HW1 Face Detection

0616098 黃秉茂

Part 1 : Load and prepare your dataset

Import module I need (os for path and numpy for data)

```
import os
import numpy as np
```

The function for checking if it is a pgm file

Chech the path is string or not, and check with extension(.pgm)

```
def is_pgm_file(path_file):
    if not os.path.isfile(path_file):
        return False
    if path_file is not str and not path_file.endswith('.pgm'):
        return False
    return True
```

The function that can help me to get image to numpy array from pgm file

check if it is a pgm file → Read file → check magic number to ensure the format is correct → get image's width and height → ensure the size of pixel value → get image content pixel value and store into a numpy array

magic number in pgm file is "P5P

first line in pgm file has magic number, second line is image's width and height, and the rest is pixel values in grayscale

use "with open" to read file and use readline to get content of the line

```
def get_image(path_file):
    # check if it is pgm file first
    if not is_pgm_file(path_file):
        raise Exception("%s is not a PGM file" % path_file)
    # read file
    with open(path_file, 'rb') as f:
        # magic number(the code in the beginning of pgm file) is often "P5"
        magic_number = f.readline().strip().decode('utf-8')
        if not magic_number == "P5":
            raise Exception("The wrong image")
        # get W and H with the pgm format
        width, height = f.readline().strip().decode('utf-8').split(' ')
        width = int(width)
        height = int(height)
        maxval = f.readline().strip()
        # the image pixel value may store than a byte
        if int(maxval) < 256:
            one_reading = 1
        else:
            one_reading = 2
        # read the content to the numpy array
        image = np.zeros((height, width))
        image[:][:] = [[ord(f.read(one_reading)) for j in range(width)] for i in range(height)]
        return image
```

folder: “non-face” → label = 0, “face” → label = 1

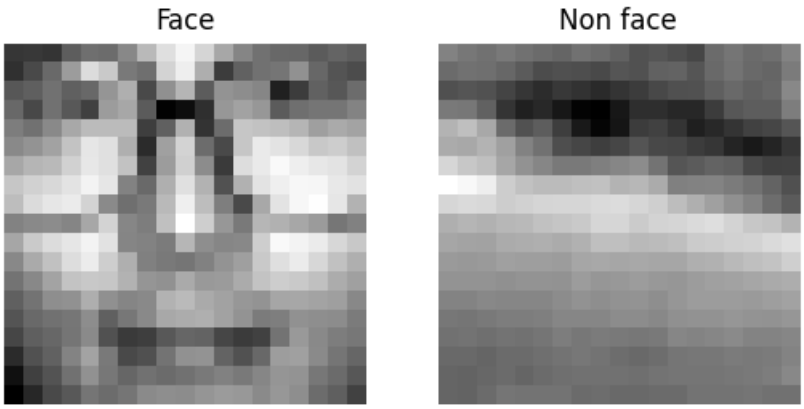
travel each folder with os.listdir() and generate image path with “os.path.join(.)”

call the function I wrote above to get image to numpy array from pgm file

the image and label are appended into the list

```
# Begin your code (Part 1)
# raise NotImplementedError("To be implemented")
dataset = []
directory = os.listdir(dataPath)
for d in directory:
    # non-face
    if 'non' in d:
        label = 0
    # face
    else:
        label = 1
    path = os.path.join(dataPath, d)
    for f in os.listdir(path):
        file = os.path.join(path, f)
        image = get_image(file)
        dataset.append((image, label))
# End your code (Part 1)
return dataset
```

Appear the follow picture means successed (face, non-face)



Part 2: Implement Adaboost algorithm

Initialize

create weak classifier with each feature

use iis to get the result the classifier predict

use absolute value to be error evaluation and multiply the L1-norm by weights

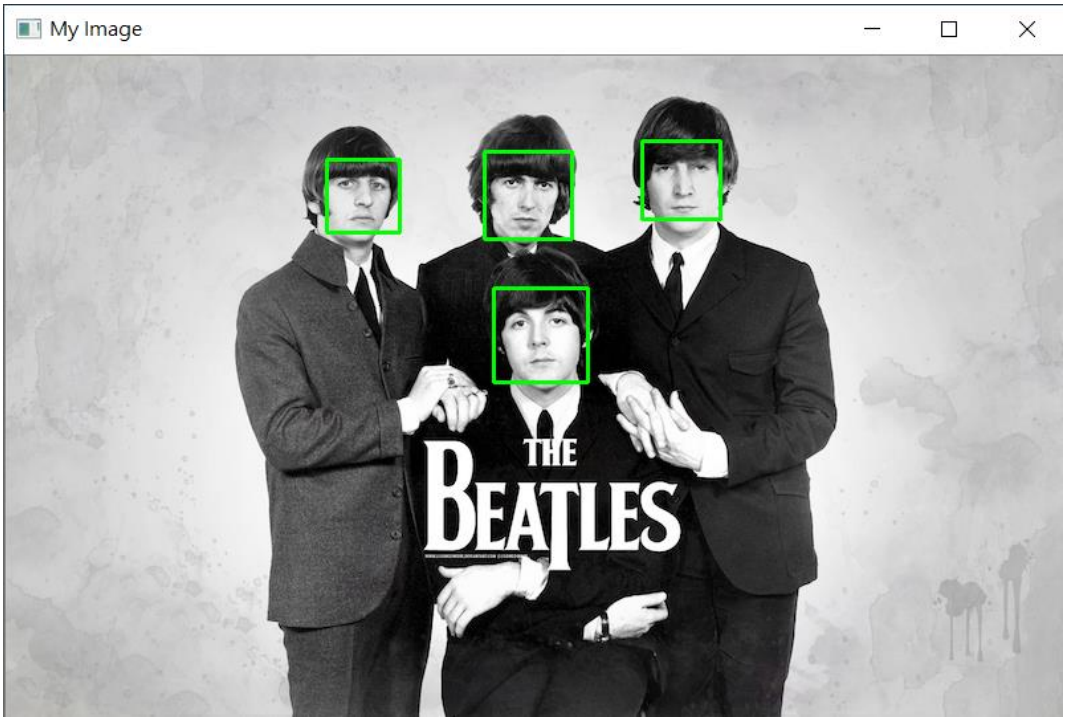
choose the minimum error as the best error and choose weak classifier according to the error

```
# Begin your code (Part 2)
# raise NotImplementedError("To be implemented")
# initialize
bestClf = WeakClassifier(features[0])
bestError = np.inf
# create weak classifier with each feature
for i in range(len(features)):
    WC = WeakClassifier(features[i])
    h_method = 1
    # homework method
    if h_method == 1:
        h = np.zeros(np.shape(labels))
        for j in range(len(labels)):
            h[j] = WC.classify(iis[j])
        # error is L1-norm
        error = np.sum(np.abs(h - labels) * weights)
    # my method
    else:
        # sigmoid
        h = 1 - 1 / (1 + np.exp(featureVals[i] * -1))
        # avoid log(0)
        h = np.clip(h, 0.00001, 0.99999)
        # binary cross entropy
        labels = np.array(labels)
        BCE = (labels * np.log(h) + (1 - labels) * np.log(1 - h)) * -1 / len(labels)
        # error is L2-norm and binary cross entropy
        error = np.sum((h - labels) ** 2 * weights + BCE * weights)
    if error < bestError:
        bestError = error
        bestClf = WC
# fix error of my method
if h_method != 1:
    h = np.zeros(np.shape(labels))
    for j in range(len(labels)):
        h[j] = bestClf.classify(iis[j])
    # error is L1-norm
    bestError = np.sum(np.abs(h - labels) * weights)
# End your code (Part 2)
return bestClf, bestError
```

Part 3: Additional experiments

T = 1

```
C:\WINDOWS\system32\cmd.exe - python main.py
C:\Users\user\Desktop\HW1-code>python main.py
Loading images
The number of training samples loaded: 200
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
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(8, 0, 1, 3), RectangleRegion(7, 3, 1, 3)], ne
ns=[RectangleRegion(7, 0, 1, 3), RectangleRegion(8, 3, 1, 3)]) with accuracy: 162.000000 and alpha: 1.450010

Evaluate your classifier with training dataset
False Positive Rate: 28/100 (0.280000)
False Negative Rate: 10/100 (0.100000)
Accuracy: 162/200 (0.810000)

Evaluate your classifier with test dataset
False Positive Rate: 49/100 (0.490000)
False Negative Rate: 55/100 (0.550000)
Accuracy: 96/200 (0.480000)

Detect faces at the assigned lacation using your classifier
```



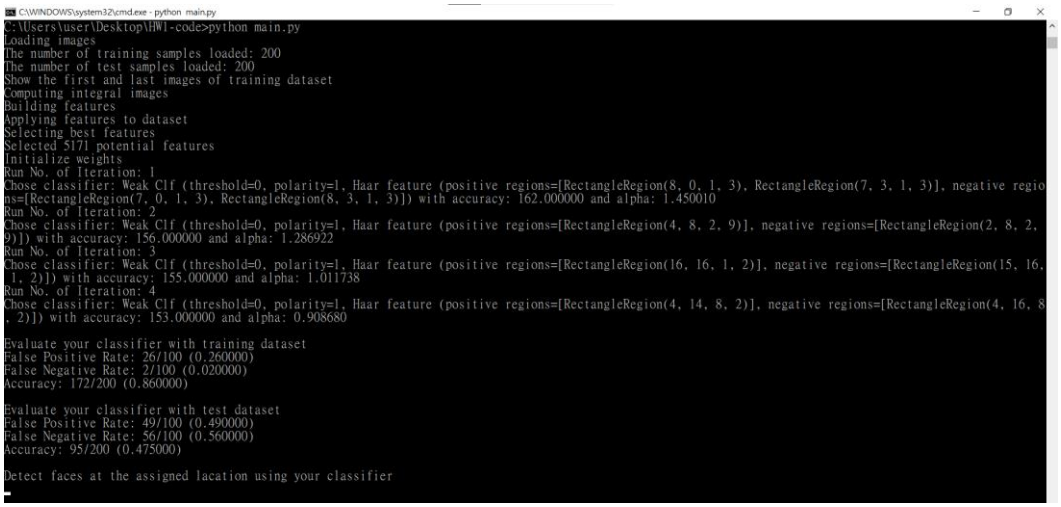
T = 2

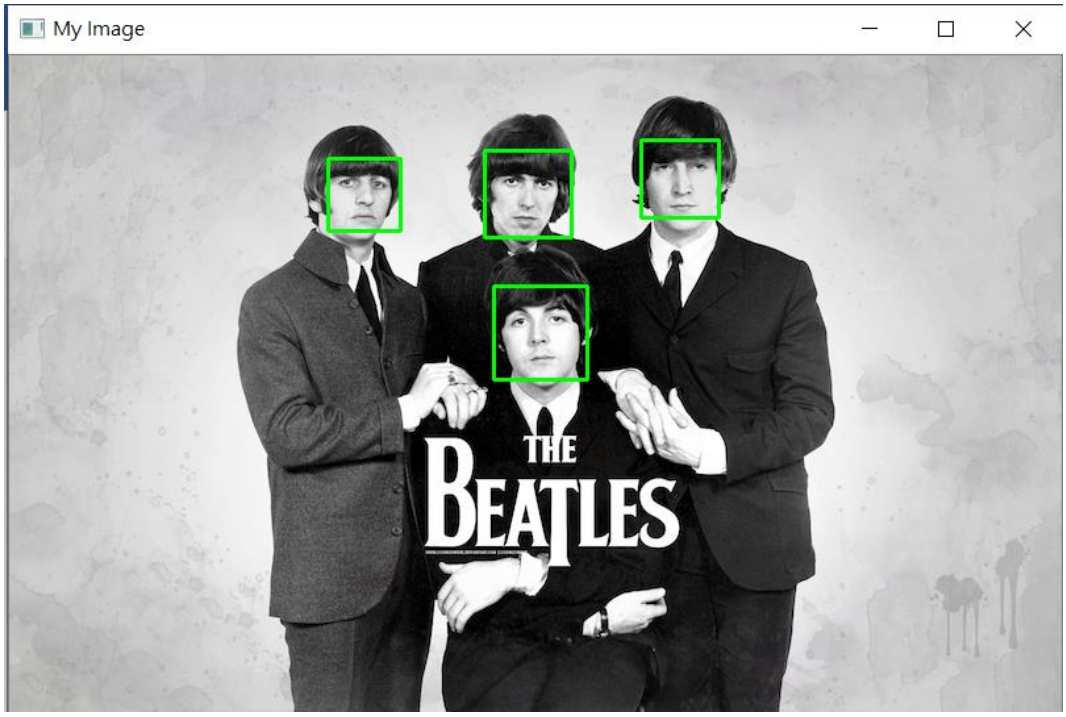
```
C:\WINDOWS\system32\cmd.exe - python  main.py
C:\Users\user\Desktop\HW1-code>python main.py
Loading images
The number of training samples loaded: 200
The number of test samples loaded: 200
Show the first and last images of training dataset
Computing integral images
Building features
Applying features to dataset
Selecting best features
Selected 5171 potential features
Initialize weights
Run No. of Iteration: 1
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(8, 0, 1, 3), RectangleRegion(7, 3, 1, 3)], ne
ns=[RectangleRegion(7, 0, 1, 3), RectangleRegion(8, 3, 1, 3)]) with accuracy: 162.000000 and alpha: 1.450010
Run No. of Iteration: 2
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(4, 8, 2, 9)], negative regions=[RectangleRegi
9)]) with accuracy: 156.000000 and alpha: 1.286922

Evaluate your classifier with training dataset
False Positive Rate: 28/100 (0.280000)
False Negative Rate: 10/100 (0.100000)
Accuracy: 162/200 (0.810000)

Evaluate your classifier with test dataset
False Positive Rate: 49/100 (0.490000)
False Negative Rate: 55/100 (0.550000)
Accuracy: 96/200 (0.480000)

Detect faces at the assigned lacation using your classifier
```

T = 3

```
C:\WINDOWS\system32\cmd.exe - python main.py

C:\Users\user\Desktop\HW1-code>python main.py
Loading images
The number of training samples loaded: 200
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
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(8, 0, 1, 3), RectangleRegion(7, 3, 1, 3)], negative regio
ns=[RectangleRegion(7, 0, 1, 3), RectangleRegion(8, 3, 1, 3)]) with accuracy: 162.000000 and alpha: 1.450010
Run No. of Iteration: 2
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
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

Evaluate your classifier with training dataset
False Positive Rate: 23/100 (0.230000)
False Negative Rate: 1/100 (0.010000)
Accuracy: 176/200 (0.880000)

Evaluate your classifier with test dataset
False Positive Rate: 48/100 (0.480000)
False Negative Rate: 46/100 (0.460000)
Accuracy: 106/200 (0.530000)

Detect faces at the assigned lacation using your classifier
```

T = 4

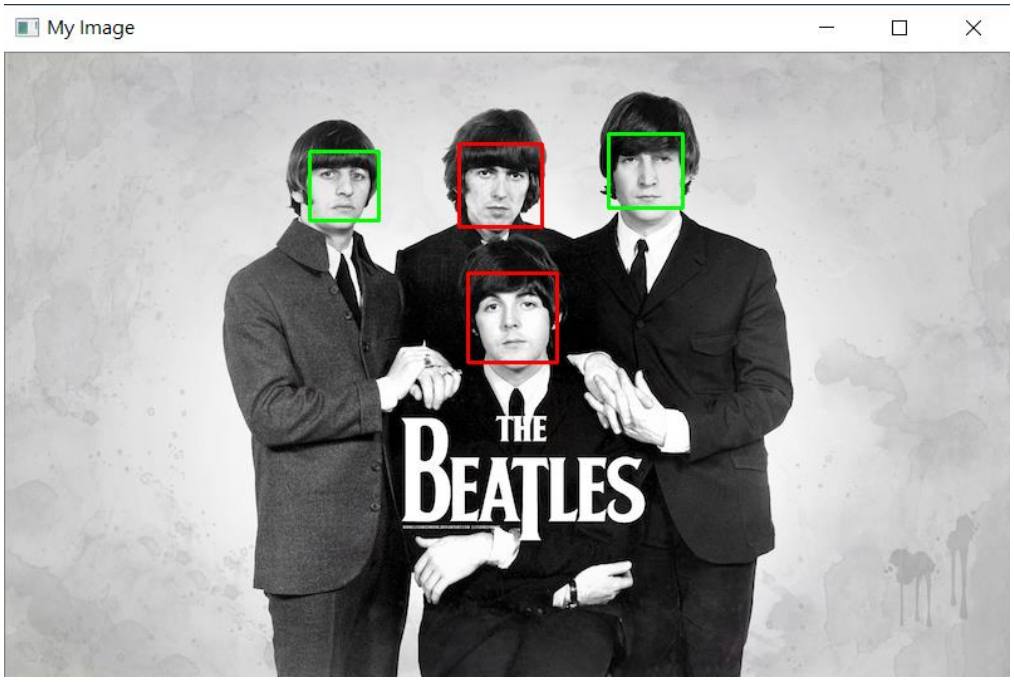

C:\WINDOWS\system32\cmd.exe - python main.py
C:\Users\user\Desktop\HW1-code>python main.py
Loading images
The number of training samples loaded: 200
The number of test samples loaded: 200
Show the first and last images of training dataset
Computing integral images
Building features
Applying features to dataset
Selecting best features
Selected 5171 potential features
Initialize weights
Run No. of Iteration: 1
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(8, 0, 1, 3), RectangleRegion(7, 3, 1, 3)], negative regio
ns=[RectangleRegion(7, 0, 1, 3), RectangleRegion(8, 3, 1, 3)]) with accuracy: 162.000000 and alpha: 1.450010
Run No. of Iteration: 2
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
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
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
Evaluate your classifier with training dataset
False Positive Rate: 26/100 (0.260000)
False Negative Rate: 2/100 (0.020000)
Accuracy: 172/200 (0.860000)
Evaluate your classifier with test dataset
False Positive Rate: 49/100 (0.490000)
False Negative Rate: 56/100 (0.560000)
Accuracy: 95/200 (0.475000)
Detect faces at the assigned lacation using your classifier


My Image
THE
BEATLES


My Image

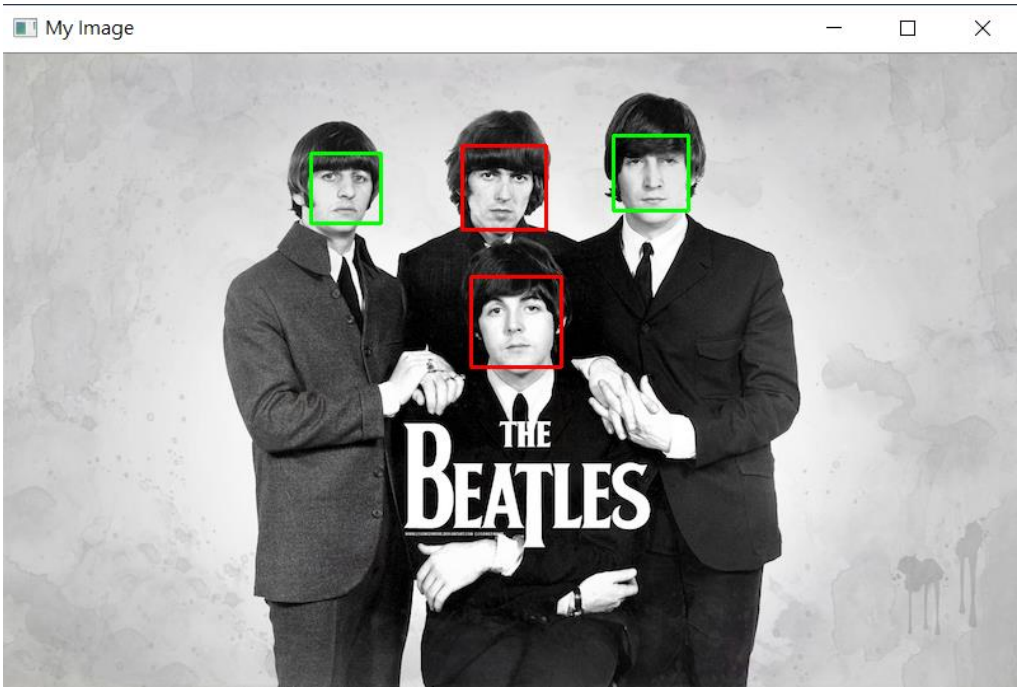
T = 5

```
C:\Users\user\Desktop\HW1-code>python main.py
Loading images
The number of training samples loaded: 200
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
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(8, 0, 1, 3), RectangleRegion(7, 3, 1, 3)], negative regio
, RectangleRegion(8, 3, 1, 3)]) with accuracy: 162.000000 and alpha: 1.450010
Run No. of Iteration: 2
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(4, 8, 2, 9)], negative regions=[RectangleRegion(2, 8, 2,
and alpha: 1.286922
Run No. of Iteration: 3
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(16, 16, 1, 2)], negative regions=[RectangleRegion(15, 16,
000 and alpha: 1.011738
Run No. of Iteration: 4
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(4, 14, 8, 2)], negative regions=[RectangleRegion(4, 16, 8
0 and alpha: 0.908680
Run No. of Iteration: 5
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(10, 8, 1, 1)], negative regions=[RectangleRegion(9, 8, 1,
 and alpha: 0.924202

Evaluate your classifier with training dataset
False Positive Rate: 23/100 (0.230000)
False Negative Rate: 0/100 (0.000000)
Accuracy: 177/200 (0.885000)

Evaluate your classifier with test dataset
False Positive Rate: 49/100 (0.490000)
False Negative Rate: 43/100 (0.430000)
Accuracy: 108/200 (0.540000)

Detect faces at the assigned lacation using your classifier
```



T = 6

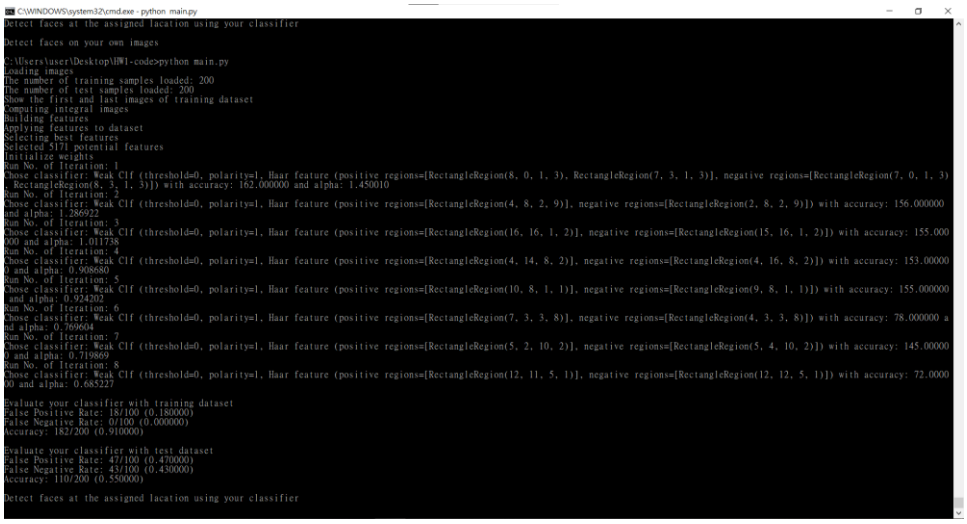
```
C:\Users\user\Desktop\HW1-code>python main.py
Loading images
The number of training samples loaded: 200
The number of test samples loaded: 200
Show the first and last images of training dataset
Computing integral images
Building features
Applying features to dataset
Selecting best features
Selected 5171 potential features
Initialize weights
Run No. of Iteration: 1
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
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
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: 155.000
000 and alpha: 1.011738
Run No. of Iteration: 4
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: 153.00000
0 and alpha: 0.908680
Run No. of Iteration: 5
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
Run No. of Iteration: 6
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(7, 3, 3, 8)], negative regions=[RectangleRegion(4, 3, 3, 8)]) with accuracy: 78.000000 a
nd alpha: 0.769604

Evaluate your classifier with training dataset
False Positive Rate: 22/100 (0.220000)
False Negative Rate: 0/100 (0.000000)
Accuracy: 178/200 (0.890000)

Evaluate your classifier with test dataset
False Positive Rate: 50/100 (0.500000)
False Negative Rate: 48/100 (0.480000)
Accuracy: 102/200 (0.510000)

Detect faces at the assigned location using your classifier
```



T = 7

```
C:\Users\user\Desktop\HW1-code>python main.py
Loading images
The number of training samples loaded: 200
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
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
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
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: 155.000
000 and alpha: 1.011735
Run No. of Iteration: 4
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: 153.00000
0 and alpha: 0.908680
Run No. of Iteration: 5
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
Run No. of Iteration: 6
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(7, 3, 3, 8)], negative regions=[RectangleRegion(4, 3, 3, 8)]) with accuracy: 78.000000 a
nd alpha: 0.769604
Run No. of Iteration: 7
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(5, 2, 10, 2)], negative regions=[RectangleRegion(5, 4, 10, 2)]) with accuracy: 145.00000
0 and alpha: 0.719869

Evaluate your classifier with training dataset
False Positive Rate: 20/100 (0.200000)
False Negative Rate: 0/100 (0.000000)
Accuracy: 180/200 (0.900000)

Evaluate your classifier with test dataset
False Positive Rate: 52/100 (0.520000)
False Negative Rate: 39/100 (0.390000)
Accuracy: 109/200 (0.545000)

Detect faces at the assigned location using your classifier
```

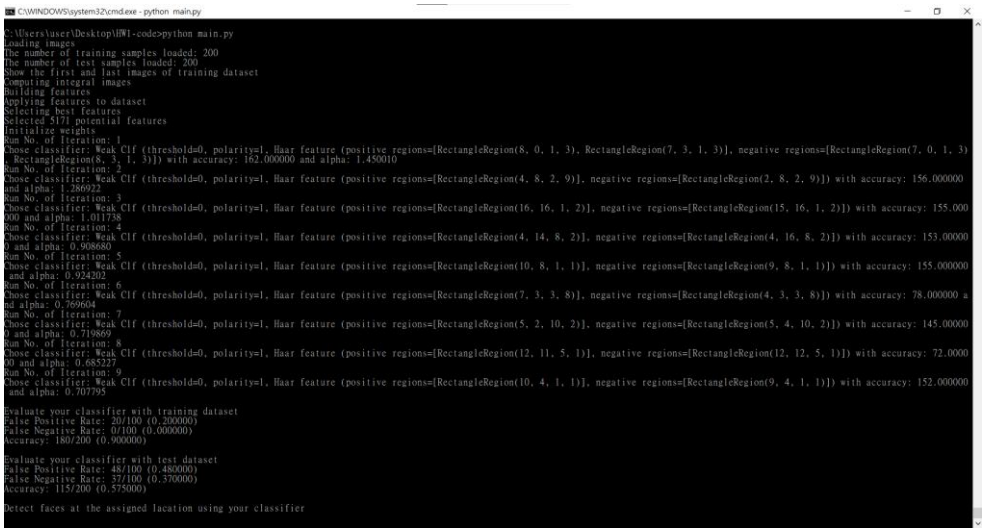
T = 8

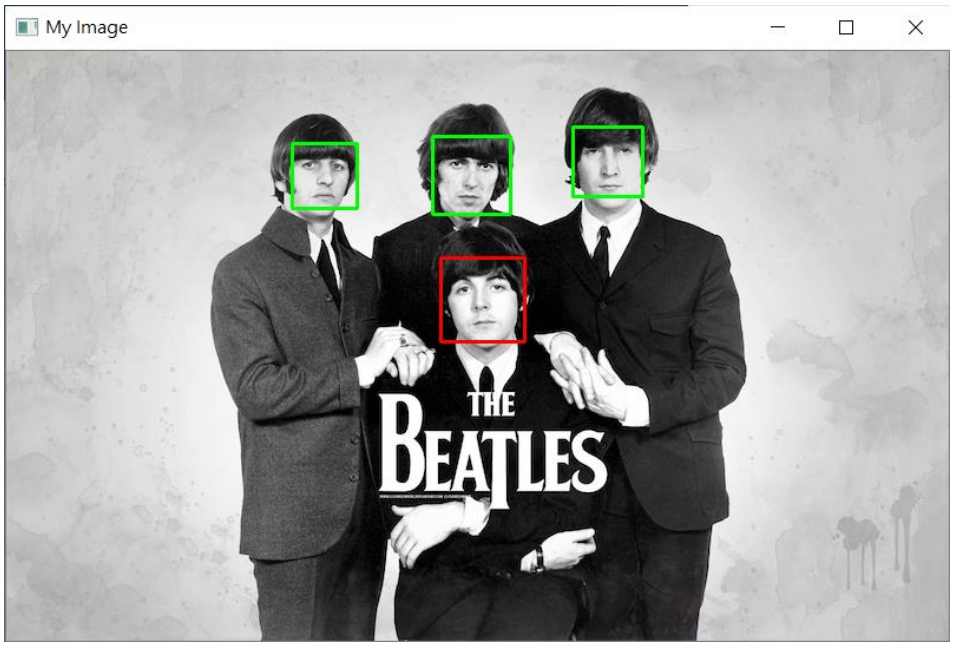

C:\WINDOWS\system32\cmd.exe - python main.py
Detect faces at the assigned location using your classifier
Detect faces on your own images
C:\Users\user\Desktop\HW1-code>python main.py
Loading images
The number of training samples loaded: 200
The number of test samples loaded: 200
Show the first and last 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
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
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
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
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
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
Run No. of Iteration: 6
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(7, 3, 3, 8)], negative regions=[RectangleRegion(4, 3, 3, 8)]) with accuracy: 78.000000 and alpha: 0.769604
Run No. of Iteration: 7
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(5, 2, 10, 2)], negative regions=[RectangleRegion(5, 4, 10, 2)]) with accuracy: 145.000000 and alpha: 0.719869
Run No. of Iteration: 8
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(12, 11, 5, 1)], negative regions=[RectangleRegion(12, 12, 5, 1)]) with accuracy: 72.000000 and alpha: 0.685227
Evaluate your classifier with training dataset
False Positive Rate: 18/100 (0.180000)
False Negative Rate: 0/100 (0.000000)
Accuracy: 182/200 (0.910000)
Evaluate your classifier with test dataset
False Positive Rate: 47/100 (0.470000)
False Negative Rate: 43/100 (0.430000)
Accuracy: 110/200 (0.550000)
Detect faces at the assigned location using your classifier

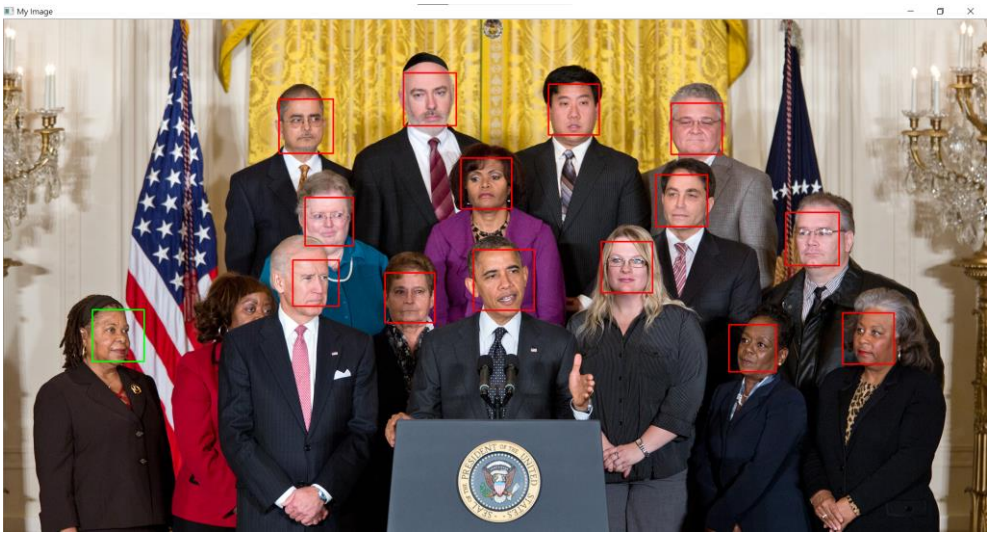

My Image
THE
BEATLES


My Image

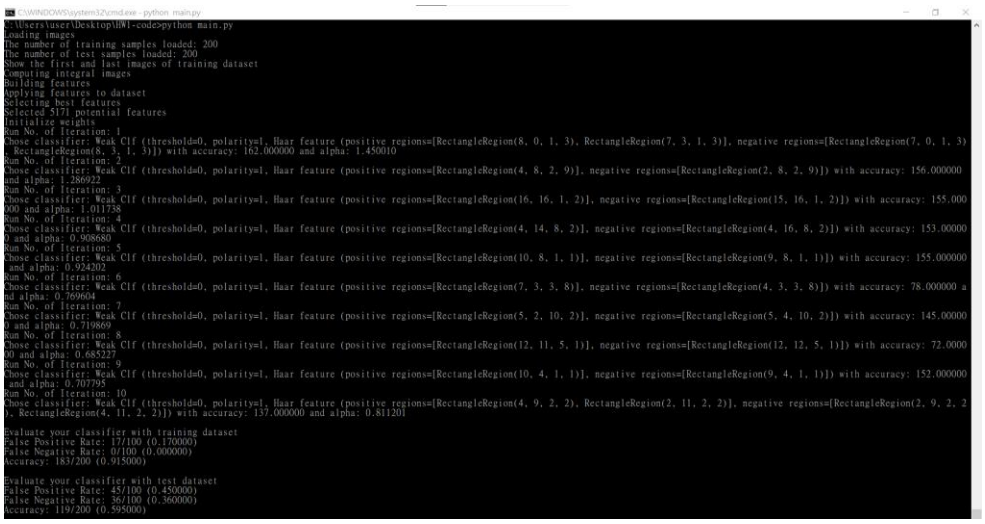
T = 9


C:\WINDOWS\system32\cmd.exe - python main.py
C:\Users\user\Desktop\HW1-code>python main.py
Loading images
The number of training samples loaded: 200
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
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
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(4, 8, 2, 9)], negative regionx=[RectangleRegion(2, 8, 2, 9)]) with accuracy: 156.000000 and alpha: 1.286922
Run No. of Iteration: 3
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
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
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
Run No. of Iteration: 6
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(7, 3, 3, 8)], negative regions=[RectangleRegion(4, 3, 3, 8)]) with accuracy: 78.000000 and alpha: 0.769604
Run No. of Iteration: 7
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(5, 2, 10, 2)], negative regions=[RectangleRegion(5, 4, 10, 2)]) with accuracy: 145.000000 and alpha: 0.710869
Run No. of Iteration: 8
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(12, 11, 5, 1)], negative regions=[RectangleRegion(12, 12, 5, 1)]) with accuracy: 72.000000 and alpha: 0.685327
Run No. of Iteration: 9
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(10, 4, 1, 1)], negative regions=[RectangleRegion(9, 4, 1, 1)]) with accuracy: 152.000000 and alpha: 0.707795
Evaluate your classifier with training dataset
False Positive Rate: 20/100 (0.200000)
False Negative Rate: 0/100 (0.000000)
Accuracy: 180/200 (0.900000)
Evaluate your classifier with test dataset
False Positive Rate: 48/100 (0.480000)
False Negative Rate: 37/100 (0.370000)
Accuracy: 115/200 (0.575000)
Detect faces at the assigned location using your classifier

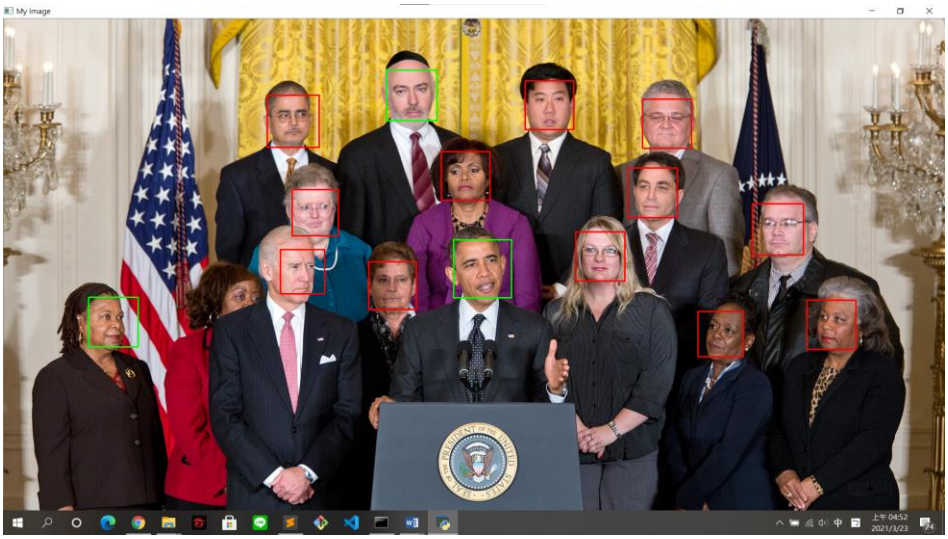

My Image
THE
BEATLES


My Image

T = 10


C:\WINDOWS\system32\cmd.exe - python main.py
C:\Users\user\Desktop\HW1-code>python main.py
Loading images
The number of training samples loaded: 200
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
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
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
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
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
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
Run No. of Iteration: 6
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(7, 3, 3, 8)], negative regions=[RectangleRegion(4, 3, 3, 8)]) with accuracy: 78.000000 and alpha: 0.769604
Run No. of Iteration: 7
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(5, 2, 10, 2)], negative regions=[RectangleRegion(5, 4, 10, 2)]) with accuracy: 145.000000 and alpha: 0.719869
Run No. of Iteration: 8
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(12, 11, 5, 1)], negative regions=[RectangleRegion(12, 12, 5, 1)]) with accuracy: 72.000000 and alpha: 0.685227
Run No. of Iteration: 9
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(10, 4, 1, 1)], negative regions=[RectangleRegion(9, 4, 1, 1)]) with accuracy: 152.000000 and alpha: 0.707795
Run No. of Iteration: 10
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(4, 9, 2, 2), RectangleRegion(2, 11, 2, 2)], negative regions=[RectangleRegion(2, 9, 2, 2), RectangleRegion(4, 11, 2, 2)]) with accuracy: 137.000000 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)


My Image
THE
BEATLES


My Image

train

T	1	2	3	4	5	6	7	8	9	10
FPR(%)	28	28	23	26	23	22	20	18	20	17
FNR(%)	10	10	1	2	0	0	0	0	0	0
acc(%)	81	81	88	86	88.5	89	90	91	90	91.5

test

T	1	2	3	4	5	6	7	8	9	10
FPR(%)	49	49	48	49	49	50	52	47	48	45
FNR(%)	55	55	46	56	43	48	39	43	37	36
acc(%)	48	48	53	47.5	54	51	54.5	55	57.5	59.5

Accuracy increase with the growth of T, the phenomenon happens not only on training but also testing.

FNR decrease to 0 when T is bigger than 4. It means face won't be classified as non-face in training.

The accuracy of training and that of testing differ a lot, they are almost 30% different. It implies that the classifier is sensitive to the dataset.

We should continue to train the model because training accuracy and testing accuracy are still tend to grow up.

Part 4: Detect face

Import module I need

```
import os
import cv2
```

open the file and store the data into the relative lists

use “with open” and “readlines” traversal each lines

2 data → filename and # faces 4 data → x, y, w, h

Data procceeing: get each face in the image

```
# Begin your code (Part 4)
# raise NotImplementedError("To be implemented")
detection_path = os.path.split(dataPath)[0]
n_face = []
image = []
face_loc = []
# open the file and store the data into lists
with open(dataPath, 'rb') as f:
    for line in f.readlines():
        img_info = line.decode('utf-8').split()
        # filename, n_face
        if len(img_info) == 2:
            filename = img_info[0]
            n_face.append(int(img_info[-1]))
            image.append(cv2.imread(os.path.join(detection_path, filename)))
        # x, y, w, h
        elif len(img_info) == 4:
            face_loc.append([int(img_info[0]), int(img_info[1]), int(img_info[2]), int(im
face_id = 0
for image_id in range(len(image)):
    img = image[image_id]
    num_face = n_face[image_id]
    for _ in range(num_face):
        x, y, w, h = face_loc[face_id]
        face_img = img[y : y + h, x : x + w]
```

Get face location and risize the face image to 19 * 19

Convert face image to grayscale and than classify with the model

Predict with clf.classify

Draw the rectangle on the image and its color is according to the prediction is correct or not, and then show the result on the image.

Shrink the image which is too big too show

```
    for _ in range(num_face):
        x, y, w, h = face_loc[face_id]
        face_img = img[y : y + h, x : x + w]
        # resize to 19 * 19
        face_img = cv2.resize(face_img, (19, 19))
        # BGR --> GRAY
        face_img = cv2.cvtColor(face_img, cv2.COLOR_BGR2GRAY)
        # predict with the model
        prediction = clf.classify(face_img)
        # face --> right --> green
        if prediction == 1:
            color = (0, 255, 0)
        # non-face --> wrong --> red
        else:
            color = (0, 0, 255)
        # draw the rectangle on the image
        cv2.rectangle(img, (x, y), (x + w, y + h), color, 2)
        face_id += 1
    height, width, channels = img.shape
    # shrink the image if it is too large to show
    if height > 2000:
        img = cv2.resize(img, None, fx=0.5, fy=0.5)
    cv2.imshow('My Image', img)
    cv2.waitKey(0)
    cv2.destroyAllWindows()
# End your code (Part 4)
```

Example of the image with the predict result
(green: classify as face, red: classify as non-face)

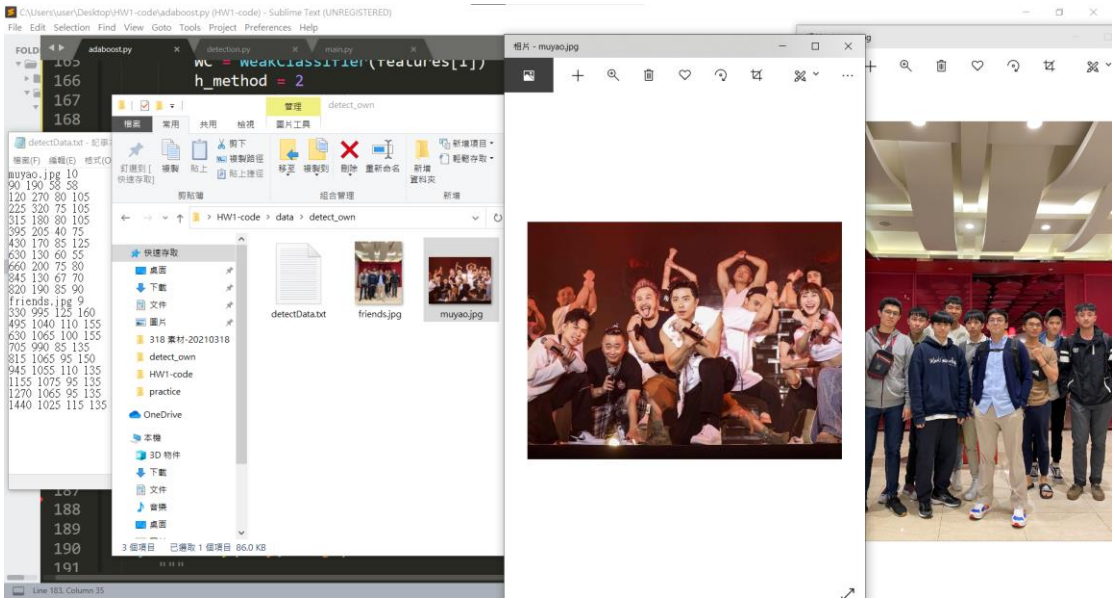
T = 2

Part 5: Test classifier on your own images

My image path: data/detect_own

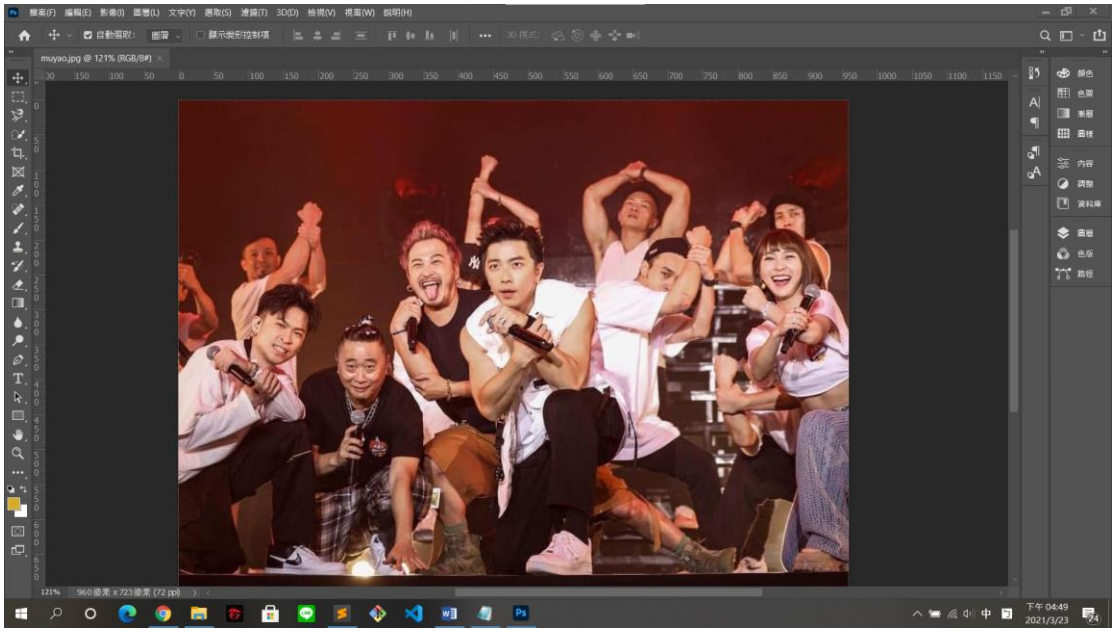
```
# Part 5: Test classifier on your own images
print('\nDetect faces on your own images')
detection.detect('data/detect_own/detectData.txt', clf)
```

create a text file with the same format



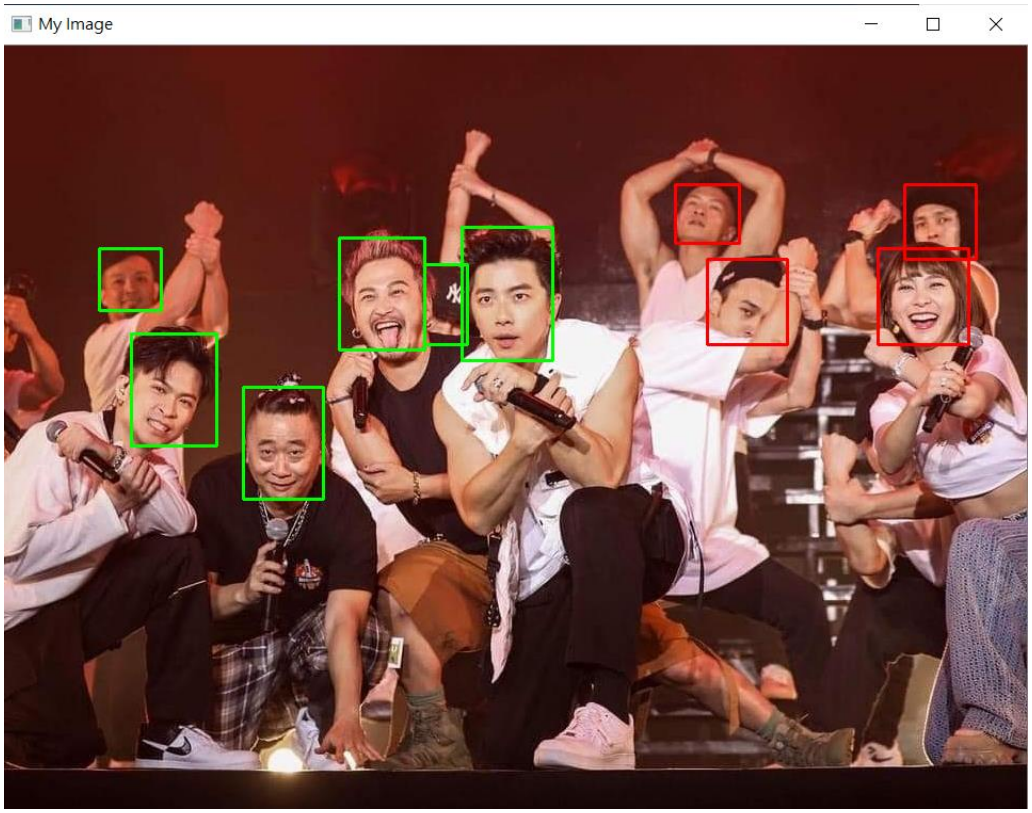
record the face location with photoshop, since PS can change the unit to pixels and enable you find the location in pixel conveniently.

handcraft labelling



Example of the image with the predict result

(green: classify as face, red: classify as non-face)



Part 6: Implement another classifier

Sigmoid + clip + binary cross entropy for binary classification

Sigmoid: 1 / (1 + e ^ x), other method for 0, 1 binary classification

[-inf, inf] → [0, 1]

1 - sigmoid since large number is predict to be 0, and negative number is predict to be 1

Clip h to avoid log(0)

binary cross entropy: - (1 / n) * sigma(y * log(f(x)) + (1 - y) * log(1 – f(x)))

other error method for binary classification

Knowing the best classifier, fix the best error to the adaboost need

Combinate L2-norm and BCE to calculate error and choose the best classifier with the comprehensive error

Fix the error that can be used in adaboost

```
# Begin your code (Part 2)
# raise NotImplementedError("To be implemented")
# initialize
bestClf = WeakClassifier(features[0])
bestError = np.inf
# create weak classifier with each feature
for i in range(len(features)):
    WC = WeakClassifier(features[i])
    h_method = 2
    # homework method
    if h_method == 1:
        h = np.zeros(np.shape(labels))
        for j in range(len(labels)):
            h[j] = WC.classify(iis[j])
        # error is L1-norm
        error = np.sum(np.abs(h - labels) * weights)
    # my method
    else:
        # sigmoid
        h = 1 - 1 / (1 + np.exp(featureVals[i] * -1))
        # avoid log(0)
        h = np.clip(h, 0.00001, 0.99999)
        # binary cross entropy
        labels = np.array(labels)
        BCE = (labels * np.log(h) + (1 - labels) * np.log(1 - h)) * -1 / len(labels)
        # error is L2-norm and binary cross entropy
        error = np.sum((h - labels) ** 2 * weights + BCE * weights)
    if error < bestError:
        bestError = error
        bestClf = WC
# fix error of my method
if h_method != 1:
    h = np.zeros(np.shape(labels))
    for j in range(len(labels)):
        h[j] = bestClf.classify(iis[j])
    # error is L1-norm
    bestError = np.sum(np.abs(h - labels) * weights)
# End your code (Part 2)
return bestClf, bestError
```

T = 1

```
C:\Users\user\Desktop\HW1-code>python main.py
Loading images
The number of training samples loaded: 200
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
C:\Users\user\Desktop\HW1-code\adaboost.py:177: RuntimeWarning: overflow encountered in exp
  h = 1 - 1 / (1 + np.exp(featureVals[i] * -1))
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(8, 0, 1, 3), RectangleRegion(7, 3, 1, 3)], ne
gative regions=[RectangleRegion(7, 0, 1, 3), RectangleRegion(8, 3, 1, 3)]) with accuracy: 162.000000 and alpha: 1.450010

Evaluate your classifier with training dataset
False Positive Rate: 28/100 (0.280000)
False Negative Rate: 10/100 (0.100000)
Accuracy: 162/200 (0.810000)

Evaluate your classifier with test dataset
False Positive Rate: 49/100 (0.490000)
False Negative Rate: 55/100 (0.550000)
Accuracy: 96/200 (0.480000)

Detect faces at the assigned lacation using your classifier

Detect faces on your own images

C:\Users\user\Desktop\HW1-code>
```

T = 2

```
C:\Users\user\Desktop\HW1-code>python main.py
Loading images
The number of training samples loaded: 200
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
C:\Users\user\Desktop\HW1-code\adaboost.py:177: RuntimeWarning: overflow encountered in exp
  h = 1 - 1 / (1 + np.exp(featureVals[i] * -1))
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(8, 0, 1, 3), RectangleRegion(7, 3, 1, 3)], ne
, 1, 3)]) with accuracy: 162.000000 and alpha: 1.450010
Run No. of Iteration: 2
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(4, 8, 2, 9)], negative regions=[RectangleRegi

Evaluate your classifier with training dataset
False Positive Rate: 28/100 (0.280000)
False Negative Rate: 10/100 (0.100000)
Accuracy: 162/200 (0.810000)

Evaluate your classifier with test dataset
False Positive Rate: 49/100 (0.490000)
False Negative Rate: 55/100 (0.550000)
Accuracy: 96/200 (0.480000)

Detect faces at the assigned lacation using your classifier

Detect faces on your own images

C:\Users\user\Desktop\HW1-code>
```

T = 3

```
C:\Users\user\Desktop\HW1-code>python main.py
Loading images
The number of training samples loaded: 200
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
C:\Users\user\Desktop\HW1-code\adaboost.py:177: RuntimeWarning: overflow encountered in exp
  h = 1 - 1 / (1 + np.exp(featureVals[i] * -1))
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(8, 0, 1, 3), RectangleRegion(7, 3, 1, 3)], negative regio
, 1, 3)]) with accuracy: 162.000000 and alpha: 1.450010
Run No. of Iteration: 2
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(4, 8, 2, 9)], negative regions=[RectangleRegion(2, 8, 2,
Run No. of Iteration: 3
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(16, 16, 1, 2)], negative regions=[RectangleRegion(15, 16,
8

Evaluate your classifier with training dataset
False Positive Rate: 23/100 (0.230000)
False Negative Rate: 1/100 (0.010000)
Accuracy: 176/200 (0.880000)

Evaluate your classifier with test dataset
False Positive Rate: 48/100 (0.480000)
False Negative Rate: 46/100 (0.460000)
Accuracy: 106/200 (0.530000)

Detect faces at the assigned lacation using your classifier

Detect faces on your own images

C:\Users\user\Desktop\HW1-code>
```

T = 4

```
C:\Users\user\Desktop\HW1-code>python main.py
Loading images
The number of training samples loaded: 200
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
C:\Users\user\Desktop\HW1-code\adaboost.py:177: RuntimeWarning: overflow encountered in exp
  h = 1 - 1 / (1 + np.exp(featureVals[i] * -1))
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(8, 0, 1, 3), RectangleRegion(7, 3, 1, 3)], negative regio
, 1, 3)]) with accuracy: 162.000000 and alpha: 1.450010
Run No. of Iteration: 2
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(4, 8, 2, 9)], negative regions=[RectangleRegion(2, 8, 2,
Run No. of Iteration: 3
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(16, 16, 1, 2)], negative regions=[RectangleRegion(15, 16,
8
Run No. of Iteration: 4
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(10, 3, 1, 8)], negative regions=[RectangleRegion(9, 3, 1,

Evaluate your classifier with training dataset
False Positive Rate: 28/100 (0.280000)
False Negative Rate: 3/100 (0.030000)
Accuracy: 169/200 (0.845000)

Evaluate your classifier with test dataset
False Positive Rate: 48/100 (0.480000)
False Negative Rate: 45/100 (0.450000)
Accuracy: 107/200 (0.535000)

Detect faces at the assigned lacation using your classifier

Detect faces on your own images

C:\Users\user\Desktop\HW1-code>
```

T = 5

```
C:\Users\user\Desktop\HW1-code>python main.py
Loading images
The number of training samples loaded: 200
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
C:\Users\user\Desktop\HW1-code\adaboost.py:177: RuntimeWarning: overflow encountered in exp
  h = 1 - 1 / (1 + np.exp(featureVals[i] * -1))
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(8, 0, 1, 3), RectangleRegion(7, 3, 1, 3)], negative regio
, 1, 3)]) with accuracy: 162.000000 and alpha: 1.450010
Run No. of Iteration: 2
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(4, 8, 2, 9)], negative regions=[RectangleRegion(2, 8, 2,
Run No. of Iteration: 3
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(16, 16, 1, 2)], negative regions=[RectangleRegion(15, 16,
8
Run No. of Iteration: 4
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(10, 3, 1, 8)], negative regions=[RectangleRegion(9, 3, 1,
Run No. of Iteration: 5
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(7, 1, 9, 4)], negative regions=[RectangleRegion(7, 5, 9,

Evaluate your classifier with training dataset
False Positive Rate: 24/100 (0.240000)
False Negative Rate: 1/100 (0.010000)
Accuracy: 175/200 (0.875000)

Evaluate your classifier with test dataset
False Positive Rate: 49/100 (0.490000)
False Negative Rate: 18/100 (0.180000)
Accuracy: 133/200 (0.665000)

Detect faces at the assigned lacation using your classifier

Detect faces on your own images

C:\Users\user\Desktop\HW1-code>
```

T = 6

```
C:\Users\user\Desktop\HW1-code>python main.py
Loading images
The number of training samples loaded: 200
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
C:\Users\user\Desktop\HW1-code\adaboost.py:177: RuntimeWarning: overflow encountered in exp
  h = 1 - 1 / (1 + np.exp(featureVals[i] * -1))
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(8, 0, 1, 3), RectangleRegion(7, 3, 1, 3)], negative regio
, 1, 3)]) with accuracy: 162.000000 and alpha: 1.450010
Run No. of Iteration: 2
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(4, 8, 2, 9)], negative regions=[RectangleRegion(2, 8, 2,
Run No. of Iteration: 3
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(16, 16, 1, 2)], negative regions=[RectangleRegion(15, 16,
8
Run No. of Iteration: 4
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(10, 3, 1, 8)], negative regions=[RectangleRegion(9, 3, 1,
Run No. of Iteration: 5
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(7, 1, 9, 4)], negative regions=[RectangleRegion(7, 5, 9,
Run No. of Iteration: 6
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(7, 3, 2, 10)], negative regions=[RectangleRegion(5, 3, 2,

Evaluate your classifier with training dataset
False Positive Rate: 22/100 (0.220000)
False Negative Rate: 1/100 (0.010000)
Accuracy: 177/200 (0.885000)

Evaluate your classifier with test dataset
False Positive Rate: 49/100 (0.490000)
False Negative Rate: 31/100 (0.310000)
Accuracy: 120/200 (0.600000)

Detect faces at the assigned lacation using your classifier

Detect faces on your own images

C:\Users\user\Desktop\HW1-code>
```

T = 7

```
C:\Users\user\Desktop\HW1-code>python main.py
Loading images
The number of training samples loaded: 200
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
C:\Users\user\Desktop\HW1-code\adaboost.py:177: RuntimeWarning: overflow encountered in exp
  h = 1 - 1 / (1 + np.exp(featureVals[i] * -1))
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(8, 0, 1, 3), RectangleRegion(7, 3, 1, 3)], negative regions=[RectangleR
, 1, 3)]) with accuracy: 162.000000 and alpha: 1.450010
Run No. of Iteration: 2
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 accu
Run No. of Iteration: 3
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
8
Run No. of Iteration: 4
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(10, 3, 1, 8)], negative regions=[RectangleRegion(9, 3, 1, 8)]) with acc
Run No. of Iteration: 5
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(7, 1, 9, 4)], negative regions=[RectangleRegion(7, 5, 9, 4)]) with accu
Run No. of Iteration: 6
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(7, 3, 2, 10)], negative regions=[RectangleRegion(5, 3, 2, 10)]) with ac
Run No. of Iteration: 7
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 ac

Evaluate your classifier with training dataset
False Positive Rate: 24/100 (0.240000)
False Negative Rate: 0/100 (0.000000)
Accuracy: 176/200 (0.880000)

Evaluate your classifier with test dataset
False Positive Rate: 51/100 (0.510000)
False Negative Rate: 38/100 (0.380000)
Accuracy: 111/200 (0.555000)

Detect faces at the assigned lacation using your classifier

Detect faces on your own images

C:\Users\user\Desktop\HW1-code>
```

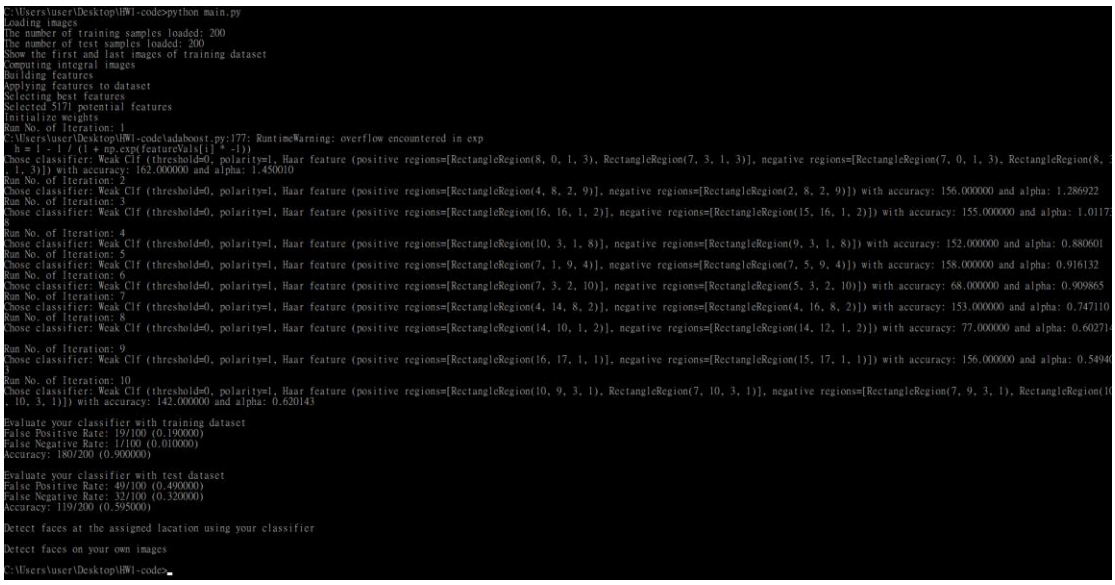
T = 8

```
C:\Users\user\Desktop\HW1-code>python main.py
Loading images
The number of training samples loaded: 200
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
C:\Users\user\Desktop\HW1-code\adaboost.py:177: RuntimeWarning: overflow encountered in exp
  h = 1 - 1 / (1 + np.exp(featureVals[i] * -1))
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)
, 1, 3)]) with accuracy: 162.000000 and alpha: 1.450010
Run No. of Iteration: 2
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
Run No. of Iteration: 3
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: 155.000
8
Run No. of Iteration: 4
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(10, 3, 1, 8)], negative regions=[RectangleRegion(9, 3, 1, 8)]) with accuracy: 152.000000
Run No. of Iteration: 5
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(7, 1, 9, 4)], negative regions=[RectangleRegion(7, 5, 9, 4)]) with accuracy: 158.000000
Run No. of Iteration: 6
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(7, 3, 2, 10)], negative regions=[RectangleRegion(5, 3, 2, 10)]) with accuracy: 68.000000
Run No. of Iteration: 7
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
Run No. of Iteration: 8
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(14, 10, 1, 2)], negative regions=[RectangleRegion(14, 12, 1, 2)]) with accuracy: 77.000000


Evaluate your classifier with training dataset
False Positive Rate: 21/100 (0.210000)
False Negative Rate: 1/100 (0.010000)
Accuracy: 178/200 (0.890000)

Evaluate your classifier with test dataset
False Positive Rate: 49/100 (0.490000)
False Negative Rate: 37/100 (0.370000)
Accuracy: 114/200 (0.570000)

Detect faces at the assigned lacation using your classifier

Detect faces on your own images

C:\Users\user\Desktop\HW1-code>
```

T = 9

```
C:\Users\user\Desktop\HW1-code>python main.py
Loading images
The number of training samples loaded: 200
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
C:\Users\user\Desktop\HW1-code\adaboost.py:177: RuntimeWarning: overflow encountered in exp
  h = 1 - 1 / (1 + np.exp(featureVals[i] * -1))
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
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
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: 1.01173
8
Run No. of Iteration: 4
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(10, 3, 1, 8)], negative regions=[RectangleRegion(9, 3, 1, 8)]) with accuracy: 152.000000 and alpha: 0.880601
Run No. of Iteration: 5
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(7, 1, 9, 4)], negative regions=[RectangleRegion(7, 5, 9, 4)]) with accuracy: 158.000000 and alpha: 0.916132
Run No. of Iteration: 6
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(7, 3, 2, 10)], negative regions=[RectangleRegion(5, 3, 2, 10)]) with accuracy: 68.000000 and alpha: 0.909865
Run No. of Iteration: 7
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: 0.747110
Run No. of Iteration: 8
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(14, 10, 1, 2)], negative regions=[RectangleRegion(14, 12, 1, 2)]) with accuracy: 77.000000 and alpha: 0.602714

Run No. of Iteration: 9
Chose classifier: Weak Clf (threshold=0, polarity=1, Haar feature (positive regions=[RectangleRegion(16, 17, 1, 1)], negative regions=[RectangleRegion(15, 17, 1, 1)]) with accuracy: 156.000000 and alpha: 0.5494
3

Evaluate your classifier with training dataset
False Positive Rate: 19/100 (0.190000)
False Negative Rate: 1/100 (0.010000)
Accuracy: 180/200 (0.900000)

Evaluate your classifier with test dataset
False Positive Rate: 47/100 (0.470000)
False Negative Rate: 38/100 (0.380000)
Accuracy: 115/200 (0.575000)

Detect faces at the assigned lacation using your classifier

Detect faces on your own images

C:\Users\user\Desktop\HW1-code>
```

T = 10



train

T	1	2	3	4	5	6	7	8	9	10
FPR(%)	28	28	13	28	24	22	24	21	19	19
FNR(%)	10	10	1	3	1	1	0	1	1	1
acc(%)	81	81	88	84.5	87.5	88.5	88	89	90	90

test

T	1	2	3	4	5	6	7	8	9	10
FPR(%)	49	49	48	48	49	49	51	49	47	49
FNR(%)	55	55	46	45	18	31	38	37	38	32
acc(%)	48	48	53	53.5	66.5	60	55.5	57	57.5	59.5

Accuracy increase with the growth of T, the phenomenon happens not only on training but also testing.

FNR decrease to almost 1 when T is bigger than 2. It means face won't be classified as non-face in training.

The accuracy of training and that of testing differ a lot, they are almost 30% different. It implies that the classifier is sensitive to the dataset.

Overfitting after T = 5 since testing accuracy is decreasing but training accuracy is increasing when T > 5.

train (part2, 3)

T	1	2	3	4	5	6	7	8	9	10
FPR(%)	28	28	23	26	23	22	20	18	20	17
FNR(%)	10	10	1	2	0	0	0	0	0	0
acc(%)	81	81	88	86	88.5	89	90	91	90	91.5

test (part2, 3)

T	1	2	3	4	5	6	7	8	9	10
FPR(%)	49	49	48	49	49	50	52	47	48	45
FNR(%)	55	55	46	56	43	48	39	43	37	36
acc(%)	48	48	53	47.5	54	51	54.5	55	57.5	59.5

They doesn't differ a lot in accuracy between Part 6 and Part 2, 3, but they are very different at T = 4, 5, 6.

Part 6 is overfitting, but Part 2, 3 isn't

Accuracy increase with the growth of T in both part in training

Describe problems you meet and how you solve them.

How to get the face location in pixel?

Use PS to support locating face.

The concept of another classifier

Surf the Internet and find that Sigmoid and binary cross entropy are good for binary classification

Sigmoid isn't good enough

Should apply 1 – Sigmoid instead of using sigmoid, since large number should be prediced to 0, and negative number should be prediced to 1.

Alpha < 0 in my former method

The error calculated by the best classifier which decided by the error with binary cross entropy is more than 0.5.

The error which choose the best classifier should consider not only binary cross entropy but also distence like L2-norm.