111550129 林彥亨

Homework 1：Face Detection

Part I. Implementation：

Part 1：Load and prepare the dataset

dataset.py -> load_data_small()

- **Line 24、25 to initialize the lists**
- **Line 27、28 to define the path**
- **Line 30~44 the load data from the train folder and test folder. Marking the image face with ‘1’, and non-face with‘0’.**

```
21 # Begin your code (Part 1-1)
22 #raise NotImplementedError("To be implemented")
23 # Initialize lists to store data
24 train_data = []
25 test_data = []
26 # Define the path for training and testing images
27 train_path = 'data/data_small/train'
28 test_path = 'data/data_small/test'
29
30 root = str(train_path) + "/face/"
31 for image in os.listdir( root ):
32     train_data.append( (cv2.imread(root+image, cv2.IMREAD_GRAYSCALE), 1) )
33
34 root = str(train_path) + "/non-face/"
35 for image in os.listdir( root ):
36     train_data.append( (cv2.imread(root+image, cv2.IMREAD_GRAYSCALE), 0) )
37
38 root = str(test_path) + "/face/"
39 for image in os.listdir( root ):
40     test_data.append( (cv2.imread(root+image, cv2.IMREAD_GRAYSCALE), 1) )
41
42 root = str(test_path) + "/non-face/"
43 for image in os.listdir( root ):
44     test_data.append( (cv2.imread(root+image, cv2.IMREAD_GRAYSCALE), 0) )
45
46 # Compile the datasets
47 dataset = (train_data, test_data)
48 # End your code (Part 1-1)
49 return dataset
```

Dataset.py -> load_data_FDDB()

- Line 108-110,first choose a no-face region
- Line 112 to 115, if the image is not cropped, and select a position to crop.
- Line 117-126, check the region whether intersect the face box.
- Line 128-132, add non-face data.

```
106    for i in range(num_faces):
107        # Randomly choose a non-face region
108        img_height, img_width = img_gray.shape
109        non_face_cropped = False
110        while not non_face_cropped:
111            # Randomly select a position for cropping
112            start_x = np.random.randint(0, img_width - 19)
113            start_y = np.random.randint(0, img_height - 19)
114            end_x = start_x + 19
115            end_y = start_y + 19
116
117            # Check if the cropped region intersects with any face bounding box
118            intersect = False
119            for face_box in face_box_list:
120                face_left_top, face_right_bottom = face_box
121                face_start_x, face_start_y = face_left_top
122                face_end_x, face_end_y = face_right_bottom
123                if not (end_x < face_start_x or start_x > face_end_x or end_y < face_start_y
124                        or start_y > face_end_y):
125                    intersect = True
126                    break
127
128            # If no intersection, add non-face data
129            if not intersect:
130                non_face_img = img_gray[start_y:end_y, start_x:end_x].copy()
131                nonface_dataset.append((cv2.resize(non_face_img, (19, 19)), 0))
132                non_face_cropped = True
```

Part 2：Implement Adaboost Algorithm

- **Line 167 to 169, we initialize the parameter to 0 and infinity.**
- **Line 170 to 177, check the featureVales and labels.**
- **if the epsilon is less than bestError, than let bestError equal to epsilon.**
- **Finally, return bestError and bestClf.**

```
163         # Begin your code (Part 2)
164         # raise NotImplementedError("To be implemented")
165 
166 
167         col_num, row_num  = featureVals.shape
168         bestError = float('inf')
169         bestClf = [0]
170         for i in range(col_num):
171             epsilon = 0
172             for j in range(row_num):
173                 if (featureVals[i][j]<0 and labels[j]==0) or (featureVals[i][j]>=0 and labels[j]==1):
174                     epsilon+=weights[j]
175             if epsilon < bestError:
176                 bestError = epsilon
177                 bestClf=WeakClassifier(features[i])
```

Part 4：Detect face：Detection.py -> detect()

- **Line 24、25, open the .txt file to read the format of image.**
- **Line 27 to 50, first, check the image and doing the processing and adding the face green box.**
- **Line 52 to 56, show the results.**

```python
24  with open(dataPath, 'r') as file:
25      lines = file.readlines()
```

```python
26
27  current_image = None
28  for line in lines:
29      parts = line.strip().split(' ')
30      if len(parts) == 2:  # This is a new image
31          if current_image is not None:
32              # Display the result for the previous image before moving on to the next
33              plt.figure()
34              plt.imshow(cv2.cvtColor(current_image, cv2.COLOR_BGR2RGB))
35              plt.show()
36          # Load new image
37          img_path = os.path.join('data/detect/', parts[0])
38          current_image = cv2.imread(img_path)
39          num_faces = int(parts[1])
40      elif len(parts) == 4 and current_image is not None:  # This is a face region
41          x, y, w, h = map(int, parts)
42          # Crop and resize face region to 19x19, then convert to grayscale
43          face_region = cv2.resize(current_image[y:y+h, x:x+w], (19, 19))
44          face_region_gray = cv2.cvtColor(face_region, cv2.COLOR_BGR2GRAY)
45
46          # Classify the face region
47          is_face = clf.classify(face_region_gray)
48          # Draw a green box if it's a face, red otherwise
49          color = (0, 255, 0) if is_face else (0, 0, 255)
50          cv2.rectangle(current_image, (x, y), (x+w, y+h), color, 2)
```

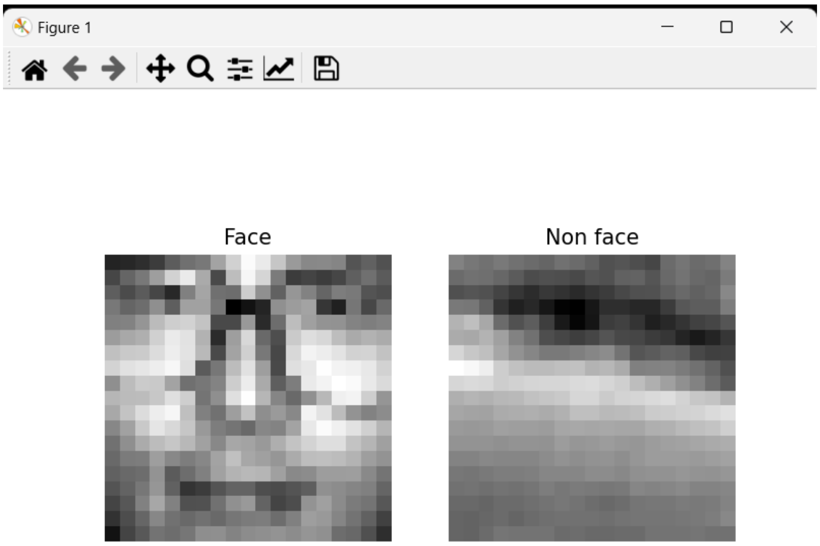
```python
51
52  # Display the result for the last image
53  if current_image is not None:
54      plt.figure()
55      plt.imshow(cv2.cvtColor(current_image, cv2.COLOR_BGR2RGB))
56      plt.show()
57
58  # End your code (Part 4)
```

Part II. Results & Analysis：

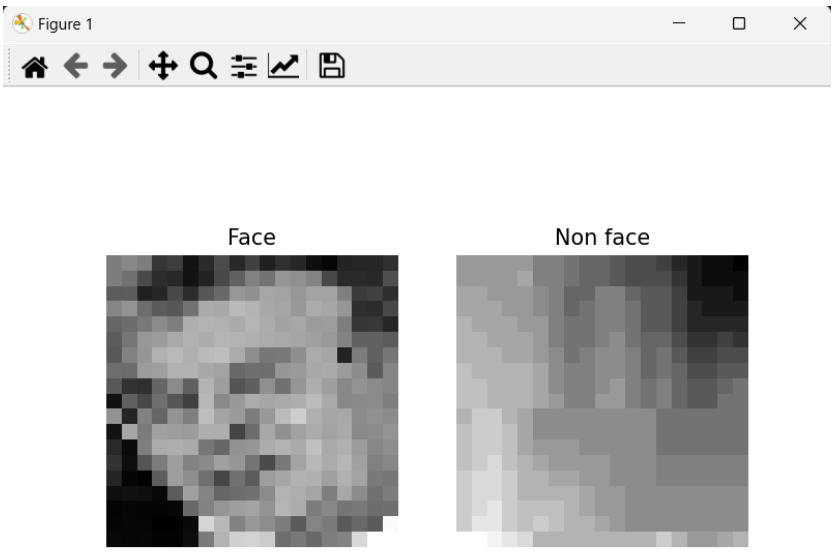
Part 1：

- **The output of running the dataset.py.**

The data_small：



The data_FDDB：



Part 3：

- **The results with the method of threshold = 0 , polarity = 1 and T=10.**

  data_small

```
Run No. of Iteration: 10
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (
positive regions=[RectangleRegion(4, 9, 2, 2), RectangleRegion(2, 1
1, 2, 2)], negative regions=[RectangleRegion(2, 9, 2, 2), Rectangle
Region(4, 11, 2, 2)]) with accuracy: 199.685000 and alpha: 0.811201

Evaluate your classifier with training dataset
False Positive Rate: 17/100 (0.170000)
False Negative Rate: 0/100 (0.000000)
Accuracy: 183/200 (0.915000)

Evaluate your classifier with test dataset
False Positive Rate: 45/100 (0.450000)
False Negative Rate: 36/100 (0.360000)
Accuracy: 119/200 (0.595000)
```

  data_FDDB：

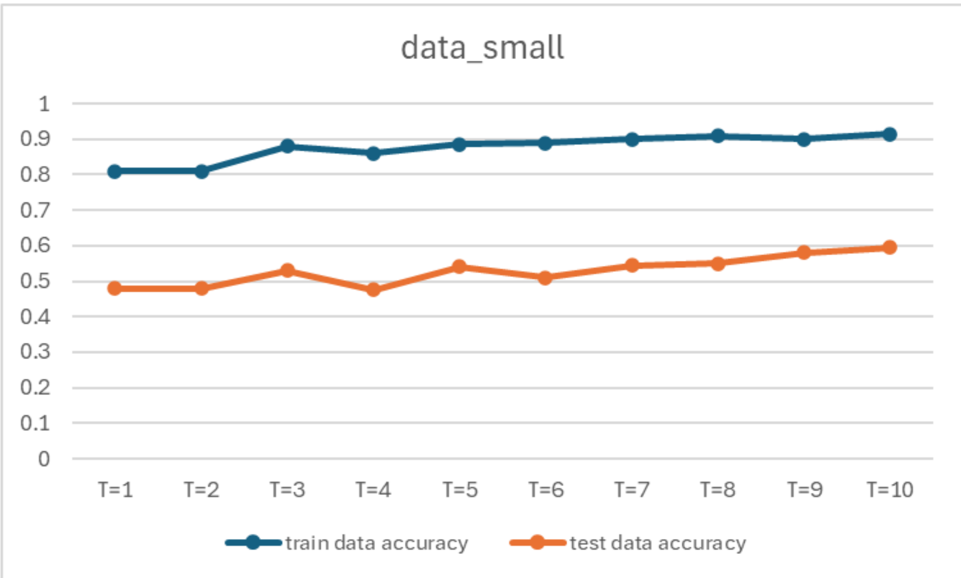
```
Run No. of Iteration: 10
Chose classifier: Weak Clf (threshold=0, pola
rity=1, Haar feature (positive regions=[Recta
ngleRegion(14, 2, 3, 6), RectangleRegion(11,
8, 3, 6)], negative regions=[RectangleRegion(
11, 2, 3, 6), RectangleRegion(14, 8, 3, 6)])
with accuracy: 719.626389 and alpha: 0.304619

Evaluate your classifier with training datase
t
False Positive Rate: 98/360 (0.272222)
False Negative Rate: 35/360 (0.097222)
Accuracy: 587/720 (0.815278)

Evaluate your classifier with test dataset
False Positive Rate: 50/155 (0.322581)
False Negative Rate: 17/155 (0.109677)
Accuracy: 243/310 (0.783871)
```

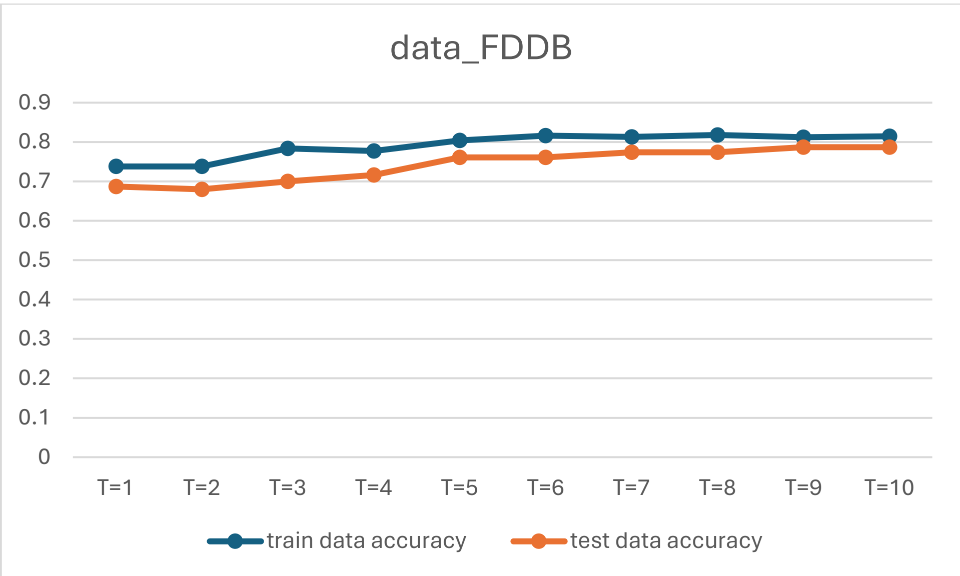
- **The table is the result of our training model with the data (data_small) and the T from 1 to 10. We can see that when T increasing, the training data accuracy is increasing.**

200 張	train data accuracy	test data accuracy
T=1	0.81	0.48
T=2	0.81	0.48
T=3	0.88	0.53
T=4	0.86	0.475
T=5	0.885	0.54
T=6	0.89	0.51
T=7	0.9	0.545
T=8	0.91	0.55
T=9	0.9	0.58
T=10	0.915	0.595



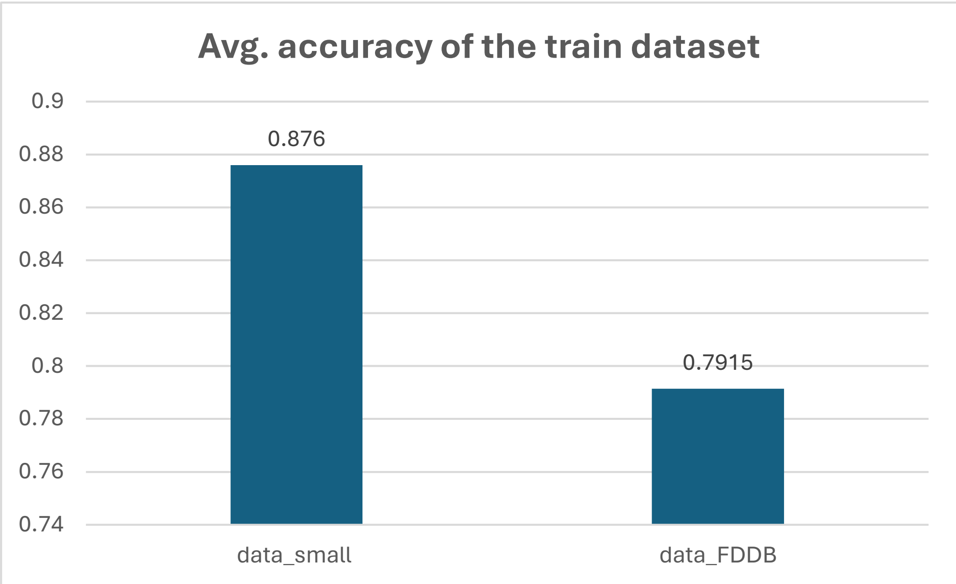
- **The table is the result of our training model with the data (data_FDDB) and the T from 1 to 10. We can see that when T increasing, the training data accuracy is increasing.**

Train：700 張 Test：310 張	train data accuracy	test data accuracy
T=1	0.738	0.687
T=2	0.738	0.68
T=3	0.784	0.7
T=4	0.777	0.716
T=5	0.804	0.761
T=6	0.816	0.761
T=7	0.813	0.774
T=8	0.818	0.774
T=9	0.812	0.787
T=10	0.815	0.787



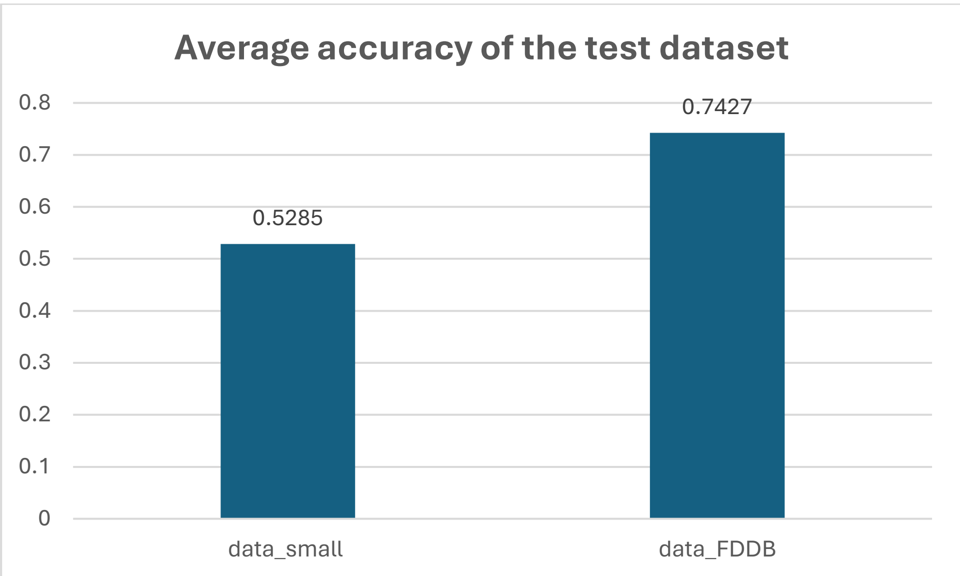
- **Compare the results of different dataset.**

In train dataset：



The average accuracy of data_small is better than the accuracy of data_FDDB.

In test dataset：

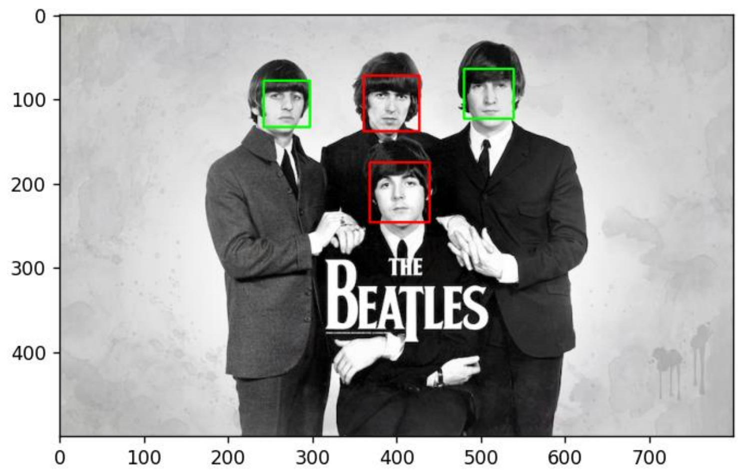


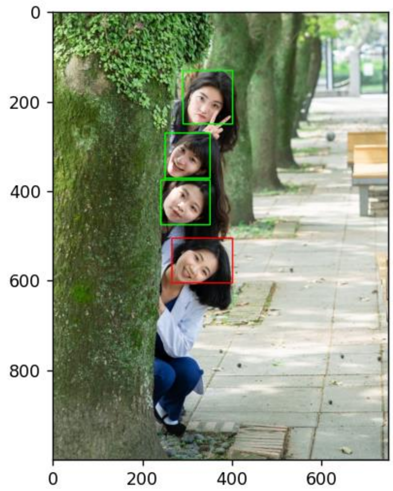
The average accuracy of data_FDDB is better than the accuracy of data_small.

Part 4、5：Face Detection

data_small：

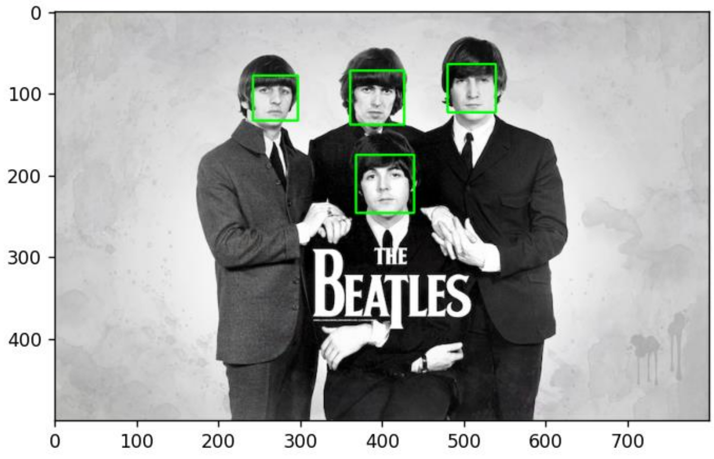
The result about "the-beatles.jpg", "p110912sh-0083.jpg" and our oen photo "test.jpg"of face detection with T =10.

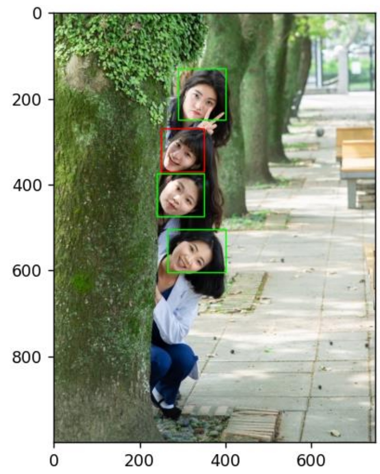




111550129 林彥亨

The result about "the-beatles.jpg", "p110912sh-0083.jpg" and our oen photo "test.jpg"of face detection with T = 2.

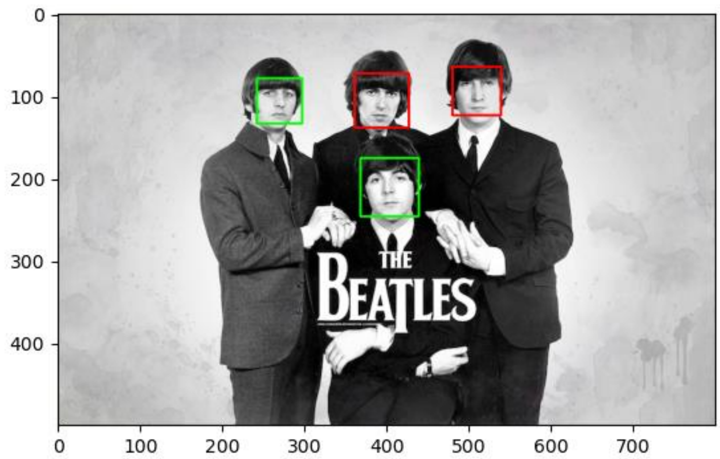


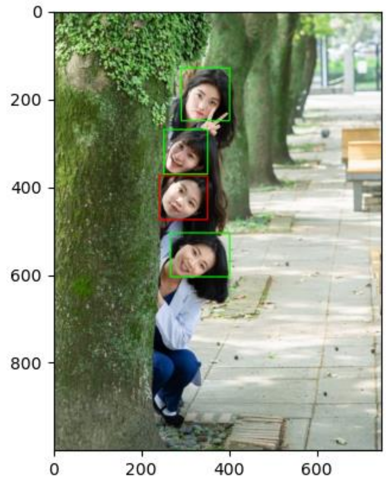


Due to the output, when T=2, the face detection has the higher accuracy.

data_FDDB：

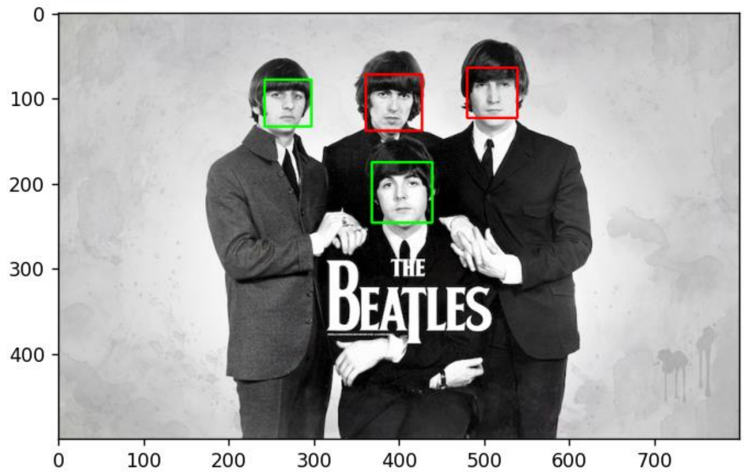
The result about “the-beatles.jpg”, “p110912sh-0083.jpg” and our oen photo “test.jpg”of face detection with T = 10.

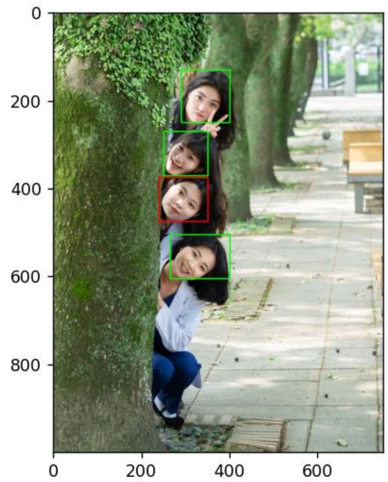




111550129 林彥亨

The result about "the-beatles.jpg", "p110912sh-0083.jpg" and our oen photo "test.jpg"of face detection with T = 5.





Due to the output, when T=5, the face detection has the higher accuracy.

111550129 林彥亨

Part III. Answer the questions：

1. ***Please describe a problem you encountered and how you solved it.***

A：

The problem I encountered is the practicing the Databoost algorithm. I never do the program about AI face detection before. So I spend a lot of time to understand the structure of the whole project. Whatsmore, I read some articles Adaboost, Viola Jones' algorithm on the Internet to understand the each part of these algorithm and know the whole process of the code.

2. ***How do you generate "nonface" data by cropping images？***

A：

For each detected face, it calculates a bounding box and crops the area from the image, resizing it to 19x19 pixels and labeling it as a face. Similarly, it generates non-face data by selecting random 19x19 pixel areas that do not overlap with any face regions, cropping these from the image, resizing, and labeling them as non-face. This process aims to create a balanced dataset, useful for training a face detection model.

3. ***What are the limitations of the Viola-Jones' algorithm***

A：

The algorithm is primarily designed for frontal face detection and may not perform as well on faces that are tilted, turned at a sharp angle, or partially obscured. And the training phase of the Viola-Jones algorithm can be computationally intensive and time-consuming, requiring a large number of positive and negative samples to effectively train the cascade classifiers.

4. ***Based on Viola-Jones' algorithm, how to improve the accuracy expect changing the training dataset and parameter T？***

A：

We can increase the size of the input image. In this way, we can avoid the mistake of detecting the pictures.

5. ***Other than Viola-Jones' algorithm, 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.***

A：

In my opinion, while CNN-based methods may offer superior accuracy and adaptability, they come with increased computational costs and complexity. And CNNs are most commonly applied to analyzing visual imagery. The network would learn to recognize various facial features and their configurations through multiple layers of processing.

The choice between a CNN approach and the Viola-Jones algorithm would depend on the specific requirements of the application, including the need for real-time processing, the availability of computational resources, and the desired level of accuracy.