COMP 425/COMP6341 Computer Vision (Winter 2024)

Homework 3

Due Date: March 29 at 3pm Montreal time

Note

- You are encouraged to meet with other students to discuss the homework, but all write-ups and codes must be done on your own. Do not take notes from those meetings. You should know how to work out the solutions by yourself.
- Please acknowledge other students with whom you discussed the problems and what resources (other than the instructor/TAs, lecture notes, and textbook) you used to help you solve the problem. This won't affect your grade.
- For programming questions, please make sure your code is clearly commented on and easy to read. If the marker cannot understand your code and it does not run correctly, you might not be able to get any partial marks.
- Follow the instructions exactly. We reserve the right to refuse to grade the homework or deduct marks if the instructions are not followed.
- Use your grace days wisely.
- Make sure your code is sufficiently optimized. In particular, do not use unnecessary loops.
- Some of the questions require setting certain parameters. You should play with these parameters to find suitable values.
- **For this homework, you can use any function in Numpy, Scipy, and OpenCV**

Part 1 (20 pt): Recognition Using Bag-of-Words

In this problem, you will implement a simple visual recognition system using the bag-of-words (BoW) approach. In particular, we will look at the problem of texture recognition. But a similar bag-of-words pipeline can be used in other recognition problems (e.g. object recognition, scene recognition) as well. You might want to review the lecture slides in week2 and week6.

You are given 7 training images (train1.jpg, traing2.jpg, … train7.jpg) where each image belongs to a texture class (i.e. there 7 classes in total). You are also given 7 test images (test1.jpg, test2.jpg, …, test7.jpg). Our goal is to build a BoW representation of an image, so that we can directly calculate the distance between two images using their BoW representation. A test image is then assigned its class label according to the class label of the closest training image -- in machine learning, this is called the nearest-neighbor (NN) classifier.

The recognition system involves the following:

1) First, we are going to define a set of filters (called "filter bank"). The provided code "LMFilters.py" will generate the filer bank of 48 filters, where each filter is 49*49.
2) For each training image (convert to gray scale if needed), apply those 48 filters in the filter bank, so each pixel is represented as a 48-dimension vector indicating the response to each of those 48 filters.
3) Collect all 48-dimensional vectors of all pixels from all training images (7 in our case). For example, if you have T training images where each image has H*W dimension, you should have T*H*W vectors. But for computational and memory reasons, you usually only randomly choose a subset of pixels from a training image. For example, you can randomly sample 100 pixels per image, then you will get T*100 vectors.
4) Run a clustering algorithm (e.g. K-means) on these vectors into K clusters (K is the number of visual words you choose), where each cluster center is a 48 dimensional vector. Each cluster is

called a “visual word” (for textures, it is also called “texton”). Then each pixel is assigned to one of the K visual words according to which cluster it is closest to (where “closeness” is measured by the L2 distance between the 48-dimensional vector representation of this pixel and the cluster center). An image is then represented as a histogram of dimension K where each element representing the frequency of a particular visual words in this image. This histogram representation is known as the BoW representation.

5) For a test image, we similarly calculate its BoW representation. Then we compare it with each of the training images in terms of the L2 distance between their BoW representations. We can classify this test image according the closest training image.

Your task is to implement the following two functions in “utils.py”

createTextons(F, file_list, K): the inputs of this function include the filter bank “F”, a list of filenames “file_list” corresponding to training images, and the number of clusters “K”. It returns a K*48 matrix where each row corresponds to a cluster center

computeHistogram(img_file, F, textons): the inputs of this function include the filename of an image “img_file”, the filter bank “F”, and the “textons”. It returns a K dimension vector for the BoW representation of this image.

If your implementation is correct, executing “run_train.py” will create the textons and BoW representations of training images, then store them in a file called “model.pkl”. Then executing “run_test.py” will load information from “model.pkl”, and classify each of the test images. Note that due to the randomness in the algorithm, your result might be slightly different in each run. But you should be able to classify most test images correctly.

If you want to use BoW for other recognition problem (e.g. object recognition), the pipeline is similar. The only differences are: (1) in object recognition, usually we only consider features at some interest points instead of all pixels; (2) in object recognition, we often use SIFT-like features instead of filter responses. Also in the homework, each class only has one training image and the classifier is very simple. In real recognition problem, you will probably have much bigger training data and you will use some more advanced machine learning classifiers (e.g. support vector machine, neural network, etc).

How to Submit

Put your codes (including data files and others that are provided as part of the assignment) in a folder named lastname_firstname (replace with your last and first names). Create a ZIP file (name it lastname_firstname.zip, again, replace with your last and first names accordingly) containing the top-level directory.

For example, if I were to submit the homework, I would create a ZIP file named “wang_yang.zip”. After running “unzip wang_yang.zip”, I should get the following:

```
wang_yang/
wang_yang/utils.py
wang_yang/run_train.py
wang_yang/…
```

If I run “python run_train.py” in the “wang_yang/” folder, it should generate the model file “model.pkl”

Go to this course in Moodle, then click “HW3 submission”. Submit the zip file containing your codes (see above). The submission server is configured to only accept ZIP files. If you try to submit other file format, the submission server will not accept it. So make sure you know how to create ZIP files ahead of time.

Note that you can make multiple submissions. But we will only consider the last submission. We will use the time stamp recorded on Moodle to calculate the late days you have used for this assignment.