Homework 1: Face Detection

109550136 邱弘竣

Part I. Implementation (6%):

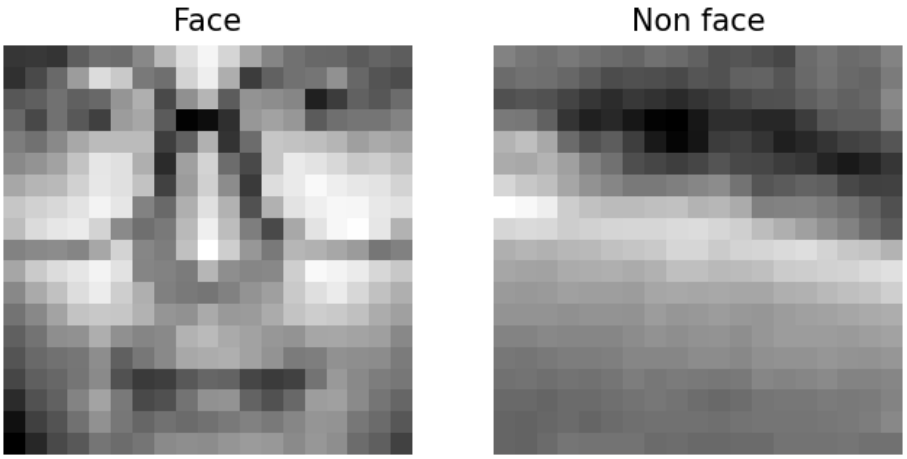
```
# Begin your code (Part 1)
# raise NotImplementedError("To be implemented")
"""
List all the path by os.listdir().
Load image by cv2.imread with grayscale.
Append img which is a numpy array to “temp”, a tuple, and append its label to “temp”.
Finally, append “temp” to “dataset”.
"""
dataset=[]
rootDir = dataPath
for lists in os.listdir(rootDir):
  path = os.path.join(rootDir, lists)
  for subpath in os.listdir(path):
    temp=[]
    ans = os.path.join(path, subpath)
    img = cv2.imread(ans,0)
    np.array(img)
    if lists=='face':
      temp.append(img)
      temp.append(1)
    else:
      temp.append(img)
      temp.append(0)
    dataset.append(temp)
# End your code (Part 1)
```

```
# Begin your code (Part 2)
# raise NotImplementedError("To be implemented")
"""
If the "featureVals" smaller than 0, it means the classifier thinks that the image is a face.
Therefore, "ans"=1, and vice versa.
weight * (ans-label) = error.
Add all the images' error to sum, and find the smallest. -> weak classifier
Finally, turn the HaarFeature class that we chose to WeakClassifier Class.

"""
min=float('inf')
for i in range(len(featureVals)):
  sum=0
  for j in range(len(featureVals[0])):
    if featureVals[i][j]<0:
      ans=1
    else:
      ans=0
    sum=sum+weights[j]*abs(ans-labels[j])
  if min>sum:
    min=sum
    minindex=i
weakclassifier_class = classifier.WeakClassifier(features[minindex])
bestClf = weakclassifier_class
bestError=min
print(min)
# End your code (Part 2)
```

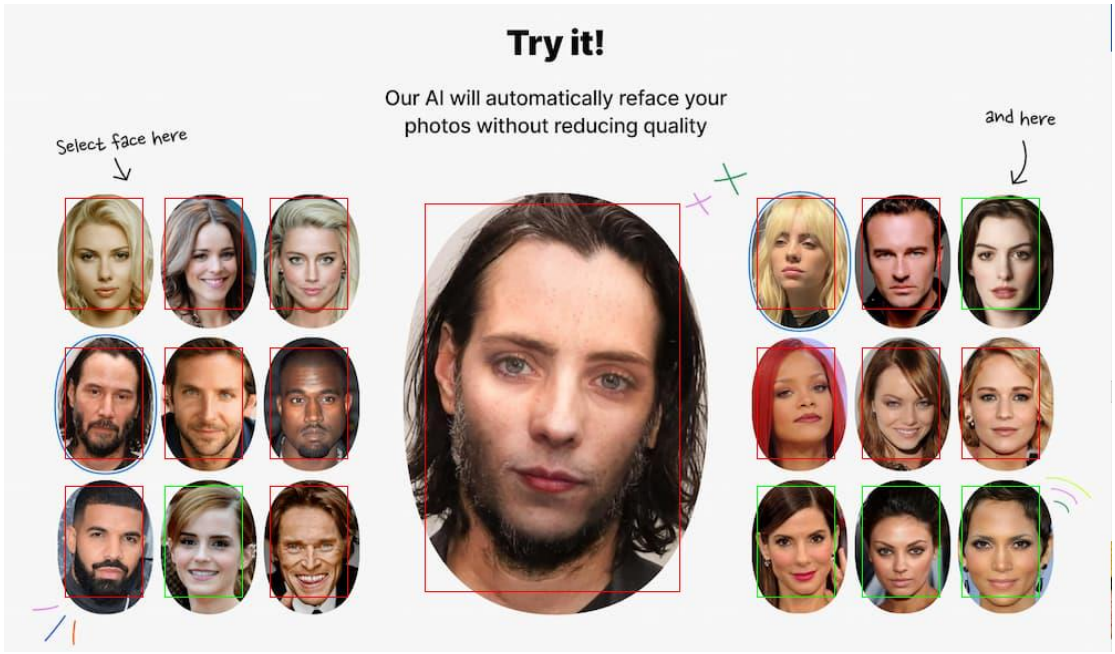
```
# Begin your code (Part 4)
# raise NotImplementedError("To be implemented")
"""
The top number in .txt means how many face we have to detect.
the following ones means the position of the face, we use cv2.rectangle to draw the rectangle.
Use classify in clssifier.py to see whether the classifier thinks it is a face.
To comform to different data in the .txt,
I use a for loop to see which image is the one that I have to deal with now.
It is moe convenient for me to doo part five.

"""
rootDir=os.path.abspath(os.path.join(dataPath, os.path.pardir))
jpg = glob.glob(os.path.join(rootDir, "*.jpg"))
image=cv2.imread(jpg[0],0)#grayscale
image_ans=cv2.imread(jpg[0])#彩色

f = open(dataPath,'rb')
for k in range(len(jpg)):
  small_image_position=[]
  line = f.readline()
  (name, size) = [i for i in line.split()]
  #print(name)
  for i in range(int(size)):
    temp=[]
    line = f.readline()
    for j in line.split():
      temp.append(int(j))
    small_image_position.append(temp)
```

```
  for i in jpg:
    name_str=str(name, 'utf-8')
    if name_str==os.path.basename(i):
      image=cv2.imread(i,0) #grayscale
      image_ans=cv2.imread(i) #彩色
      #print("success")
      break
  for i in range(len(small_image_position)):
    x1=small_image_position[i][0]
    y1=small_image_position[i][1]
    x2=small_image_position[i][2]
    y2=small_image_position[i][3]
    smallimage=cv2.resize(image[y1:y1+y2,x1:x1+x2],(19,19),interpolation=cv2.INTER_NEAREST)
    #clf = classifier.WeakClassifier()
    smallimage_array = np.array(smallimage)
    #print(smallimage_array)
    ans = clf.classify(smallimage_array)
    if ans==1:
      cv2.rectangle(image_ans, (x1, y1), (x2+x1, y2+y1), (0, 255, 0))
    else:
      cv2.rectangle(image_ans, (x1, y1), (x2+x1, y2+y1), (0, 0, 255))
  cv2.imshow('My Image', image_ans)
  cv2.waitKey(0)
  cv2.destroyAllWindows()


# End your code (Part 4)
```

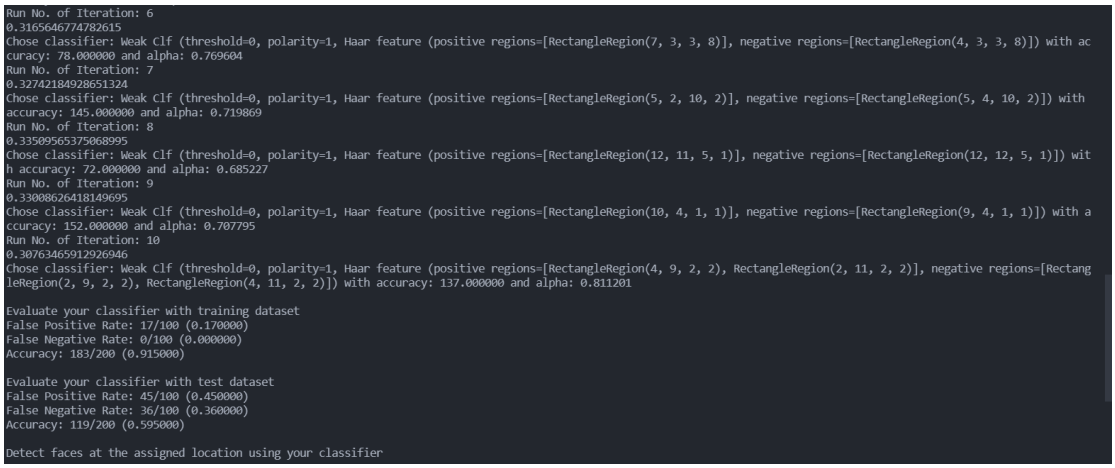
Part II. Results & Analysis (12%):


Face
Non face


THE
BEATLES




```
Loading images
(19, 19)
The number of training samples loaded: 200
(19, 19)
The number of test samples loaded: 200
Show the first and last images of training dataset
Computing integral images
Building features
Applying features to dataset
Selecting best features
Selected 5171 potential features
Initialize weights
Run No. of Iteration: 1
0.1900000000000014
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(8, 0, 1, 3), RectangleRegion(7, 3, 1, 3)], negative regions=[RectangleRegion(7, 0, 1, 3), RectangleRegion(8, 3, 1, 3)]) with accuracy: 162.000000 and alpha: 1.450010
Run No. of Iteration: 2
0.21637426900584811
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(4, 8, 2, 9)], negative regions=[RectangleRegion(2, 8, 2, 9)]) with accuracy: 156.000000 and alpha: 1.286922
Run No. of Iteration: 3
0.26663977410246065
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(16, 16, 1, 2)], negative regions=[RectangleRegion(15, 16, 1, 2)]) with accuracy: 155.000000 and alpha: 1.011738
Run No. of Iteration: 4
0.28727004067022943
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(4, 14, 8, 2)], negative regions=[RectangleRegion(4, 16, 8, 2)]) with accuracy: 153.000000 and alpha: 0.908680
Run No. of Iteration: 5
0.28410245052361843
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(10, 8, 1, 1)], negative regions=[RectangleRegion(9, 8, 1, 1)]) with accuracy: 155.000000 and alpha: 0.924202
```



200張	train data accuracy	test data accuracy
mathos 1 T=1	81%	48%
mathos 1 T=2	81%	48%
mathos 1 T=3	88%	53%
mathos 1 T=4	86%	47.50%
mathos 1 T=5	88.50%	54%
mathos 1 T=6	89%	51%
mathos 1 T=7	90%	54.50%
mathos 1 T=8	91%	55%
mathos 1 T=9	90%	57.50%
mathos 1 T=10	91.50%	59.50%

Discuss：

Training data accuracy is much higher than testing data accuracy since the model trains for the training data not the latter one.
Besides, the higher the parameter is, the higher the accuracy is. It's because parameter means the number of weak classifiers which should be used.

Part III. Answer the questions (12%):

1. Please describe a problem you encountered and how you solved it.
   Ans. In the beginning, error was always bigger than one so that beta was always negative, it led to error in python. I tried to understood the code and found "np.linalg.norm(weights, ord=1) " might be the problem that stop my footstep, so I rewrite it to "np.sum(weights)".

2. What are the limitations of the Viola-Jones' algorithm?
   Ans. Training time is very slow, and restricted to binary classification.

3. Based on Viola-Jones' algorithm, how to improve the accuracy except increasing the training dataset and changing the parameter T?
   Ans. We should find a high-qualified photo.

4. Please propose another possible face detection method (no matter how good or bad, please come up with an idea). Please discuss the pros and cons of the idea you proposed, compared to the Adaboost algorithm.
   Ans. Pros of Knowledge-based system is that it can accept a large amount of data in a short time. However, the disadvantage is that it doesn't accept the data that is different from the rules.