CS 484/555, Spring 2022

BONUS ASSIGNMENT:

This homework assignment will have a +10% contribution to your total grade over and above 100%. If you prefer not to do it, it will not affect you grade adversely.

Homework Assignment 3: Image Segmentation

Due: Monday, 9 May 2022, 11:59 PM

Image segmentation is a fundamental problem in computer vision. The goal of this assignment is to obtain an accurate segmentation of images texture information. A common problem in segmentation algorithms is that it is usually hard to find a fixed set of parameters that work well for all images in a data set. The same set of parameters may provide an oversegmentation for some images while they may cause undersegmentation for the others.

In this assignment, we will try to obtain a good segmentation by, first, oversegmenting the images, and then, combining the regions that have common texture characteristics using clustering. The data set for this assignment consists of 10 images, shown in Figure 1, that are taken from the Breast Cancer Histopathological Database (BreakHis) (`https://web.inf.ufpr.br/vri/databases/breast-cancer-histopathological-database-breakhis/`).

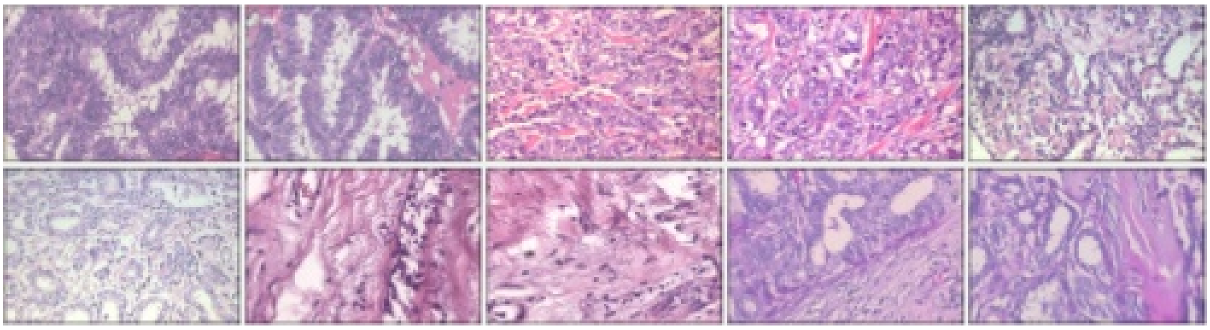

Figure 1: 10 images in the data set of this assignment.

Part 1:

The first step is to obtain an oversegmentation for each image. First, read the following paper (provided along with this assignment):

> R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua, S. Susstrunk, "SLIC Superpixels Compared to State-of-the-art Superpixel Methods," IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 34, num. 11, pp. 2274–2282, May 2012.

The algorithm described in the paper can be used to obtain an oversegmentation of an image into superpixels that are locally homogeneous groups of pixels that preserve the object boundaries. Oversegmentation using superpixels can be used as a preprocessing stage that provides an abstraction and simplifies the computation at later stages.

Second, use the code for the above paper available here to obtain superpixels for each image in your data set. There are two versions (SLIC and SLICO). You can use either version. Read the instructions for the version you choose carefully. The output that will be necessary in the rest of the homework assignment is a matrix that contains an integer label for each pixel where

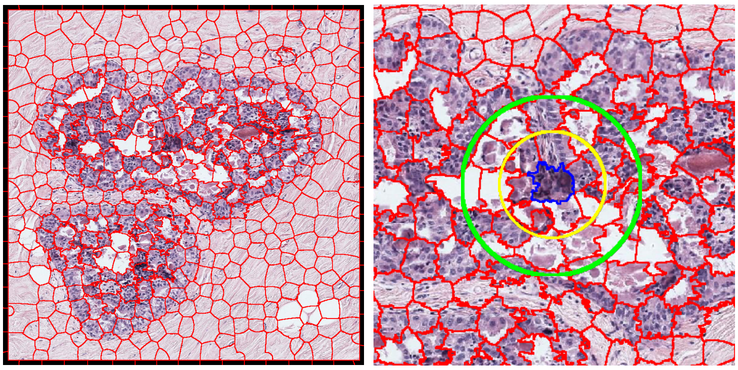

Figure 2: left: Superpixels by the SLIC0 algorithm; right: Circular neighbourhoods around the blue superpixel.

each label indicates the corresponding pixel's superpixel id. You may need to experiment with the parameters in the code to obtain a meaningful oversegmentation.

After trying the code on several images, fix the parameters and use the same settings for segmenting all images. The output of this part for each image is an integer label image where each pixel stores the id of the region it belongs to.

Part 2:

The second step is to compute Gabor texture features for all images. You can use the Gabor filter implementation in Matlab's image processing toolbox (`https://www.mathworks.com/help/images/ref/imgaborfilt.html` or the Matlab implementation by Dr. Peter Kovesi (`http://www.peterkovesi.com/matlabfns/PhaseCongruency/gaborconvolve.m`). You can adjust the parameters so that you obtain a filter bank of 4 scales and 4 orientations. You can use the suggested default values for most of the parameters. Note that Gabor filter responses should be computed on the grayscale version of the input image, and the magnitude of the output response should be used in the rest of the assignment. At this step, for each pixel, you will have one response for each filter, e.g., you will have a total of 16 responses per pixel if you use 4 scales and 4 orientations.

Once you have Gabor representations for each pixel in an image, compute Gabor features for all superpixels by simply computing the average of Gabor features of the pixels inside a particular superpixel. Assume that you have N_i superpixels extracted from an image i, after this step you will have an $N_i \times 16$ feature matrix for image i.

Part 3:

Now, each superpixel is represented as a point in the 16-dimensional feature space. You can use any clustering algorithm (e.g., k-means) to group the superpixels. You can use clustering code from other sources but you are required to know the details of the particular implementation used. The clustering algorithm must be applied to the whole data set (not separately for each individual image), i.e., feature vectors of all superpixels of all images (or a random sample if you have memory issues) should be given to the clustering algorithm. The output at this step is a matrix that contains an integer (cluster) label for each superpixel. You can use these labels to produce a pseudo colour representation of the clustering result by colouring the pixels corresponding to each superpixel according to the cluster label of that superpixel.

You need to experiment with the design parameters (e.g., superpixel size, Gabor filter bank size, number of clusters) to obtain a meaningful clustering. You should provide an output pseudo colour image for each image in the data set.

Part 4: The final step is to build a contextual representation for each superpixel and repeat the clustering process. First, you will need to find the neighbouring superpixels for each superpixel. We define two levels of neighbourhoods. For a given superpixel, the first level neighbours consist of the superpixels that have a number of pixels in a ring around the given superpixel. In Figure 2, this first level ring is represented by the yellow circle. That is, the superpixels that are in the first level neighbourhood of the blue superpixel are the ones that have a number of pixels within the yellow circle (except the blue superpixel). Given the expected size of a superpixel and the diameter of its equivalent circle, the yellow ring corresponds to the circle which is centred at the centroid of the blue superpixel and have a radius of 1.5 times this diameter. You can set a threshold for the number of pixels required to accept a superpixel to be inside a given ring. The second level neighbourhood is defined similarly and is shown by the green ring in Figure 2. The superpixels that have a number of pixels within the ring bordered from outside by the green circle and from inside by the yellow circle are the ones in the second level neighbourhood of the blue superpixel. The radius of the green circle can be set as 2.5 times the expected diameter of a superpixel. Once the neighboring superpixels are identified, you should compute an average feature vector from the vectors of the superpixels within the first level neighbourhood and a separate average feature vector from the vectors of the superpixels within the second level neighbourhood. Then, the final contextual feature representation for the superpixel is constructed by concatenating the feature vector of this superpixel, the average vector computed for the first level neighbourhood, and the average vector compared for the second level neighbourhood. For example, when the feature vectors for superpixels have lengths of 16, the concatenated feature vector will have a length of 48. You should repeat the clustering process by using the contextual feature vectors. You need to experiment with the design parameters (e.g., threshold for accepting a superpixel to be within a ring, and possibly the radius of the first and second level neighbourhoods) to obtain a meaningful clustering. You should provide an output pseudo colour image for each image in the data set.

Submit:

1. Description of the parameters used for superpixel segmentation.
2. Segmentation results for all images using the superpixel segmentation algorithm (SLIC version) that you chose. Note that you must produce oversegmentations using the same parameters for all images.
3. Description of the parameters used for Gabor texture feature extraction.
4. Gabor texture feature examples (e.g., output of the `imgaborfilt` or `gaborconvolve` functions) for several scales and orientations for at least two different images.
5. Clustering/segmentation results for all images. You must show the obtained regions by overlaying segment boundaries on each image or overlaying the pseudo colouring result with some transparency on the image as shown in the lecture slides.
6. Any source code that you wrote for the assignment.
7. Citation for any external code used.
8. Detailed discussion of the results with respect to different parameter settings (including all parameters listed above and descriptions of how and why you chose the particular values).

Notes:

This assignment is due by Monday, 9 May 2022, 11:59 PM. An important part of this assignment is the discussion of the results where you must comment on the effectiveness of the segmentation/clustering for different images with respect to different settings.

You should submit your solutions as a **single archive file** that contains your **code** and **report** (a pdf file that contains the description of the methodology, the resulting images, descriptions of how you obtained them, and discussion of the results) to Moodle. Please note that for this bonus assignment, no deadline extensions, or late submissions will be applied.