CS 470: Assignment 04

Programming Assignments (95%)

The goal of this assignment is to write code to **extract LBP features from a set of images**.

A04.py

You must provide code for the following functions in the file:

- **getOneLBPLabel(subimage)**
- **getLBPImage(image)**
- **getOneRegionLBPFeatures(subImage)**
- **getLBPFeatures(featureImage, regionSideCnt)**

You MUST implement the LBP algorithm yourself; however, you may use OpenCV and/or Numpy histogram functionality.

If you are not sure whether a library function is permitted, ASK ME!

getOneLBPLabel(subimage)

Given a (grayscale, uint8) 3x3 subimage (basically the neighborhood of a single pixel), return the correct LBP label for that pixel. **Please note that, for thresholding, value > center to be 1 (otherwise 0).**

You will only implement the **uniform, rotation-invariant version of LBP** (10 labels total)

getLBPImage(image)

Given a (grayscale, uint8) image, generate and return the uniform, rotation-invariant LBP label image.

- Radius will be 1 and the sample count is 8, so you may directly use the 8 pixel neighbors (strong and weak) for each pixel.
- The output label image will be the same size and type as the input image.
- If you extend out of bounds, assume zero padding. You may use cv2.copyMakeBorder (with a padding of 1 on all sides).
- Loop through each pixel in the image, cut out the appropriate subimage, and call getOneLBPLabel to get the correct output label per pixel.

getOneRegionLBPFeatures(subImage)

Given an **LBP label image** (NOT the original image!), compute and return the correct LBP histogram.

Do NOT forget to normalize the histogram by dividing by the total number of pixels!

Make sure that the histogram is the correct length: **UNIFORM** = 10 elements

getLBPFeatures(featureImage, regionSideCnt)

Given an **LBP label image** (NOT the original image!) and the number of subregions on each side of the image, you will need to do the following:

- Compute the subregion width and height in pixels. **Use either integer division or floor(). This means that your subregions may not cover the entire image; that is acceptable.**
- Start with an empty list to hold all of the individual histograms.
- Loop through each possible subregion, ***going row by row and then column by column***.
    - Extract the subimage (using the precomputed subregion width and height as well as the starting point). **Subregions do NOT overlap!**
    - Call getOneRegionLBPFeatures() to get the one subimage's histogram.
    - Append this histogram to your list of all histograms.
- Convert your list of histograms to an np.array() and then reshape so that it is a flat array:
    - allHists = np.array(allHists)
    - allHists = np.reshape(allHists, (allHists.shape[0]*allHists.shape[1],))
- Return allHists

Testing Screenshot (5%)

I have provided several files:

- **Test_A04.py** – the test program
- **General_A04**.py – code shared by the test programs
- **assign04/**
    - **images/** – folder containing the input images
    - **ground/** – folder containing the ground truth LBP label images and the csv files

Copy these files/folders into the SAME directory.

Grading

Your OVERALL assignment grade is weighted as follows:

- 95% - Programming assignments
- 5% - Testing screenshot