# Computer vision : Mid-term

Due on

*Dr. Xiu Wen Liu 11:00am*

**Jian Wang**

# Contents

# [Purpose]

This mid-term project helps us to review the knowledge that we learned from the computer vision class.

# [Problems]

**Problem 1:**
1) describe how SIFT features are detected and how SIFT descriptors are computed. On typical images, how would you characterized the local patches centered at identified SIFT features in terms of the eigenvalues of the autocorrelation matrices? Briefly explain.
**Solution:** a) To compute SIFT features, first we need to compute the difference of Gaussian images. Then we detect local extreme by comparing each pixel by its eight neighborhoods in the same scale and nie neighbors in the scale above and below. Last step is to compute the descriptor s we first compute the gradient magnitude and orientation at each sample point around feature point. Then the samples are accumulated into orientation, histograms. Summarizing the contents over 4x4 subregions. Each oriental histogram has 8 bins for 8 directions. b) If the patch has two large eigenvalues, it represents a vertex. If it has one large eigenvalue, it is an edge. Otherwise, it is not a feature point.

2) In our version of the agglomerative hierarchical clustering algorithm, we sue the single-link as the between-clusters distances in the odd steps(2,4, and so on ) Show the step-by-step clustering results on the pairwise distances given on the following. In each step, you need to show which clusters are merged and what is the distance between the merged clusters.

$$\begin{bmatrix} .000 & .480 & .658 & .456 & .642 & .183 \\ .480 & .000 & .425 & .117 & .391 & .485 \\ .658 & .425 & .00 & .418 & .161 & .644 \\ .456 & .117 & .418 & .000 & .382 & .464 \\ .642 & .391 & .161 & .382 & .000 & .607 \\ .183 & .485 & .644 & .464 & .607 & .000 \end{bmatrix}$$

**Solution:**
In *single-link* clustering or *single-linkage* clustering, the similarity of two clusters is the similarity of their most similar member. In *completed-link* clustering or *complete-linkage* clustering, the similarity is the similarity of the most dissimilar members.
Step 1: we use the single-link method, sample 2 and sample 4 are merged, the distance is =.117
Step 2: We use the complete-link method, note here, we should use the **maximum** distance between the cluster {2,4} and other samples. for example, the distance between sample 1 and cluster {2,4} should be 0.480 instead of 0.456. In this step , sample 3 and sample 5 are merged , the distance is 0.161.
Step 3: use *single-link*,sample 1 and sample 6 are merged, the distance is 0.183.
Step 4: use *completed-link*,sample (2,4) and (3,5) are merged, the distance is 0.425.
Step 5: use *single-link* sample(2,3,4,5) and (1,6) are merged.

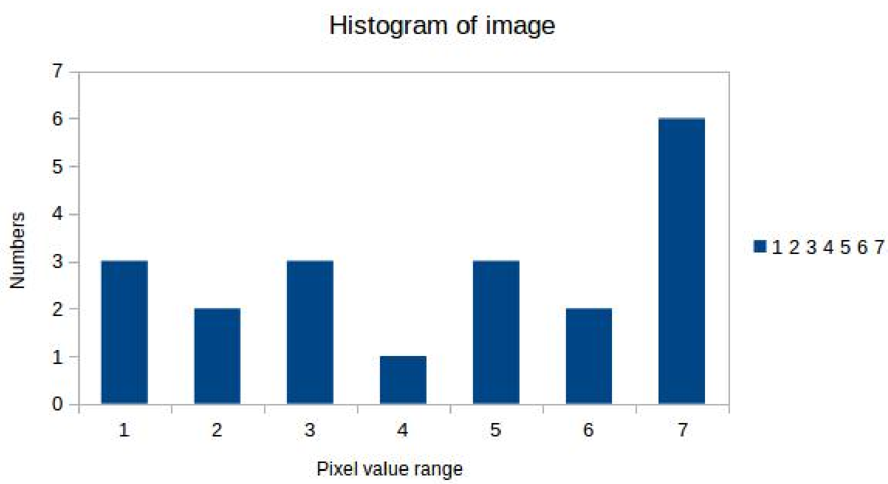
3) Given the following image(on the left side), compute its histogram, assuming that a bin's range is [x-0.5,x+0.5], where x is the value at the center of the bin. Suppose that the pixel values must be between 0 and 7 inclusively, given an image with the same dimension (as the given image) so that the histogram intersection between the histograms of the two images is the smallest. You need to briefly explain your answer.

| | | | | |
|---|---|---|---|---|
| 2 | 5 | 2 | 0 | 7 |
| 4 | 6 | 6 | 0 | 1 |
| 3 | 2 | 7 | 6 | 5 |
| 1 | 0 | 4 | 0 | 7 |

**Solution:**

| Pixel value | Numbers |
|---|---|
| [0, 1) | 3 |
| [1,2) | 2 |
| [2,3) | 3 |
| [3,4) | 1 |
| [4,5) | 3 |
| [5,6) | 2 |
| [6,7) | 6 |

The histogram of the image is given by the following chart:




4)W use the Canny edge detector to detect edges in an image. At pixel (x=150,y=200), $\frac{\partial f}{\partial x}$ is estimated to be -120 and $\frac{\partial f}{\partial y}$ is estimated to be 80; for non-maximum suppression, which two pixels shall be used for comparison at the given pixel? Justify your answer. We assume that the eight nearest neighbors are used.

**Solution:**

Since $(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}) = (-120, 80)$, so the derivation of gradient can be estimated by $\beta = arctan(-\frac{2}{3}) \approx 146.31^o$, which is rounded to $135^o$. Hence we compare it with the upper left and the lower right pixel.]

**Problem 2** To train a real-time face detector, there are 8 face training images and 12 non-face training images, In the process of selecting the first optimal weak classifier, for a particular Haar feature, its values for all the face training images are 0.48,0.67,0.60,0.64,0.64,0.88,0.74,0.35 and its values for all the non-face training images are -0.09,0.31,0.15,0.38,0.27,0.45,0.10,0.28,0.29,0.37,0.58,0.15.Answer the following questions:

1) A common way to implement a face detector is to use a classifier to classify each local window whether it is a face. Explain the key differences of the real-time face detection method that enables it to achieve real-time detection compared to the common way.

Answer:
The real-time face detector contains the following key components:
a) A new image representation:The features can be computed fast and in constant time regardless of the scales
b)A simple and efficient (but also effective) classifier:Based on AdaBoost for feature selection and learning classifiers
c)Cascade of classifiers to reduce the average detection time for an image:A degenerated decision tree; often known as the twenty-question approach

2) Describe clearly how to efficiently compute the optimal weak classifier for this feature. Then plot the weighted error against the threshold for this feature and give ALL the parameters for the resulting optimal weaker classier.
Answer:
The algorithm to efficiently compute the optimal weak classifier is described in algorithm 1.

**Algorithm 1:** AdaBoost for feature selection

Given training example images:$(x_1, y_1), \ldots, (x_n, y_n)$ `// assume two dimensions`
  where $y_i = 0, 1$ for negative and positive examples respectively.
Initialize weights $w_{1,i} = \frac{1}{2m}, \frac{1}{2l}$ for negative and positive training example respectively.
  where m and l are the number of negatives and positives respectively
**for** $t = 1, \ldots, T$ **do**
  Normailize weights so that $w_t$ is a distribution
  For each feature j train a classifier $h_j$ and evaluate its error $\epsilon_j$ with respect to $w_t$
  Chose the classifier $h_j$ with lowest error
  Update weights according to :
  $w_{t+1,j} = w_{t,j}\beta_t$ where $e_i = 0$ is $x_i$ is classified correctly, 1 otherwise, and $\beta_t = \frac{\epsilon_t}{1-\epsilon_t}$
**end**
The final strong classifier is:

$$h(x) = \begin{cases} 1 & \sum_{t=1}^{T} \alpha_t h_t(x) >= \frac{1}{2}\sum_{t=1}^{T} \alpha_t, where\ \alpha_t - log(\frac{1}{\beta_t}) \\ 0 & \text{otherwise} \end{cases}$$

For each weak classifier is determined by a threshold and sign $h(x, f, p, \theta) = \begin{cases} 1 & \text{if } pf(x) < p\theta \\ 0 & \text{otherwise} \end{cases}$ So the problem is how to find the best threshold for the criterion used in AdaBoost. To make this process in a fast way, we can use the following algorithm 2

In this problem, we let the weights for positive sample as 1/2*8=1/16 and the weights for negative numbers as 1/2*12=1/24.
Now change the threshold and calculate the different sum of weights to find the error.
After we sort the list, it becomes $--------+---+-++++++$, if we put the threshold between the first and the second element, $-|-------+---+-++++++$, the $T^+$ is 0.5 and the $T^-$ is also 0.5,the $S^+$ is 0, and the $S^-$ is 1/24, so the error for the threshold is:$e = min(S^+ + (T^- - S^-), S^- + (T^+ - S^+)) = min(11/24, 13/24) = 11/24$., similarly, the error for the threshold between the second and the third element is:10/24,9/24,8/24,7/24,6/24,5/24,4/24,11/48,9/48,7/48,5/48,8/48,6/48,9/48, 12/48,15/48,18/48. The minimum threshold is 5/48, which occurs between 0.45 and 0.48, so we can choose the threshold 0.465.

3)Suppose that this Haar feature gives the lowest weighted error among all the haar features, what are the weights used to choose the second optimal weak classifier? You need to specify the weight for each sample.

**Algorithm 2:** How to choose threshold

Training samples are sorted based on the current feature.;
Optimal threshold is computed in a single pass over list.;
Four sums are maintained and evaluated:;
    1)The total sum of positive example weights $T^+$
    2)The total sum of negative example weights $T^-$
    3)The sum of positive weights below the current example $S^+$
    4)The sum of negative weights below the current example $S^-$
The error for a threshold which splits the range between the current and previous example in the sorted list.

$$e = min(S^+ + (T^- - S^-), S^- + (T^+ - S^+))$$

Answer: if we take threshold as 0.465, the sample bigger than this threshold will be classified as 1, and -1 vice versa. Only two samples are misclassified, the 9th sample(positive misclassified as negative) and the 14th(negative misclassified as positive) sample. So we only need to change the weights of these two samples and leave others unchanged.