**CPSC 425 - Assignment 1**

Eloise Peng - 13812169

Oct. 8, 2019

```
from PIL import Image, ImageDraw
import numpy as np
import math
from scipy import signal
import ncc

scale = 0.75
line_width = 2
box_color = "red"
```

Part 2

```
# q2
def MakePyramid(image, minsize):
    "Scaled representation of the input image."
    "Creates a list of PIL images that each image is 0.75 of the previous image size"
    "and each image should not be smaller than minsize."
    im = Image.open(image)

    x, y = im.size

    pyramid = [];

    while x >= minsize and y >= minsize:
        # im.resize((int(x),int(y)), Image.BICUBIC).show() # show and check if the resize() works
        pyramid.append(im.resize((int(x),int(y)), Image.BICUBIC))
        x *= scale
        y *= scale

    return pyramid
```

Part 3

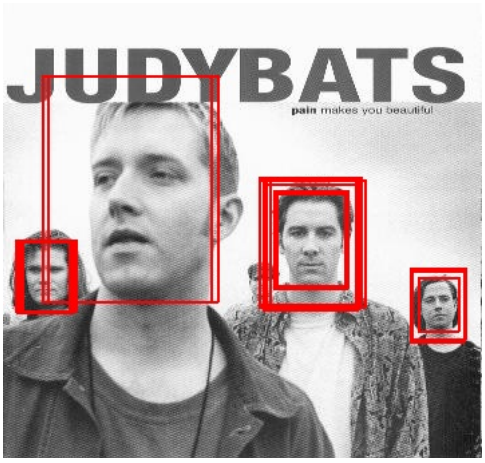
```
# q3
def ShowPyramid(pyramid):
    "Return one horizontal image that joins all the images in the pyramid."
    assert (len(pyramid) > 0), "The pyramid must have at least one image element."

    # since the first image is always the original image, which has the largest height
    width, height = 0, pyramid[0].size[1]
    for i in pyramid:
        x, y = i.size
        width += x

    # create a new image with white background
    image = Image.new("L", (width, height), 255)

    offset_x, offset_y = 0, 0
    for i in pyramid:
        x, y = i.size
        image.paste(i, (offset_x, height - y))
        offset_x += x
    image.show()
```

```
>>> import a2
>>> a2.ShowPyramid(MakePyramid("faces/judybats.jpg", 20))
>>>
```


Part 4

```python
# q4
drawRedLine = lambda draw, x1, y1, x2, y2 : draw.line((x1, y1, x2, y2), fill = box_color, width = line_width)

def FindTemplate(pyramid, template, threshold):
    "Find and mark all locations in the pyramid at which the NCC of the template with the image is above the threshold."
    assert (len(pyramid) > 0), "The pyramid must have at least one image element."

    template_width = 15
    template = Image.open(template)
    tx, ty = template.size
    t = template.resize((template_width, ty * template_width / tx), Image.BICUBIC)

    im = pyramid[0].copy().convert('RGB')
    draw = ImageDraw.Draw(im)

    for i, image in enumerate(pyramid):
        arr_nmcorrl = ncc.normxcorr2D(image, t)

        upscale = (1 / scale) ** i # scale to be multiple regards to original image
        height, width = arr_nmcorrl.shape

        # To find every pixel that is above the threshold, and output the peaks scaled to the original image
        peaks = []
        for x in range(width):
            for y in range(height):
                if (arr_nmcorrl[y][x] > threshold): peaks.append([x * upscale, y * upscale])

        for p in peaks:
            x_left = max(p[0] - t.size[0] * upscale, 0)
            y_top = max(p[1] - t.size[1] * upscale, 0)
            x_right = min(p[0] + t.size[0] * upscale, im.size[0] - 1)
            y_bottom = min(p[1] + t.size[1] * upscale, im.size[1] - 1)
            # draw a red rectangle box for matchings
            drawRedLine(draw, x_left, y_top, x_right, y_top) #top
            drawRedLine(draw, x_left, y_top, x_left, y_bottom) #left
            drawRedLine(draw, x_left, y_bottom, x_right, y_bottom) #bottom
            drawRedLine(draw, x_right, y_top, x_right, y_bottom) #right

    del draw
    im.show()
    return im
```

```
>>> import a2
>>> threshold = 0.6
>>> pyramid = a2.MakePyramid("./faces/judybats.jpg", 20)
>>> template = "./faces/template.jpg"
```




Part 5

First try,
for threshold = 0.6
we count:

| Images | **False positive** | **False negative** |
|---|---|---|
| Family | 0 | 2 |
| Tree | 2 | 0 |
| Fans | 1 | 2 |
| Judybats | 0 | 1 |
| Sports | 0 | 1 |
| Students | 2 | 4 |
| Total | 5 | 10 |

for threshold = 0.55
we count:

| Images | **False positive** | **False negative** |
|---|---|---|
| Family | 0 | 1 |
| Tree | 4 | 0 |
| Fans | 2 | 2 |
| Judybats | 1 | 1 |
| Sports | 1 | 1 |
| Students | 4 | 3 |
| Total | 12 | 8 |

As we saw above, the threshold of 0.55 is equally rate comparing to 0.6, let's take 0.575 to try again.

for threshold = 0.575
we count:

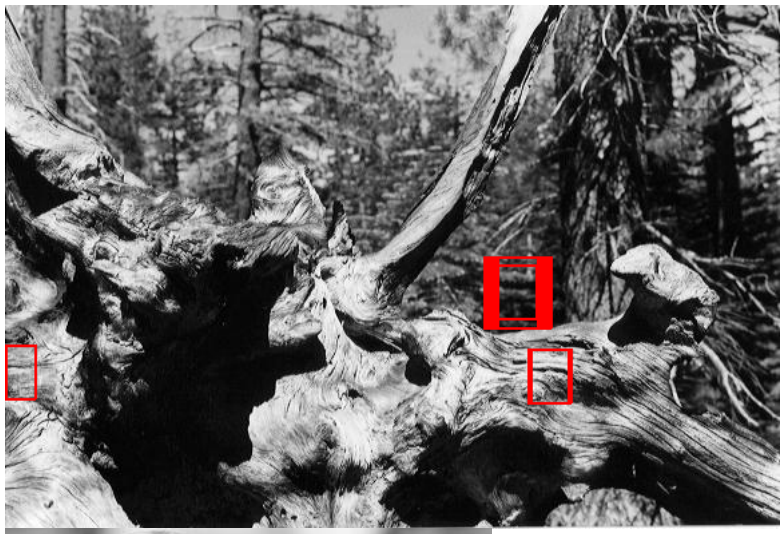
| Images | **False positive** | **False negative** |
|---|---|---|
| Family | 0 | 1 |
| Tree | 3 | 0 |
| Fans | 1 | 2 |
| Judybats | 1 | 1 |
| Sports | 0 | 1 |
| Students | 3 | 4 |
| Total | 8 | 9 |

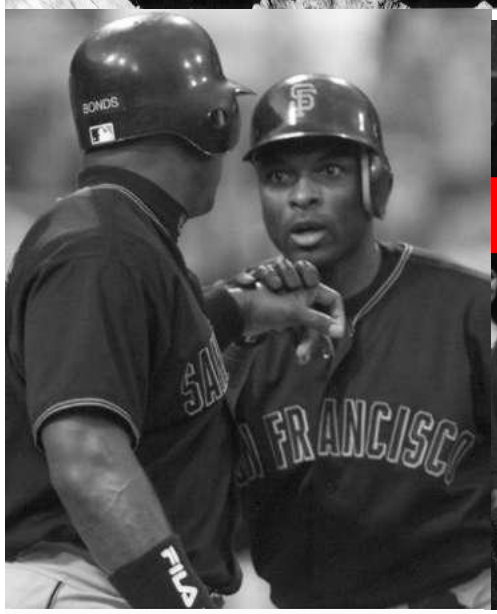
As we see, the threshold 0.575 is close, let's take 0.571,

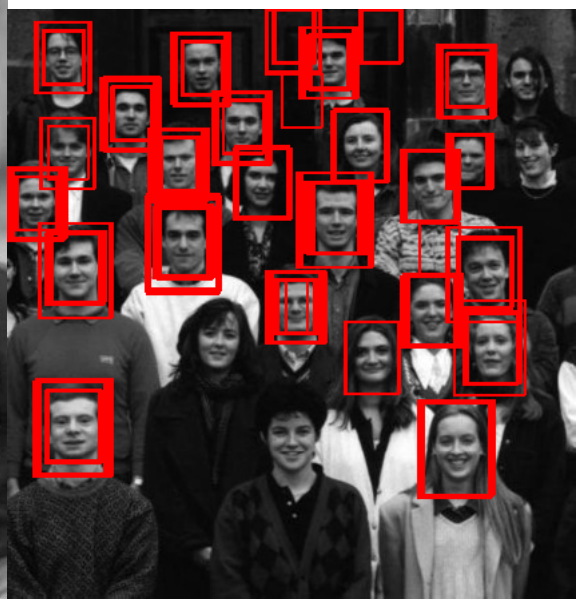
**for threshold = 0.571**
we count:

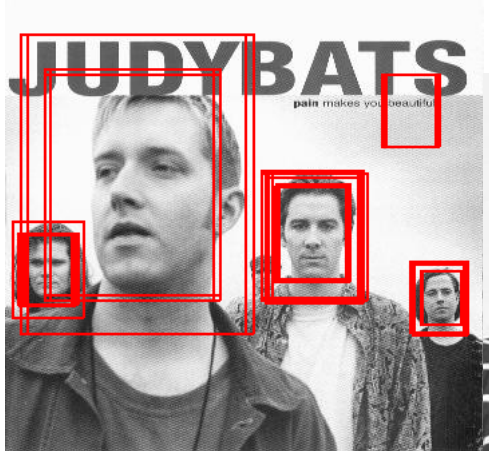
| Images | **False positive** | **False negative** |
|---|---|---|
| Family | 0 | 1 |
| Tree | 3 | 0 |
| Fans | 1 | 2 |
| Judybats | 1 | 1 |
| Sports | 0 | 1 |
| Students | 4 | 4 |
| **Total** | **9** | **9** |

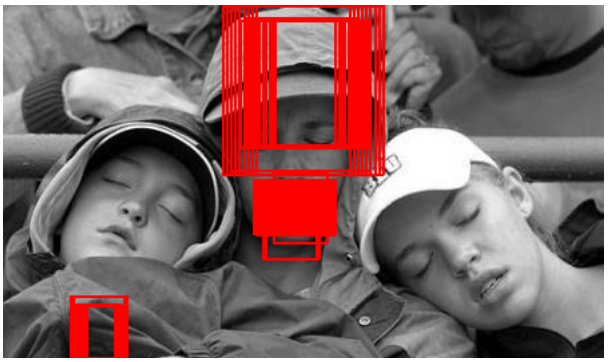
**Therefore, my final threshold is 0.571.**




BONDS
FRANCISC
FILA




JUDYBATS
pain makes you beautiful



Part 6

Recall rate:

| Images | **Recall rate** |
|---|---|
| Family | 2/3 |
| Tree | 0/0 |
| Fans | 1/3 |
| Judybats | 4/5 |
| Sports | 0/1 |
| Students | 23/27 |

NCC method has very low rate on both Fans.jpg and Sports.jpg. As we observed, they faces that are tilted, hiding in the dark, or covered by helmet/hats not picked up. We found that all false negative cases exist in common as they are having low light condition, different orientation, or partial occlusion, as NCC method is not robust to above cases.