The Hong Kong University of Science and Technology
Department of Computer Science and Engineering
CSIT 5410 (Spring 2021)

Assignment 3

Total = 100 marks
Due: 11:55pm, 14 May 2021
Assignments must be submitted via Canvas
Late Policy: 10% reduction; only one day late is allowed, i.e., 11:55pm, 15 May 2021

Overview

This assignment consists of two sections: programming section and written section. Both sections should be submitted via the Canvas system.

In the programming section, you need to finish two tasks. The skeleton code is prepared in MATLAB and can be obtained from CANVAS. You need to complete the missing implementations in the corresponding M-files.

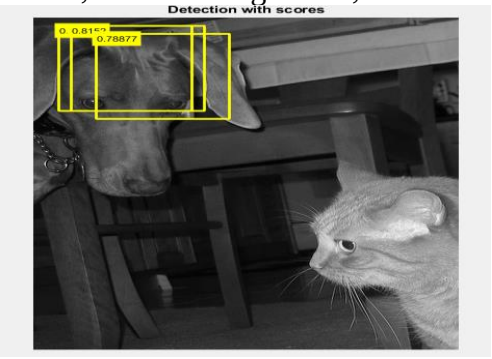
In the written section, you need to answer one question about local binary pattern. If you would like to finish the written assignment with handwriting, you may scan and upload it.

You must compress all your files with the following filename format: [your 8-digit student ID]_assign3.zip, e.g.: 12345678_assign3.zip, into one file. Your compressed file should include: (1) All M-files, mat-file, and test images related to the programming section, (2) a PDF file "report.pdf" including the report of programming section and the answer of the written section, and (4) a README.txt file indicating the programming software (Octave/MATLAB) that you are using for this assignment. Please **Do Not Include** the VOC 2007 dataset into your submission.

If your compressed file has been submitted multiple times (including late submission date), the newer version will replace the old version in marking.

Programming section (85%)

In this section, you are required to implement a simple object detection system with the Adaboost algorithm. Specifically, the object detection system can find out all the dogs inside an image and label it with a bounding box. The object detection system consists of feature extraction, a set of weak classifiers, Adaboost algorithm, and sliding window.



Prerequisite

Before you start, you need to download the PASCAL VOC 2007 dataset and the toolkit of the dataset at http://host.robots.ox.ac.uk/pascal/VOC/voc2007/index.html.

The PASCAL VOC 2007:
http://host.robots.ox.ac.uk/pascal/VOC/voc2007/VOCtrainval_06-Nov-2007.tar

The toolkit:
http://host.robots.ox.ac.uk/pascal/VOC/voc2007/VOCdevkit_08-Jun-2007.tar

And download the starting code and the testing images on CANVAS.

After that, create an empty folder "{{your 8-digits student ID}}_assignment3", unzip the above files into the folder and put the provided csit5410_test.txt into the folder VOC2007\ImageSets\Main.. Your workspace for assignment 3 should contain:

"{{your 8-digits student ID}}_assignment4"

```
-test_images            % Images for testing the object detection system
-VOC2007                % VOC 2007 dataset with annotations
-VOCcode                % Some helper functions for VOC 2007
-csit5410_assignment4.m  % Main function, Programming Task 3
-feature_extract.m       % Programming Task 1
-train_weak_classifier.m % Programming Task 2
-viewanno.m              % Example of reading an image with its annotation from VOC 2007
-and some other *.m files % Not necessary for this assignment
```

Programming task 1 - Feature extraction (20%)

In this task, you are required to complete the function in `feature_extract.m`. The function `feature_extract`, with format `fea= feature_extract(Im)`, computes the feature vector `fea` from the input image `Im`. The feature type must be either the Harr-like feature (Rectangle features) or the Local binary patterns (LBP) described in the lecture. You can modify the arguments list of the function. Third-party libraries or any related build-in functions **Are Not Allowed**.

If you are not able to finish this function, you may use "`extractLBPFeatures`" from MATLAB. In that case, no marks will be given for this task.

Programming task 2 - Weak classifiers (20%)

In this task, you are required to complete the function in `train_weak_classifier.m`. The function `train_weak_classifier.m` with format `model=train_weak_classifier(feature_type, ...)`, returns a trained weak classifier `model`, trained with feature specified by the input string `feature_type`. In this assignment, you are required to create ***at least five unique weak classifiers.*** Each weak classifier should be different in feature type/feature-length/classifier. For example, Harr-like features computed on different set of spatial positions can be regarded as different features.

2.1 Training data

For each weak classifier, it must be trained with images specified in VOC2007\ImageSets\Main\dot_train.txt. The example of visualizing the training data can be found in `viewanno.m`, e.g.: Run the command `viewanno('Main/dog_train')`. Note that it is not necessary to utilize all the images specified in dog_train.txt.

2.2 Pre-processing (optional)

As the images may vary in light condition or size, pre-processing may be necessary for object detection. A suggested pre-processing pipeline is shown below:

1. Convert the RGB image into grayscale using rgb2gray function.
2. Histogram equalization.
3. Resize the bounding box images into fixed size h x w, e.g.: 128 x 128.

2.3 Training a classifier

Given a pre-processed image cropped by the annotation (bounding box), the purpose of the classification is to classify the image into two categories, "dog" or "non-dig". Firstly, you are required to compute the feature vectors from a set of pre-processed images using the function in `feature_extract.m` (Programming task 1). After that, fed the computed feature vector and the ground truth ("dog" or "non-dog") into a custom classifier, e.g.: Support-vector machine (SVM) or other classifiers. For the classifier, you **Are Allowed** to use any built-in function related to it. The example of using SVM for binary classification is available in https://www.mathworks.com/help/stats/support-vector-machines-for-binary-classification.html.

2.4 Model size limitation

To facilitate the grading process, you are required to save your trained classifier into "{{model name}}_model.mat" using the command "save(filename,variables)". The model/variable size for each weak classifier **must not exceed 15MB**. If your size of the model is exceeding the size limit, you may need to reduce the length/dimension of the computed feature vector.

Programming task 3 - Adaboost algorithm and sliding window (30%)

In this task, you are required to complete the script in `csit5410_assignment3.m`. Specifically, this script first loads all the trained weak classifiers (the mat-file that you saved in part 2.4) and classifies the unseen validation data specified in dog_val.txt. Then, an Adaboost algorithm is applied to all the weak classifiers to compute a strong classifier. Finally, the strong classifier classifies the image patches, extracted by a sliding window algorithm, to locate the object "dog" within the images (provided in the "test_images" folder).

3.1 Weak classifiers

The script `csit5410_assignment3.m` first load all the pre-trained weak classifiers, e.g.: using the function "`load(filename)`". Then, these weak classifiers are used to classify all the annotated image patches specified in the given csit5410_test.txt. You are required to report the classification result for all the weak classifiers in the followings format:

```
>> Correctness (Weak Classifier 1): {{number of correct prediction}}/{{number of image patches}}
>> Correctness (Weak Classifier 2): {{number of correct prediction}}/{{number of image patches}}
...
```

3.2 Adaboost Algorithm

After section 3.1, you are required to implement the Adaboost algorithm described in the lecture. This Adaboost algorithm takes all the weak classifiers and its prediction results on images in "VOC2007\ImageSets\Main\dog_val.txt" as input and outputs a final strong classifier, which formed by at least 5 weighted weak classifiers. Similarity, you are required to report and print out the classification result for the strong classifier in the followings format:

```
>> Correctness (Strong Classifier): {{number of correct prediction}}/{{number of image patches}}
```

3.3 Sliding window

You are required to implement the sliding window algorithm to extract fixed size, e.g.: 128x128, image patches from images in folder "test_images". The concept of the sliding window is described in https://www.pyimagesearch.com/2015/03/23/sliding-windows-for-object-detection-with-python-and-opencv/. Noted that the stride for the sliding window is flexible. You can fine-tune the stride size to speed up your algorithm. For each image patches extracted by the sliding window, a strong classifier (computed in section 3.2) is applied to classify whether the image patch contains a "dog" object or not. Finally, you are required to display (e.g. imshow()) the detection results overlay with three top-3 scores/confidences bounding boxes for each image in the folder "test_images" as followings:

Note that we will test your algorithm with a set of new testing images.

3.4 Running time

The script `csit5410_assignment3.m` must be completed **within 10 minutes** in Virtual Barn's MATLAB. Running time for `csit5410_assignment3.m` exceed the time limit will subject to marks deduction. Note that you must make sure all your codes can be run without a runtime error. Otherwise, **no marks will be given.**

Programming task 4 – Report (15%)

You are required to submit a report describing the details and results of your object detection system in PDF format and rename it as "report.pdf". Your report should cover the followings:

- A brief explanation of your weak classifiers
- A brief explanation of your preprocessing methods (if any)
- A screen capture of the selected weak classifiers and its weight after the Adaboost algorithm
- A screen capture of the classification accuracy of each weak classifier and strong classifier on images specified in csit5410_test.txt
- A screen capture of the detection results of the given images in the "test_images" folder, a maximum of 3 bounding boxes per image.

Written section (15%)

Local Binary Pattern

You can include your answer for the written section in "report.pdf"

(1) Given one local patch with intensity values as follows, please calculate the feature vector for the center point using LBP. (Let the right-bottom pixel is the most significant bit. The counting is in clockwise direction.)

20	67	30
15	67	84
65	86	**21**

(2) If we rotate this input patch by 180 degree, what is the LBP for the center point then?

(3) Due to the changes of illumination condition, the intensity values of this patch have been affected and changed to

21	70	32
16	69	82
66	86	**20**

Please give the LBP for this effected patch.

~~ End of Assignment 2 ~~