

PRML（ProjectⅠ）

Face Recognition and Detection



学 院（系）： 电子信息与电气工程学部

专 业： 计算机科学与技术

班 级： 电计1704

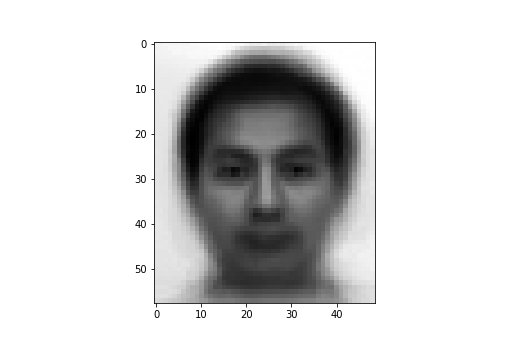
学 生 姓 名： 谢玉宁

学 号： 201792260

完 成 日 期： 2020年5月20日

# Ⅰ. Face Recognition

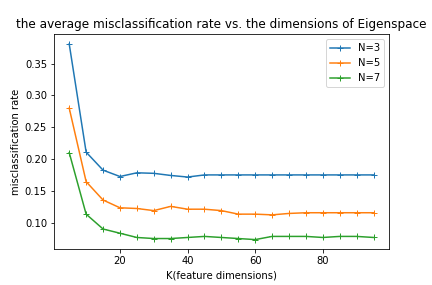
## Average face



**Fig 1** Average face

the simulation results of Eigenfaces and Fisherfaces are as follows.

## Eigenfaces



**Fig 2** Average misclassification though Eigenfaces

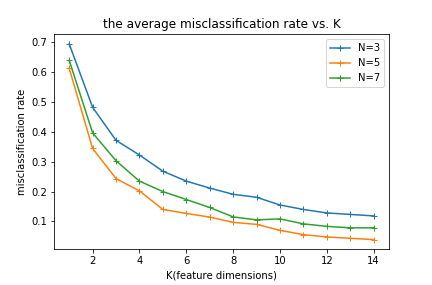
Explanation:

1. As can be seen from the above figure, the three functions of K (feature dimensions) are generally monotonically decreasing, because more features we use, better the test face can be described, so that lower the Average misclassification rates are. However, the rates hardly change when the main component features are greater than 40.

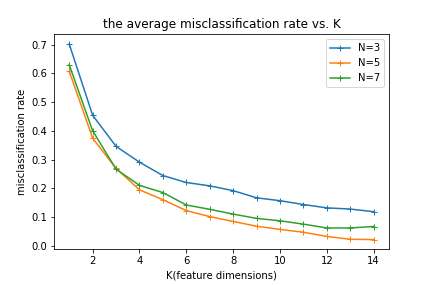
Theoretically speaking, each rate won’t change at the position of K1=15\*3, K2=15\*5, K3=15\*7 respectively, because the main components features with K > Ki (i = 1,2,3) contribute nothing to describe the test face, as the rank of the within-class scatter matrix is 45, 75, 105.

1. Average misclassification rate when the number of train sample is 7, is lower than any other rate. That’s because the performance of the classifier gets better as the samples are more.
2. The performance is not bad as the lowest average misclassification rate is about 4% with N=7.

## Fisherfaces



**Fig 3** Average misclassification though Fisherfaces with down sampling ratio



**Fig 4** Average misclassification though Fisherfaces with down sampling ratio

1. As can be seen from the above figure, the three functions of K (feature dimensions) are generally monotonically decreasing, because more reduced features we use, better the test face can be described, so that lower the Average misclassification rates are.

Because there are 15 subjects to classify, we can only reduce the feature space up to 14 dimensions.

1. The performance of classifier though fisherface is obviously better than eigenface, that’s because This is because fisherface has the best projection that can separate the three categories to the greatest extent as it takes into count the distance information between different classes, while eigenface doesn’t.
2. The performance of classification is so powerful that the lowest average misclassification rate is about 2% with down sampling ratio = .

# Ⅱ. Face Detection

## *Challenges: LBP, HOG, IoU, dominant hue*

To conclude, Challenges I met in this project are as follows:

1. What features can I use to well describe the face and immune to some influence like light intensity?
2. How to merge the two rectangles indicating the same face?
3. How to distinguish fake face from the candidate faces whose shapes are like a face?

Below are how I overcome the challenges:

## *Keywords*: *LBP, Hamming distance, HOG, IoU,* *dominant hue*

To complement face detection, first we should find a way to extract some face features from the segments of picture and then compare the distances between the features of segments and the features we already have and choose the closest ones as the faces we detect.

At first, I used eigenfaces to extract the features from segments, and the results were not satisfactory as the detector considered black segments as faces because the training data are darker than the picture from which we want to detect faces.

And then, I tried to use PCA to rebuild a face though reduced data of each segment, and calculate the distance from average face, and then find a threshold to determine whether the segment is a face or not. But it still didn’t work.

As the results I mentioned are usually dark picture, **I tried to apply an algorithm which is not influenced by luminosity effect.** Therefore, I chose **LBP**, which meets the requirement, as well as **Hamming distance** as a way to extract the features and calculate the difference between features of training data (average face) and features of detected areas. And the result is not bad.

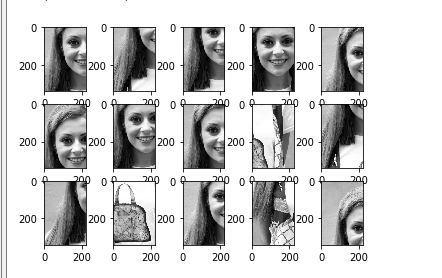


Figure 1 Top 15 likeliest areas for human faces in picture1 detected by LBP

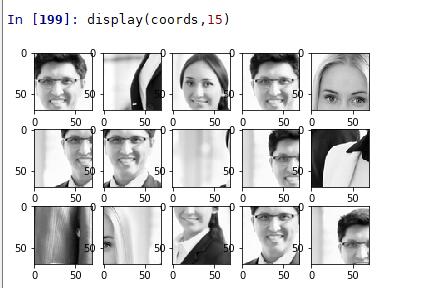


Figure 2 Top 15 likeliest areas for human faces in picture2 detected by LBP

In those two figures above, although it has great progress on detection obviously, **we can find that the detector based in LBP still has a flaw as it ignores the information of gradient of the areas**, which is important for computer to recognize the objects in picture. Therefore, HOG algorithm suddenly occurred to me!

Then **I built a detector based on HOG** which stands for Histogram of Oriented Gradients. HOG uses the gradient information as features and **eliminates the luminosity effect by Gamma Correction and normalization** as well. And the result is really fascinating.

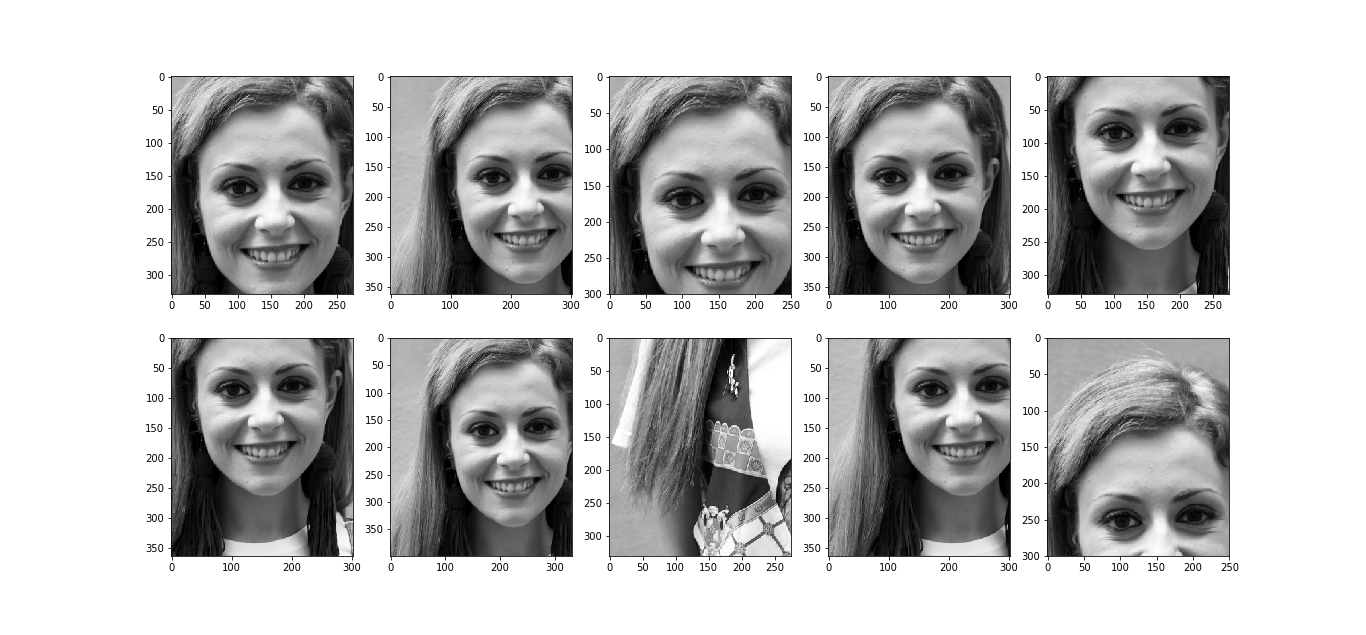


Figure 3 Top 10 likeliest areas for human faces in picture1 detected by HOG

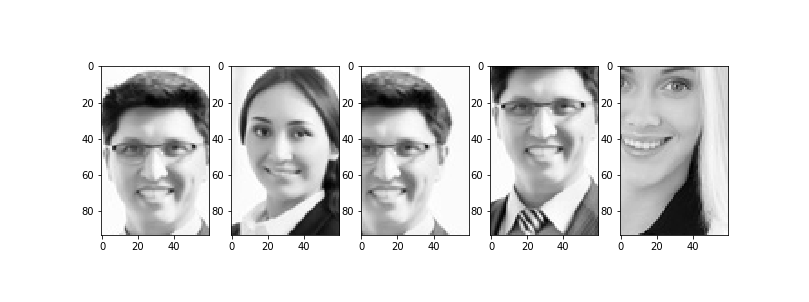


Figure 4 Top 5 likeliest areas for human faces in picture2 detected by HOG

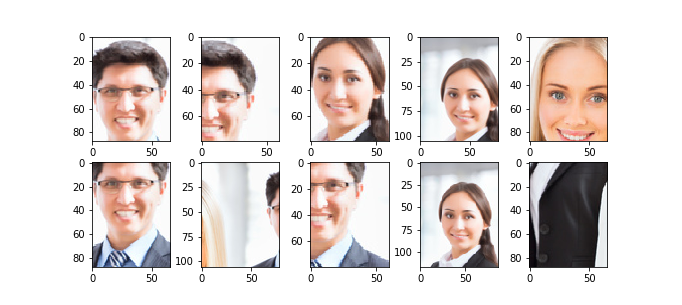
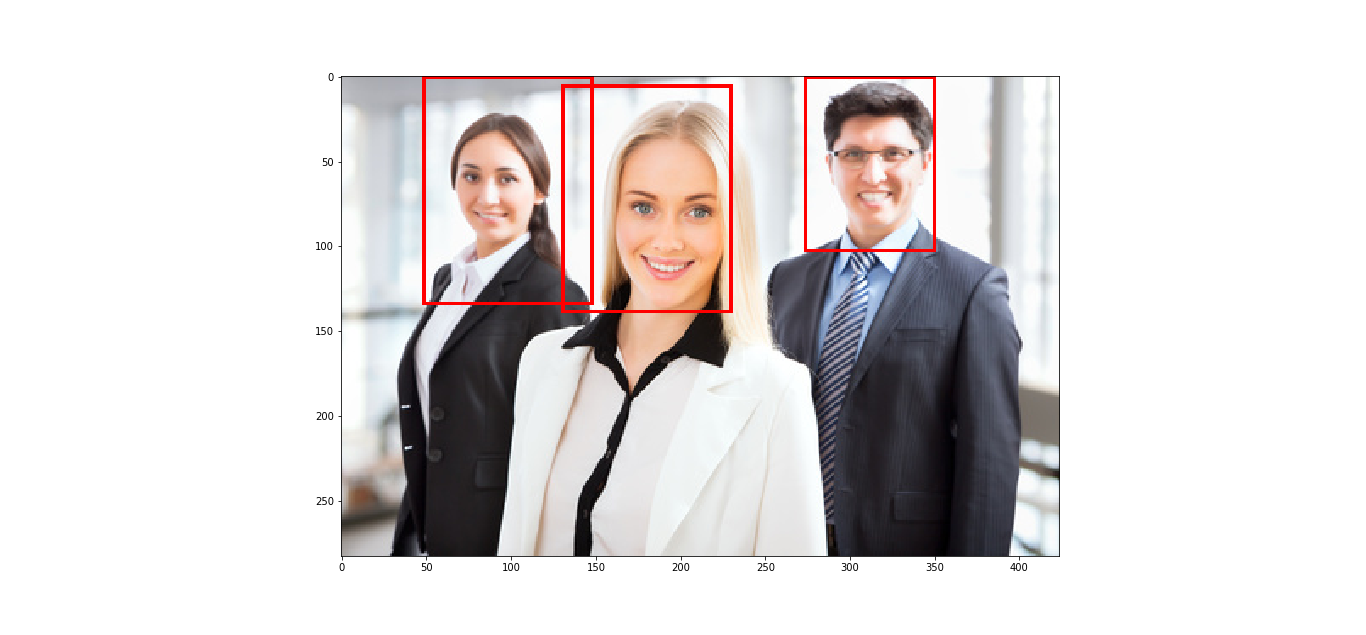
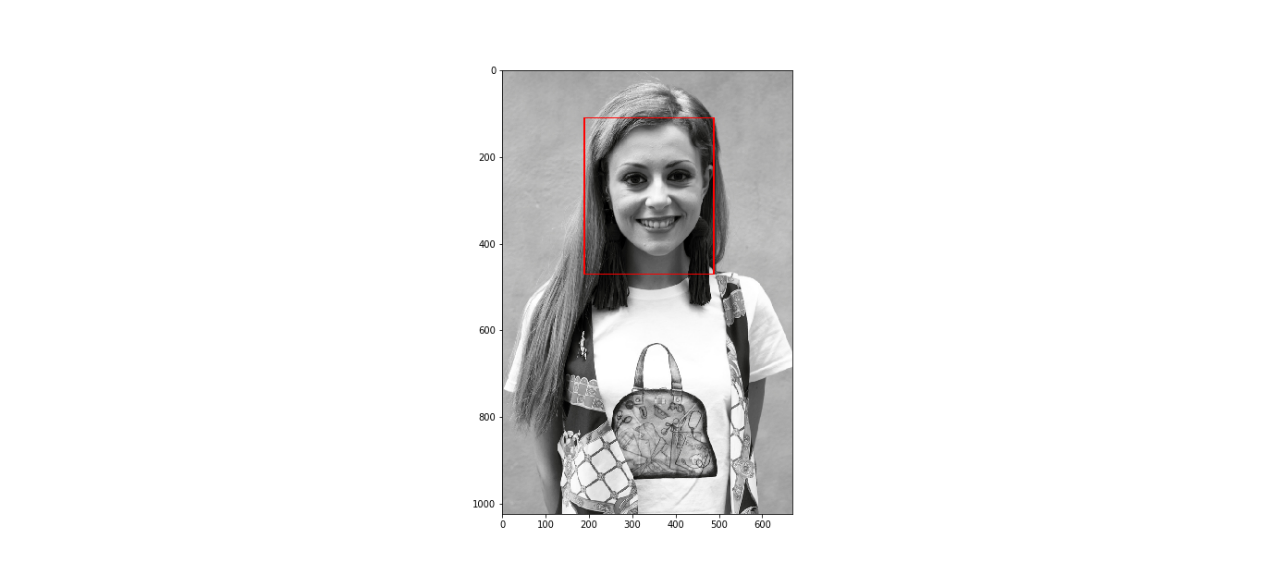


Figure 5 Top 5 likeliest areas for human faces in picture2 detected by HOG

Seeing that the figures above show a good recognition result, I got down to apply the algorithm to frame faces in detected picture by rectangle. Here comes the question: How to determine if there are more than one detected picture indicating the same face? To solve this problem, **I introduced *IoU*.**

IoU stands for Intersection over Union, which means the area of overlap divided by the area the 2 pictures occupy. Though IoU we can distinguish if the new area contains a new face or an old face we have detected. If the area contains an old face, **the algorithm will revise the coordinate and size of the area containing face.** The threshold of IoU by default is 0.3.

The results are as follows:



Since we have already gotten the candidate faces from the top N detected areas. How to determine if there are more than one detected picture indicating the same face? To solve this problem, **I introduced *IoU*.** IoU stands for Intersection over Union, which means the area of overlap divided by the area the 2 pictures occupy. Though IoU we can distinguish if the new area is a new face or an old face we have detected. And the threshold of IoU by default is 0.3.

Besides, I use the **color information** to determine whether detected areas are real faces. I count the hues of different parts of the areas, if the **dominant hue** is not like the color of face, the algorithm will consider the detected area as a fake face, which will be left off.



The left figure shows top 15 likeliest areas for human faces in picture2 without using color information, so it contains pictures that are not real faces while the right figure shows the correct result. This is because the 2 “fake faces” in left figure has a dark blue dominant hue while real face has skin color.

## Flaws of the detecting algorithm:

The algorithm shows very poor performance in small target detection such as picture3. It can’t distinguish faces and the complex background. The result of picture3 is as follows.



Figure 6 Top 20 likeliest areas for human faces in picture3 detected by HOG

One possible solution is to let the results of different algorithms vote for the final result.

To conclude, in this project I learned the traditional pattern recognition algorithms like eigenfaces and fisher faces as well as object detection algorithms such as HOG and LBP. It took me 3 days to finish this project and it turned out that I really enjoy doing this job. Although there are some flaws in my “detector” but I’m satisfied with it as it shows good results in some certain circumstance. Hope I can get a good grade in this project.