Image and Vision Computing Mini-Project Assignment

In this mini-project you will explore building a face detection and recognition/verification system. The assignment consists of there parts: Part 1: Face detection (40% marks), Part 2: Face recognition/verification (40% marks), Part 3: Bonus (20% marks).

You will do the assignment in teams of 2. Make your own groups and put down your choice link to spreadsheet. If you cant find a partner you can work on your own, but you get no extra credit for doing all the work yourself. The assignment issue date is Wed 8th Nov, and due date is 4pm, Thu 23rd Nov. The assignment is expected to take up to 12 hours work per person in the group.

Part 1: Face Detection

Face-detection addresses detection of human faces in images. A standard approach to face detection is to use labelled images to build a binary classifier for face/non-face images. This classifier can then be applied in sliding window fashion across a test image, the coordinates of bounding foxes at which the detector obtains a high score can be returned as putative detected faces. As you know from the lecture, detection systems are usually evaluated by sliding a threshold over detection scores, and generating a precision-recall curve.

In the first part of the assignment you will implement a face detector. In the assignment package, you are given:

- A subset of cropped face images from LFW dataset[1] as positive face images.
- A random subset of cropped LFW background images as negative non-face images.
- A subset of uncropped LFW images[2] as validation images.
- A demo script 'Assignment1.m' that shows the concept of face detection by loading up the images, resizing to 64x64, extracting some features (raw pixels), trains a classifier (KNN), attempts to detects faces (by classifying random bounding boxes), performs non-max suppression, and then evaluates the results quantitatively in terms of a precision-recall (PR) curve with Average Precision (AP) summary statistic.

Run the demo script. You will see that it tries to detect faces in a small subset of the data for visualisation (blue detections vs red ground truth) as illustrated in Figure 1, and then classifies the full provided validation set in order to compute the AP.

The features, classifier, and window generator used in the demo script are not very good. Your task is to use what you have learned to build an improved face detector, as measured by the final PR curve and AP summary. The initial proof of concept detector you are provided with should obtain about 0.02 AP. A good solution which will obtain full marks should obtain about 0.35 AP. An excellent solution which would attract bonus marks should obtain 0.43 AP.

To obtain a better solution, you are suggested to upgrade the provided solution as follows:

1. Apply a better feature extractor. You can use vl_feat[3] to get HoG[4] and LBP features instead

[1] http://conradsanderson.id.au/lfwcrop/
[2] http://vis-www.cs.umass.edu/lfw/
[3] http://www.vlfeat.org/
[4] Good starting parameters: Cell size = 4

of raw pixels.

2. Get a faster and more accurate classifier. A good choice would be libsvm, and more specifically it's fast linear SVM variant liblinear[5][6].
3. Write a sliding window algorithm to systematically test the patches.

If you do the above, you should be able to get 0.35-0.43 AP.

If you want to go further, you can try:

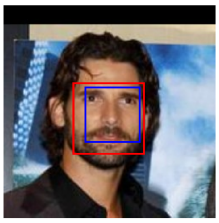
- Download more of the LFW or other face images, and more background images to obtain more training data.
- Tune HoG, LBP and SVM well.
- Tune your sliding window well.
- Use Deep features.

Then you should be able to get significantly more than 0.43 AP.

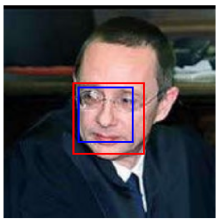
Rules:

Allowed: Any feature extractor. Any classifier.

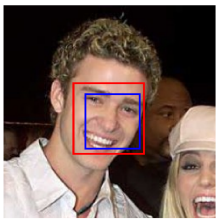
Disallowed: Any off-the-shelf object detector. You must implement your own sliding window and extract features/classify each sub-window.



(a) Eric Bana



(b) Hans Leistritz



(c) Justin Timberlake

Figure 1: Seen LFW Faces

Part2 2: Face Recognition and Verification

Recognition Face recognition addresses multi-class classification of a face image into one of C known classes. Face recognition is typically built on a C-way multi-class classifier. To train a face recognition system there must be (typically many) train images of each of the C classes of people to recognise in order to train the classifier.

Verification Face verification addresses taking two face images and reporting whether they are the same person or not. (For example comparing your passport photo to the access gate image in immigration). Face verification is typically built on a binary classifier that inputs two face images and reports +1 (same) or -1 (different). To train a face verficiation system there must be many pairs of face images, some of which are the same identity (+1) and some are different identity (-1).

In the second part of the assignment, you will implement a face recognizer and verifier. In the assignment package you are given:

[5]https:// www.csie.ntu.edu.tw/ cjlin/liblinear/
[6]Good starting parameters cost 1

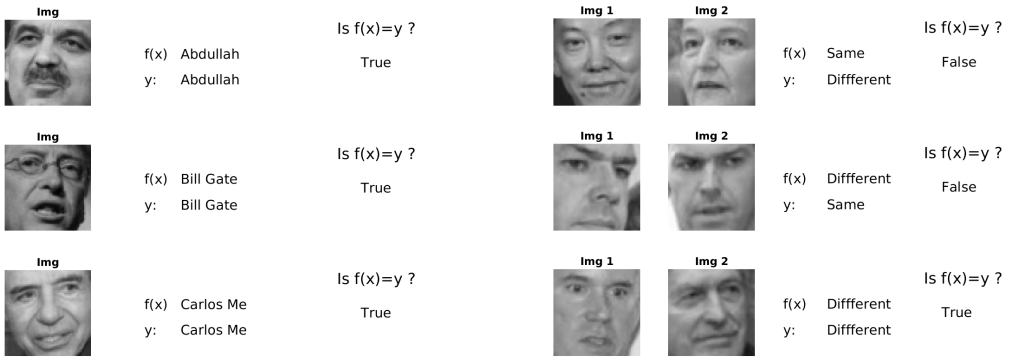
- Data for these tasks are in 'face_recognization.mat' and 'face_verification.mat' respectively (drawn from LFW dataset).
- An example script 'Assignment2.m' that again implements some simple baselines for recognition and verification.

If you run the demo script you will see it performs a poor job of solving these tasks. It obtains around 16% recognition accuracy and 54% verification accuracy on these tasks. Note that here there are $C = 35$ classes so random chance for recognition is $1/35 = 2.8\%$ and random chance for verification is 50%.

To obtain a better solution you are suggested to upgrade the provided solution as follows:

1. Extract better features. The HoG and LBP features suggested in the previous exercise would be a good start.
2. You can also try to enhance them by making a bag of words with normalization.
3. Build a better classifier. The linear SVM from the previous task would be a good start.

With these updates, you should be able to obtain around 78% recognition accuracy and 64% verification accuracy on these tasks



(a) Visualization of Recognition

(b) Visualization of Verification

Figure 2: Visualization of Some Images of Assignment2

Part 3: Bonus

The final part of the assignment is open ended. Building on what you have done before, do something cool. For example:

- B1: Surpass the suggested accuracy target on detection or recognition/verification. For example: (i) upgrade the face detector to be a more realistic multi-scale detector, (ii) download more training data to build a better model, (iii) use deep features.
- B2: Connect up the detector and the recogniser and verifier. Show a complete system that detects faces and says who they are (recognition), or searches the database for all examples of a query face (verification).
- B3: Make some other cool application of the technology.

Submission and Grading

The grading will be based primarily on (1) Your score on the testing set (Part 1&2). Supported by: (2) A brief report/presentation, (3) A mini demo/viva of your implementation (Parts 1-3).

Test Set: You have been provided initially with the validation set for development. Shortly before the deadline we will release the test set (Wed 22nd Nov, TBC), which the assessable score will be based on.

Report: The report should take the form of a short 7 page powerpoint presentation (template to be provided). Page 1: Your team members, Page 2-3: The methodology implemented for detection, the detection result obtained (test set score). Page 4-5: The methodology implemented for recognition & verification, the recognition & verification results obtained (test set score). Page 6-7: Description of your attempt at bonus marks (eg advanced methodology implementation, system integration), the result or screenshot of the outcome.

Live Demo/Viva Session: Fri 24th Nov, (TBC). Each group will have 5 minutes to present their report slides and give a mini demo. Then answer some questions of the marker/lecturer.

Submission You should submit a zip file containing your code (source files only) and report/presentation by Thu 23rd Nov, 4pm. The command for the DICE-based submission is:

```
submit ivc cw1 PRESENTATION CODE
```

where PRESENTATION is your (pdf or ppt) presentation, and CODE is the zip file of your code.

Good Scholarly Practice

This assignment is expected to be in your own words and code. Short quotations (with proper, explicit attribution) are allowed, but the bulk of the submission should be your own work. Use proper citation style for all citations, whether traditional paper resources or web-based materials.

Please remember the University requirement as regards all assessed work for credit. Details about this can be found at:

http://www.ed.ac.uk/academic-services/students/undergraduate/discipline/academic-misconduct

http://www.ed.ac.uk/academic-services/students/postgraduate-taught/discipline/academic-misconduct

http://web.inf.ed.ac.uk/infweb/admin/policies/academic-misconduct

Furthermore, you are required to take reasonable measures to protect your assessed work from unauthorised access. For example, if you put any such work on a public repository then you must set access permissions appropriately (by permitting access only to yourself, or your group in the case of group practicals).