381V Visual Recognition, Fall 2017
Assignment 2: Recognizing scene categories

http://vision.cs.utexas.edu/381V-fall2017/

Out: Wed Sept 19
Due: Wed Oct 11, 11:59 pm,
with follow-up result due Friday Oct 13, 11:59 pm (see below)

Goal

For this assignment, the goal is to implement an image recognition system based on methods we have studied in class so far. You will evaluate your system on natural images containing 25 scene categories from SUN (Xiao et al., CVPR 2010).

`http://vision.cs.princeton.edu/projects/2010/SUN/`

Feel free to use existing code for *components* of your system; however, it is not permissible to use existing executables or complete end-to-end systems you might find online to compute your results. Cite any components you have borrowed clearly in your writeup.

There will be extra credit for the top 3 best performing systems, as gauged on a held-out test set we will provide later (see below).

Data

We have posted a local copy of the SUN database here:

`/projects/cs381V.grauman/`

There is also limited temporary disk space available here for class members for writing features, etc., if you need it. Please note the space is only temporary and will be reclaimed after our course ends.

We will be operating with a subset of the SUN Database for this assignment, consisting of 25 categories. So that results are comparable, you are to use the subset of training and test images specified in this file: **`/projects/cs381V.grauman/filenames.mat`** This mat file contains these variables:

- `classnames` is a cell array of 25 strings. It enumerates the category names for the 25 categories we randomly selected for inclusion in this assignment.

- `trainImNames` is a 25 x 100 cell array of JPEG file names. The {i,j}-th entry in `trainImNames` specifies the j-th training image for the i-th selected category. There are 100 training images for each of the 25 categories. Your code will need to read this list to load the training images, which as the file names show, are located in subdirectories of `/projects/cs381V.grauman/`. Note that `classnames`{i} gives the descriptive label for all images specified by `trainImNames`{i,:}.

- `test1ImNames` is a 25 x 25 cell array of JPEG file names, formatted as above. The {i,j}-th entry specifies the j-th testing image for the i-th category.

- `extraTrainImNames` is a cell array formatted like `trainImNames`, and points to extra training images that are optional for use for training. The number of extra training images varies per category.

What to implement

You are to implement an image recognition system that trains models for the 25 provided categories, and can predict the class label for a novel test image. The approach you take must be derived from one of the papers and methods we have studied in class so far, but the details are up to you. We will evaluate the assignment based on 1) your design is technically correct, 2) it follows a basic paradigm we've studied in class, and 3) the accompanying report demonstrates careful analysis and understanding of the method you have employed. For example, your system might be based on one of the following:

- Bag of words representation, computed on sparse or dense local features, combined with a chosen classifier (e.g., SVM with a spatial pyramid kernel).
- Boosting with rectangular features computed from integral images (following Viola-Jones)
- GIST or Histogram of Oriented Gradients (HoG) descriptors and SVMs
- Pre-trained AlexNet/VGG/etc. CNN features, combined with a chosen classifier.

- Fine-tuned CNN descriptors for our target task. Please note - for any of the CNN variants, you may *not* use any pre-trained networks that were trained using our database source, e.g., you may *not* use a network pre-trained with Places data `http://places.csail.mit.edu/` since SUN and Places overlap. You may use ImageNet pre-training.
- Nearest neighbors (if you choose this one, we will expect to see some effort put on the feature design and/or distance function used for finding neighbors)

See the course website for links to available code, feature extraction toolkits, etc.

We are posing two **challenge tasks** for this assignment:

Challenge #1) *Specified training*:
Train with the specified provided training images only (trainImNames).

Challenge #2) *Open training*:
Train with arbitrary external training data---whether that is additional labeled images from SUN given in `extraTrainImNames`, web images you gather, or a pre-trained network (though see note above about avoiding any network pre-trained with SUN content).

For the assignment, you must participate in Challenge #1 above. Challenge #2 is optional. We will compare accuracies across all submissions on each challenge separately.

What to report and how to submit

Train your system on the specified training images, and test it on the specified test images. Write a pdf report **called `report.pdf`** that includes the following:

- A description of precisely how your approach was implemented, listing the steps and details of any choices you made.
- For Challenge #1: Compute and display a 25 x 25 confusion matrix, and explain the result. Report the average of the diagonal of this matrix as a summary of the accuracy for the 25-way multi-class task. We will create a leaderboard using this average accuracy number for challenge #1.
- [optional] Do the above for Challenge #2. Please indicate clearly in your submission which results go with which challenge/training set, and explain specifically how the training data or procedures differ between Challenge #1 and Challenge #2.

- Discuss the overall performance of your system, such as where it works and where it doesn't, and briefly why. The `classnames` variable will be useful for qualitative analysis, and you may also want to examine image examples to get a feel for what is working or not.
- Test two variants of your system, and design a comparative experiment to show how they differ. Carefully *explain* the impact on accuracy. Illustrating the impact on accuracy could go beyond the overall accuracy number, such as impact on particular classes, types of examples, etc. *Note: please avoid simply reporting the impact of changing some hyper-parameter, unless there is an interesting conceptual explanation for the impact*.

Grades will be determined based on implementation completeness as well as the insights demonstrated by the report discussion. Use figures or images to make your point clear when appropriate.

Submit your tarred code and the report.pdf file together in a single zip file on Canvas. Please do NOT include any train/test data in the submission.

Follow up result with withheld test set: due Wed Oct 11

After the assignment deadline, on Thursday Oct 12, we will release a second test set on Piazza. You will need to run your approach, exactly as it was submitted (**no additional training, parameter tweaking, etc. allowed!).**

Report the results for Challenge #1 and Challenge #2 via email to Wei-Lin, in the form of

- the average accuracy per class, and
- a one-liner about the approach you've taken to achieve that accuracy (e.g., "SVM on GIST descriptors, trained with provided training images only" etc.) This will help us all get a quick view of how the variants tested in our class performed.

Please put CS 381V in the subject line of the email. Those results are due by 11:59 PM Friday Oct 13. We will base the contest/extra credit winners on these results. If you choose not to participate in Challenge #2, we will simply use your Challenge #1 result as your Challenge #2 entry.