The University of Saskatchewan

Saskatoon, Canada

Department of Computer Science

487/819– Computer Vision and Image Processing

Assignment 5

Date Due: November 22, 2018

Total Marks: 58

1 Submission Instructions

- Assignments are to be submitted via the Moodle class page.
- Programs must be written in Python 3.
- No late assignments will be accepted. See the course syllabus for the full late assignment policy for this class.

2 Background

The purpose of this assignment is to explore the use of shape and texture descriptors to classify images.

2.1 Assignment Synposis

In Question 1 you will classify leaf shapes described by the Histogram of Curvature Scale feature. In Question 2 you will classify texture images using GLCM texture features and local binary patterns.

2.2 Datasets

Datasets for question 1

For question 1, you will use images similar to the ground truth images in assignment 3. You will download two zip files:

`leaftraining.zip` These files naming scheme is `threshimage_????.png` where the wildcard characters represent digits of a four-digit number. The first ten files (in lexicographic order) are of class 1, the next ten files are of class 2, and the final ten files are of class 3. The images are binary PNG images.

`leaftesting.zip` This zip contains 129 test images. These are binary PNG images with filenames of the form `image_????.png` where the wildcard characters represent digits of a four-digit number. The first 50 files (in lexicographic order) are of class 1, the next 27 are of class 2, and remaining 52 are of class 3.

Datasets for question 2

For Question 2 you will use images of texture. Each image contains the same texture throughout.

`brodatztraining.zip` This archive contains 15 *training images* for each of 8 texture classes (120 images in total). These are PNG images with the names `trainingPatch-??????.png` where the wildcard

entries represent a six-digit number. The first digit of the number indicates the class label for the training patch. The remaining digits are random. When you read these filenames with `os.walk()` function, the first 15 files will be samples of texture class 1, the next 15 will be for texture class 2, and so on. The training samples are *unrotated* samples of the texture classes.

`brodatztesting.zip` You are also provided with 40 *test images* for each of the 8 texture classes (320 images in total). These are PNG images with the names `patch-??????.png`. Again, the wildcard entries are a six-digit number. The first digit indicates the class label. Similar to the training data, when you read the filenames with `os.walk()`, the first 40 elements of the array will be of texture class 1, the next 40 will be of texture class 2, and so on. To make things interesting, each test image is a sample of the original texture but rotated by a random angle between 0 and 360 degrees. This will enable us to get a good feeling of how invariant to rotation our texture features are. Note: the test images contain some imperfections that were deliberately left in the data set to make the problem a bit more interesting and challenging (but not too challenging!).

2.3 Confusion Matrix

A confusion matrix is a quick way to visualize both how well a classifier is performing, and which pairs of classes are causing the most problems (in terms of mis-classifications). It is considerably more informative than the overall classification rate. The confusion matrix is a square matrix of size N where N is the number of classes and entry (i, j) (row i, column j) is the number of times an image of class i was classified as class j. Naturally then, if the classification rate is 100%, the confusion matrix is a diagonal matrix with all off-diagonal entries equal to zero.

Example 1
In the following confusion matrix, most samples were classified correctly. 4 samples of class 1 were incorrectly classified as class 2. 5 samples of class 6 were incorrectly classified as class 7, etc. At a glance, we can see that there were a few instances of problems distinguishing between classes 1 and 2, 1 and 3, and 6 and 7. Additionally there were some rare, isolated confusion between other classes.

	1	2	3	4	5	6	7	8
1	35	4	0	0	0	0	0	0
2	0	36	0	0	0	0	0	0
3	4	0	39	0	0	0	0	1
4	0	0	0	39	0	0	0	0
5	0	0	0	0	40	0	0	0
6	0	0	1	0	0	40	5	0
7	1	0	0	1	0	0	35	0
8	0	0	0	0	0	0	0	39

2.4 Classification Rate

The *classification rate* is the percentage of images (feature vectors) that were correctly classified. In Example 1, there were a total of 17 misclassifications over 320 images, for a classification rate of $303/320 \times 100\% = 94.7\%$.

3 Problems

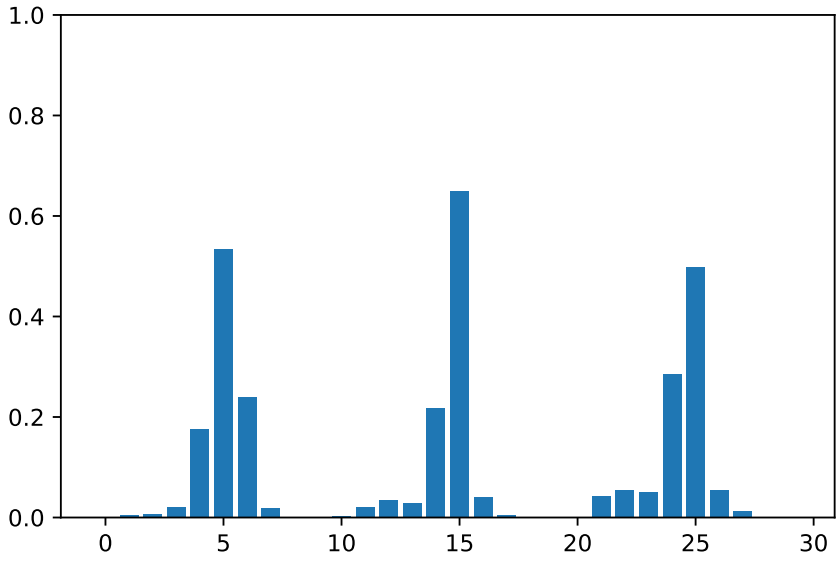
Question 1 (29 points):

In this question you will classify pre-segmented images of leaves using the histogram of curvature scale feature.

Detailed instructions are provided `asn5-q1.ipynb`. You should be able to get a classification rate of more than 95%.

Step 2 Sample Output

If your output from step 2 looks like this, then your `HoCS` function is probably working.



Question 2 (29 points):

In this question you'll classify images of various texture into eight different texture classes. Again we will use a K-nearest neighbour classifier.

Detailed instructions are provided in asn5-q2.ipynb.

Using GLCM features, if you're beating 65% correct classfication rate, you're doing excellently. For LBP/VAR8 features you should be able to get much higher; it shouldn't be too hard to exceed 95%.

4 Files Provided

`asn5-qX.ipynb`: These are iPython notebooks, one for each question, which includes instructions and in which you will do your assignment.

`leaftraining`: Folder containing training images for classifier in Question 1.

`leaftesting`: Folder containing testing images for classifier in Question 1.

`brodatztraining`: Folder containing training images for the classifiers in Question 2.

`brodatztesting`: Folder containing testing images for the classifiers in Question 2.

5 What to Hand In

Hand in your completed iPython notebooks, one for each question.

6 Appendix A — Grading Rubric

The grading rubric can be found on Moodle.