INM460: Computer Vision

Instructor: Dr Sepehr Jalali
Email: sepehr.jalali.1@city.ac.uk

Coursework: Face Recognition and OCR

Submission Details:

You must submit your project (as a single .zip file) to Moodle by **5pm, Sunday 22 April**. The submission must include both your project report (PDF or Word format) as well as source code.

Standard lateness penalties (late submission = 0 marks) and extensions policy applies. This assignment constitutes **50% of the total marks for this module**.

Concept and Task:

Have you ever been introduced to a number of people, but forget their names? Humans are exceedingly capable of recognising faces previously seen before, but matching names to faces can be difficult. Imagine having an app running on your device (smartphone, glasses) that recognises people and prompts you with their name, to help you avoid blurting an awkward "*I'm sorry but I forgot your name?*"

Person identification is important for a variety of practical tasks, including cybersecurity and authentication. In this project, you will develop a computer vision system to detect and identify individuals using a database of known face images (specifically, students / lecturers in this module). In addition, you will implement detection of detecting numbers each person is holding on a piece of paper in order to group these images for training your classifiers.

Technology:

It is recommended you use Matlab (preferably 2017b, although earlier versions are acceptable) for the development of the coursework in this module. Matlab is an exceedingly powerful programming language and environment for imaging and computer vision, and includes a variety of toolboxes that provide functionality of interest to this project, including the Computer Vision System Toolbox, Image Processing Toolbox, Optimization Toolbox, Neural Networks and Statistics and Machine Learning Toolbox.

As an alternative to Matlab, you are permitted to use OpenCV with an appropriate language binding if you prefer (e.g. C++,Python); however, please bear in mind the lecture notes, code samples, and lab exercises will be based on Matlab. For simplicity, there will be limited academic support for OpenCV.

Spend some time familiarising yourself with Matlab, particularly if you haven't used it before. Also be sure to spend sufficient time working with the labs and code samples. Look at online sources (documentation, blogs, tutorials, videos) on computer vision in Matlab, including use of the Computer Vision System Toolbox, and refer to the books recommended for the module. At times you may wish to discuss the methods used with your colleagues. This is encouraged. The coursework is, however, **individual**. Plagiarism will be dealt with directly by the department's senior staff and may lead to course disqualification. Your work must be your own and you must cite any and all resources used.

Marking:

The coursework is marked out of 100%, with the final mark scaled to 50% for the module.

1. Deliverables -- Project report and source code (25%):

- Prepare a project report (maximum 15 pages) in Word or PDF format. The report should include: (20%)
    - An overview of your project; a description of your final approach(es). Include a block diagram showing the key algorithmic components that are part of your solution.
    - For each section (Parts 2 - 3 below), list briefly the requirements that were implemented. Describe your implementation, review and discuss any relevant theory underpinning the methodology. If a requirement was skipped, say so.
    - If you initially tried approaches that didn't work, mention these briefly, and provide a rationale on why they were ineffective.
    - Provide an analysis of when the final approaches implemented are successful -- and conditions when they may fail to achieve the desired result.
    - Reference any external source code used.
    - Include a discussion section reflecting on the project. Consider the strengths and weaknesses of what you achieved in the time provided. Also discuss how you might expand the project to improve performance or take it in new directions.
- Source code to be commented and follow a logical design, organisation, and coding style. (5%)

2. Face recognition (50%):

Objective: Develop a Matlab function,

```
P = RecogniseFace(I, featureType, classifierName)
```

that returns a matrix **P** representing the people present in an RGB image **I**. **P** should be a matrix of size Nx3, where N is the number of people detected in the number of image. The three columns should represent

1. `id`, a unique number associated with each person that matches the number in database provided
2. x, the x location of the person detected in the image (central face region)
3. y, the y location of the person detected in the image (central face region)

For example, if the function detects two people in the image, with person 3 at position [144, 153], and person 13 at position [312, 123], the **P** matrix would be

```
P =
   3   144   153
  13   312   123
```

The `RecogniseFace` function should have two additional arguments, `featureType` and `classifierName`, which allow different features or classifiers to be used (discussed in more detail below). Evaluate your function on an independent evaluation dataset (**not** be used in training), using a single fold or cross-validation strategy.

- For the images in your training set, use a face detector to find faces, and save extracted faces as images. Each person has a unique number associated with him or her. This will produce a database of images with their labels. (10%)
    - -Use of Optical Character Recognition to automatically label the data is also necessary as a separate project section explained in detail below.
- Use (at least) **three** different classification methods (such as CNN, SVM, MLP, etc.) to recognise individuals in an image. For each classification method, use (at least) **two** different types of features (e.g., SURF, HOG, etc.) as applicable. (30%: 5% for each classifier-feature combination).

-Note that at least one CNN implementation is required. In case of using a CNN the feature-type argument will be nil.

- For your best performing method, in your report, provide a characterisation of the accuracy (percentage of faces correctly identified) as well as a confusion matrix on the evaluation dataset. Show examples that are classified correctly and some that are incorrect (if applicable). (10%)

- *Optional task*: detect the person's emotional state, using the following labels: happy = 0; sad = 1; surprised = 2; angry = 3. Return this value as a fourth column in the **P** matrix returned by your `RecogniseFace` function. *(up to 5 points extra credit)*

Notes:

- Your `RecogniseFace` function should run automatically on an image. If your function does not detect any faces, return an empty matrix []. Note the image 'I' may have more than one face present (i.e., group photo).

- In your report, be sure to provide all the valid ways to call your `RecogniseFace` function (e.g., arguments to pass in as `featureType` and `classifierName`).

- In addition to the `RecogniseFace` function, include any training code, network weights, etc. in the deliverable you submit to Moodle, so that the entire system is reproducible.

- During marking, your code will be tested on unseen images.

3. OCR (25%)

Objective: Develop a function,

```
detectNum (filename)
```

that accepts an image/video file (`filename`) of a person holding a number in hand (as in your image dataset) and your detectNum will return the number seen in that image/video. You can use the built-in OCR function in Matlab but you will notice that you will need to deal with many pre-processing and modifications to the code so that your code runs well on the images. You can use a CNN (or any other machine learning model) trained on different numbers and apply it to your images using some modifications you find necessary.

- Detect the numbers correctly in test images. (15%)
- Detecting numbers in videos. (10%)
- During marking, your function will be tested with unseen images/videos.
- Each image/video will only contain one number. If you can modify your code in a way that it can detect multiple numbers you will earn up to 5 extra points.

Notes:

All source code should be commented (per block is ok). Clearly mark any code that you have written and any code that is externally produced. Use appropriate naming conventions and display an organised code strategy through the use of functions.

If additional functions are used (beyond those in the standard City, University of London Matlab distribution), make sure that these are sufficiently packaged with both the source code and the final code can run on a machine with Matlab without having to set up any complicated file paths.

Deliverables:

You should deliver your project as a single .zip file (note this must be less than 200MB!) uploaded to Moodle by the deadline. The file should include:

1. Code. The code must be at a stage where calling the methods described above works without exceptions. Code that does not execute will receive a **FAIL**.
2. Report. The report will be a single file in PDF or Word format.

Tips:

- Maximise your marks by making some attempt at every section.
- Install Matlab 2017b (from https://www.city.ac.uk/current-students/it-support/resources-and-facilities/matlab-nvivo-oxmetrics-spss) as soon as you can, especially if you're working from home or on your laptop. Explore code, tutorials, and demos provided given in class, as well as online sources.
- Google any of the terms you are unfamiliar with. Put aside time each week to cover material and practice coding frequently. Explore the exciting world of computer vision and try topics you've learned about. Build up a good portfolio of example code for you to keep and explore best practices.