

Project 1: face classification and detection

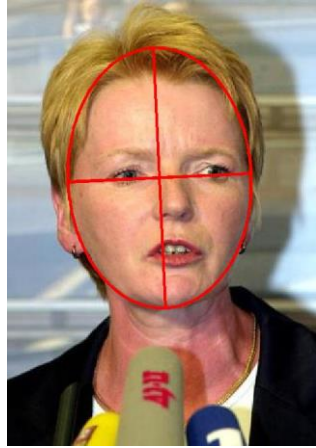
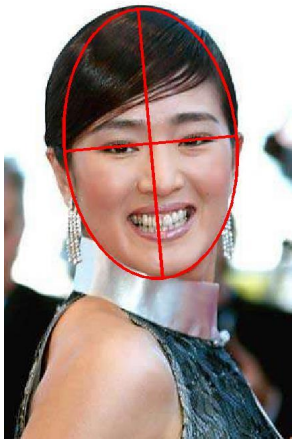
CS 385

Objective

- The objective is to help you have an deep insight of current models, including the intuition about the feature and the model, and the discrimination power of different models.
 - Implement face classification using different models
 - Implement face detection
 - Combining face classification and the sliding window
 - Visualize face features
 - Visualize models
 - Write a report

Dataset: FDDB: Face Detection Data Set and Benchmark

- A dataset before the era of deep learning
 - Relatively easy samples
- You need to download the data from <http://vis-www.cs.umass.edu/fddb/>
 - Training images
 - Face annotations



Most faces are captured from a frontal view without a significant rotation.

Face classification

- Given training samples and their labels $\{x_i, y_i\}$
- For the input face image x_i
 - 5171 faces in a set of 2845 images taken from the Faces in the Wild data set.
 - There are ten annotation folders
 - FDDB-fold-01, FDDB-fold-02, ..., FDDB-fold-10
 - **Please use face images in the first eight folders as training samples, and use face images in the last two folders as testing samples.**

Face classification

- For the input face image x_i
 - Annotations on each image are represented as follows
 - <image name i>
 - <number of faces in this image =im>
 - <face i1>
 - <face i2>
 - ...
 - <face im>
 - Each face <face im> is annotated as
 - <major_axis_radius minor_axis_radius angle center_x center_y 1>



2002/07/28/big/img_416

3

60.933903 44.468305 -1.530098 98.276900 82.268230 1

52.564676 35.774346 -1.460712 321.103651 63.475043 1

23.958142 18.064426 1.553739 364.859591 103.521632 1

2002/08/07/big/img_1393

2

163.745742 109.136126 1.236962 132.225074 172.180404 1

134.196339 96.693777 -1.276447 204.178610 271.741124 1

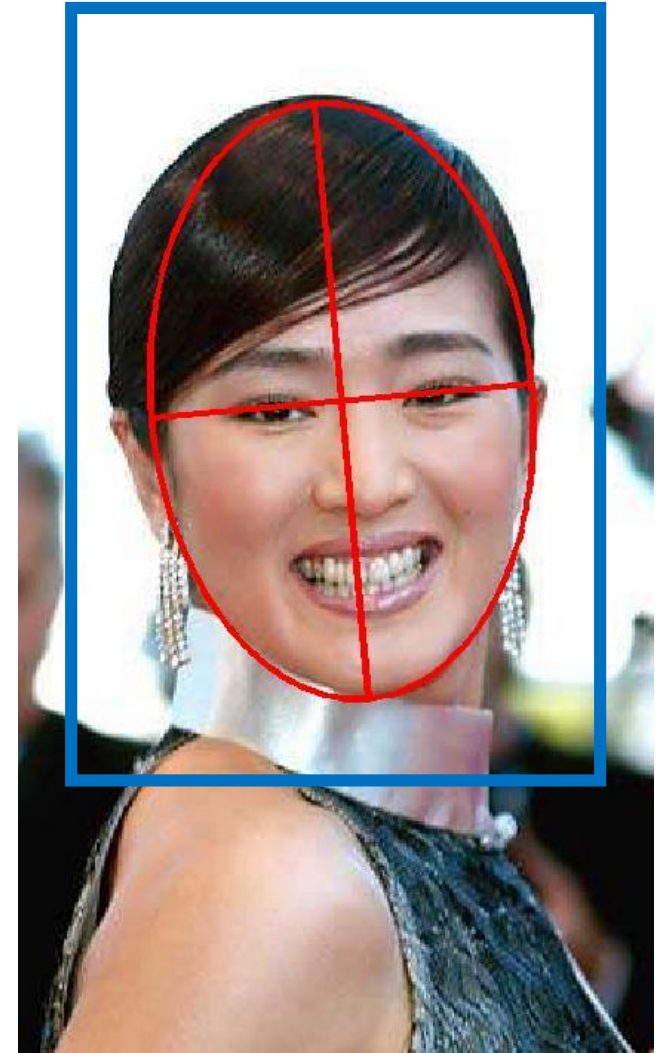
2002/08/26/big/img_292

1

163.778521 101.169600 1.543931 148.203875 181.913193 1

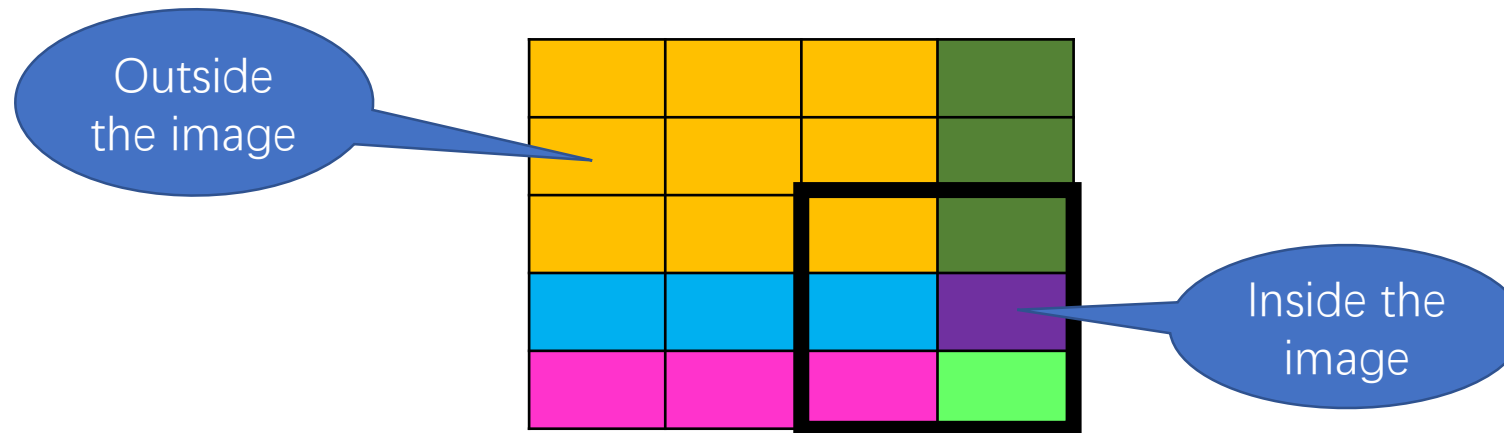
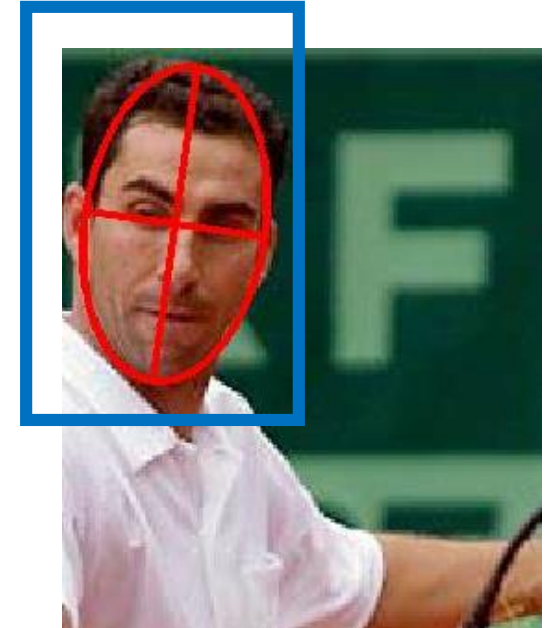
Face classification

- For the input face image x_i , we need to simplify the data input
 - Use a rectangle **without** any rotations
 - We extend the scale of the bounding box by $1/3$
 - Width = (horizontal axis * 2) * (1+1/3)
 - Height = (horizontal axis * 2) * (1+1/3)
 - Crop the face within the bounding box as a positive sample.



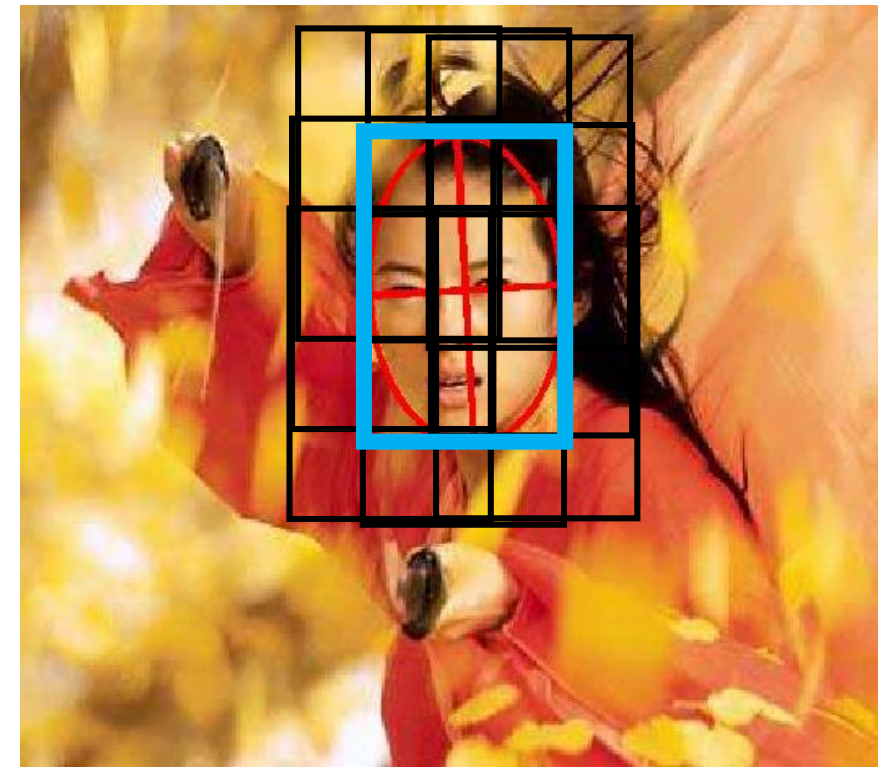
Face classification

- What if the face is on the border of the image?
 - Please fill the empty space using the nearest pixel.



Face classification

- Given training samples and their labels $\{x_i, y_i\}$.
 - For each face image, i.e., a positive image, $y_i = +1$
 - For each negative image, $y_i = -1$
- How to generate negative images?
 1. Resize original images into 96 x 96 pixels
 2. For simplicity, please generate eight negative images based on each face by sliding the bounding box by 1/3
- Using face images in the first four folders to generate negative images for training.
- Using face images in the last one folders to generate negative images for testing.



Face detection

- Enumerate all potential bounding boxes of faces
 - At all locations
 - Using all scales
- Use the face classifier to identify the true face from the background.

To do

- Resize all samples (including clipped faces and negative samples) into 96 pixels x 96 pixels
- Extract HOG features from each sample using Matlab
 - Paper for HOG features
 - <http://www.cs.toronto.edu/~fleet/courses/2503/fall11/Handouts/cvpr05.pdf>
 - Extract HOG features using Matlab
 - <https://www.mathworks.com/help/vision/ref/extracthogfeatures.html>
 - Block size = 2 x 2
 - Cell size = 16 x 16
 - Number of bins = 9
 - Block overlap = 1 x 1
 - → in this way, you may extract a 900-d feature vector from each sample.
 - You may also select other platform to extract HOG features
 - You may save and transfer the extracted HOG features to another platform, if the platform for feature extraction is different from that for model learning.

To do

- Visualize bounding boxes of positive and negative samples on images.
- Visualize the extracted HOG features
 - Please see <https://www.mathworks.com/help/vision/ref/extracthogfeatures.html> if you extract HOG features using Matlab
 - Show the original image on the left and the HOG feature on the right
- Learning a logistic model for classification
 - Optimizing the model via SGD
 - Optimizing the model via Langevin dynamics
 - Report the accuracy
 - Do not use off-the-shelf codes

To do

- Learning a fisher model for classification
 - Report the accuracy, the intra-class variance and the inter-class variance
 - Do not use off-the-shelf codes
- Learning SVMs for classification
 - Learning a linear SVM
 - Learning an RBF SVM (with RBF kernel) for classification
 - Learning a SVM with another kernel
 - Report the accuracy
 - List samples of support vectors (i.e., the samples whose the margin =1)
 - You may use off-the-shelf codes to train the SVM
- Learning convolutional neural networks for classification
 - I will introduce this in several weeks
- Visualize face detection results and feature distribution
 - Based on logistic model, linear SVM, RBF SVM, CNN
 - Visualization of the feature distribution will be introduced later.

Submit

- A report for this project
- Code
 - **Do not copy others' codes**
 - The teaching assistant will check the code.
 - **Make sure your code can run**
 - I will randomly run codes of 30% students.
 - **Do not create numbers without writing a code**