Joy Zhuge

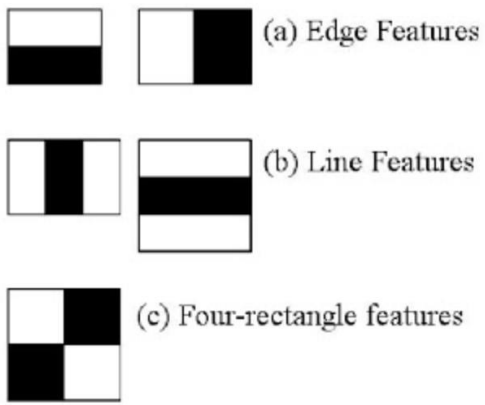
Simple Face Detection, Classification and Alignment

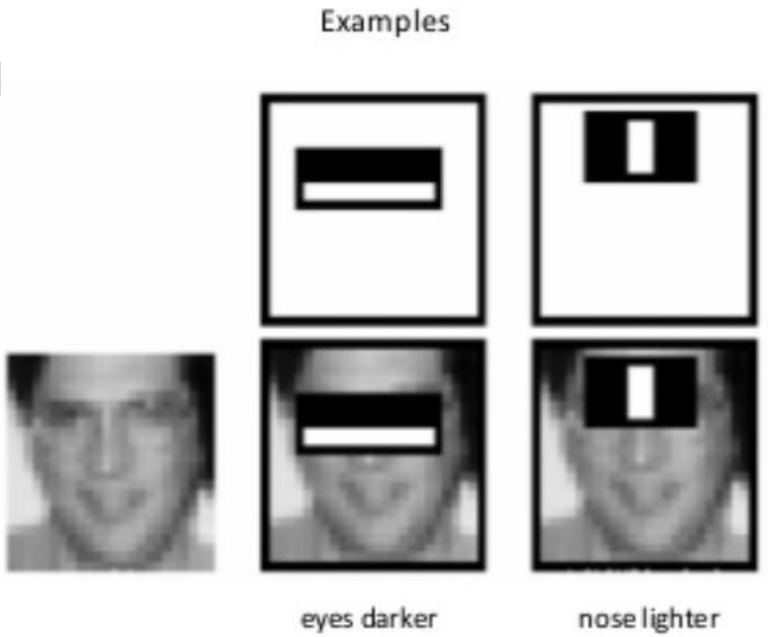
Face Detection

Face Detection: Camera focus, Digital make-up Application, Facebook tags, CSI identify 'bad guy' from security footage, Phone unlocking.

OpenCV: Open Source Computer Vision, a library of programing functions mainly aimed at real-time computer vision.

Haar feature-based: Old School





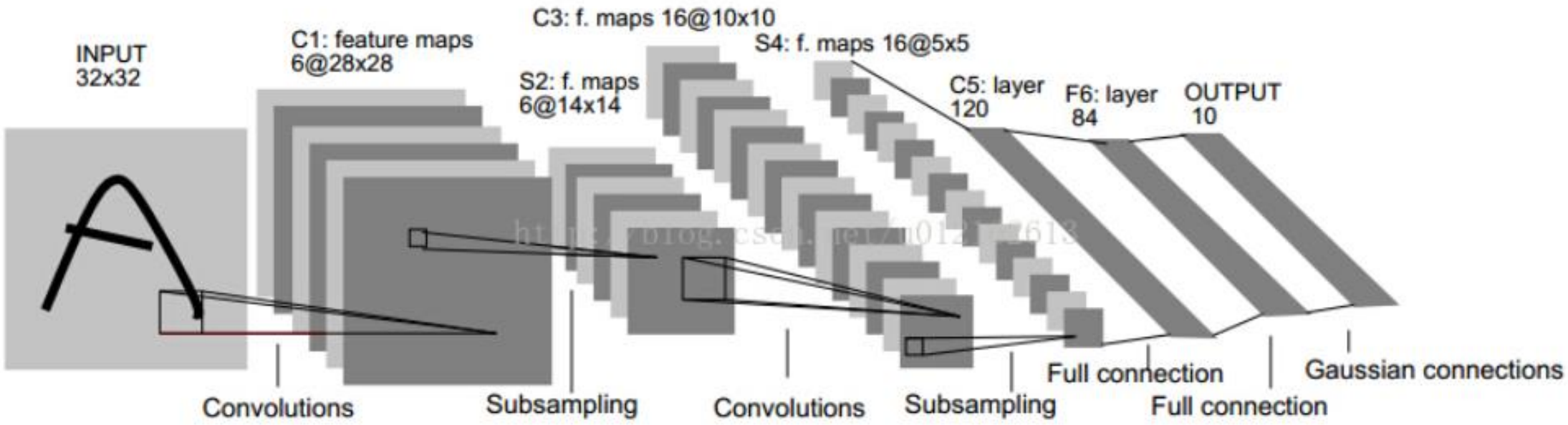
https://docs.opencv.org/3.4.1/d7/d8b/tutorial_py_face_detection.html

```python
import cv2
import sys

# Get user supplied values
imagePath = "ironman.jpg"
cascPath = "haarcascade_frontalface_default.xml"

# Create the haar cascade
faceCascade = cv2.CascadeClassifier(cascPath)

# Read the image
image = cv2.imread(imagePath)
cv2.imshow("Original", image)
gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)

# Detect faces in the image
faces = faceCascade.detectMultiScale(
    gray,
    scaleFactor=1.1,
    minNeighbors=5,
    minSize=(30, 30),
    flags=cv2.CASCADE_SCALE_IMAGE
)

 print("Found {0} faces!".format(len(faces)))

 # Draw a rectangle around the faces
 for (x, y, w, h) in faces:
     cv2.rectangle(image, (x, y), (x+w, y+h), (0, 255, 0), 2)

 cv2.imshow("Faces found", image)
 cv2.imwrite('ironman_face.jpg',image)
 cv2.waitKey(0)
```

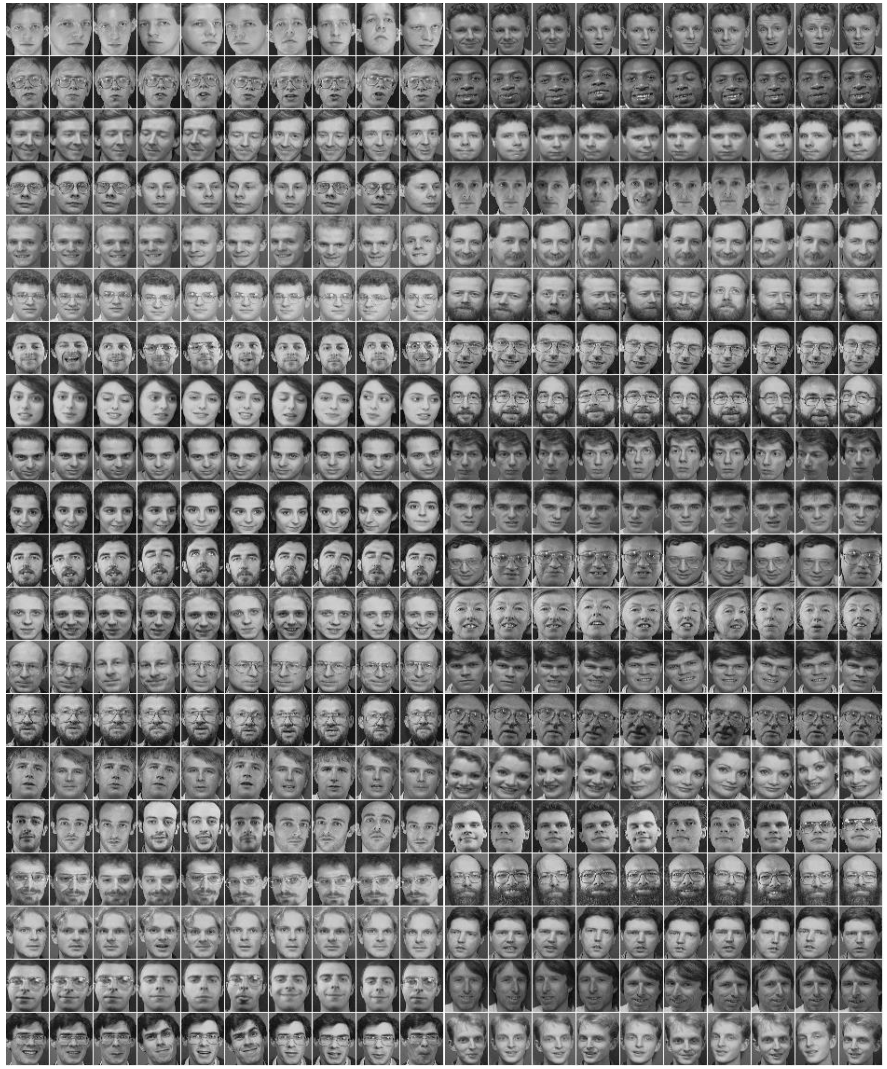
Found 8 faces!

Face Detection

Face Classification- LeNet5



Olivetti-faces

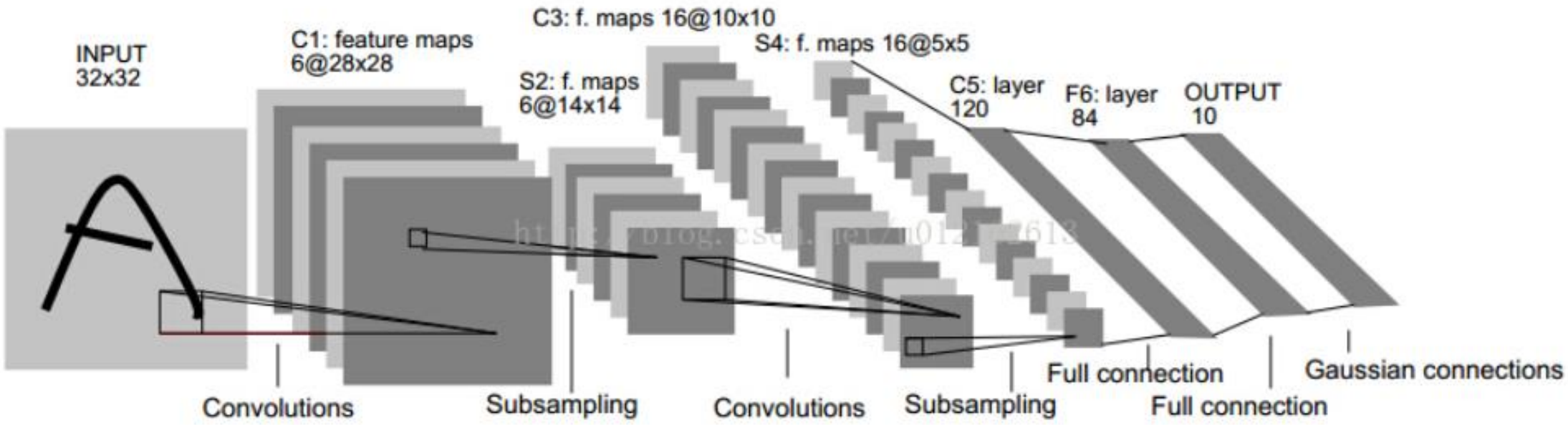


Small
Different time, varying the light,facial expression(open/closed eyes, smiling/not smiling), facial details(glasses/no glass)

Data_size: 40 individuals * 10= 400 images
Image_size: 64*64*8bits

Shuffle
Training_set: 320
Validation_set: 40
Testing_set 40

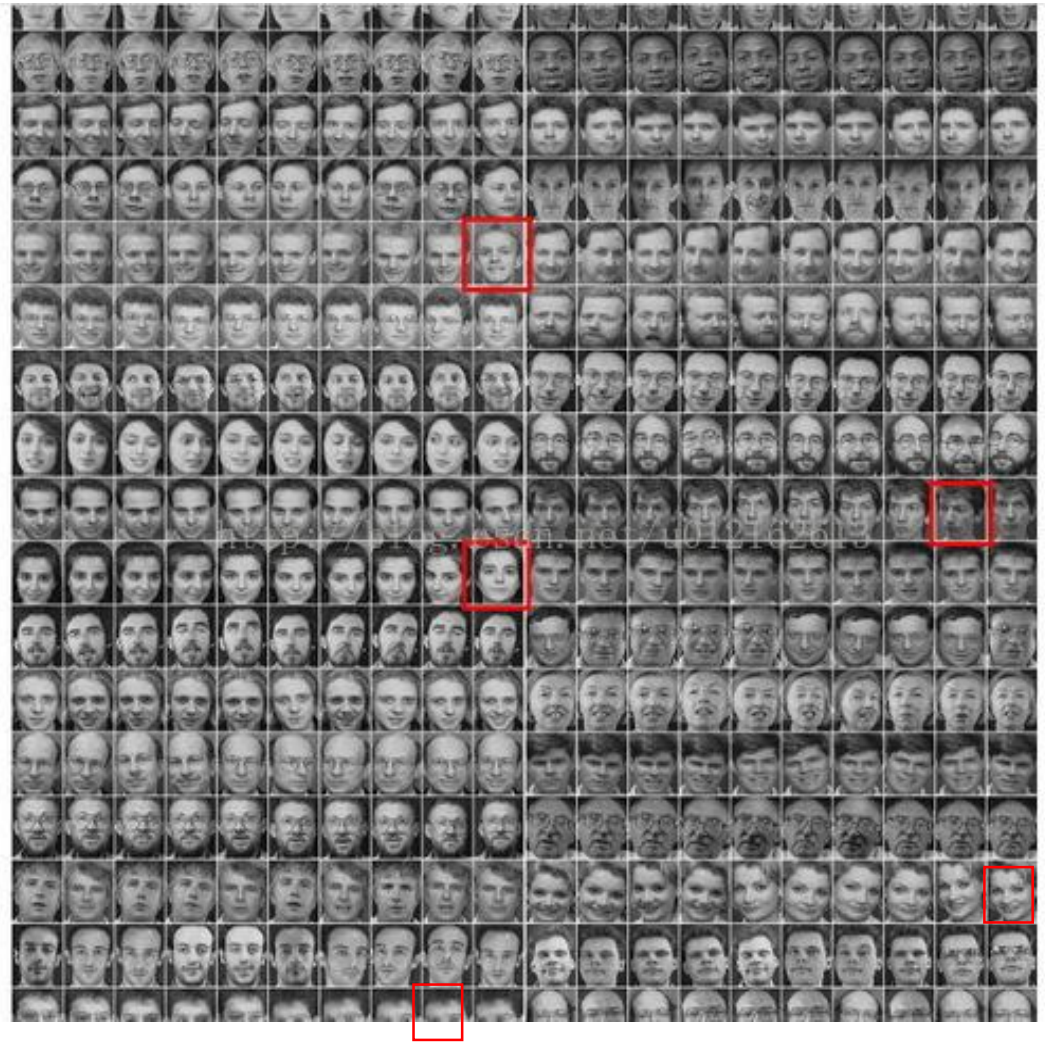
Face Classification- LeNet5



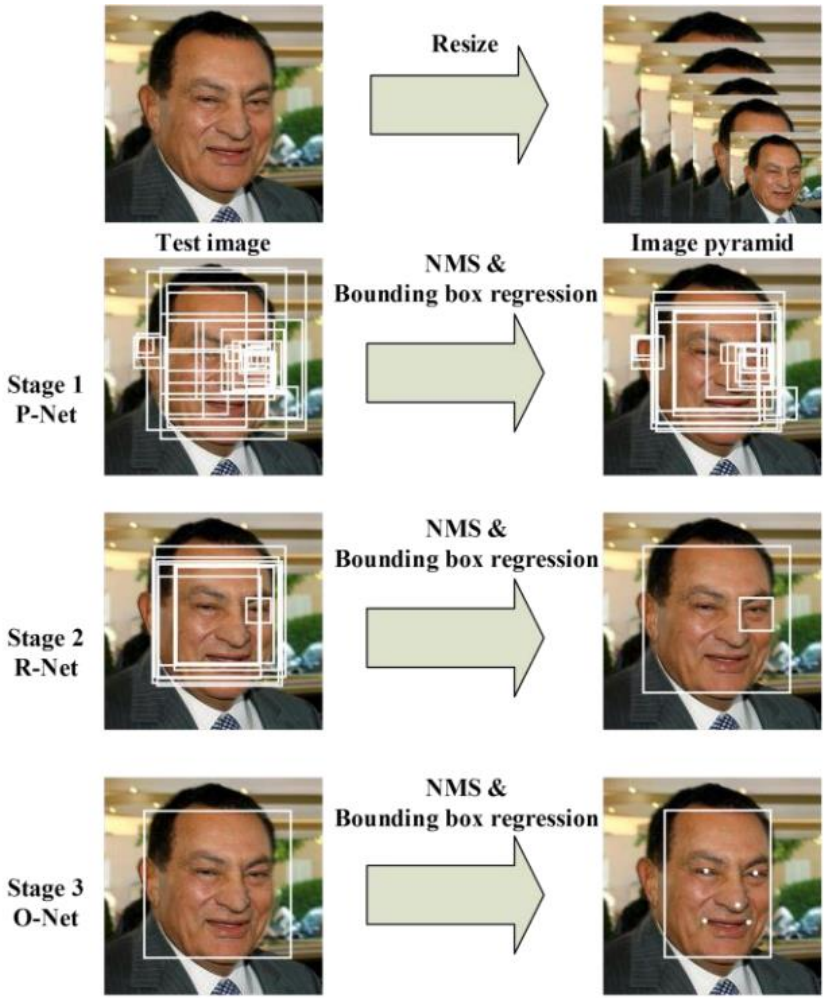
Face Classification

Learning_rate=0.05
Batch_size=10
max_epochs=500
Poolsize=(2,2)
Convolution kernel number Nkerns=[20,50]

validation error=5.00 % obtained at
iteration 304, with test error 7.500000 %
The code for CNN_face_detection_test.py ran for 2.17m
picture: 89 is person 8, but mis-predicted as person 34
picture: 178 is person 17, but mis-predicted as person 37
picture: 189 is person 18, but mis-predicted as person 6
picture: 299 is person 29, but mis-predicted as person 39
picture: 368 is person 36, but mis-predicted as person 20



Face-Alignment MTCNN Pipeline



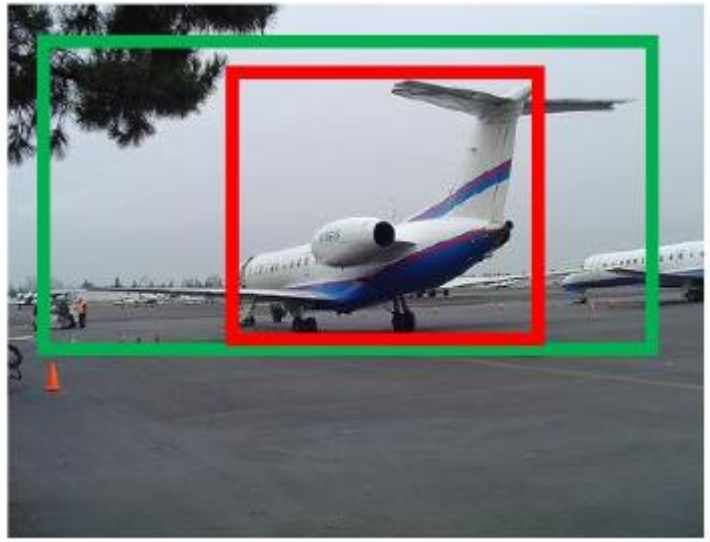
Initially, resize the image to different scales to build an pyramid

Proposal network: predicting potential face positions and their bounding boxes. The result of this step is a large number of face detections and lots of false detections.

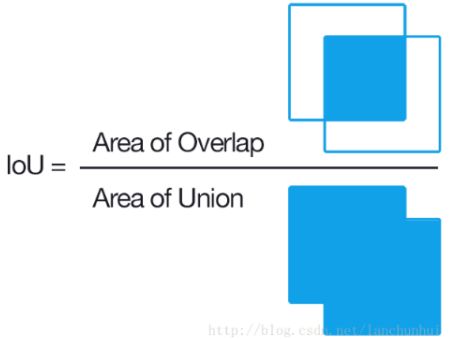
Refinement network: using images and outputs of the first prediction. It makes a refinement of the result to eliminate most of false detections and aggregate bounding boxes.

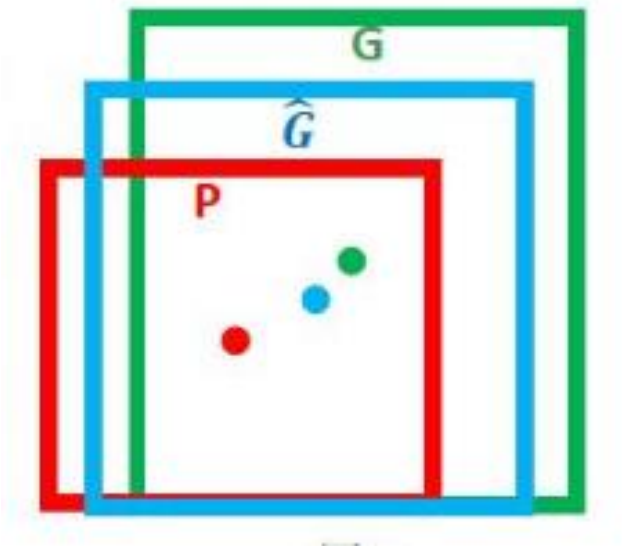
Output network: refining even more the predictions and adds five facial landmarks predictions including the bounding box, confidence, left eye, right eye, nose, mouth left and mouth right.

Bounding Box Regression



Green: ground truth
Red: Region proposal
IoU(intersection over union)<0.5





$$\Delta x = P_w d_x(P), \Delta y = P_h d_y(P)$$

$$\hat{G}_x = P_w d_x(P) + P_x \quad (1)$$

$$\hat{G}_y = P_h d_y(P) + P_y \quad (2)$$

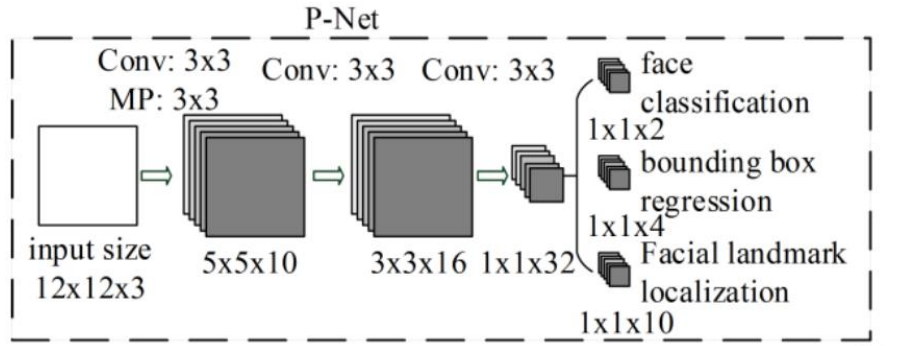
$$S_w = P_w d_w(P), S_h = P_h d_h(P)$$

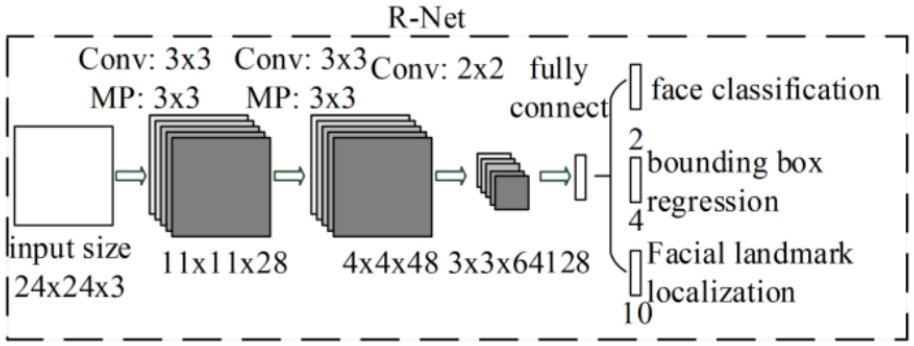
$$\hat{G}_w = P_w \exp(d_w(P)) \quad (3)$$

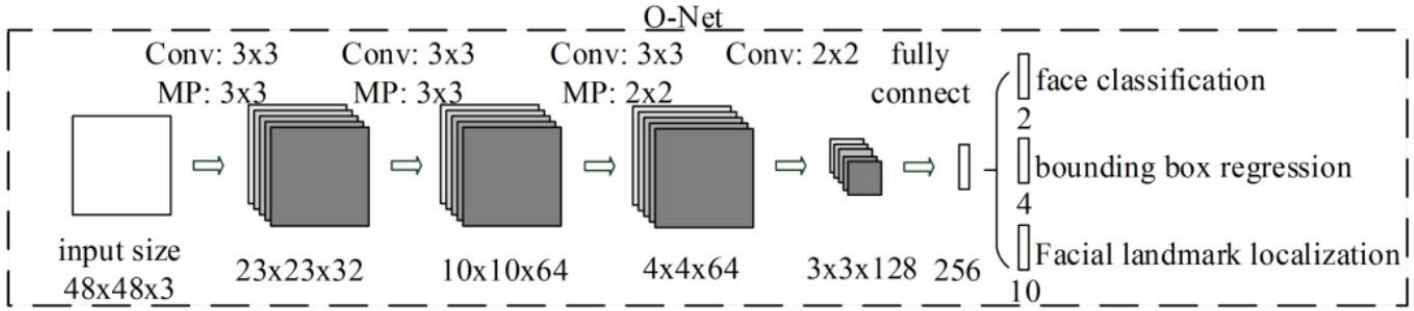
$$\hat{G}_h = P_h \exp(d_h(P)) \quad (4)$$

Non-Maximum Suppression(NMS)

P-Net, R-Net, O-Net







1) Face classification: The learning objective is formulated as a two-class classification problem. For each sample x_i, we use the cross-entropy loss:

$$L_i^{det} = -(y_i^{det} \log(p_i) + (1 - y_i^{det})(1 - \log(p_i))) \quad (1)$$

2) Bounding box regression: For each candidate window, we predict the offset between it and the nearest ground truth (i.e., the bounding boxes' left top, height, and width). The learning objective is formulated as a regression problem, and we employ the Euclidean loss for each sample x_i:

$$L_i^{box} = \left\| \hat{y}_i^{box} - y_i^{box} \right\|_2^2 \quad (2)$$

3) Facial landmark localization: Similar to the bounding box regression task, facial landmark detection is formulated as a regression problem and we minimize the Euclidean loss:

$$L_i^{landmark} = \left\| \hat{y}_i^{landmark} - y_i^{landmark} \right\|_2^2 \quad (3)$$

Wider Face & CelebA

Wider Face--detection
-- 32K images
-- 494K faces

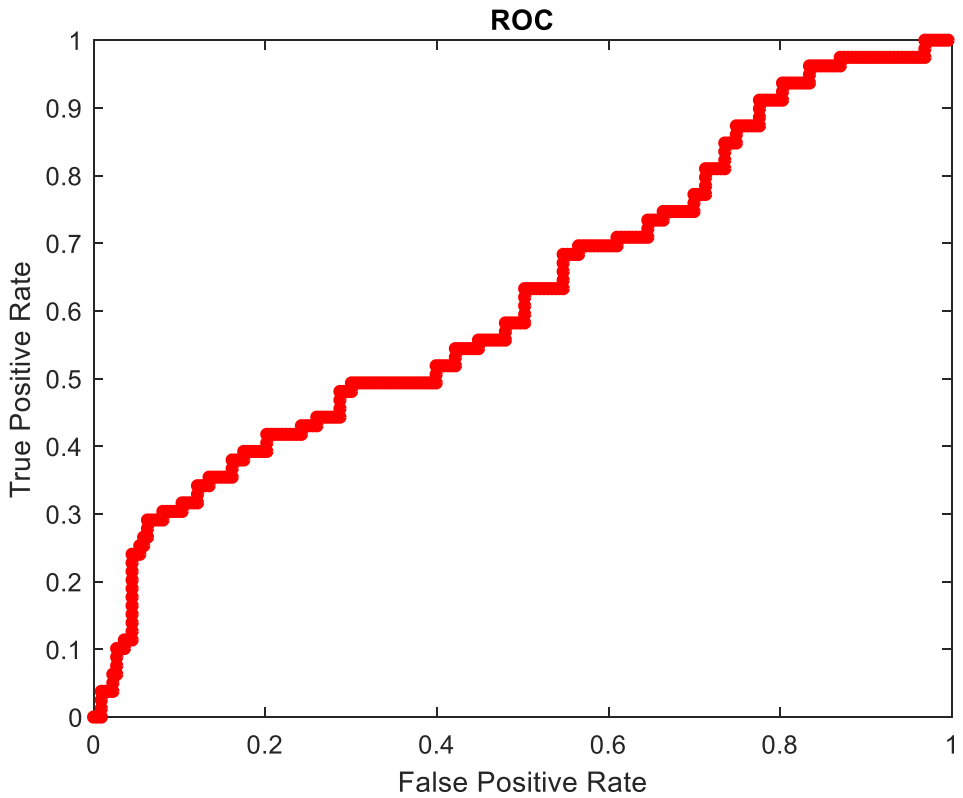
CelebA--alignment
-- 200K images, 10K persons
-- 5 facial landmarks, 40 binary attributes

Training Data

Training: 2000+2000
Testing: 300+300

Data annotation:
Negatives: Regions that the IoU ration less than 0.3 to any ground truth faces
Positives: IoU above 0.65 to a ground truth
Part faces: IoU between 0.4 to 0.65 to a ground truth
Landmark faces: faces labeled 5 landmarks' positions.

Face classification: Negatives, Positives
Bounding box regression: Positives, part faces
Facial landmark localization: landmark faces



AOC=0.6154

Found 11 faces!

Face Alignment