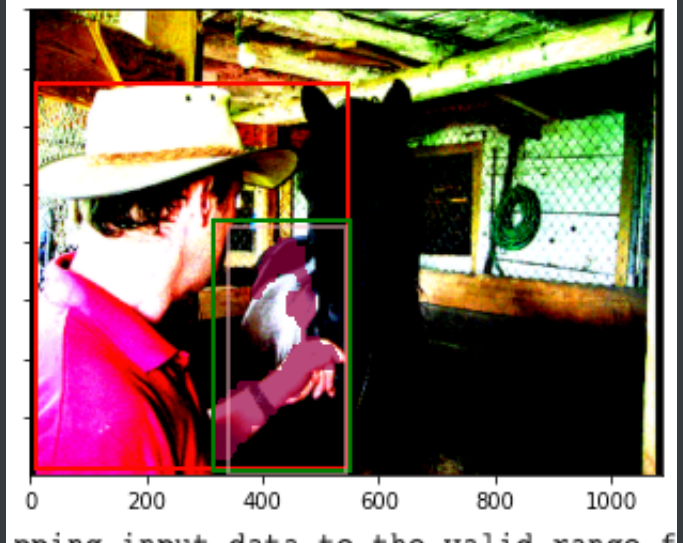



0
200
400
600
800
1000

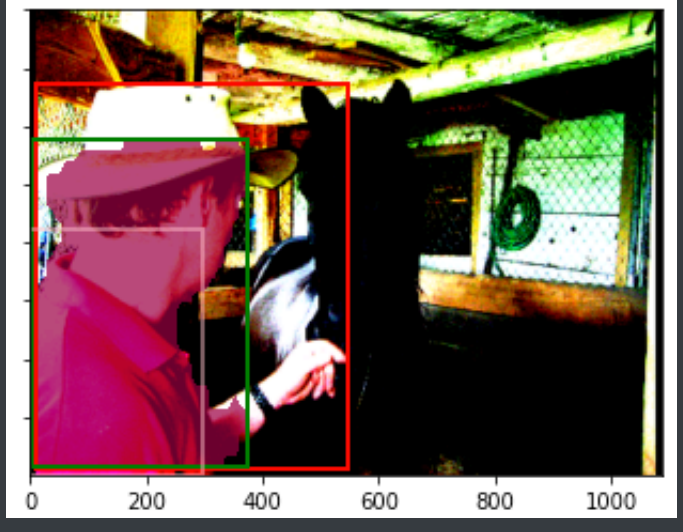

0
200
400
600
800
1000

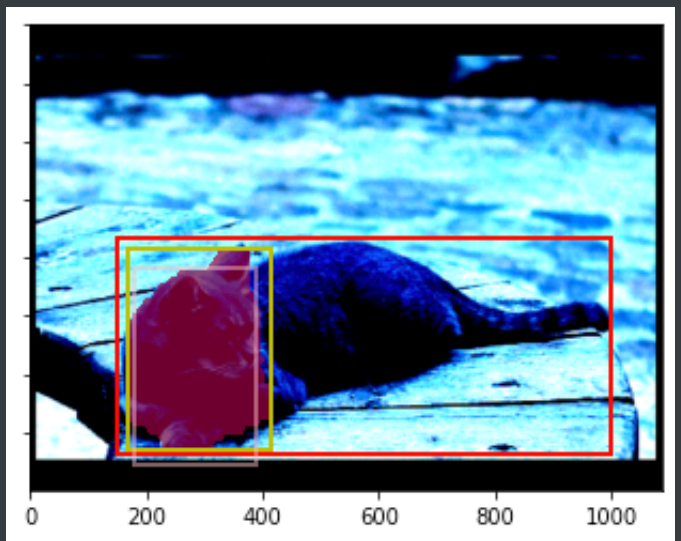



```
pping input data to the valid range f
```

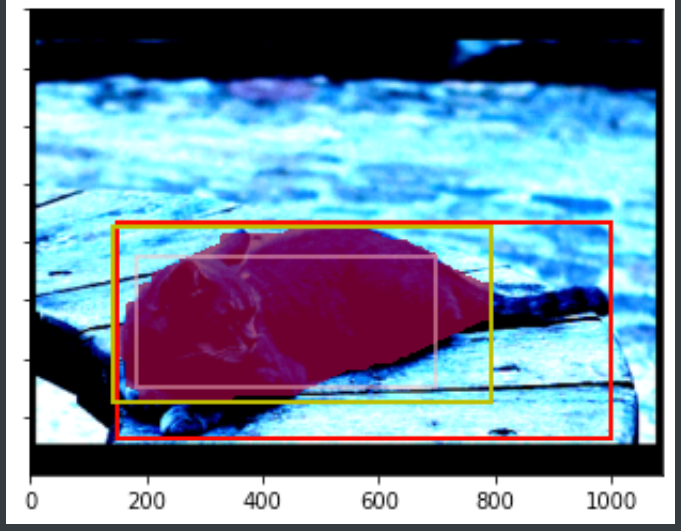


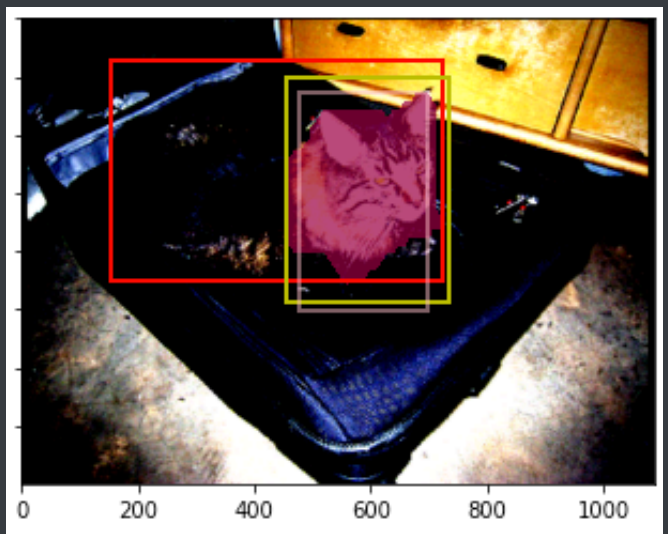

```
pping input data to the valid range f
```



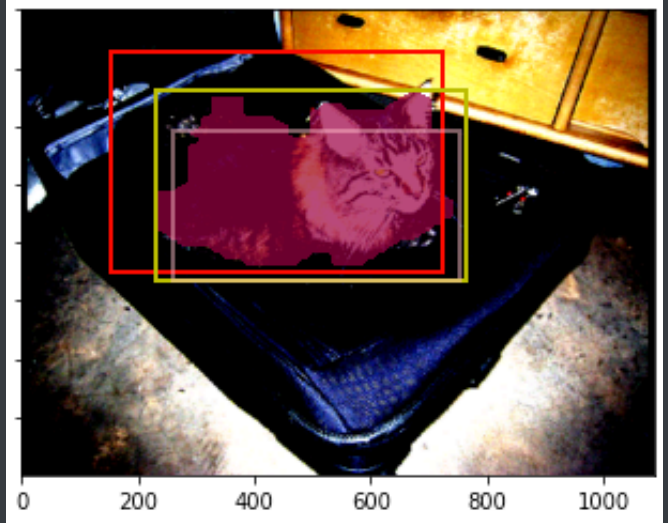


pping input data to the valid range f

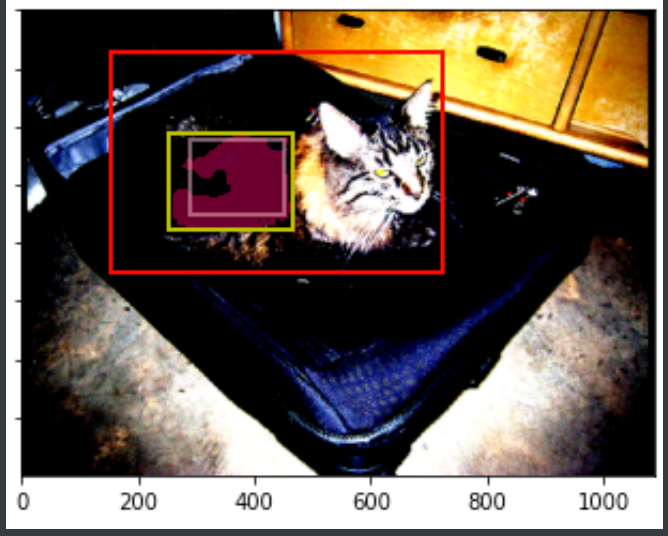


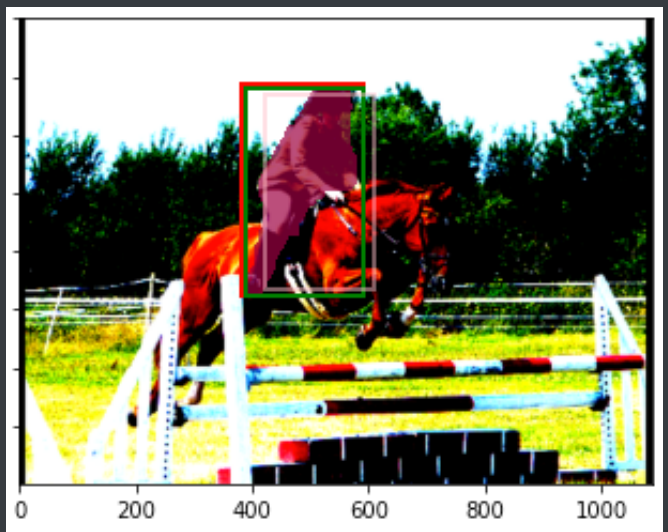


pping input data to the valid range f

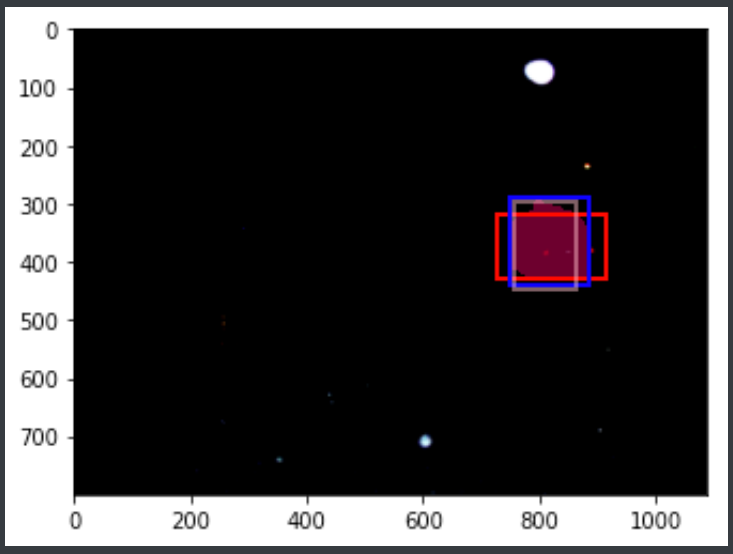


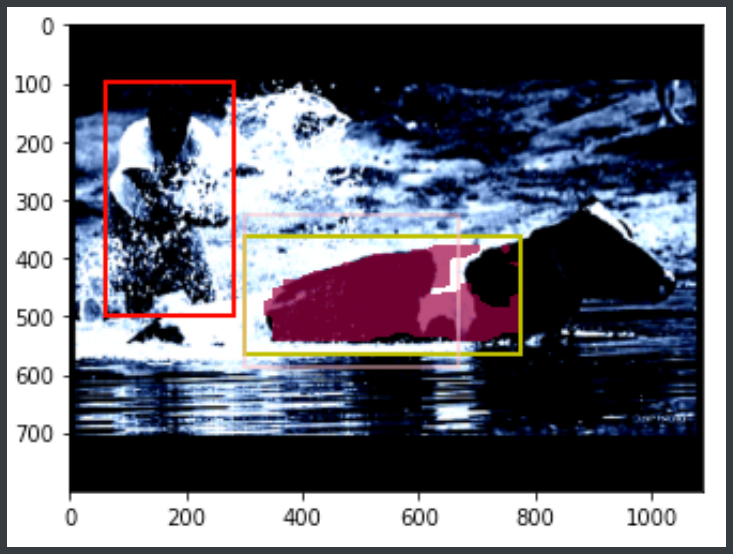
pping input data to the valid range f

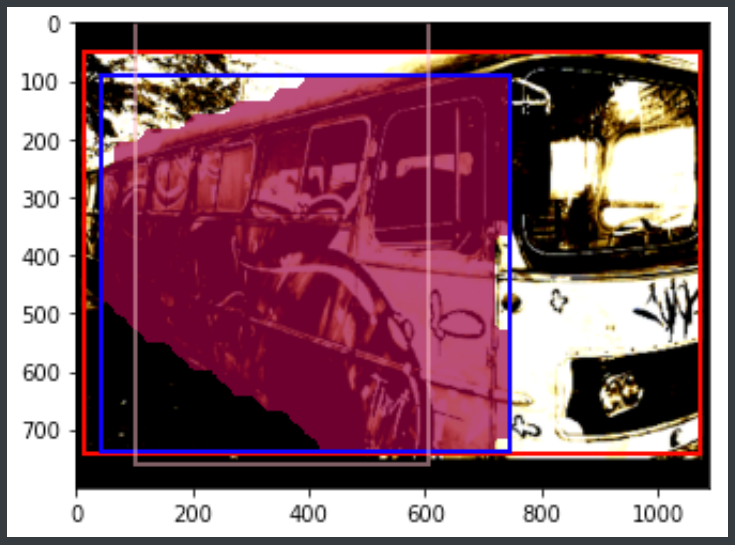




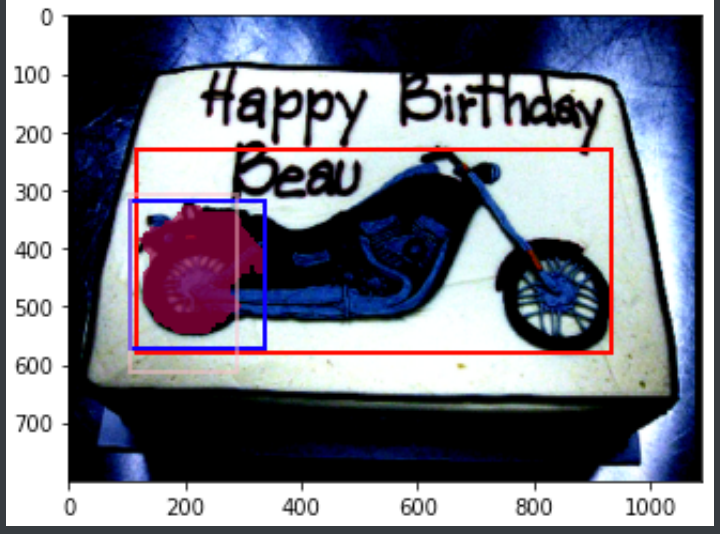
pping input data to the valid range f

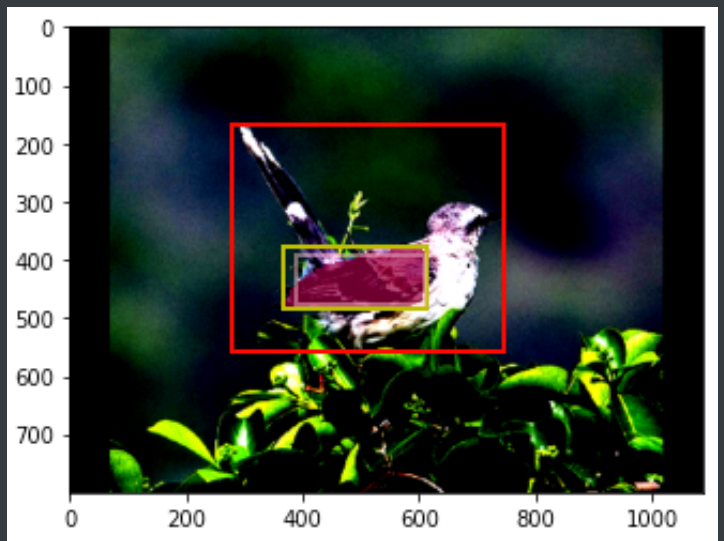



0
100
200
300
400
500
600
700
0
200
400
600
800
1000

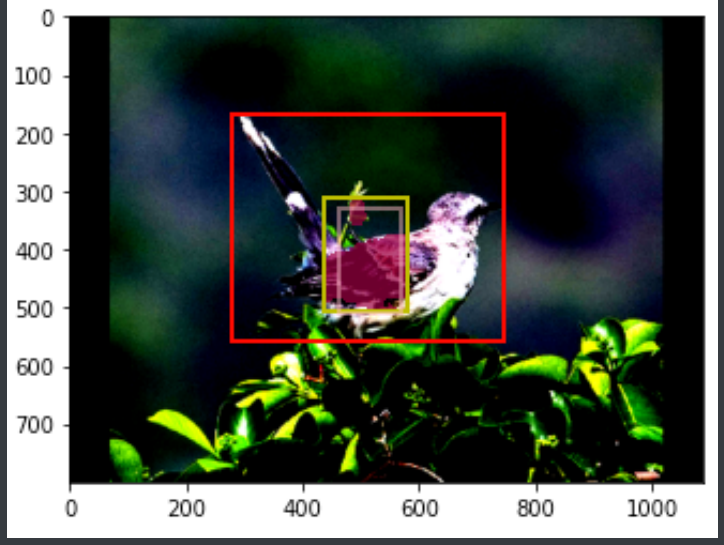

0
100
200
300
400
500
600
700
0
200
400
600
800
1000

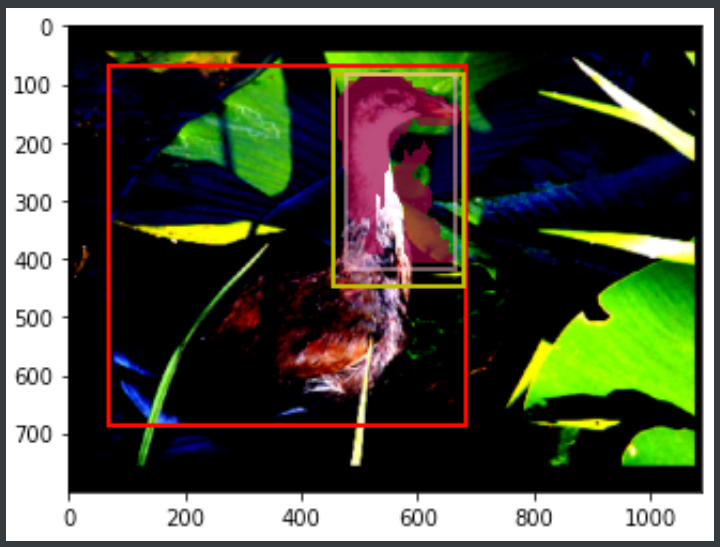

```
!lipping input data to the valid range f
```

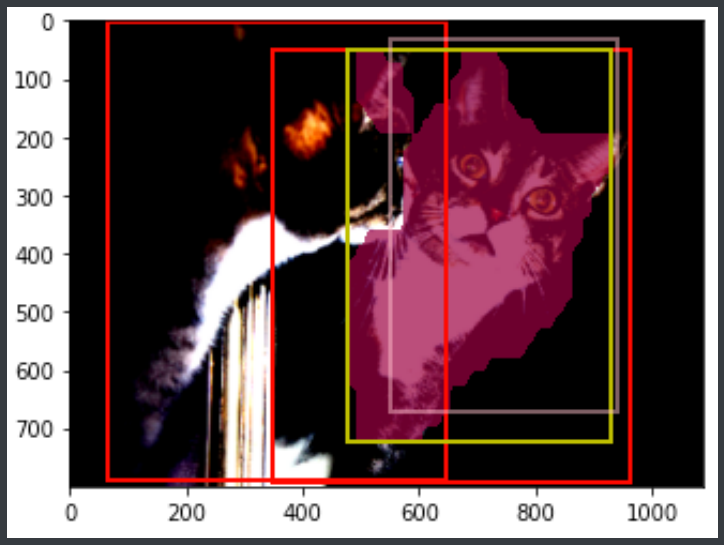


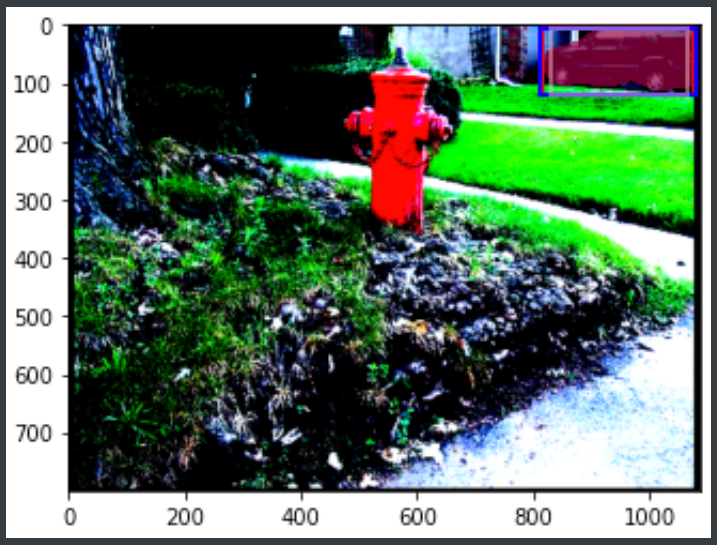


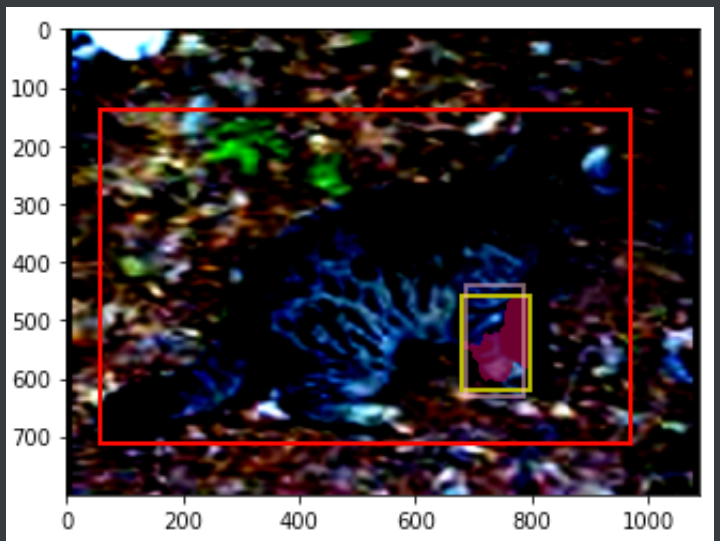
Clipping input data to the valid range f



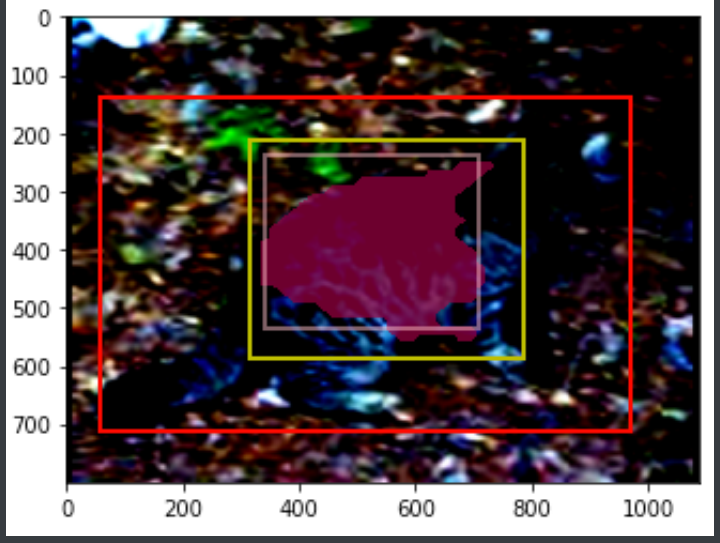


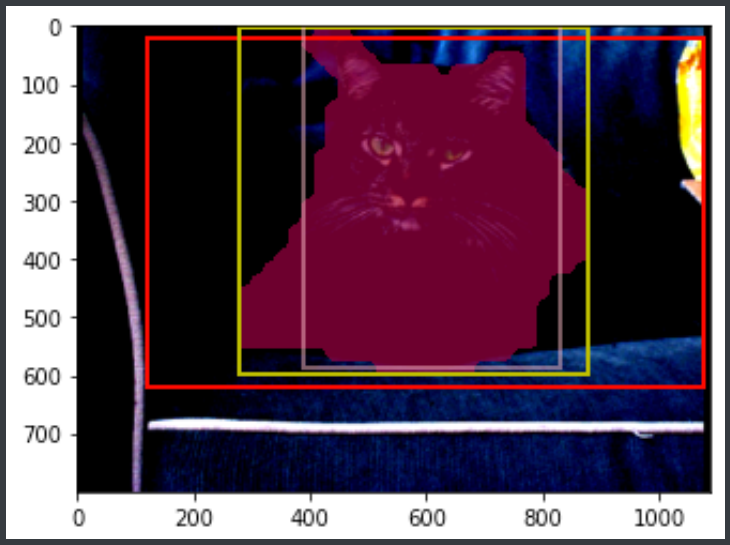

0
100
200
300
400
500
600
700
0
200
400
600
800
1000

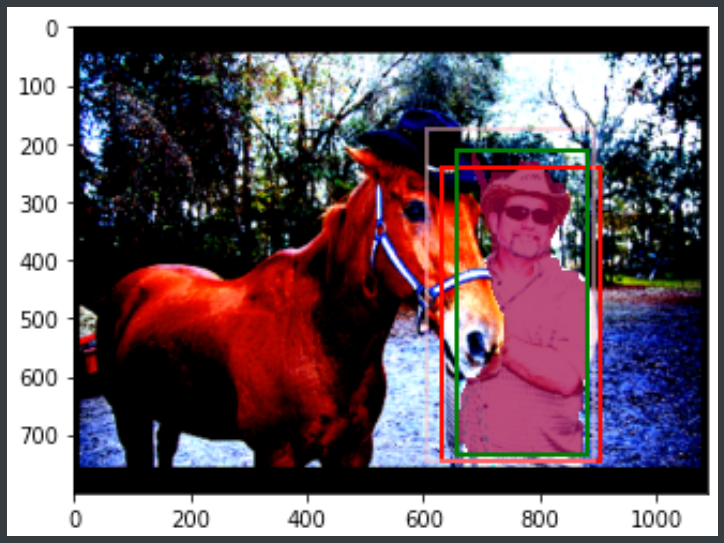

0
100
200
300
400
500
600
700
0
200
400
600
800
1000

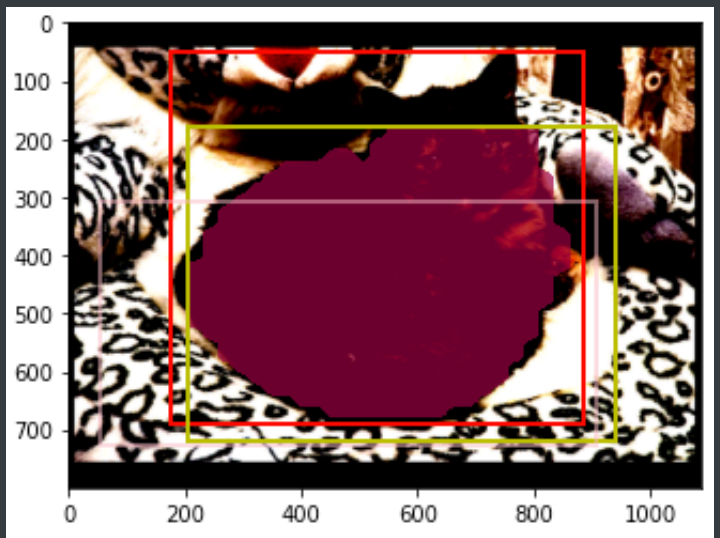



```
Clipping input data to the valid range f
```

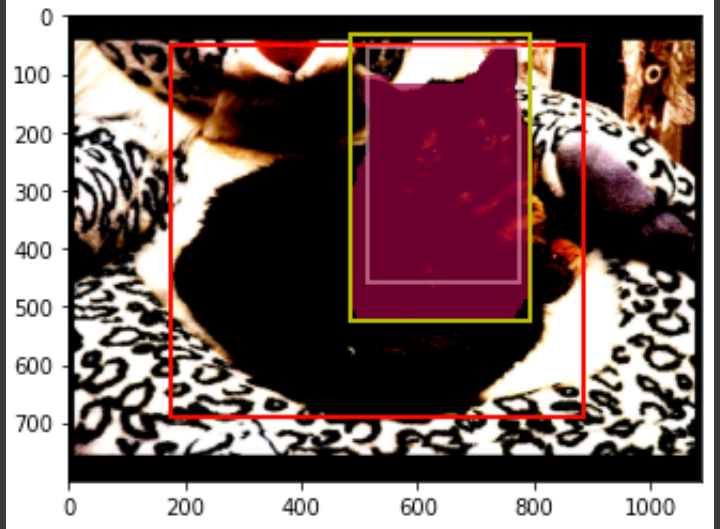




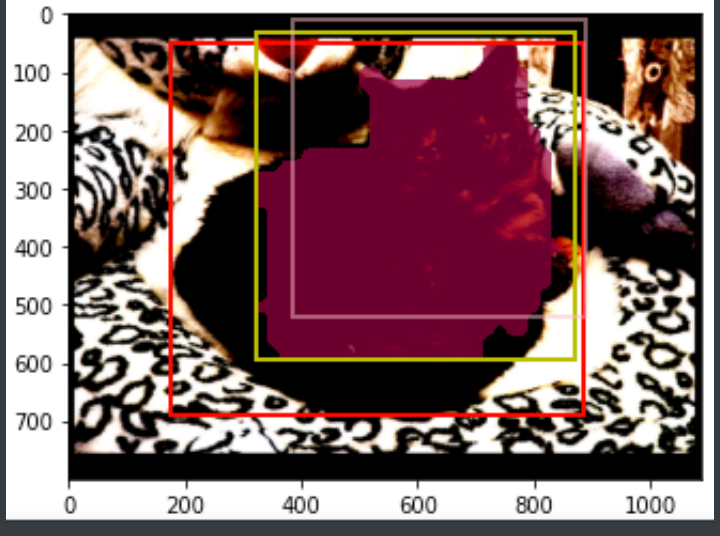

0
100
200
300
400
500
600
700
0
200
400
600
800
1000

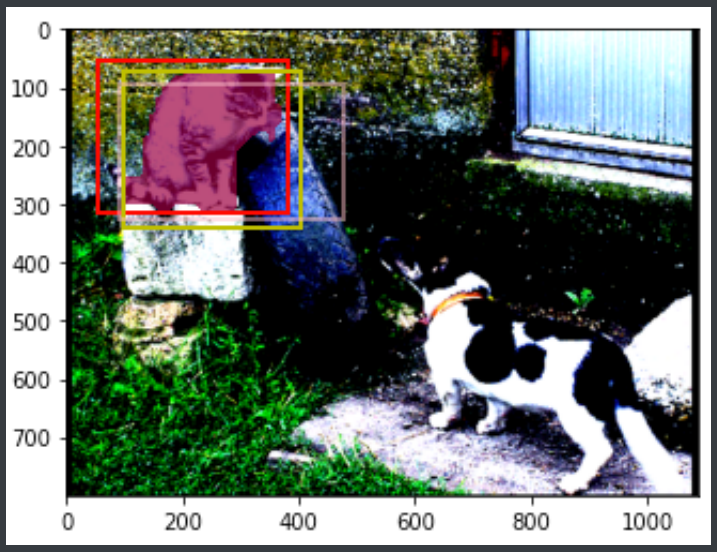


Clipping input data to the valid range f



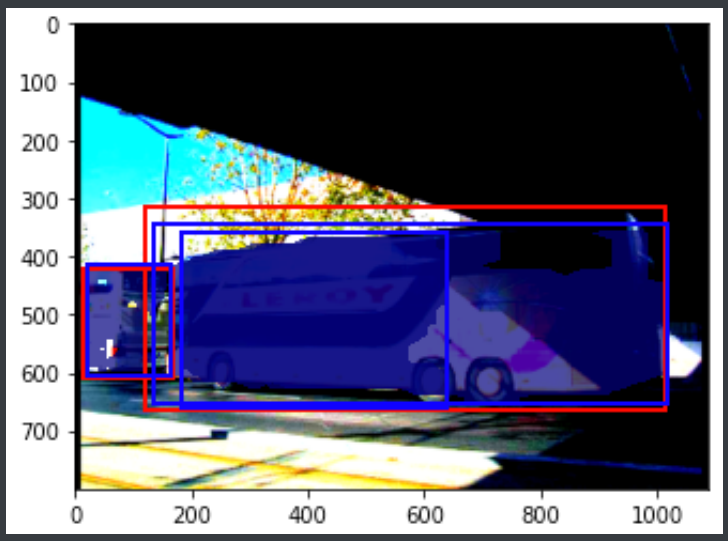
Clipping input data to the valid range f

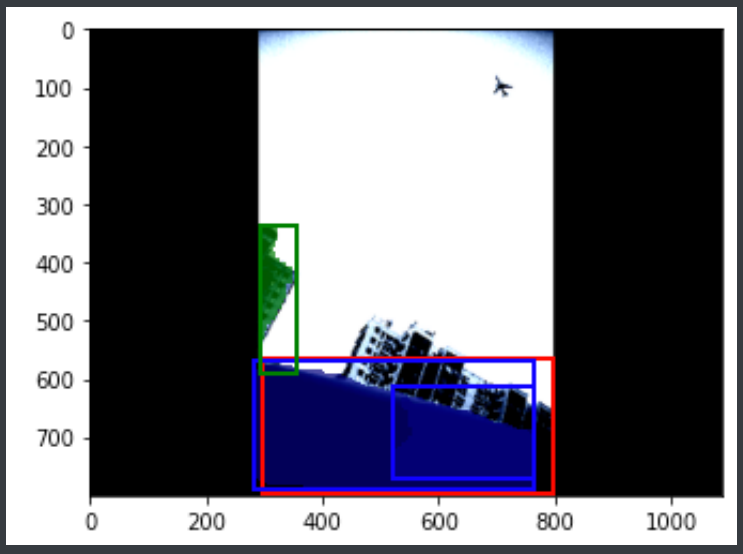


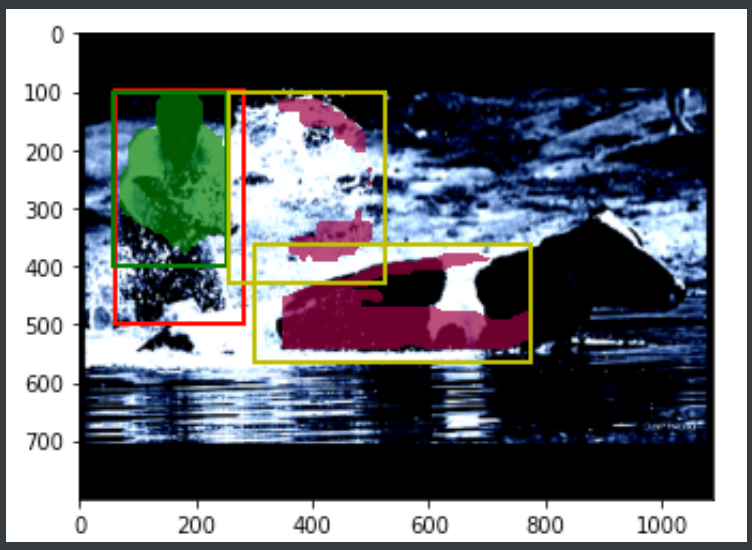


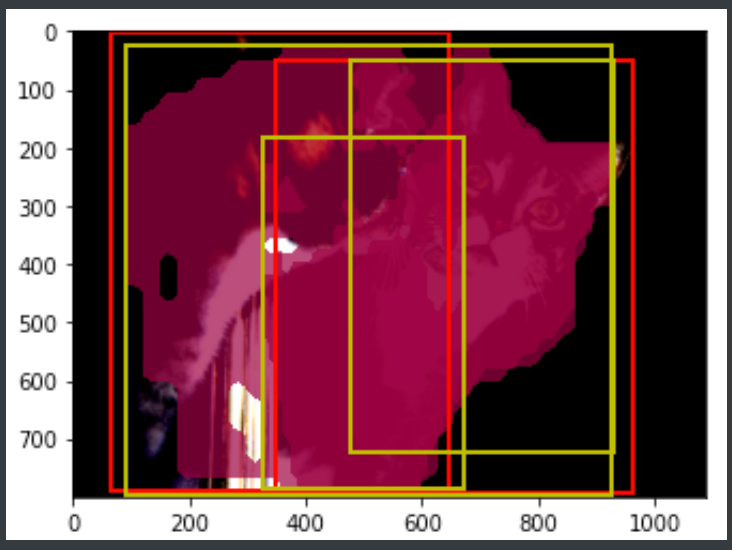
Multiple objective detection(No engineering involved):

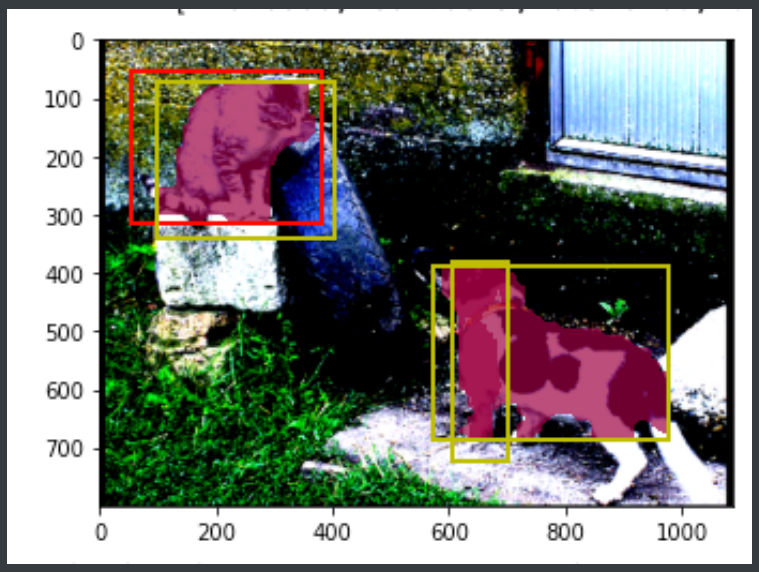
- No bias and manully inference, results comes from the first 21 images include multiple objects, and some generalization examples.
- I pick the first 3 most confident detection

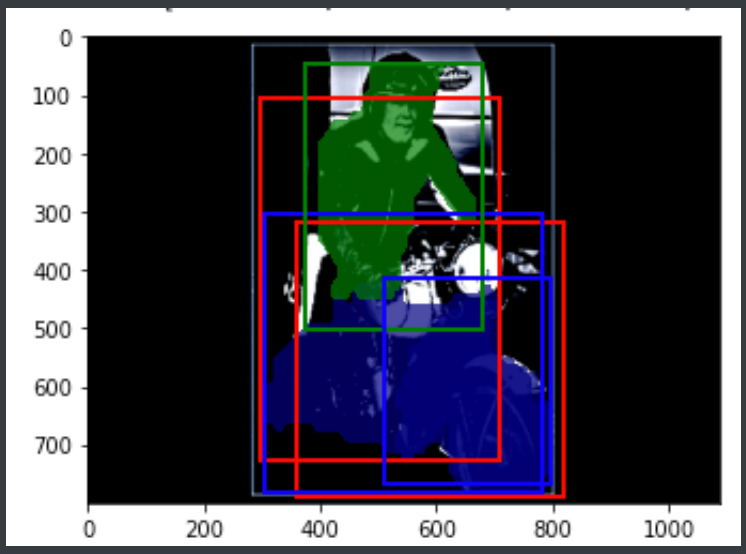


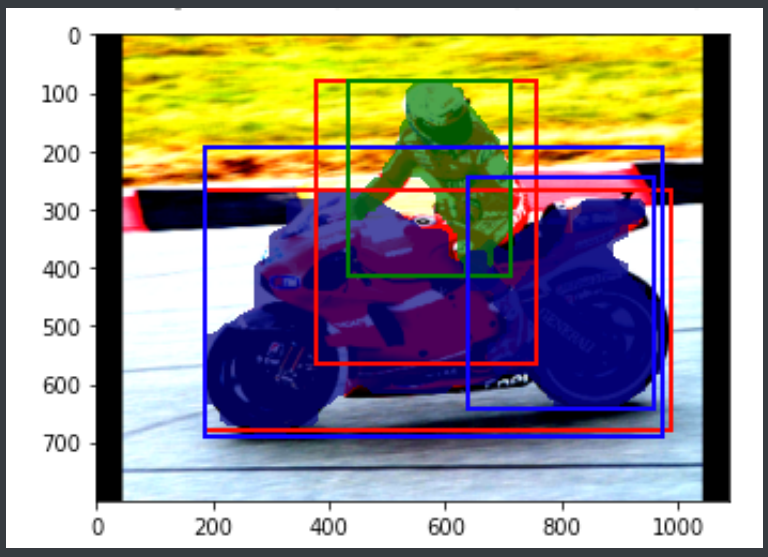

0
100
200
300
400
500
600
700
0
200
400
600
800
1000

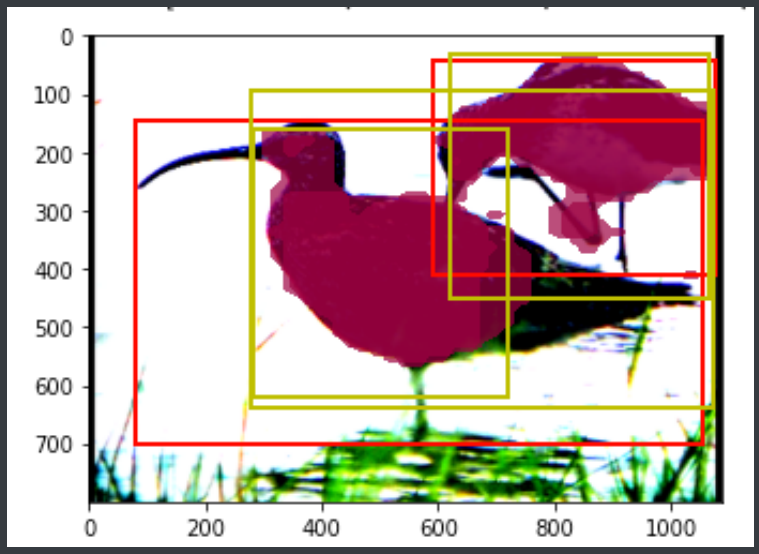

0
100
200
300
400
500
600
700
0
200
400
600
800
1000

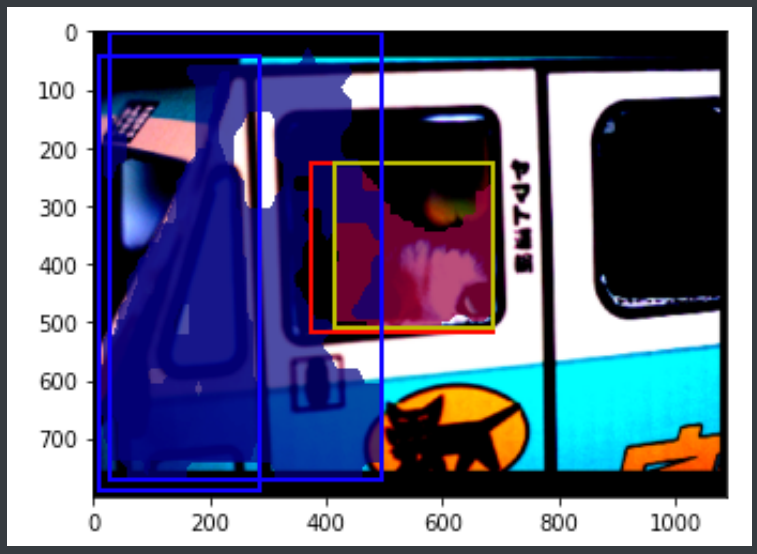

0
100
200
300
400
500
600
700
0
200
400
600
800
1000

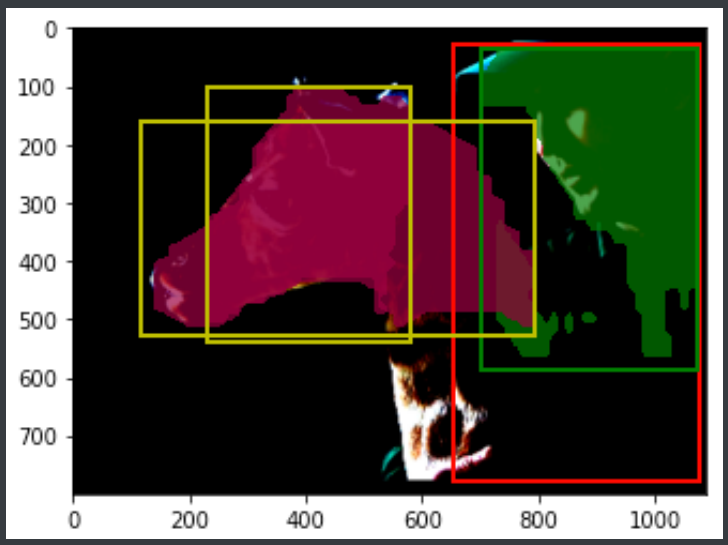

0
100
200
300
400
500
600
700
0
200
400
600
800
1000

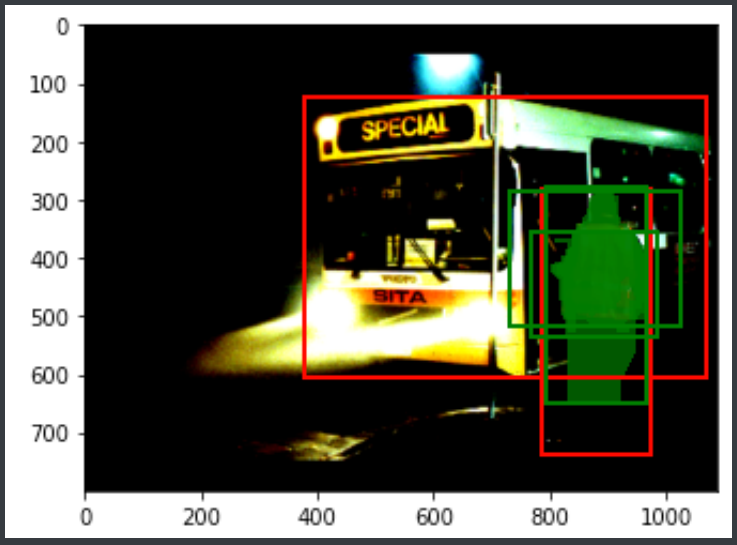

0
100
200
300
400
500
600
700
0
200
400
600
800
1000

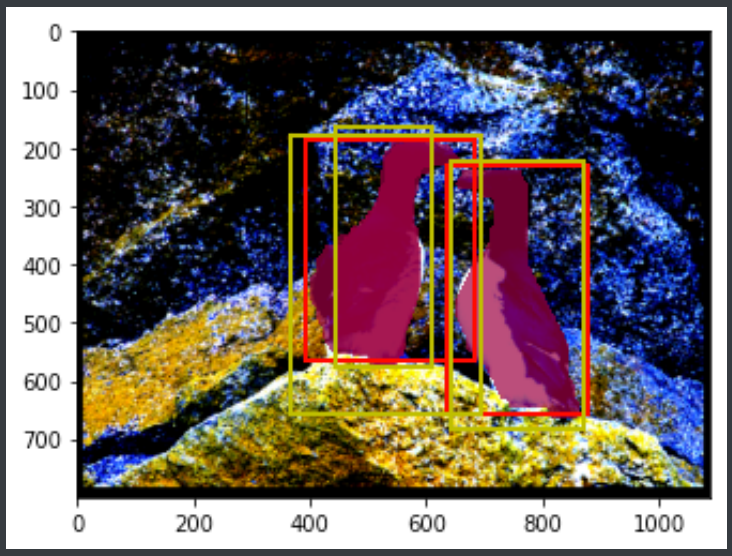

0
100
200
300
400
500
600
700
0
200
400
600
800
1000

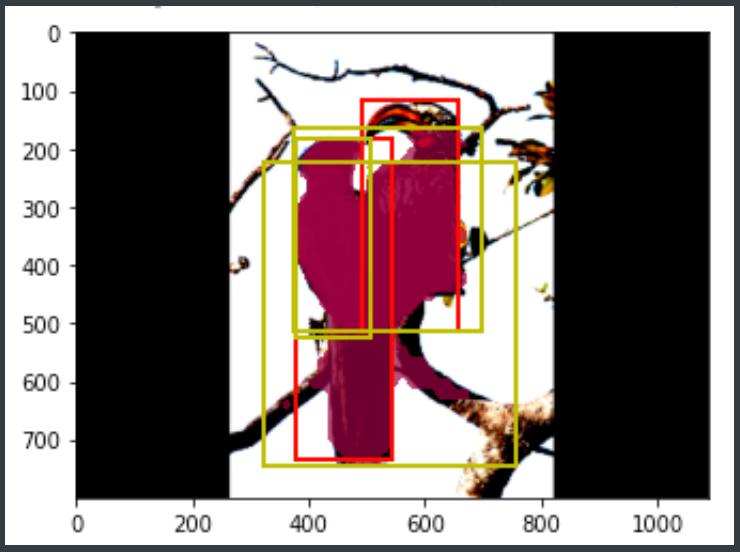

0
100
200
300
400
500
600
700
0
200
400
600
800
1000

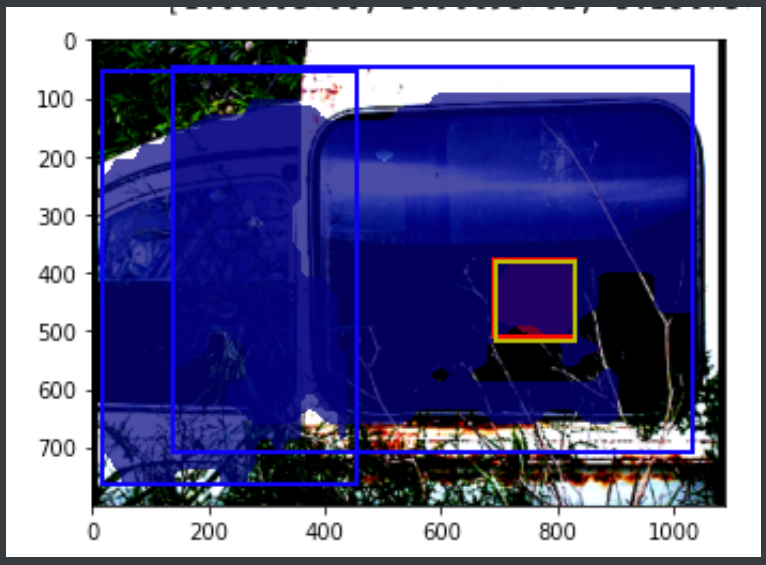

0
100
200
300
400
500
600
700
0
200
400
600
800
1000


0
100
200
300
400
500
600
700
0
200
400
600
800
1000

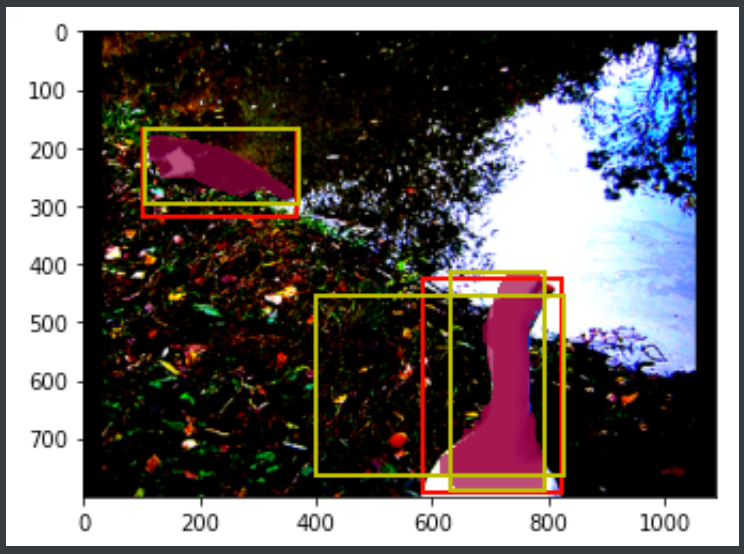

SPECIAL
SITA
0
100
200
300
400
500
600
700
0
200
400
600
800
1000

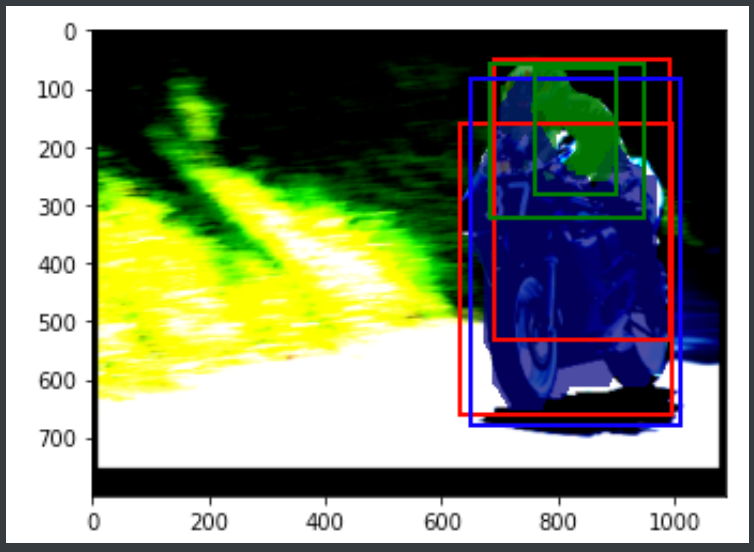

0
100
200
300
400
500
600
700
0
200
400
600
800
1000

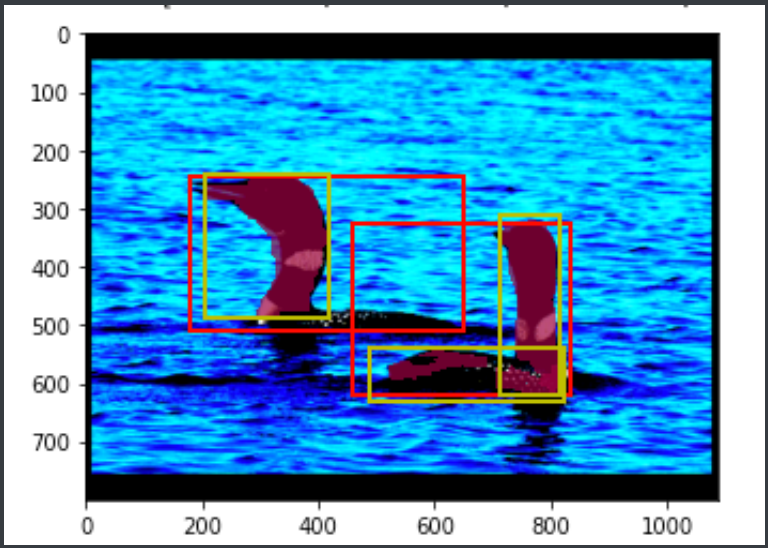

0
100
200
300
400
500
600
700
0
200
400
600
800
1000

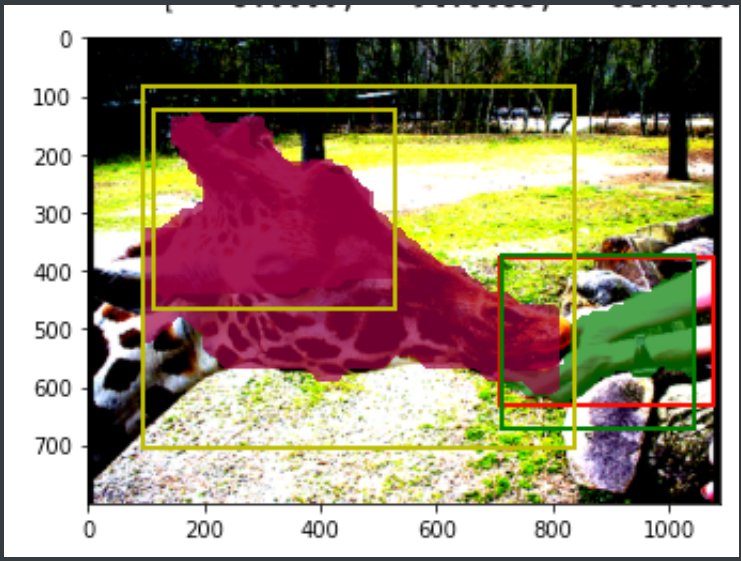


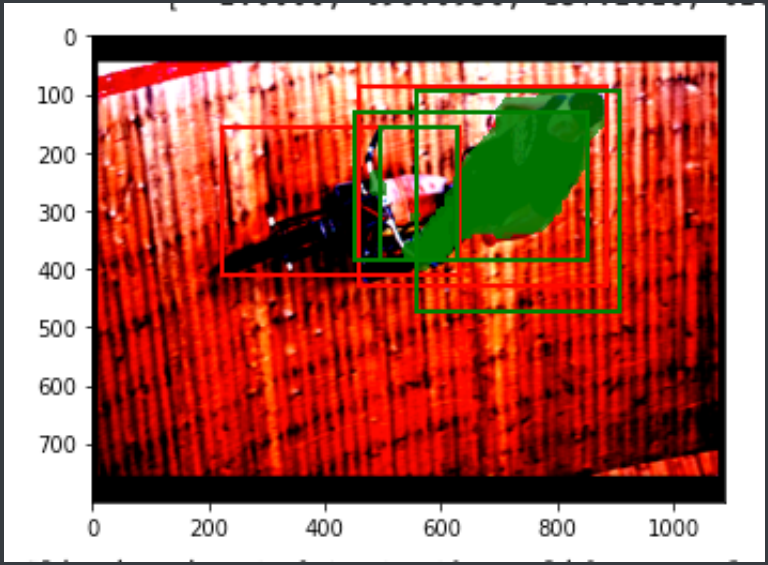
in the above image the red color is cover by blue, this is actually a great case of generalization



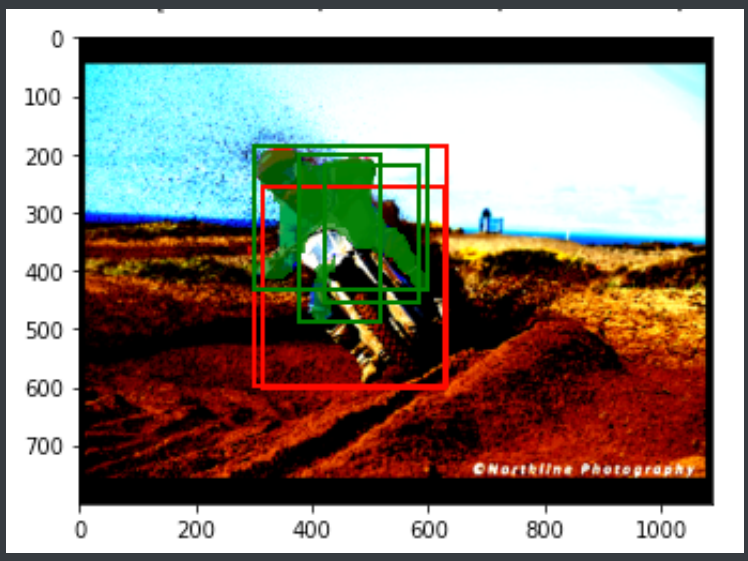


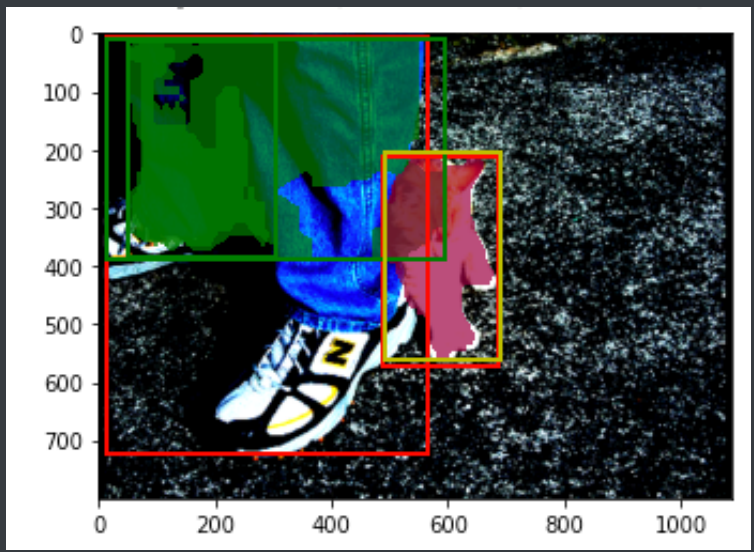

0
100
200
300
400
500
600
700
0
200
400
600
800
1000

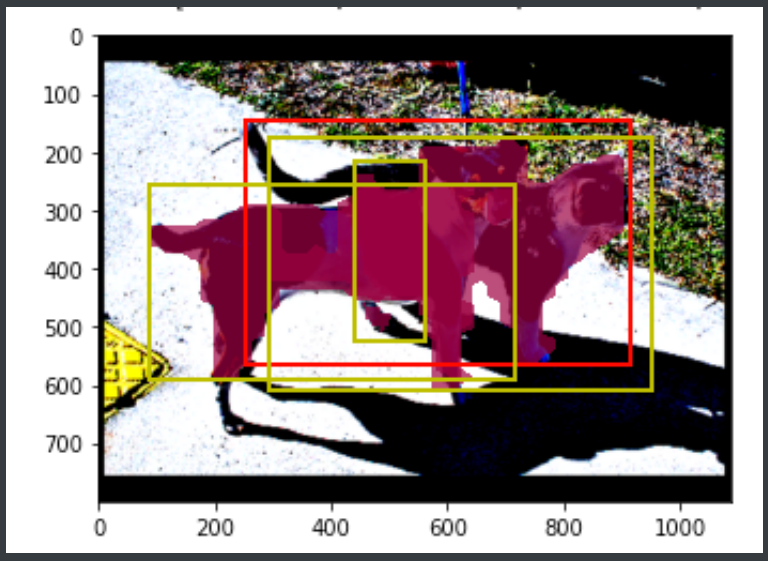

0
100
200
300
400
500
600
700
0
200
400
600
800
1000

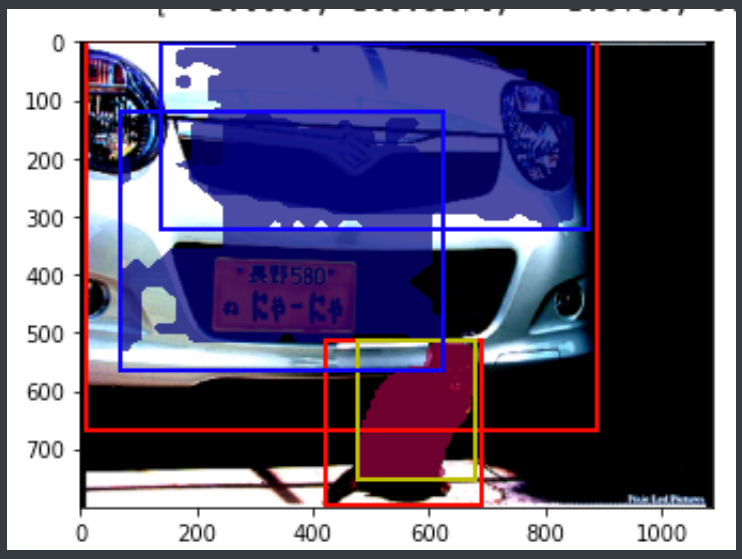

0
100
200
300
400
500
600
700
0
200
400
600
800
1000

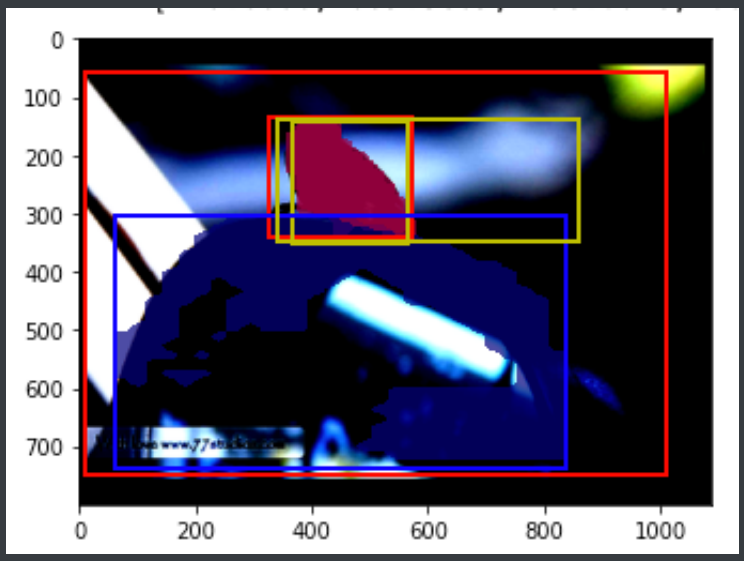

0
100
200
300
400
500
600
700
0
200
400
600
800
1000

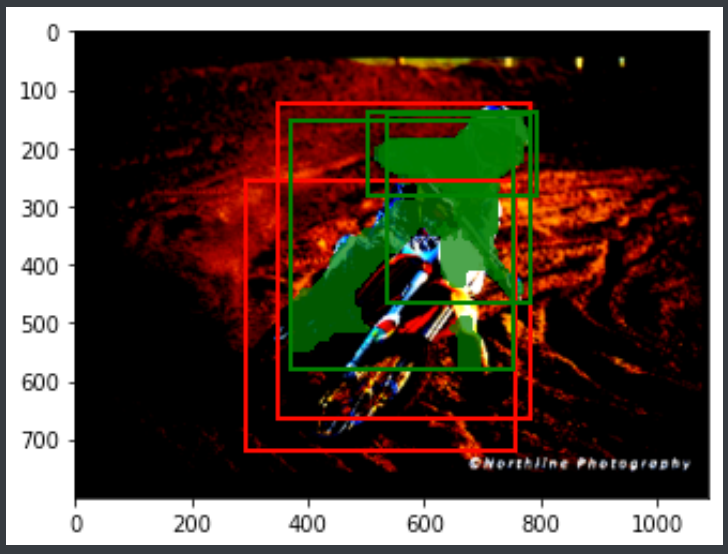

0
100
200
300
400
500
600
700
0
200
400
600
800
1000
©Northline Photography

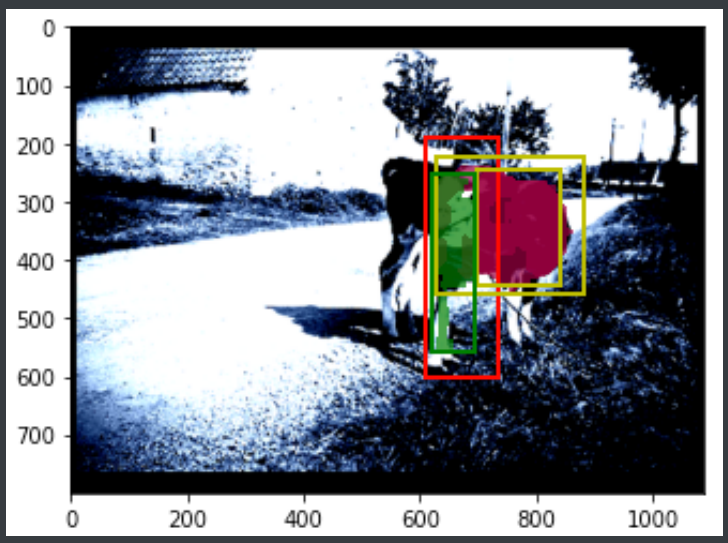

0
100
200
300
400
500
600
700
0
200
400
600
800
1000

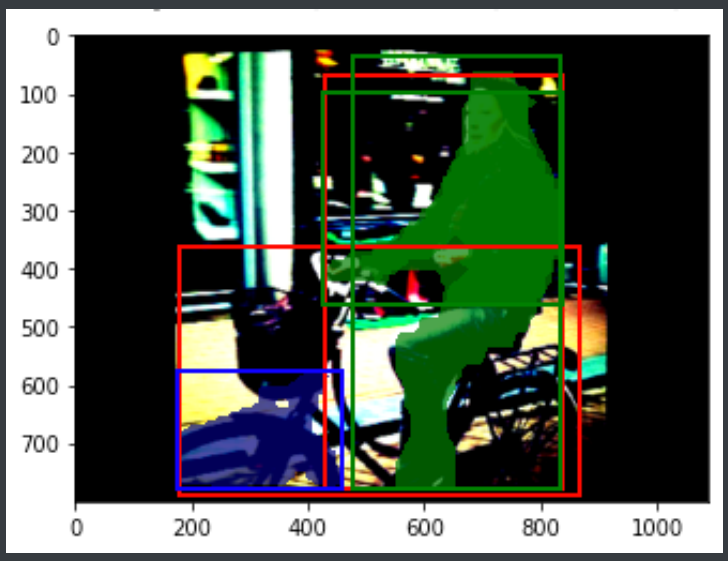

0
100
200
300
400
500
600
700
0
200
400
600
800
1000


0
100
200
300
400
500
600
700
0
200
400
600
800
1000


0
100
200
300
400
500
600
700
0
200
400
600
800
1000



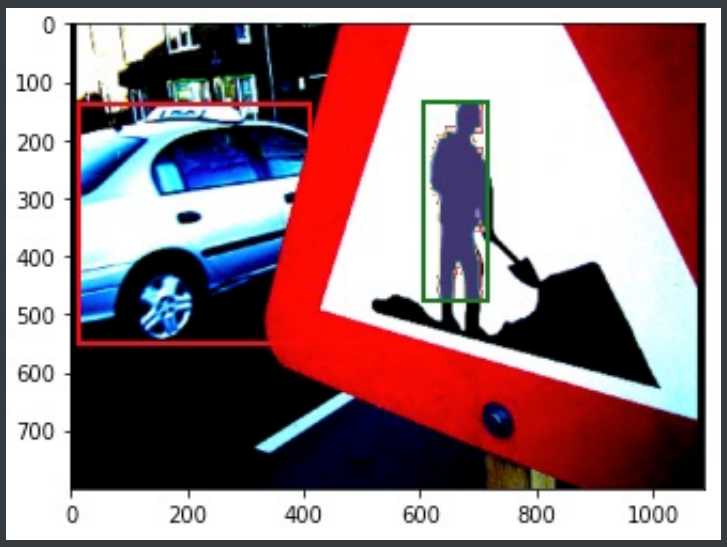


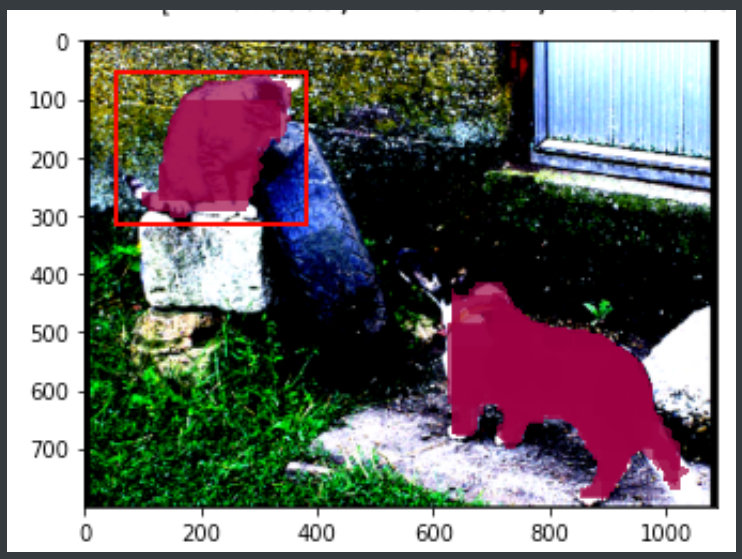


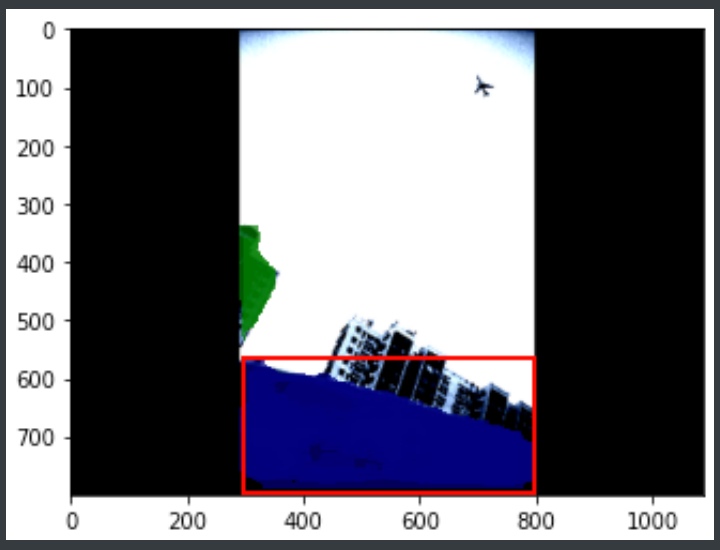
generalization result:

Some objects are not initially labeled in the image could been found by the network

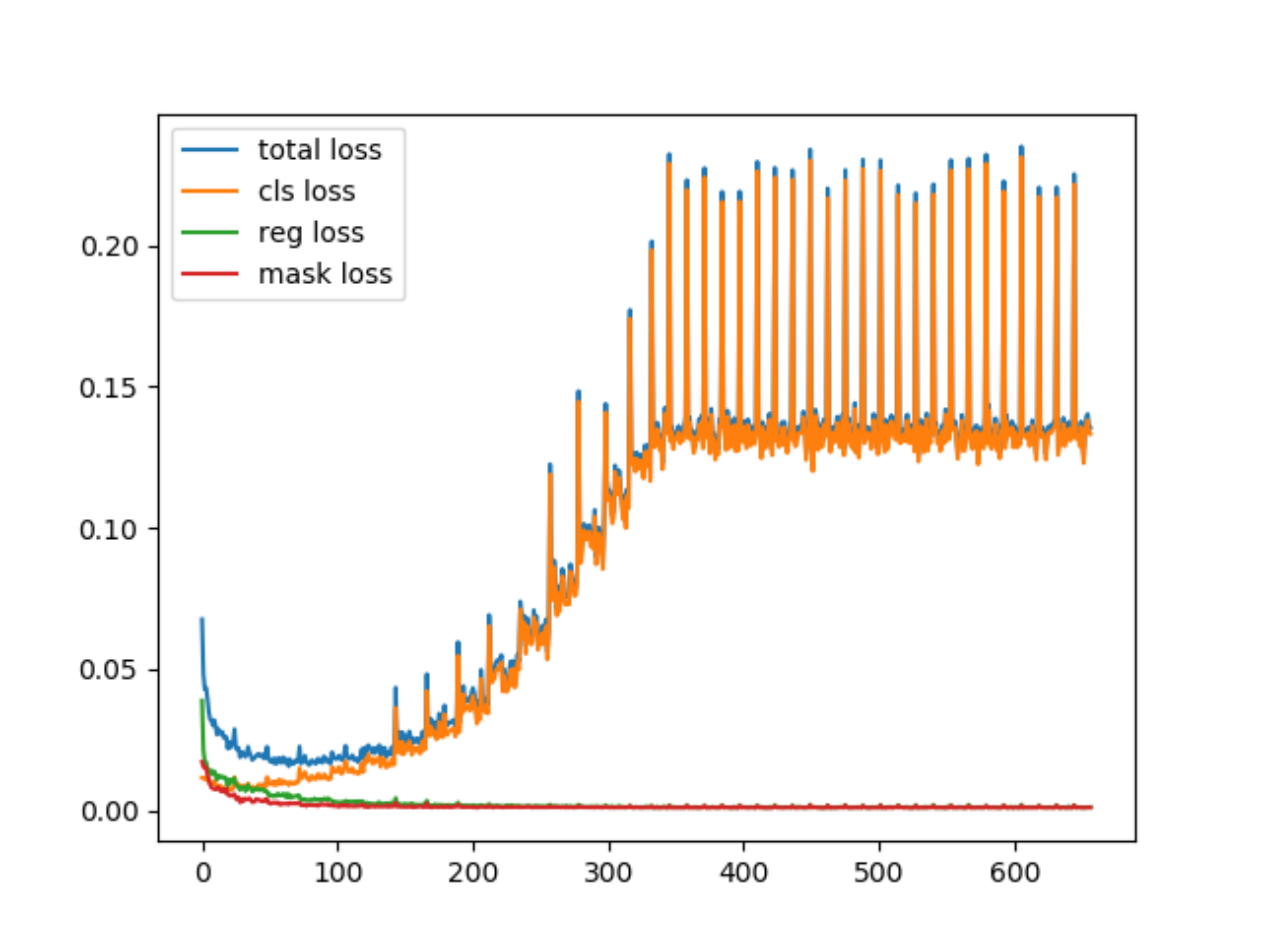
Here are some cases: first example comes from a different setup(find it interesting, so put it here)







Objective function values convergence trajectory:



This is beautiful!

Basically, two-stage training phases.

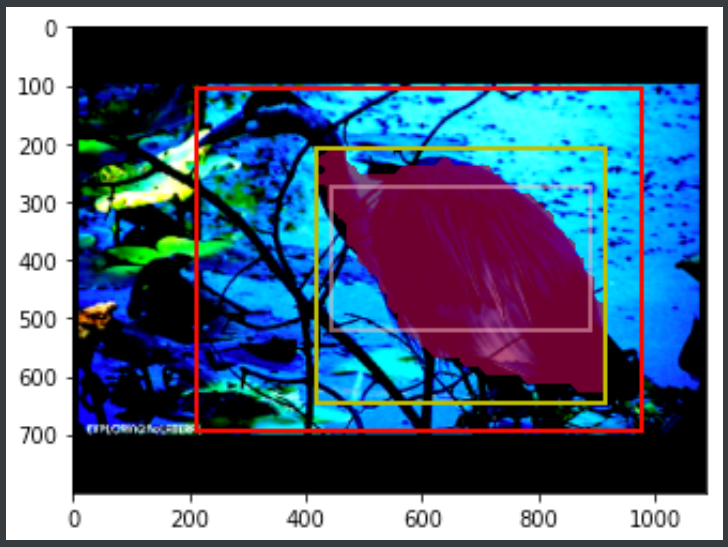
At the beginning, focus more on the regression, and take care of cls and mask also.

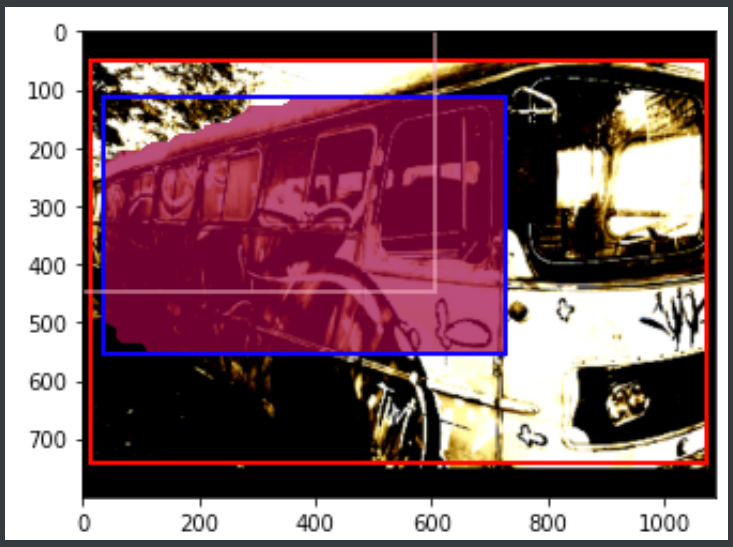
As the training goes on, once we have the reasonalable reg&mask part of the network, focus on cls which is directly relate to the confidence criterion. The purpose is to identify the right proposal in the inference phase.

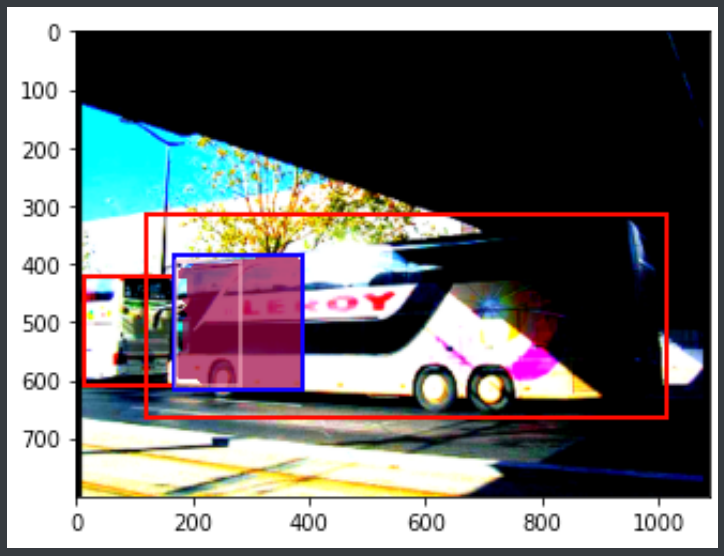
Discussion

Refinement show case (regression part of the network):

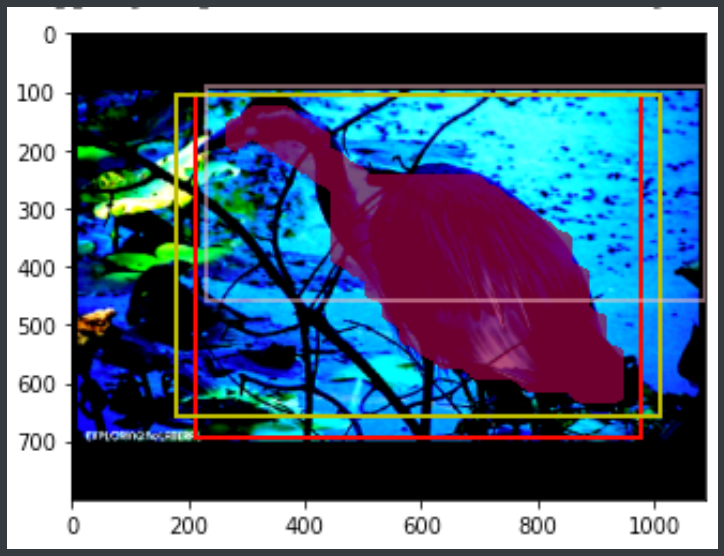
- light pink represent the proposal.
- if the original proposal is too small, the network is not capable to refine it into a big detect. Some examples as follows where the proposals with refinement could not find the detection well:

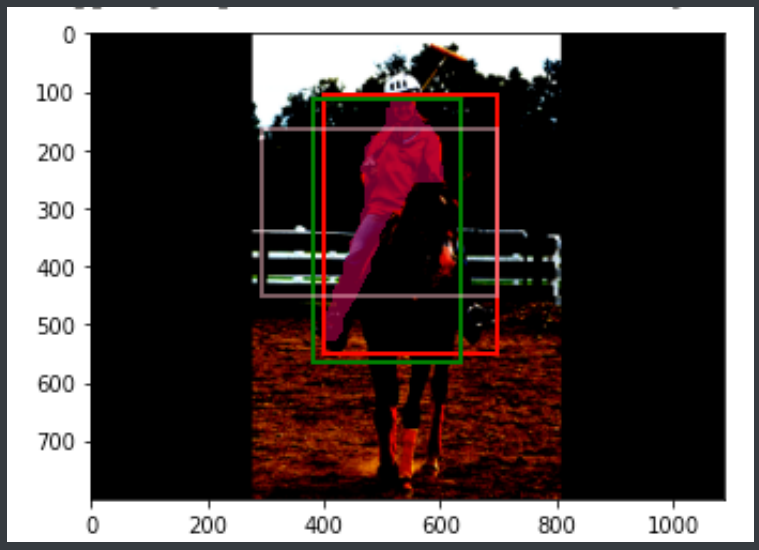


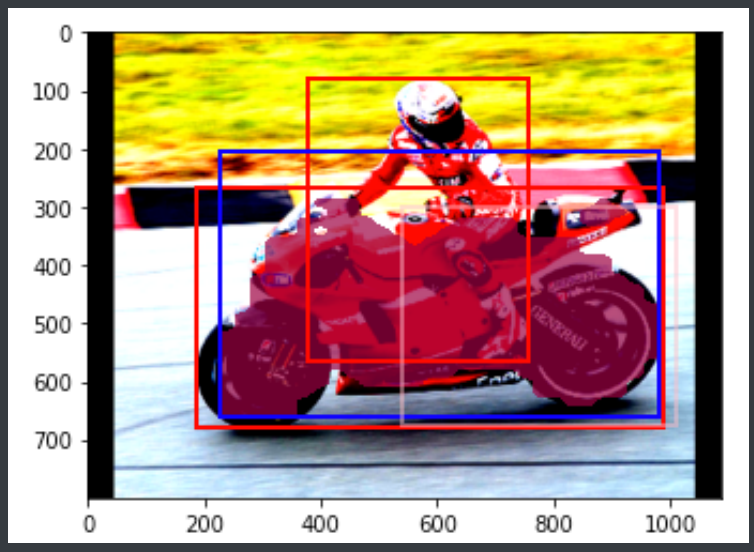




If the proposal the network assigns to is big enough, then the refinement could sucessfully refine the proposal to find a good detection:







So, the size of the proposal is very critical, the regression part of the network could do a good job given the right-sized proposal.

implement details

Setup

- Weights: $\lambda_{cls} = 1/5$; $\lambda_{reg} = 1$; $\lambda_{mask} = 1/250$

Hacking tools if needed

Special enginner in the Single object detection:

- select first 10 detections based on objectness(cls scores).
- feedwards to mask branch and get mask matrix.
- add up mask matrix for each detection, identify the max mask area as $mask_{max}$.
- among detections which have at least 90% area of $mask_{max}$, choose the one with the smallest detection area.

Mask fusion

- select first 10 detections based on objectness(cls scores) and their corresponding masks.
- conbine all these mask and represent in the original image.

Parameter tunning

- weight of λ_{cls}, λ_{reg}, λ_{mask}.

  these weights control the learning rate between different tasks.

- bg_threshold (0.5): background threshold.

  This parameter controls how strict we want the proposals in the training phase to be. Higher this value means that

  note here: lower this parameter could speed the training, basically less training sample, we could increase this value along the training.

- optimizer learning rate

  An easy adaptive setup:

  $lr = 0.0001 * \frac{1}{2^{epoch}}$

Potential problems

confidence criterion choosing:

- used in sorting the detection results, only incode the cls information.

Disconsistency in training

- In the RPN, used the all layers from FPNs to propose the close objects.
- But in the following training, only used one layer from FPN.

Potential option to improve the performance

- code the regression information into confidence criterion.
- adaptively changing the bg_threshold throughout the whole network training: low --> high.
- fusion the masks: how to combine the first few predictions. One mask should be enough for one object if the network is truly working!
- Use all layers of FPNs in the detection and mask training, I am reading the FPN paper now...
- tunning around the weights maybe...