ECEN 649 Project Report

Name: Peixin Liu, UIN: 128002261

1 Code

The code is on GitHub: https://github.com/jackcspn/Viola-Jones

I also attached code with my report.

To run the code, please use python 3.7.2. Extra library include pillow and numpy. Please ensure install extra libraries with requirements.txt.

To find number of features as in part 2.1 and train single adaboost detector as in part 2.2, 2.3, run:
python train_ada.py

To find result of 2.1, 2.2, 2.3 and write them in txt file, run:
python eval_adaboost.py

To train cascade detector as in part 2.4, run:
python train_cas.py

To find result of 2.1 and 2.2 and write them in txt file, run:
python eval_cascade.py

2 Result

2.1. Extract Haar Features

Numbers of Haar Features are in 'output/n_features.txt', and also are listed below

```
There are 7440 type 1 (two vertical) features.
There are 7440 type 2 (two horizontal) features.
There are 3844 type 3 (three horizontal) features.
There are 3844 type 4 (three vertical) features.
There are 3600 type 5 (four) features.
The total number of Haar Features is: 26168.
```

As we can see, feather type with smaller basic size generates more features.

2.2 Build Your Adaboost Detector

The features information of round 1, 3, 5, 10 detectors is in 'output/feature_eval.txt', and also listed below

```
Adaboost Round: 1
```

```
Feature number 1:
Type: two horizontal
Position: (8, 2)
Width: 2
Height: 8
Threshold: -0.407845
Training accuracy: 0.848339

==================================

Adaboost Round: 3

Feature number 1:
Type: two horizontal
Position: (8, 2)
Width: 2
Height: 8
Threshold: -0.407845
Training accuracy: 0.848339

Feature number 2:
Type: two vertical
Position: (3, 3)
Width: 2
Height: 6
Threshold: 0.796080
Training accuracy: 0.807523

Feature number 3:
Type: four
Position: (13, 14)
Width: 6
Height: 4
Threshold: -0.376457
Training accuracy: 0.824730

==================================

Adaboost Round: 5

Feature number 1:
Type: two horizontal
Position: (8, 2)
```

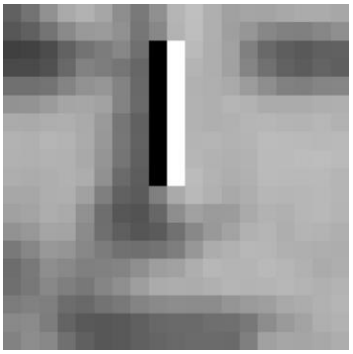
```
Width: 2
Height: 8
Threshold: -0.407845
Training accuracy: 0.848339

Feature number 2:
Type: two vertical
Position: (3, 3)
Width: 2
Height: 6
Threshold: 0.796080
Training accuracy: 0.807523

Feature number 3:
Type: four
Position: (13, 14)
Width: 6
Height: 4
Threshold: -0.376457
Training accuracy: 0.824730

Feature number 4:
Type: two horizontal
Position: (11, 9)
Width: 6
Height: 3
Threshold: -0.588230
Training accuracy: 0.815526

Feature number 5:
Type: two vertical
Position: (6, 0)
Width: 8
Height: 2
Threshold: -0.466666
Training accuracy: 0.821128

==================================

Adaboost Round: 10

Feature number 1:
Type: two horizontal
Position: (8, 2)
```

```
Width: 2
Height: 8
Threshold: -0.407845
Training accuracy: 0.848339

Feature number 2:
Type: two vertical
Position: (3, 3)
Width: 2
Height: 6
Threshold: 0.796080
Training accuracy: 0.807523

Feature number 3:
Type: four
Position: (13, 14)
Width: 6
Height: 4
Threshold: -0.376457
Training accuracy: 0.824730

Feature number 4:
Type: two horizontal
Position: (11, 9)
Width: 6
Height: 3
Threshold: -0.588230
Training accuracy: 0.815526

Feature number 5:
Type: two vertical
Position: (6, 0)
Width: 8
Height: 2
Threshold: -0.466666
Training accuracy: 0.821128

Feature number 6:
Type: four
Position: (2, 15)
Width: 6
Height: 4
Threshold: 0.129408
Training accuracy: 0.729492
```

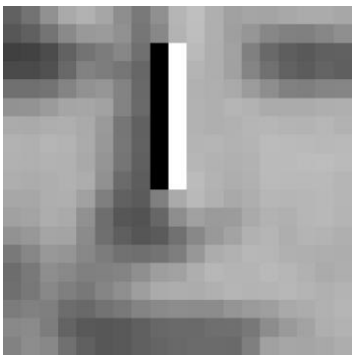
```
Feature number 7:
Type: four
Position: (0, 0)
Width: 2
Height: 6
Threshold: -0.074510
Training accuracy: 0.762705

Feature number 8:
Type: two vertical
Position: (15, 3)
Width: 4
Height: 4
Threshold: 0.431366
Training accuracy: 0.765906

Feature number 9:
Type: two vertical
Position: (8, 11)
Width: 1
Height: 2
Threshold: 0.043137
Training accuracy: 0.783914

Feature number 10:
Type: two horizontal
Position: (8, 9)
Width: 2
Height: 1
Threshold: -0.050980
Training accuracy: 0.845538

==================================
```

The feature training accuracy does not increase with number of rounds in one detector. Probably this is because trainset is not big enough to give proper accuracy, or there is something wrong when calculating the accuracy.

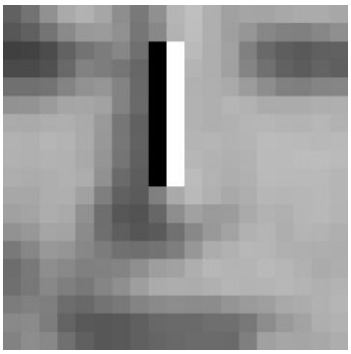
The image of top accuracy feature of four detectors are ‘output/img_feature_1.png’, ‘output/img_feature_3.png’, ‘output/img_feature_5.png’, ‘output/img_feature_10.png’, and are shown below.



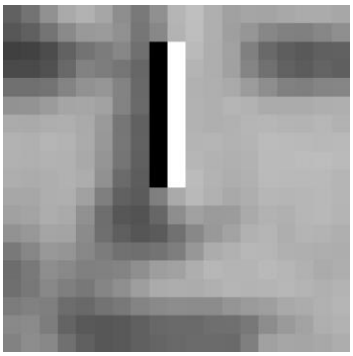
Top feature of round 1 detector



Top feature of round 3 detector



Top feature of round 5 detector



Top feature of round 10 detector

As we can see from the accuracy of features, the first feature are always the feature with high accuracy in all four detectors. So the four images show the same feature and are identical. The reason for this is the same for non-increasing accuracy of features discussed above.

Accuracy, false positive rates and false negative rates of four detectors is in 'output/ adaboost_eval_1.txt', and is also listed below

```
Adaboost Round: 1
Total Accuracy: 78.811161%
False Positive : 9.495252%
False Negative : 70.762712%

Adaboost Round: 3
Total Accuracy: 82.571775%
False Positive : 3.148426%
False Negative : 77.966102%

Adaboost Round: 5
Total Accuracy: 81.641731%
False Positive : 1.599200%
False Negative : 89.406780%
```

```
Adaboost Round: 10
Total Accuracy: 82.854832%
False Positive : 0.699650%
False Negative : 86.864407%
```

As we can see accuracies are high, but do not necessarily increase with round of Adaboost. And false positive rate is low, however false negative rate is high. This may cause by reason that implementation has error, or our adaboost detector is designed to select less positive samples but with high accuracy.

2.3 Adjust the threshold

The detectors information with different criterion is in 'output/ adaboost_eval_2.txt', and is also listed below

```
Criterion: Empirical
Total Accuracy: 81.641731%
False Positive : 1.599200%
False Negative : 89.406780%

Criterion: False_Positive
Total Accuracy: 81.439547%
False Positive : 2.348826%
False Negative : 87.288136%

Criterion: False_Negative
Total Accuracy: 71.977355%
False Positive : 21.389305%
False Negative : 56.144068%
```

We can make table out of these information

Criterion	Total Accuracy	False Positive	False Negative
Empirical Error	81.641731%	1.599200%	89.406780%
False Positive	81.439547%	2.348826%	87.288136%
False Negative	71.977355%	21.389305%	56.144068%

We can find that, the detector performs well on the evaluation value that is corresponding to its error criterion. Empirical error detector performs better in accuracy, false positive detector performs better in false positive, false negative detector performs better in false negative. This is what we expect for testing criterion. Since the detector is trained on such criterion, it is expected to performs better on that criterion.

2.4 Build the cascading system

To get better result, I trained 3 layers cascade. The detectors information with 3 layer of cascade

‘output/cascade_eval.txt’, and is also listed below

```
Accuracy: 94.397759%
Detect Rate: 72.344689%
Empirical Error: 5.602241%
False Positive : 0.100000%
False Negative : 27.655311%

Layer 0 drops negative 1720 images
Layer 1 drops negative 310 images
Layer 2 drops negative 106 images
```

As we can see, the result is much better than single Adaboost classifier. Since cascade drops negative images every layer and decrease false positive rate every layer. From the result, we can find that with increase of the layers, number of dropped image decrease, since less images are fed into deeper layer and accuracy is increasing.