Introduction

The main goal of this homework is perform semantic labeling of images from a benchmark dataset and to evaluate your labeling procedure. You will be provided with a set of 715 benchmark images of urban and rural scenes created by [1], and commonly known as the Stanford background dataset. These images have approximately approximately 320×240 pixels, they contain at least one foreground object and have the horizon positioned within the image (it need not be visible). Each image in the dataset has already been oversegmented into superpixels and the ground-truth labels for each superpixel will also be provided. There are eight (8) classes representing the semantic labels to which superpixel must be assigned. these classes are: `sky`, `tree`, `road`, `grass`, `water`, `building`, `mountains`, and `foreground objects`.

You will be required to perform 5-fold cross-validation with the dataset, which has already been randomly split into five different sets of 572 *training images* and 143 *test images* for each fold. In order to accomplish this, we have preprocessed the data for you so that each image in the dataset has already been over-segmented into superpixels and a correct label has been assigned to each superpixel. You will therefore need to extract different sets of features from the superpixels of the training images, select your own classifier(s), train the classifiers on the eight classes mentioned above and label the superpixels in the training set. You will perform this 5 different times for cross-validation. The folder accompanying this assignment contains different functions to assist you wit feature extraction and result visualization.

The Matlab binary file *cvip_image_data.mat* containing the data for this assignment can be accessed at the link below:
`https://www.dropbox.com/s/i64idp4wn1erifk/cvip_image_data.mat?dl=0`

Please note that this final project will NOT focus significantly on your code implementation, but rather, it will be graded mainly on your evaluation framework and the quality of your final report. You will still be expected though, to submit the all code that led to the results in the paper.

Requirements

You should perform this assignment in Matlab. It is due on **Monday Dec 15th by 5pm (no extensions please)**. You are strongly encouraged to start the assignment early and don't be afraid to ask for help from either the TAs or the Instructor. You are also welcome to ask questions and have discussions about the homework on the course piazza but please do not post your solutions or any closely related material. If there are parts of the assignment that are not clear to you, or if you come across an error or bug please don't hesitate to contact the TAs or the Instructor. Chances are that other students are also encountering similar issues.

You are allowed to collaborate with other students as far as discussing ideas and possible solutions. However you are required to code the solution yourself. Copying others' code and changing all the variable names is not permitted! You are allowed to use solutions from similar assignments in courses from other institutions, or those found elsewhere on the web. But if you access such solutions YOU MUST refer to them in your submission write-up. As usual, your solutions should be uploaded to the CSE server using the CSE_SUBMIT

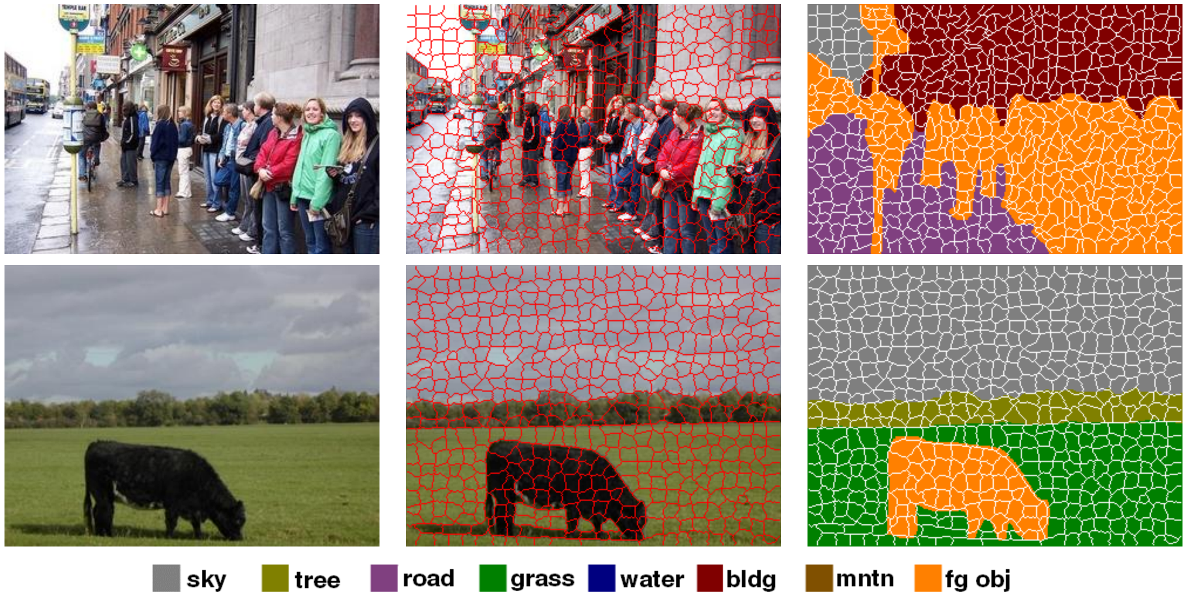

Figure 1: Shows 2 examples of images from the benchmark datset. The top row shows an urban image while the bottow shows a rural one. Left column: original image; center column: superpixels overlaid on the original; right column: superpixels label. Each superpixel is ultimately assigned a label value (from 1 - 8) and this is displayed in the right column. Lastly, the color indices for the 8 classes is shown at the bottom.

command (There is a tutorial on how to use CSE_SUBMIT) on the course piazza page.

The instructions, data and starting code for this assignment is provided in the zipped file **homework4.zip** which can be downloaded from Piazza. Your submitted zipped file for this assignment should be named **PersonNumber_hw4.zip**. Failure to follow this naming convention will result in your file not being being picked up by the grading script. Your zipped file should contain: (i) a PDF file named PersonNumber_hw4.pdf with your report, Details of the report content are described in more detail below; (ii) the source code used to generate the solutions (with code comments). Please include a working runfile for this assignment **(runfile_hw4.m)**. For grading, we will be evaluating the quality of your final report although we might run your test code in some instances.

Details of the *cvip_image_data.mat* file

The Matlab binary file *cvip_image_data.mat* containing the data for this assignment can be accessed at the link below:
`https://www.dropbox.com/s/i64idp4wn1erifk/cvip_image_data.mat?dl=0`

Below are the brief descriptions of the 4 variables contained in the binary *.mat* file provided:

(a) `image_data`: all image related information is stored here (e.g. super pixels, raw images, labels, etc)

(b) `label_color_map`: a color map to display labeled images

(c) `train_idx`: indices for the training images for 5-fold cross-validation

(d) `test_idx`: indices for the testing images for 5-fold cross-validation

The image_data variable is a Matlab structure containing the following data fields:

(a) `raw_image`: the original image, can view it with imshow

(b) `pixel_level_label`: the pixel level labels, used to compute the pixel level labeling accuracy

(c) `super_pixels`: the super pixels generated from the image

(d) `nsegs`: the number of super pixels generated from the image

(e) `npixels_in_super_pixels`: number of pixels contained in the corresponding super pixels

(f) `adjmat`: adjacent matrix of the superpixels

(g) `super_pixel_labels`: the label for corresponding super pixels

(h) `label_names`: the semantic label names for the corresponding label number

To visualize the contents of the files, for example, to view the files displayed in Figure 1, you can run the following commands in Matlab:

(a) To see the original image: `imshow(image_data(53).raw_image);`

(b) To see the pixel level labels: `imshow(image_data(53).pixel_level_label, label_color_map);`

(c) To see the super pixel level labels: `super_pixels = image_data(53).super_pixels;` `super_pixel_labels = image_data(53).super_pixel_labels;` `imshow(super_pixel_labels(super_pixels), label_color_map);`

(d) To overlay an image with the contours of superpixels, run `maskedim = drawregionboundaries(super_pixels, uint8(super_pixel_labels) , [255,255,255]);` See more information on `drawregionboundaries` by typing `'help drawregionboundaries'` in Matlab, after adding the subfolders to your path

Instructions

Feature extraction

For this project, we are interested in investigating how location and shape features, color features, and texture features affect the efficacy of our labeling. We will therefore extract and use different features in our classification task. Some suggestions of features to use as labeling cues are provided below. Several of these cues have already been extracted for you in the Matlab function `getSPixelFeatures` provided assignment folder. Your task here is to evaluate performance based on these cues and also to augment the cues to obtain as high accuracies as possible.

LABELING CUES
Location and shape
L1. Location: normalized y position, 10th and 90th percentile in the segment
L2. Shape: number of pixels in segment
L3. Shape: normalized area in image
Color
C1. RGB values: mean R, G, and B values in the segment
C2. HSV values: mean H, S, and V values in the segment
C3. Hue: histogram with 5 bins
C4. Saturation: histogram with 3 bins
Texture
T1. LM filters: mean absolute response (15 filters)
T2. LM filters: histogram of maximum responses (15 bins)

Training the classifier

Our approach to this problem is to compute all cues that might be useful for labeling and then allow our classifier to to decide which cues to use and how to use them. You are free to select your own classifier, but a default implementation has been provided for you in the Matlab scripts `trainSceneLabels.m` and `testSceneLabels.m`, which trains and tests on just one fold of the data using the built-in Matlab implementation of support vector machines (SVM).

Your task here is to tune your classifier or even try different classifiers on a good set of features to obtain as high accuracies as possible. You can do this only on you best performing validation fold.

Final report

Please see the new tab on the course webpage.

Also, the assignment folder contains two interesting papers that perform similar tasks to this one, albeit using more complicated labeling techniques. It might be useful to review how these papers are laid out and how they evaluate and present their results.

References

[1] S. Gould, R. Fulton, and D. Koller, "Decomposing a scene into geometric and semantically consistent regions," in *Proc. Int. Conf. Comput. Vis.,* 2009, pp. 18.