CSCI 332 Fall 2022 - Software Assignment #2 (v4)

Major changes since the last version are indicated in ***bolded*** blue font.

Prompt: Implement (i) Ford-Fulkerson algorithm, and then use it in (ii) an implementation of the segmentation algorithm described in Section 7.10 of the Kleinberg/Tardos (KT) textbook. For full credit, you must implement the algorithm as described in Section 7.10 of the Kleinberg/Tardos textbook. Let $x \in N^2$ be row/column coordinates of a pixel, and $I(x) \in N^3$ RGB values of a pixel at location x. Per the implementation in Section 7.10, each pixel is treated as a graph node and must have weights for (i) the likelihood of that pixel being background, denoted here $b(x)$; (ii) the likelihood of that pixel being a foreground, denoted here $a(x)$; and (iii) the weights between neighboring pixels, denoted $p(x_i, x_j)$, where x_i and x_j are the coordinates of two pixels, i and j. You must set these values in the following way:

$$a(x) = 442 - round\left(d\left(I(x), I(x_a)\right)\right) \tag{1}$$

$$b(x) = 442 - round\left(d\left(I(x), I(x_b)\right)\right) \tag{2}$$

$$p(x_i, x_j) = \begin{cases} p_0, & d(x_i, x_j) < 2 \\ 0, & otherwise \end{cases} \tag{3}$$

In these equations $d(I(x), I(x_a))$ is the Euclidean distance between the RGB values at pixel coordinate x and x_a, for example. The coordinates $x_b, x_a \in N^2$ are coordinates of one background and one foreground pixel, respectively. This is analogous to the way segmentation software is often used in practice; the user manually chooses one location in the image (i.e., a pixel coordinate) that represents the foreground they want to isolate, and one location in the image that represents the background.

You can only utilize the following primitive python data structures inside your python class definition: python lists, python dictionaries, and numpy arrays. You can only utilize primitive python and numpy functions, such as sum, multiplication, division, exponentiation, logarithms, etc. *One exception* is that you may use someone else's implementation of breadth-first-search or depth-first-search for use in Ford-Fulkerson. If you are in doubt about any other functions, then please ask. Regardless of what operations and data structures you use, you must conform to the API provided below. I will provide 1-2 input/output examples, aimed at ensuring that your code is compatible with the API, and working properly.

How to submit the assignment:

- Submission on Moodle by 11pm MST on 12/10/2022.
- I recommend submitting early - at 11:01pm it will be considered a day late.
- 5% off per day that the assignment is late, up to 5 days (11pm on Dec. 15th). 0% credit if submitted beyond 5 days after deadline
- Note: you are *not* permitted to drop a software assignment, but you permitted to postpone your submission, if you didn't already do so on the first software assignment. Due to end of the semester, you must submit by 11:01pm on Dec. 15th if you use your postponement.
- You must submit the following files:
    - A .py file that contains the definition for your python class, per instructions below.
    - A .py file that runs tests your python class, per instructions below.
    - A .png image file for testing your script

Additional Software Specifications and I/O:

- All code must be written in Python.
- You must use *exactly* the function names described below.
- You may have additional functions in your class, as long as the requested functions below are also included
- Your class must have the following properties that can be set by the user
  - segmentationClass.p0 – an **integer** value greater than one. This parameter will be used as p_0 in equation (3)
  - segmentationClass.x_a - a 1x2 numpy array specifying the coordinate (row and column,respectively) of a location in the image representing the foreground.
  - segmentationClass.x_b – a 1x2 numpy array specifying the coordinate (row and column,respectively) of a location in the image representing the background.

obj = segmentationClass()

- **Input**: none.
- **Output**: an object instantiated from your class. Your class should have the name as shown here 'segmentationClass'.

L = obj.segmentImage(I)

- **Input**: I is an NxNx3 numpy array representing a color (RGB) image. Each pixel intensity will be have an integer value between 0 and 255.
- **Output**: L is an NxNx1 numpy array of binary values representing whether each pixel in the image is in the foreground (value of 1) or in the background (value of 0). So, if pixel at row (i, j) is in the foreground then $L[i, j] = 1$, and if it is in the background we have $L[i, j] = 0$.

Grading criteria

- **Valid Testing Script** (50%): You must provide a .py "testing script" with your class that performs the following steps. Each of these steps must come with at least one comment that indicates what you are doing (10% points off if this is not done). You will lose 20% points for each one of the following steps that is missing, up to 50% points:
  - Import your python class, assuming the class definition .py file is in the same file directory as the test script
  - Load a test image, assuming the test image is in the same file directory as the test script
  - Display the image using matplotlib package
  - Instantiate your class, and set the hyperparameters
  - Input the image to the segmentImage() function using the API specified above
  - Display the adjacency matrix for the graph nodes corresponding to the pixels at location (0,0) and (1,0) in the image. The entries in the adjacency matrix should be positive if there is a connection between the nodes, with a value indicating the edge capacity between those nodes, and zero if there no edge connecting two nodes.
    - **You may alternatively display adjacency list output for these two pixel locations, as long as (i) the connected nodes and capacities are indicated, and (ii) it is clear where I can find this information (e.g., put in some comments explaining how to interpret it).**

    - Display the segmented image in black and white color using matplotlib, where segmentation values of 1 correspond to foreground (color of white), and values of 0 correspond to the background (color of black)
- **Correctness** (50%)**:** I will input one or two 25x25 pixel images of my choosing into your code to verify that it produces the correct output for the top-level segmentation, as well as some intermediate steps. I will also vary the user-chosen hyperparameters such as x_a, x_b, and p_0 and verify that the output changes properly.

A Note on Teamwork and Cheating:

- I may manually inspect or use software to check the similarity of your code to software online (e.g., github), or among other submitted sets of code. If I find similarities in your code to other existing/submitted code, I reserve the right to ask you questions about your submitted software. *If asked, you should be able to explain all portions of your code in detail.* If not, I may dock a substantial portion of points from your software grade, and potentially take more serious action.
- For these reasons, I strongly encourage you not to inspect a specific implementation of the algorithm created by someone on the internet, or by someone else in the class. It is crucial that you attempt to write the code yourself to ensure that you understand it well, and can explain your code, and the algorithm, if asked.
- You are welcome to discuss the assignment and your algorithmic strategy with others, but for the reasons outlined above, you should be cautious and only discuss it at a broad level.
- Conclusion: you are responsible for being capable of explaining your code, in detail, if I were to ask you about it.