The Hong Kong University of Science and Technology
Department of Computer Science and Engineering
COMP4421 (Fall 2018)

Assignment 3

Total = 100 marks
Due: 11:55pm, Dec. 7, 2018
Assignments must be submitted via Canvas
Late Policy: 10% reduction; only one day late is allowed, i.e., 11:55pm, Dec. 8.

Overview

Topics: Segmentation and Classification

In this assignment, you need to finish two parts. The first one is to answer several simple questions, while the second part requires certain programming works.
Please submit your answers of the first part in PDF format, and all relevant codes and results of your second part together in a folder. You also need to **paste your result images of the second part in the PDF file**. **Please follow the structure of "Sample" folder to place your codes and result images (otherwise marks will be deducted)**. Then you should rename the "Sample" folder by your student ID and the programming language you use and compress it into a zip file. For example, if your student ID is "12345678" and you choose to use Matlab, then your folder should be "12345678_matlab".

This assignment should be submitted via the Canvas system on or before the due date.

1. Exercises (30%)

Answer the following question.

Question 1: Lossless compression (Huffman coding).

Given a 1-D image profile as follows (intensity values range from 0 to 15):

Index	1	2	3	4	5	6	7	8	9	10	11	12	13	14	15	16
Intensity	1	1	3	12	2	3	9	7	9	7	9	15	1	1	3	3

Perform Huffman coding for the image and calculate the compression ratios.

Show all your steps clearly.

For Huffman coding, show your steps in terms of the tables as shown in the Huffman Encoding part of the lecture notes (Image Compression). You should assign label **'1'** to the intensity with higher frequency in the Huffman code assignment table.

In the calculations of compression ratios, ignore the storage overhead due to the lookup tables.

Question2: Adaboost Learning Algorithm.

It is assumed that there are five weak classifiers, h_1, h_2, h_3, h_4 and h_5; and there are nine training examples.

$y_1 = 1, y_2 = 1, y_3 = -1, y_4 = -1, y_5 = 1, y_6 = 1, y_7 = -1, y_8 = 1, y_9 = -1$, where 1 represents face image and -1 represents non-face image.

The weak classifier response are as follows:

$h_1 =$ [1 1 -1 -1 1 -1 1 -1 1];
$h_2 =$ [-1 1 -1 1 -1 -1 1 -1 -1];
$h_3 =$ [1 1 -1 1 -1 -1 -1 1 -1];
$h_4 =$ [-1 -1 1 -1 1 -1 1 -1 1];
$h_5 =$ [-1 1 1 -1 1 -1 1 -1 1];

a) Find the strong classifier H by selecting two weak classifiers.

b) Show the final response of the strong classifier toward the nine training example.

Note: If multiple weak classifiers give the same classification error, then the weak classifier with the lowest classifier index is selected. The answer are rounded to 4 decimal places. Show all steps clearly.

2. Programming Tasks (70%)

Write programs to finish the following tasks.

2.1 Pre-requirement

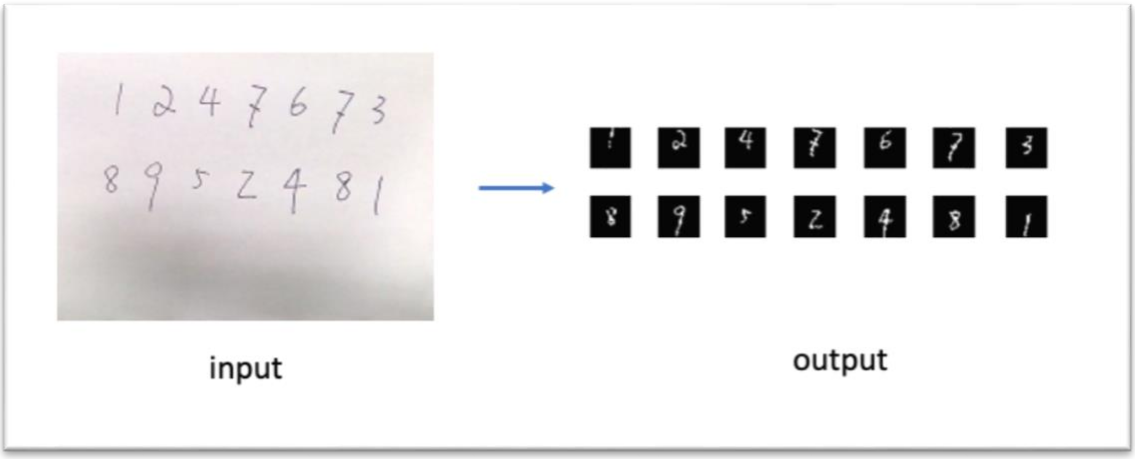
2.1.1 Input: The input images are in the folder "input_imgs" (/Sample/input_imgs). You should test all the input images and paste the result images in the PDF file. Note that we will have some other test images for grading.

2.1.2 Language: Matlab/Octave/**Python 3.6**. For the students who choose to use Python, you need to transform our provided simple MAIN function into Python so that we can run your code.

2.1.3 Functions to be used: In this assignment, you can use the built-in function for filtering and edge detection. However, you cannot use the built-in function relating to the segmentation and Adaboost algorithm.

2.2 Digits Segmentation (40%)

Followed by the assignment 2, our input images are the transformed A4 paper with some hand-written digits on it. Your task is to use a certain segmentation algorithm to segment each digit. Note that you can choose any segmentation methods. **However, no matter which algorithm you choose to use, you should implement it by yourself**. Please describe your segmentation algorithm briefly in the PDF file. If the algorithm you use is a typical segmentation algorithm, for example OSTU algorithm, you just need to write down the name of the algorithm.
The function prototype is given in the digit_segment.m.
A sample output is provided as follows,

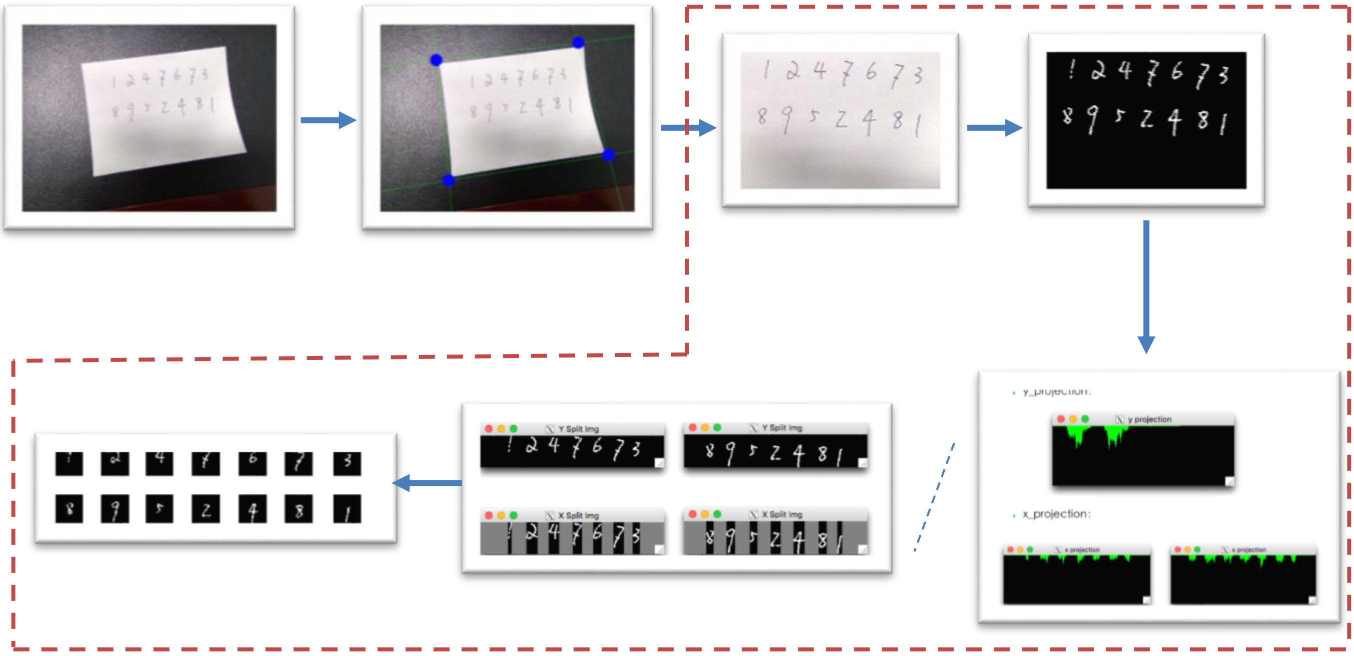


Here, I provide one of the algorithm I used in my program for the digit segmentation.

Step 1: Threshold the input images to obtain binary images;
Step 2: Horizontal projection. Count the number of white pixels along horizontal direction. The number of rows for the digits should equal to the number of peaks in the projected histogram. We can then segment the digits by rows.
Step 3: Vertical projection. For each row, we count the number of white pixels along vertical direction. The number of columns for the digits should equal to the number of peaks in the projected histogram. We can then segment the digits by columns.

[Note: for step 3 you may face the problem that some digits cannot be separated properly. Here is another idea for Step 3. Scan each column and count the number of black pixels, denoted as N. If N > H – n, where H is the height of the row_image obtained in Step 2 and n is a parameter defined by user, the column will be marked. The segmentation will be based on the marked column. This idea, which is similar to vertical projection, compares the height of the row_image and thus is more controllable.]

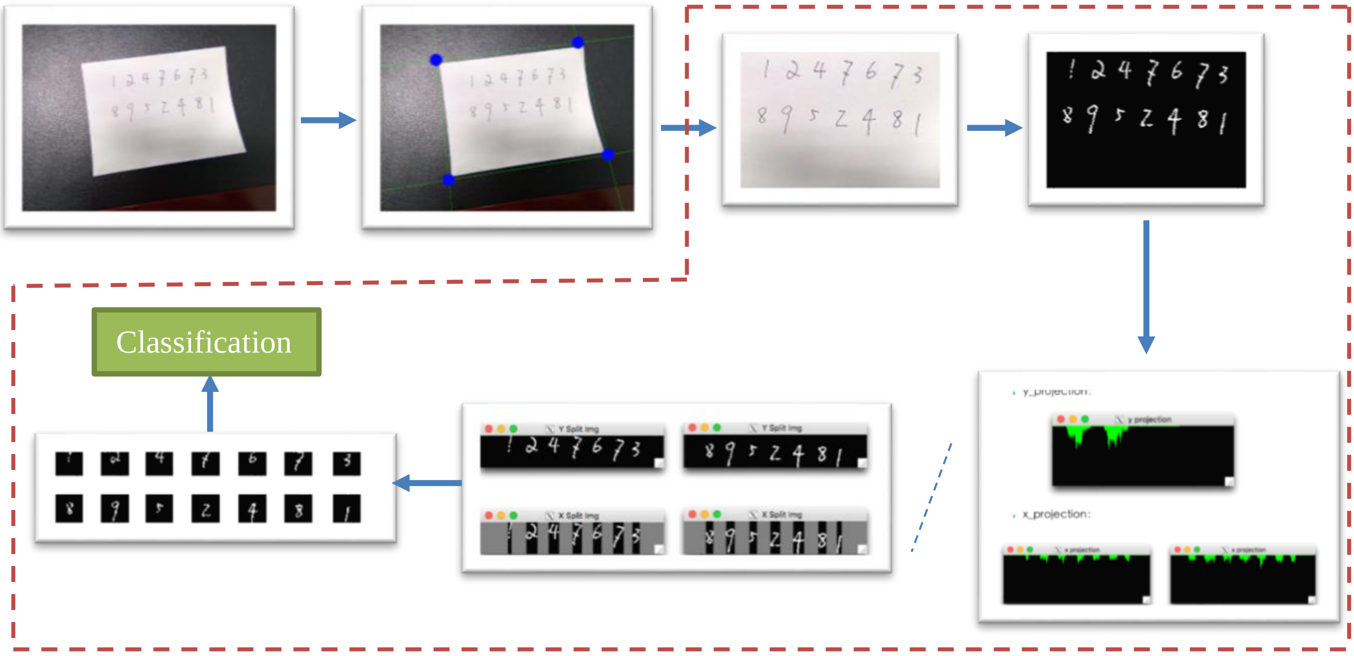
I provide the pipeline combining assignment 2 and assignment 3 as follows. Note that only the parts in the orange dash box is required in this assignment



2.3 Classification via Adaboost (30%)

In this part, you are required to implement the Adaboost algorithm to classify the segment digits. You should use MNIST dataset as training data, which contains 60000 training examples. **Please report your classified accuracy per image in the PDF file**. Instead of directly using the segmented digits, you may consider to compute some features, for example local binary pattern (LBP) or PCA, as input for the classification. Remember to compute the same features in the training data to train the classifier. You can download the MNIST dataset by http://yann.lecun.com/exdb/mnist/ .
The function prototype is given in the ada_classification.m.
For the weak classifiers, you can choose to implement by yourself or just directly use the MATLAB built-in functions or third-party libraries. Please report which classifier you use in the PDF file (Just write down the name of the classifier.)

The completed pipeline is as follows,



Classification