size

HW2-18-794: Introduction to Deep Learning and Pattern Recognition for Computer Vision

Assignment 2: Face Detection

Instructor: Marios Savvides

Due Date: Oct 19, 2023

Total Points: 100

Submission: Submit your solutions and plots on Gradescope.

START HERE: Instructions

- **Collaboration policy:** All are encouraged to work together BUT you must do your own work (code and write up). If you work with someone, please include their name in your write-up and cite any code that has been discussed. If we find highly identical write-ups or code or lack of proper accreditation of collaborators, we will take action according to strict university policies. See the Academic Integrity Section detailed in the initial lecture for more information.
- **Late Submission Policy:** There are a **total of 5** late days across all homework submissions. Submissions more than 5 days after the deadline will receive a 0.

  For those students taking Mini-1 (semi-semester), you cannot submit it later than Oct 21, 2023, as the grading deadline of mini-1 is due on Oct 23, 2023.
- **Submitting your work:**
  - We will be using Gradescope (`https://gradescope.com/`) to submit the Problem Sets. Please use the provided template only. Submissions must be written in LaTeX. All submissions not adhering to the template will not be graded and receive a zero.
  - **Deliverables:** Please submit all the `.py` files. Add all relevant plots and text answers in the boxes provided in this file. TO include plots you can simply modify the already provided latex code. Submit the compiled `.pdf` report as well.

NOTE: Partial points will be given for implementing parts of the homework even if you don't get the mentioned numbers as long as you include partial results in this pdf.

Overview

In this assignment, you will train and test a state-of-the-art face detector that is deployed in the real world applications.

1 Understand your data (POINTS : 30)

Understanding your data is always the first step in training deep learning models, which will greatly ease the following works. This step usually involves statistical analysis, preprocessing, and visualization for detection datasets.

Data path:

https://drive.google.com/file/d/1799456S54_M7CS9Gom1xrD9_tIKavPuB/view?usp=sharing

Note 1. The bounding box is indicated by the first 4 floats and is in (x1, y1, w, h). Where x1, y1 is the top-left corner, followed by the 5 landmarks points (L_x, L_y), each landmark point is followed by a 0/1 that you can ignore. The landmarks might be -1 indicating no landmark available for this face.

1.1 Write code to do the following statistical analysis

1. Total number of ground truths (GT), i.e., faces, for training and validation.

2. Average area, width, and height of bounding boxes

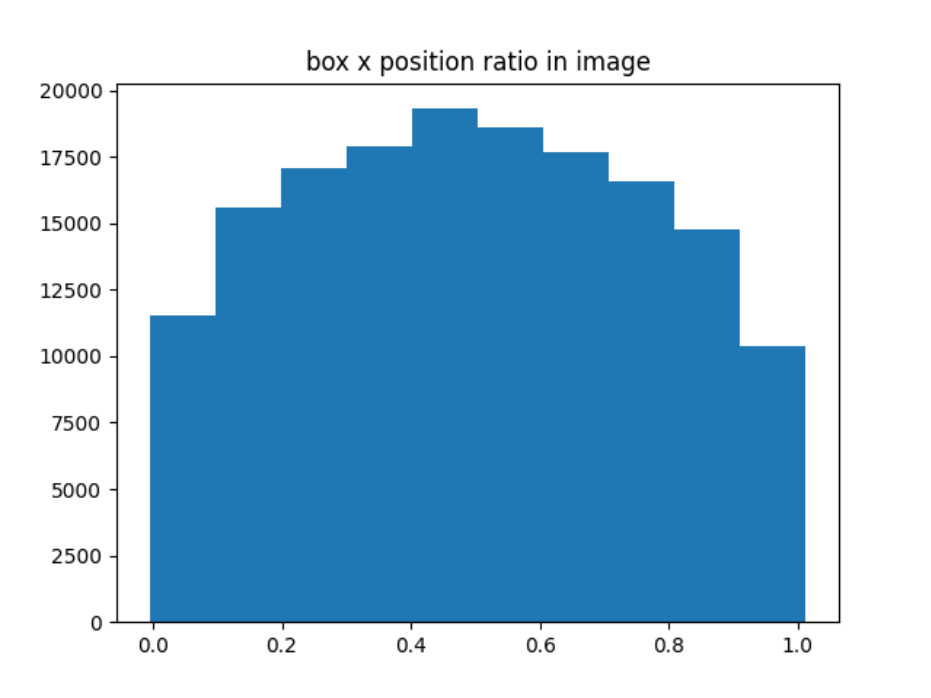
3. Are faces uniformly distributed on images? Give your analysis.

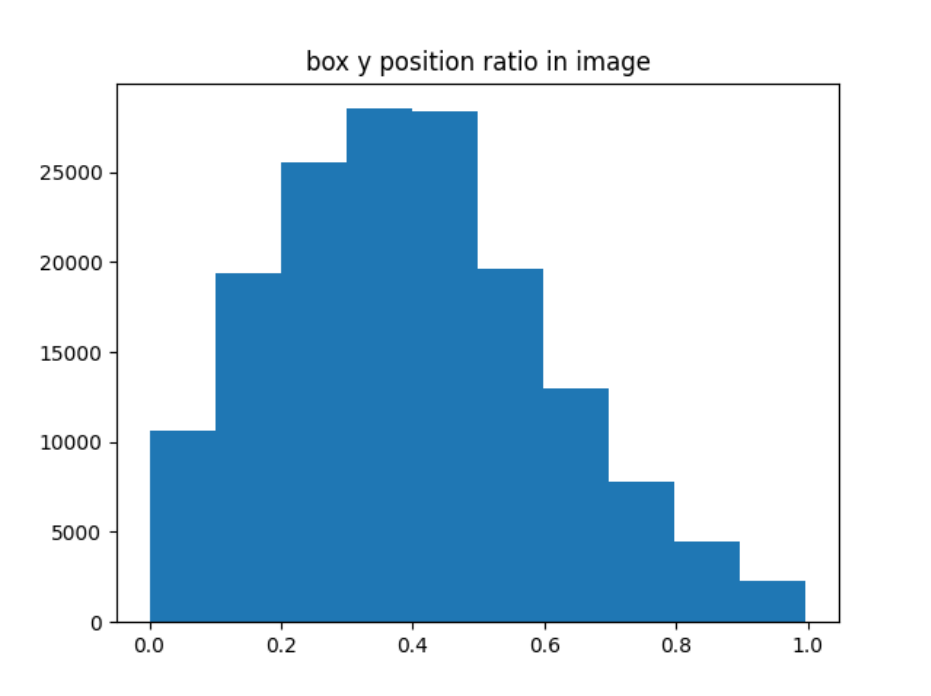
4. Check if any GT exceeds the range of the image. If yes, how do you plan to solve it?

5. How serious is the mutual overlap among GTs? Calculate the percentage of GTs that overlap with one another (i.e., overlapped GT vs total number of GT). Hint: check the overlap using IOU > 0.

Solution

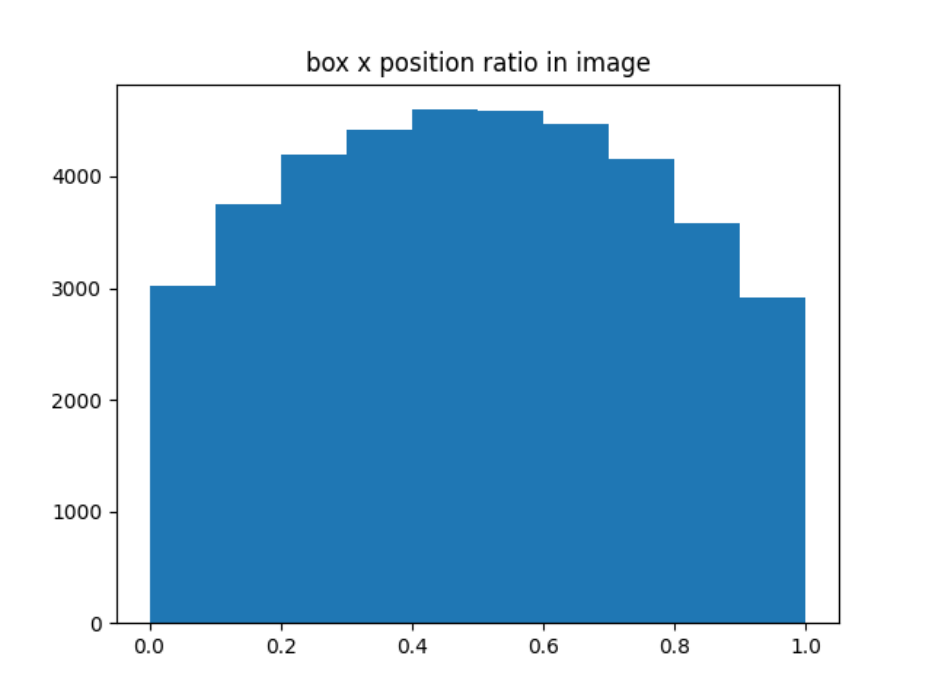
Enter your solution here
For training data:
1. GT count: 159422
2. ave area: 3851.279020461417
ave w: 28.948482643549823
ave h: 37.398778085835076
3. No, average center value:
img center: 512.0 384.00535678442395
box center: 508.95301209353676 304.4233459203131
Also, with box center x and y location in plot:

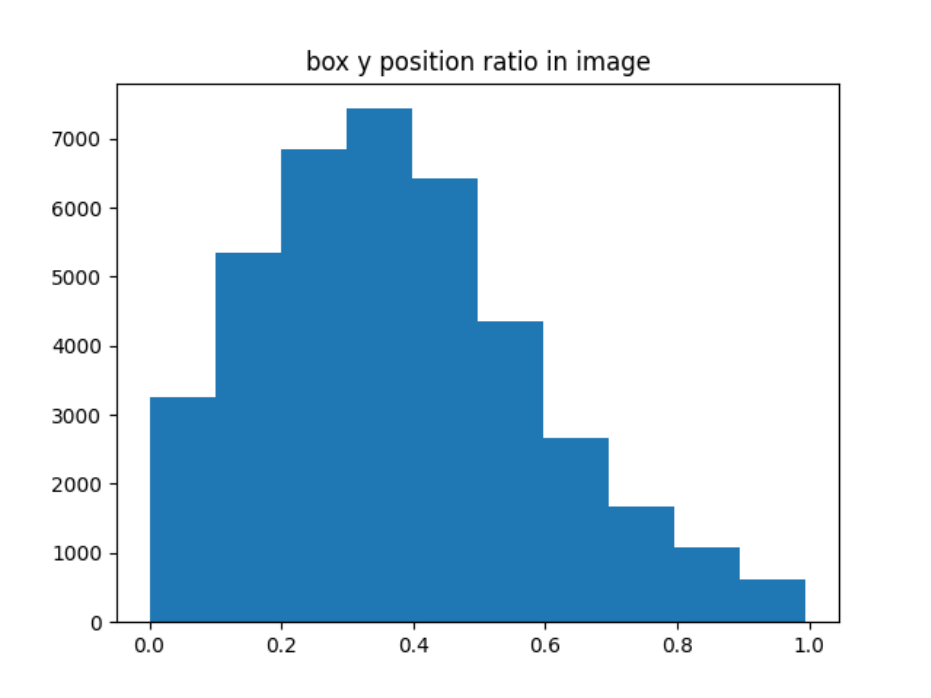




We can see the values aren't the same and uniformly distributed.
4. No exceeding:
exceed h: 0
exceed w: 0
5. percentage of GTs:

Overlap: 0.12396657926760422
For val data:
1. GT count: 39707
2. ave area: 3834.9729266879895
ave w: 29.191981262749643
ave h: 37.60183342987383
3. No, average center value:
img center: 512.0 383.90033494509925
box center: 509.62418152513345 288.8188400322353
Also, with box center x and y location in plot:





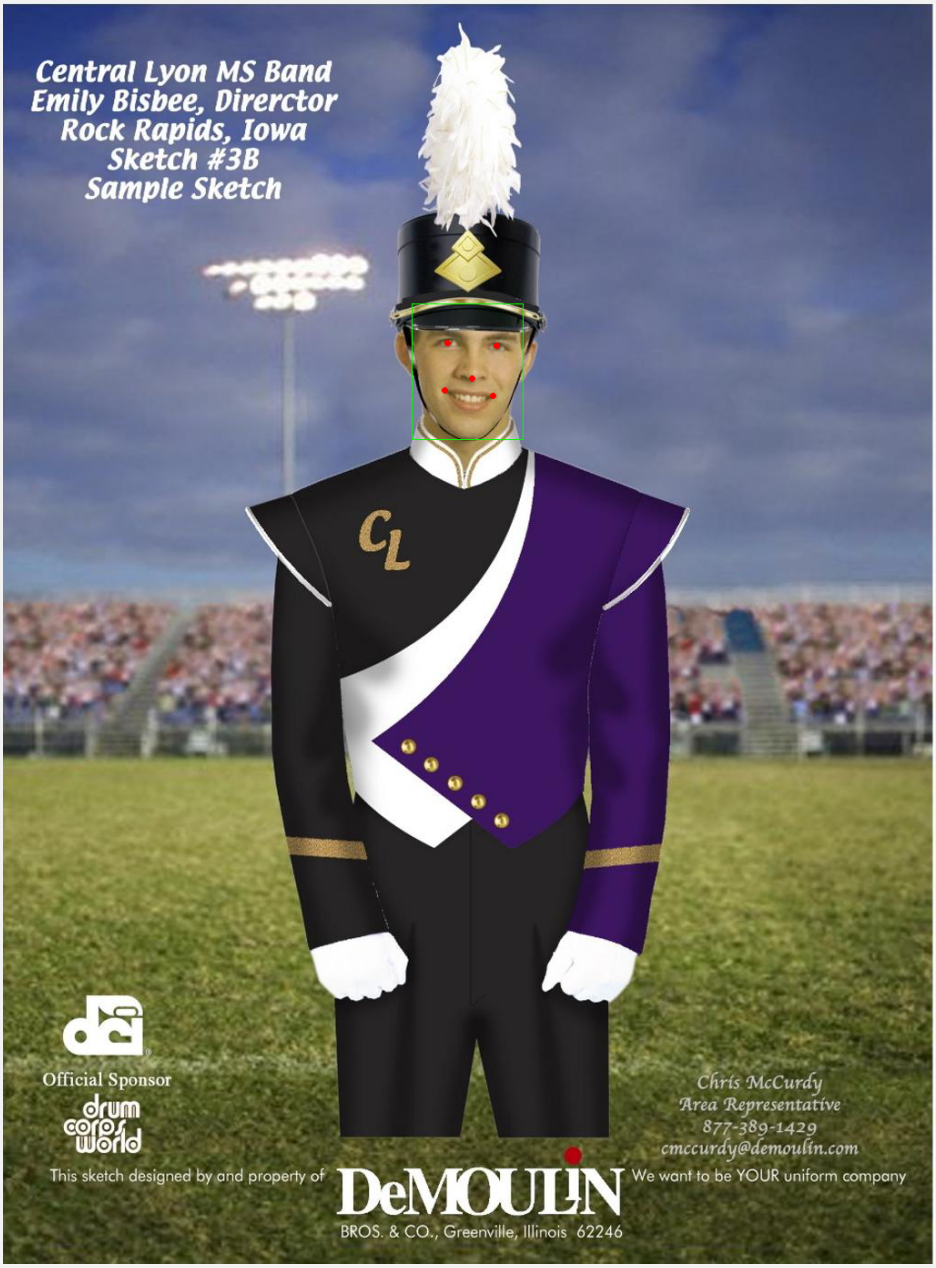
We can see the values aren't the same and uniformly distributed.
4. No exceeding:
exceed h: 0
exceed w: 0
5. percentage of GTs:
Overlap: 0.12118769990178055

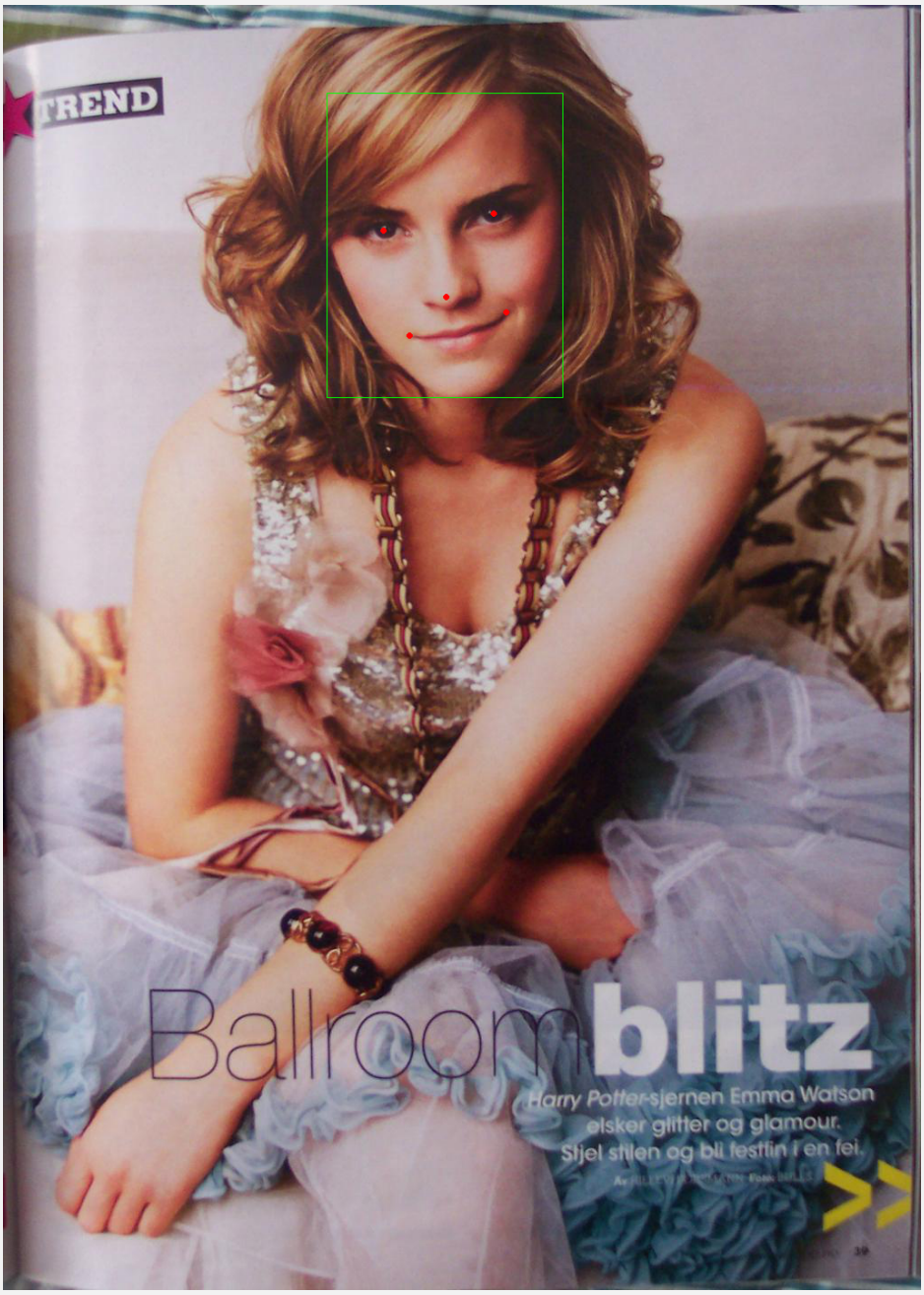
1.2 Visualize the first 4 images and their ground truths, including the faces (in green) and landmarks (in red).

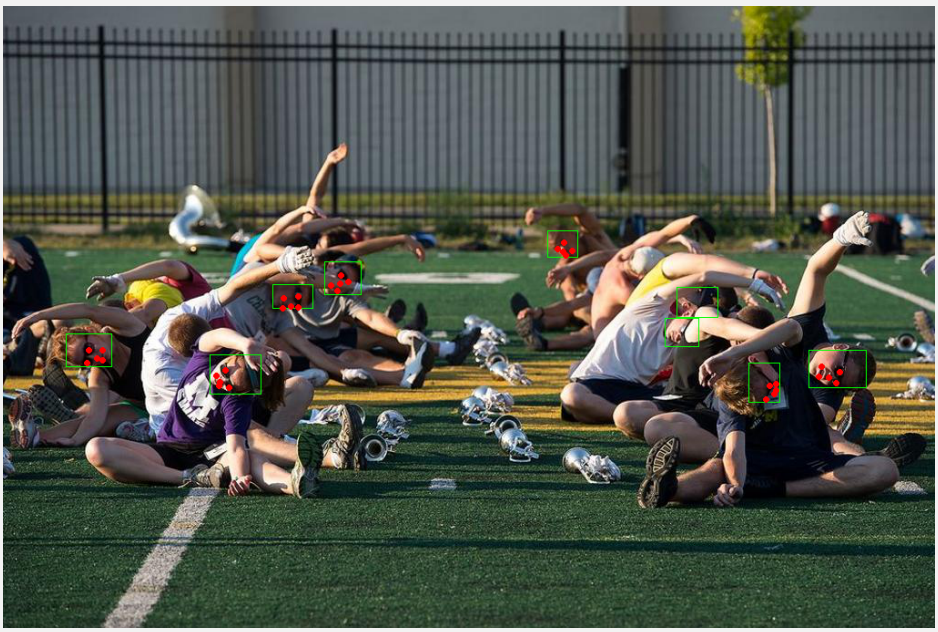
Solution

Enter your solution here


Central Lyon MS Band
Emily Bisbee, Direrctor
Rock Rapids, Iowa
Sketch #3B
Sample Sketch
Official Sponsor
drum corps world
This sketch designed by and property of
DeMOULIN
BROS. & CO., Greenville, Illinois 62246
We want to be YOUR uniform company
Chris McCurdy
Area Representative
877-389-1429
cmccurdy@demoulin.com
TREND
Ballroom blitz
Harry Potter-stjernen Emma Watson
elsker glitter og glamour.
Sjekk stilen og bli festlig i en fei.


Central Lyon MS Band
Emily Bisbee, Direrctor
Rock Rapids, Iowa
Sketch #3B
Sample Sketch
CL
Official Sponsor
drum corps world
This sketch designed by and property of
DeMOULIN
BROS. & CO., Greenville, Illinois 62246
We want to be YOUR uniform company
Chris McCurdy
Area Representative
877-389-1429
cmccurdy@demoulin.com


TREND
Ballroom blitz
Harry Potter-stjernen Emma Watson
elsker glitter og glamour.
Stjel stilen og bli festfin i en fei.



2 Initialize your detector and build up the forward path (./detector/mydetector.py).

This face detector has 4 components: 1. ResNet 50 as Backbone. 2. Feature pyramid network (FPN). 3. A special module named SSH. 4. Three heads for classification, bounding box regression, and landmarks, respectively.

Note 1: Since the FPN needs a feature pyramid extracted from the backbone, you will need to use

`torchvision.models._utils.IntermediateLayerGetter`

to gather these intermediate layers from backbone.

Note 2: The classification head aims to classify each anchor as positive or negative, because we are doing face/not face here. Regression aims to regress a bounding box for each anchor, it requires 4 degrees of freedom to do so. Landmark head aims to regress 5 landmark points, it requires 10 degrees of freedom to do so.

2.1 Initialize backbone and gather intermediate layers, the layers to-be-gather are defined in cfg['return layers']

Note: cfg is "configuration" that is specified in config.py

Solution

backbone = models.resnet50(pretrained=cfg['pretrain'])
self.body = _utils.IntermediateLayerGetter(backbone,return_layers=cfg['return_layers'])

2.2 Understand how we make the classification head. What is the output dimension of the head? And, for the binary classification, is sigmoid mathematically equivalent to softmax? Show your proof.

Solution

1. (batch_size, height * width * self.num_anchors, 2)
2. Yes, for sigmoid function, it's sig(x) = 1 / (1 + exp(-x))
For Softmax Function,it's soft(x) = exp(x)/sum(exp(xi))
For binary classification, there are only two type: y==1 or y==0, so it can be written as
P(y = 1) = exp(x) / (exp(0) + exp(x)) = exp(x) / (1 + exp(x)) = 1 / (exp(-x) + 1), which is same as sigmoid function.
Also, P(y = 0) =1 / (1 + exp(x)) = exp(-x) / (1 + exp(-x)) = 1- sig(x)

2.3 Following the code of the classification head, finish the regression head and landmarks head.

Solution

```
class BboxHead(nn.Module):
    def __init__(self, inchannels=512, num_anchors=3):
        super(BboxHead,self).__init__()
        # TODO
        self.conv1x1 = nn.Conv2d(inchannels, num_anchors * 4,
         kernel_size=(1,1),stride=1,padding=0)

class LandmarkHead(nn.Module):
    def __init__(self, inchannels=512, num_anchors=3):
        super(LandmarkHead, self).__init__()
        # TODO
        self.conv1x1 = nn.Conv2d(inchannels, num_anchors * 10,
         kernel_size=(1,1),stride=1,padding=0)
```

2.4 Complete the forward function of your detection. The process is Input image $->$ gather feature maps $->$ FPN $->$you got 3 feature pyramids $->$ for each feature pyramid, apply a predefined SSH module to it $->$for each feature pyramid, apply three heads to it.

Solution

```
# FPN
fpn = self.fpn(out)

# SSH
feature1 = self.ssh1(fpn[0])
feature2 = self.ssh2(fpn[1])
feature3 = self.ssh3(fpn[2])
features = [feature1, feature2, feature3]
```

3 Dataloader (data/dataloader.py)

Your dataloader feeds images, bounding boxes, and landmarks to the detector for training.

3.1 initialize dataloader

```
In __init__, load the widerface/train/label.txt,
and store the path of images in self.imgs_path as a list.
Store the annotations in self.words. The structure must be
self.imgs_path: [path1, path2, path3]
self.words: [[img1_anno1, img1_anno2, ...]
             [img2_anno1, img2_anno2, img2_anno3, ...]
             [img3_anno1, ...]
                 ...]
Make sure you are storing numbers in float.
```

3.2 getitem

Getitem is a commonly seen function in pytorch based dataloader. It aims to get/organize/process the training data from the pool prepared for each iteration.

Read carefully:

```
In __getitem__, complete the annotation reorganization. Each "annotation"
should be in the following format:
np.array(x1, y1, x2, y2, L0_x, L0_y, L1_x, L1_y, ..., L4_x, L4_y, M)
Where (x2, y2) is the bottom-right corner of the bounding box,
L stands for the landmark points.
M is an indicator showing whether landmark points are
available,
Yes for 1 and No for -1. The "annotation" is in shape (1, 15)
```

4 Anchor(anchor.py)

Anchor has been an important concept for detection since 2015 (Faster RCNN).

We define ”Anchor”, in this homework, as ”all reference boxes located in every feature point”

Understand the code, and answer the following

4.1 Name all hyperparameters that correspond to the number of anchors. I.e., which hyperparameters in config will affect the number of anchors? Briefly describe their physical meanings in your words

Solution

min_sizes: lower and upper bounds for minimum sizes for the anchors of image, the size(number of items) of this effects number of anchors.
steps: step size of placing anchors in feature map. A smaller step size results in more anchors.
image_size: The size of the input image, used to calculate the spatial scale of the anchors. With image_size increasing, the number of anchors also increases.

4.2 If we enlarge the width and height of the image by x, how would the number of anchors change?

Solution

If enlarging the width and height of the image by x, the number of anchors will increase. With fixed step size, the anchor with be more with larger scale of image. Also, the number of anchors will become

$$x^2$$

4.3 For anchors on different feature pyramids, are they the same size? Explain your observations and explain why it is designed in such a way.

Solution

No, anchors at different scales are different.
Larger anchors works on larger and higher-resolution feature pyramids especially for smaller objects on it, and smaller anchors works on smaller and lower-resolution feature pyramids especially for larger objects on it. Size is different for different feature pyramids.

5 Calculate the Loss

Anchor plays a key role in anchor-based object detection for deciding which feature point on the feature pyramid should be responsible for learning a certain ground-truth. Let's do a quick review:

On your multi-level feature pyramid, each pixel is a feature point. Each feature point is mapped to one or multiple anchors depending on the detector's setting. We use an anchor to compare with ground-truth via “jaccard index”, i.e., Intersection over Union. If the IoU is larger than a threshold (0.2), then the feature point corresponds to that anchor would be assign to learn the ground-truth, the learning includes positive/negative classification, bounding box regression, and landmark. If an anchor matches zero ground-truth, the corresponding feature point needs to be classified as negative.

5.1 In utils/boxutils.py, complete jaccard function.

5.2 Explain the following code in your words:

In “match”, what is the following code used for?

```
best_prior_overlap, best_prior_idx = overlaps.max(1, keepdim=True)
```

Solution

this is used to find the best matching prior box index and value for each ground truth box based on Jaccard(IOU) result.

In "match", what is the following code used for?

```
valid_gt_idx = best_prior_overlap[:, 0] >= 0.2
```

Solution

Used as filter to select the one with IOU value larger than 0.2 and ignore others. It checks best prior box for each ground truth box.

In "match", what is the following code used for?

```
best_truth_overlap, best_truth_idx = overlaps.max(0, keepdim=True)
```

Solution

this is used to find the best matching ground truth box index and value for each prior box based on Jaccard(IOU) result.

6 NMS

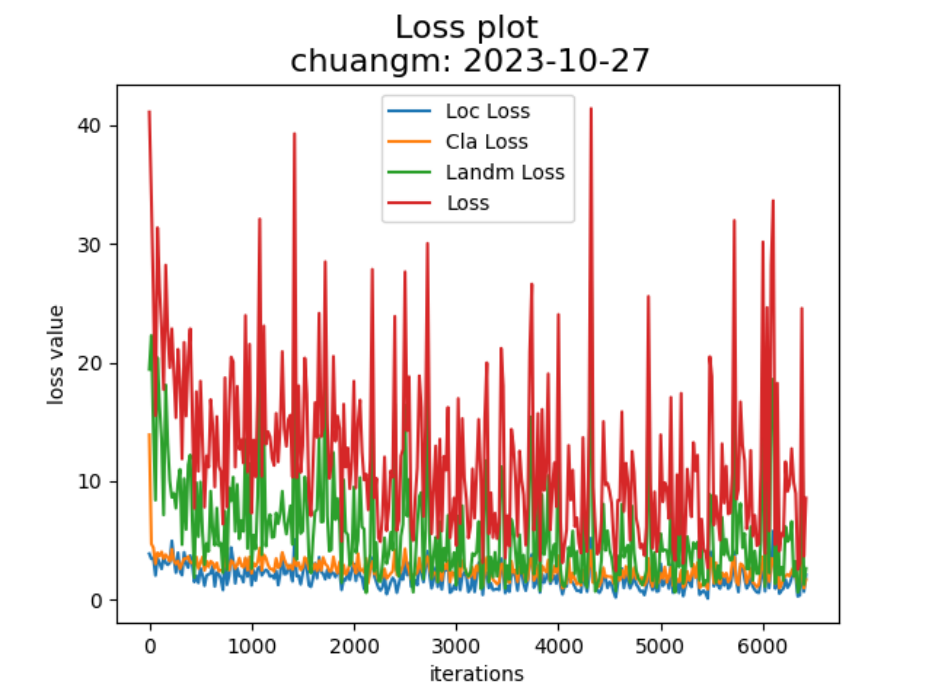
Non-maximum suppression is a must-ask question for those interviews seeking computer vision engineer positions. Prior to 2020, it is an essential component for object detection and many other tasks. Write function of NMS in nms.py

```
input: dets: ndarray (nx5), where n is number of predictions,
and each prediction is in (x1, y1, x2, u2, confidence)
        thres: float, which is IOU threshold.
output: list, the index of the predictions in dets, which passed NMS.
```

7 Training

Train your detector with 1 epoch, and plot each loss curve, the plotting step is per 20 iterations.

Solution




```
Epoch:1/1 || Epochiter: 1/6440 || Iter: 1/6440 || Loc: 3.8900 Cla: 13.9186 Landm: 19.4262 || LR: 0.00100000 || Batchtime: 15.3221 s || ETA: 1 day, 3:24:34
Epoch:1/1 || Epochiter: 301/6440 || Iter: 301/6440 || Loc: 2.3814 Cla: 2.6253 Landm: 11.0061 || LR: 0.00100000 || Batchtime: 0.3213 s || ETA: 0:32:52
Epoch:1/1 || Epochiter: 601/6440 || Iter: 601/6440 || Loc: 2.2358 Cla: 3.2574 Landm: 9.1569 || LR: 0.00100000 || Batchtime: 0.3584 s || ETA: 0:34:53
Epoch:1/1 || Epochiter: 901/6440 || Iter: 901/6440 || Loc: 1.9579 Cla: 2.8817 Landm: 6.6998 || LR: 0.00100000 || Batchtime: 0.3653 s || ETA: 0:33:43
Epoch:1/1 || Epochiter: 1201/6440 || Iter: 1201/6440 || Loc: 2.0543 Cla: 2.5807 Landm: 5.2528 || LR: 0.00100000 || Batchtime: 0.3525 s || ETA: 0:30:47
Epoch:1/1 || Epochiter: 1501/6440 || Iter: 1501/6440 || Loc: 2.2226 Cla: 2.7301 Landm: 5.8052 || LR: 0.00100000 || Batchtime: 0.3334 s || ETA: 0:27:27
Epoch:1/1 || Epochiter: 1801/6440 || Iter: 1801/6440 || Loc: 2.2389 Cla: 3.6269 Landm: 12.4245 || LR: 0.00100000 || Batchtime: 0.3464 s || ETA: 0:26:47
Epoch:1/1 || Epochiter: 2101/6440 || Iter: 2101/6440 || Loc: 0.9241 Cla: 2.2995 Landm: 5.9620 || LR: 0.00100000 || Batchtime: 0.3839 s || ETA: 0:27:46
Epoch:1/1 || Epochiter: 2401/6440 || Iter: 2401/6440 || Loc: 1.5829 Cla: 4.2119 Landm: 16.5259 || LR: 0.00100000 || Batchtime: 0.3728 s || ETA: 0:25:06
Epoch:1/1 || Epochiter: 2701/6440 || Iter: 2701/6440 || Loc: 2.6861 Cla: 3.2719 Landm: 5.8302 || LR: 0.00100000 || Batchtime: 0.3450 s || ETA: 0:21:30
Epoch:1/1 || Epochiter: 3001/6440 || Iter: 3001/6440 || Loc: 0.8829 Cla: 1.5760 Landm: 1.2438 || LR: 0.00100000 || Batchtime: 0.3376 s || ETA: 0:19:21
Epoch:1/1 || Epochiter: 3301/6440 || Iter: 3301/6440 || Loc: 2.3112 Cla: 3.5884 Landm: 11.7768 || LR: 0.00100000 || Batchtime: 0.3203 s || ETA: 0:16:45
Epoch:1/1 || Epochiter: 3601/6440 || Iter: 3601/6440 || Loc: 0.7980 Cla: 1.5307 Landm: 2.2358 || LR: 0.00100000 || Batchtime: 0.3865 s || ETA: 0:18:17
Epoch:1/1 || Epochiter: 3901/6440 || Iter: 3901/6440 || Loc: 2.6159 Cla: 3.3555 Landm: 10.4696 || LR: 0.00100000 || Batchtime: 0.3680 s || ETA: 0:15:34
Epoch:1/1 || Epochiter: 4201/6440 || Iter: 4201/6440 || Loc: 0.7935 Cla: 1.5137 Landm: 1.6873 || LR: 0.00100000 || Batchtime: 0.3356 s || ETA: 0:12:31
Epoch:1/1 || Epochiter: 4501/6440 || Iter: 4501/6440 || Loc: 1.5036 Cla: 1.9508 Landm: 4.2025 || LR: 0.00100000 || Batchtime: 0.3451 s || ETA: 0:11:09
Epoch:1/1 || Epochiter: 4801/6440 || Iter: 4801/6440 || Loc: 0.7017 Cla: 1.3746 Landm: 1.5991 || LR: 0.00100000 || Batchtime: 0.3802 s || ETA: 0:10:23
Epoch:1/1 || Epochiter: 5101/6440 || Iter: 5101/6440 || Loc: 2.9093 Cla: 2.7489 Landm: 8.5032 || LR: 0.00100000 || Batchtime: 0.3586 s || ETA: 0:08:00
Epoch:1/1 || Epochiter: 5401/6440 || Iter: 5401/6440 || Loc: 0.6278 Cla: 1.1736 Landm: 4.3687 || LR: 0.00100000 || Batchtime: 0.3804 s || ETA: 0:06:35
Epoch:1/1 || Epochiter: 5701/6440 || Iter: 5701/6440 || Loc: 2.0547 Cla: 3.1596 Landm: 3.4040 || LR: 0.00100000 || Batchtime: 0.3942 s || ETA: 0:04:51
Epoch:1/1 || Epochiter: 6001/6440 || Iter: 6001/6440 || Loc: 3.2170 Cla: 3.6431 Landm: 20.0956 || LR: 0.00100000 || Batchtime: 0.3940 s || ETA: 0:02:53
Epoch:1/1 || Epochiter: 6301/6440 || Iter: 6301/6440 || Loc: 2.5849 Cla: 2.7577 Landm: 1.9895 || LR: 0.00100000 || Batchtime: 0.4836 s || ETA: 0:01:07
```

Use any methods you feel comfortable with to remove the landmark supervision. Explain your method. And what do you expect the face detector would perform (better or worse) after the removal? Explain your answer.

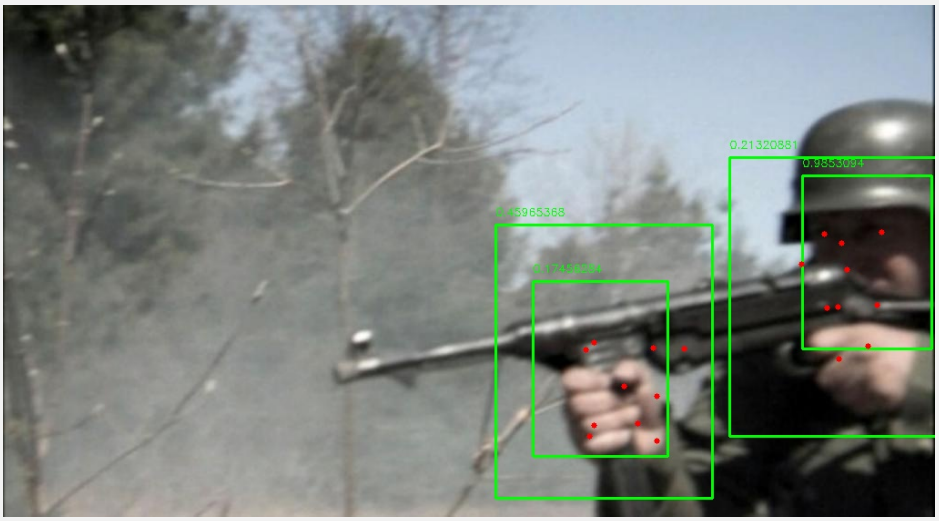
Solution

According to Piazza, can just remove landmark supervision from loss perspective. To do this, just delete remove landmark loss here:
loss = cfg['loc_weight'] * loss_l + loss_c + loss_landm
to:
loss = cfg['loc_weight'] * loss_l + loss_c
I think it will be worse since one loss is removed and the features of landmarks are also lost. Not that much information for detection.

8 Testing

Load the checkpoint we provided, it is the same as yours but trained with larger batchsize and with 180 epochs. Plot the first 4 results of your detector, with confidence, face bounding box, and landmarks.

Solution



Enjoy your face detector and feel free to deploy it for your home security.

Collaboration Survey Please answer the following:

1. Did you receive any help whatsoever from anyone in solving this assignment?

   √ **Yes**

   ◯ No

   - If you answered 'Yes', give full details, for any type of collaboration:
   - (e.g. "Jane Doe explained to me what is asked in Question 3.4")

   1. Post question in Piazza asking question logic and debugging code. 2. Visit OH to find how to use matrix calculation to make jaccard function faster rather than using loops. 3. Discuss with yichuanl about what what is asking in some question like 4.3.

2. Did you give any help whatsoever to anyone in solving this assignment?

   √ **Yes**

   ◯ No

   - If you answered 'Yes', give full details, for any type of collaboration:
   - (e.g. "I pointed Joe Smith to section 2.3 since he didn't know how to proceed with Question 2")

   Discuss with yichuanl about what what is asking in some question like 4.3.

3. Note that copying code or writeup even from a collaborator or anywhere on the internet violates the Academic Integrity Code of Conduct.