ECSE-415 Introduction to Computer Vision

Final Project: Face Recognition and Tagging

Due: 3rd December 2019, 11:59PM

1 Introduction

In this project, you will develop a software system for face recognition. Specifically, you will explore using the popular bag of quantized features and k-nearest neighbours as classifier to classify unseen face images. Your resulting face recognition system will be used to tag members of the team in group photos. You will compare the bag of words approach with a PCA based approach. It is required that you work in a team of 4-5 students. Your project report should be in pdf format and is due on 3rd December 2019 at 11:59pm on myCourses. All codes must be handed in as well. Reports submitted up to 12 hours late will be penalized by 30%. After that, the student will be given a 0 grade for the project. The project will be graded out of a total of **100 points**.

2 Submission Instructions

1. Submit (i) report in pdf format (ii) codes.
2. The report should be comprehensive but concise. It should not exceed 10 pages.
3. Submit training and testing datasets. Do not submit output images. They should be included in the report.
4. Comment your code appropriately.
5. Make sure that the submitted code is running without error. Add a `README` file if required.
6. If external libraries were used in your code please specify its name and version in the `requirement.txt` file.
7. Submissions that do not follow the format will be penalized 10%.

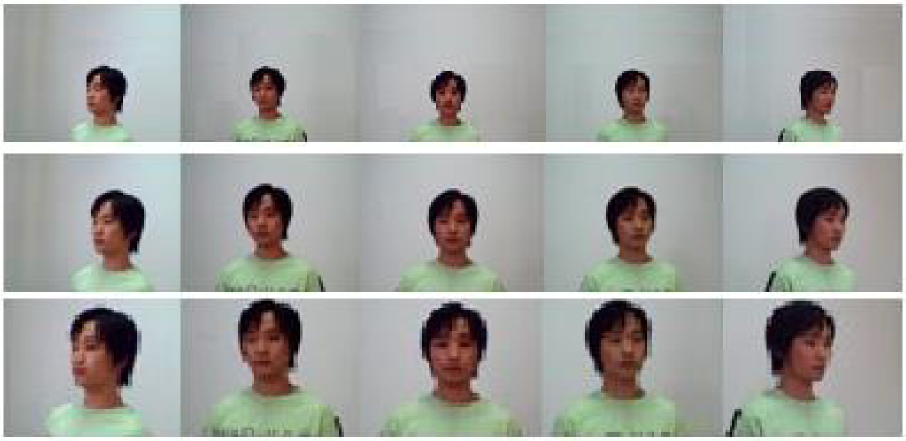


Figure 1: Training images. Each row illustrates the same head pose at a different scale (First row: small scale; second row: normal scale; third row: large scale). Each column illustrates same head pose with different head yaw rotation (from left to right: $\sim -45°, \sim -30° \sim 0°, \sim 30°, \sim 45°$.)

3 Acquiring Appropriate Datasets

For this project, you have to acquire two separate sets of images using your own cameras: a training set and a testing set. You will use only the training set for generating the visual codebook. The testing set is used to perform recognition.

3.1 Training Images

For the training dataset, you will acquire fifteen images from each of the members of your group [1] as shown in Fig 1 (total 75 images)[2]. Crop the acquired images to be square. People should be presented in five different poses and at three different scales. Note that in all cases, the pose variations consist of pure yaw rotations. Try to use a uniform white background when acquiring images. Display your training images in the report, along with their scales and poses.[3] **(5 Points)**

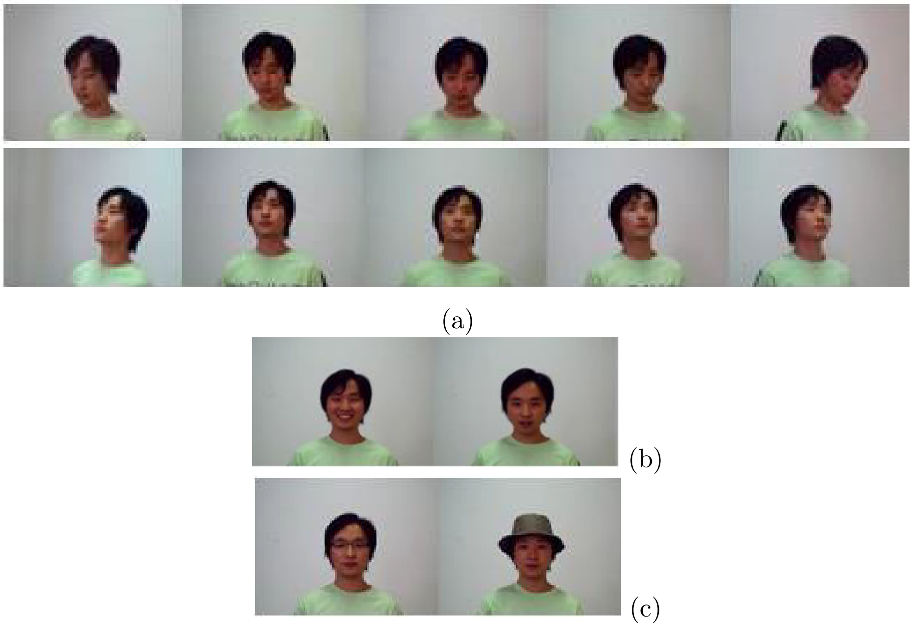
3.2 Testing Images

To assess the performance of the bag of words method in recognizing faces, you will acquire new, "unseen" images of the same subjects under different conditions (e.g. different head pose, different facial expression, and different

[1] Five subjects are the five team members. If you are working in a team of 4 students, ask your friend to become the fifth subject.

[2] Images from `http://mla.sdu.edu.cn/info/1006/1195.htm`

[3] Images should be rescaled to resolution 256×256 after acquisition.



(a)

(b)

(c)

Figure 2: Testing images. All images are in the normal scale. (a) Pose variations. In addition to the yaw rotation of the training set, there a $\sim 30^\circ$ pitch rotation. (b) Expression variations (happy and surprised). (c) Accessory variations.

accessories). The test dataset consists of fourteen images of each subject as shown in Fig 2 (total 70 images). Note that all the test images should be acquired at the normal scale (Revise scale definition from Fig 1). Crop the acquired images to be square.

Fig 2 (a) includes head pose variations; Fig 2 (b) illustrates varying facial expressions; and Fig 2 (c) shows accessory variations. Each of the above variations will be used to estimate the recognition rate of the bag of words method for unseen images. Display your testing images in the report. [4] **(5 Points)**

4 Face Recognition

You will build several vocabularies using different methods of keypoint detection and descriptor computation. Before we delineate the experiments, details for the training and testing procedure are described.

4.1 Training: Building vocabulary

Follow the steps to build visual vocabulary and compute the normalized histograms for the training set.

[4] Images should be rescaled to resolution 256 × 256 after acquisition.

- Manually define a rectangular bounding box around the face/head area on each image. [5]
- Detect keypoints in an image and discard the keypoints outside the face/head bounding box. Display keypoints on 10 selected training images.
- Compute feature descriptors for each keypoint.
- Cluster feature descriptors into K clusters using Gaussian Mixture Model. Each cluster center represents one word.
- Compute normalized histogram of words for each training image using Bag of Words (BoW) method. Store the computed histograms. Display the histograms for 3 selected images.

4.2 Testing: Face recognition

Follow the steps to recognize the identity of the person in each image.

- Manually define a rectangular bounding box around the face/head area on each image [6].
- Detect keypoints in an image and discard the keypoints outside the face/head bounding box.
- Compute feature descriptors for each keypoint.
- Compute normalized histogram of words for each test image using the method described in Section 1.
- Find one nearest neighbour histogram from the training set. Assign the face identity of the nearest histogram to the test image.

4.2.1 Evaluation

To evaluate the recognition performance of the method, you will use two quantitative measures. In order to compute this measures you will need true identities of the test images.

- **Recognition rate.** This is computed as $\frac{\text{number of correct recognitions}}{\text{total number of images}} \times 100$. The recognition rate will be 100 under the ideal performance.
- **Confusion matrix.** This is a matrix of size $N \times N$, where N is the number of subjects (for this project $N = 5$). The value of element (i, j) (i^{th} row and j^{th} column) is equal to $\frac{\text{number of images where person } i \text{ is recognised as person } j}{\text{total number of images of person } i}$. Confusion matrix will be identity matrix under the ideal performance.

[5] Optionally you can use Built-In Face Detection methods of OpenCV/MATLAB for this purpose.

[6] Optionally you can use Built-In Face Detection methods of OpenCV/MATLAB for this purpose.

4.3 Experiments

You will perform 3 sets of experiments to find out best performing keypoint detection and descriptor computation methods.

1. For this experiment use either SIFT or SURF (choose one) to detect keypoints. Extract patch of a size 15 × 15 around every keypoint and do the following[7].

   - Extract HoG descriptor for the patch as follows. Fix the block size = 2, number of bins = 9 and vary the cell size. Use cell sizes 3 × 3, 4 × 4 and 5 × 5 [8]. Build separate vocabularies for HoG descriptor with different cell sizes. Test and evaluate the vocabularies using recognition rate. Plot recognition rate (on y-axis) vs cell size of the HoG descriptor (on x-axis). Compute confusion matrix for the best performing vocabulary. **(10 Points)**
   - Extract LBP descriptor for the patch as follows. You may use `local_binary_pattern` function from `skimage` library. Use three different values of the radius 2, 7 and 12[9]. Use the number points equal to 8×radius. Build separate vocabularies using LBP descriptor with different radius values. Test and evaluate the vocabularies using recognition rate. Plot recognition rate (on y-axis) vs radius value of the LBP descriptor (on x-axis). Compute confusion matrix for the best performing vocabulary. **(10 Points)**
   - Compare best performing HoG and LBP vocabularies. Which feature performs better? Why? **(5 Points)**

2. For this experiment, use SIFT/SURF (use the same method as experiment 1) to detect keypoints and the best performing feature detection method (LBP/HoG with best performing radius/cell-size from the experiment 1) and do the following.

   - Extract patches of three different sizes around the keypoint. You can use sizes 5 × 5, 15 × 15, 25 × 25 [10]. Extract descriptors from these patches. Build separate vocabularies for different patch sizes. Test and evaluate the vocabularies using recognition rate. Plot recognition rate (on y-axis) vs patch size (on x-axis). Compute confusion matrix for the best performing vocabulary. **(15 Points)**

3. For this experiment extract keypoints using Harris corner features. Use the best performing patch size and descriptor (from the experiments 2 and 1 respectively) to build a new vocabulary for Harris corners. Test

[7] Given patch size is just a suggestion. You can use any other size. A good patch size should cover an eye of a person at normal scale in one patch.

[8] Adjust these values if you use a patch size other than the suggested one.

[9] Adjust these values if you use a patch size other than the suggested one.

[10] Given patch sizes are just a suggestion. You can use other sizes. A good set should be: the size you tried in experiment 1 and +10/-10 from that size.

and evaluate the vocabulary using recognition rate and confusion matrix. Compare the results of SIFT/SURF (from experiment 2) and Harris corners. **(15 Points)**

5 EigenFaces

In this experiment you will compare Bag-of-words approach with PCA.

- Training: Compute a reasonable number of eigenfaces from the training dataset[11]. Display first 5 eigenfaces. Find eigen representations of the training images by projecting them onto the computed eigenfaces. Store these representations. **(7 Points)**
- Testing: Find the eigen representation of the test image. Find nearest eigen representation from the training dataset. Assign identity of the nearest neighbor to the test image. **(5 Points)**

Evaluate the method using recognition rate and confusion matrix. Compare the PCA based method with best performing bag-of-words method from experiment 3. **(3 Points)**

Write a combined summary/conclusion of the above four experiments (3 in Section:3 and 1 of PCA). **(5 Points)**

6 Face Tagging

For this part, you should acquire five different group photos (each image should contain all the trained subjects). Try to have pose variations (note that the face detectors usually fail for large head rotations), facial expressions and scale variations in these images (try to be creative!). Display the original and the tagged images. **(5 Points)**

Use your best performing face recognition codebook along with a face detector to tag faces in group photos. You can use the Matlab or OpenCV built-in face detector to find all the faces in the images. Display a bounding box around each face. After detecting and locating all faces in the input image (note that there might be some undetected faces and some misdetections), you can then treat each detected face as a separate image and use your face recognition method to find the label for each face and print the name beside the head on the image. **(10 Points)**

Bonus Points: Recycle your implementation of ‘car detection using HoG’ from the second assignment to detect faces. Follow the same training and testing procedure for the collected face dataset. **(10 Points)**

[11] You mightlike to use the dimensionality estimation methods taught during te class to determine the number of eigen faces for good accuracy.