Coursework specifications for Computer Vision UG students - Sit

- **Module:** Computer Vision – IM3060
- **Module leader:** Giacomo Tarroni (giacomo.tarroni@city.ac.uk)

Table of Contents

General information

You must submit your coursework by **5pm** on **Sunday, 15th of May 2022**. Standard lateness penalties (late submission = fail mark) and extensions policy ("Extenuating Circumstances") apply. This assignment constitutes **100% of the total mark for this module**.

You will need to submit most of the produced materials on Moodle, in the dedicated "Submission Area". You will also need to provide a link to a Google Drive folder to allow the testing of your code. For the detailed list of the materials to submit, please refer to the *Submission details* section of this document.

At times you may wish to discuss your coursework implementation with your colleagues. This is encouraged. However, **the coursework is individual**. **Plagiarism** will be dealt with

directly by the Department's senior staff and may lead to **course disqualification**. More on this topic in the *Coding* section.

Concept and task

Humans are extremely good at recognising the emotion of individuals by simply looking at their faces. A quick glance is usually enough to gain a fairly accurate estimation of one's emotions, even from unfamiliar people. In fact, this skill is fundamental to effectively navigate and perform social interactions in our everyday lives.

Despite being trivial for us, this task is very complicated to automate and implement in a Computer Vision pipeline. In fact, it not only requires to correctly identify faces and their features (e.g., eyes, mouth, cheeks), but to detect the very small changes that separate one emotion from another. As a consequence, **Facial Emotion Recognition** (FER) has been subject of research in Computer Vision for over a decade. In its most common definition, it translates to an **image classification task**, where images of faces are used as input and emotion labels (e.g., "surprise", "fear", "happiness") are generated as output. A successful pipeline would have many useful applications, especially in the field of human-computer interaction.

In this coursework, you will **develop a computer vision pipeline for Facial Emotion Recognition**. You will be provided with a dataset of cropped images showing aligned faces accompanied by a numerical label (1-7) for the associated emotion. You will need to implement a series of different machine learning classification models to perform the task and compare them both qualitatively and quantitatively. You will then test your models on a few images "in the wild". You will need to submit a short report and a video presentation showcasing your code in action, as well as share a working version of your pipeline using Google Drive.

Coding

You must implement your coursework using **Python** (python3) and **relevant, open-source libraries** (e.g., scikit-image, scikit-learn, OpenCV, PyTorch, TensorFlow, Keras).

Your work must be your own and not the result of exchanges with other students. You are allowed to use external resources, but **you must always cite all resources used.** References must be present within your code (roughly at the function level), and not just at the end**.** Please note the following:

- You can use any open-source libraries/code found online or adapt code from the lab tutorials available on Moodle. Again, all sources must be referenced (labs too).
- You can use pre-trained models for any task (e.g., image classification on ImageNet, face detection) **with the exception of models already trained for Facial Emotion Recognition** (this is for you to do!).
- **You must not reuse other students' submissions** for this module's Computer Vision coursework from this or previous years, regardless of proper referencing.

Failure to follow the above guidelines will be treated as **academic misconduct** and potentially lead to course disqualification.

Additionally, you can (and should!) use personal online code repositories for version control and backup (e.g., GitHub) but **you must keep your repo private until the end of your entire course**. Afterwards, if you want to make it public, please avoid titles like "Computer Vision coursework" and use the likes of "Facial Emotion Recognition". This will protect you and your colleagues from potential plagiarism.

Deliverables

Project report (20%)

Write and submit **a report describing how you carried out the coursework**. For this, you **must use the provided template** available on Moodle. The report must be in PDF format and **not exceed 4 pages**. You are not allowed to change font size or page margins to accommodate more content. You **must include**:

- A link to *CW_Folder_UG* in Google Drive at the top of the report to allow testing your implemented functions (more details in the *Submission details, Google Drive submission* section).
- A **"data"** section, where you critically describe the data you have used, including both provided dataset and personal one (more on this in the *Facial Emotion Recognition on images "in the wild"* deliverable).
- An **"implemented methods"** section, where you list which **classification models** you decided to implement and explain why. For each of them, describe the **training process** (e.g., if you pre-processed the data, on which data the model was trained, how you optimised the hyper-parameters, etc.). There is no need to provide a theoretical description of the models but feel free to add relevant citations to the methods. Also describe the implementation of your *Facial Emotion Recognition on images "in the wild"* deliverable (more on this in the relative section).
- A **"results"** section, in which you provide **qualitative examples** of the obtained results for all implemented models. Report also **quantitative results of performance** (e.g., accuracy, speed, model size) for the different methods. Provide at least **some qualitative examples** of the results obtained by running your best performing model **on your personal dataset** (more on this in the *Facial Emotion Recognition on images "in the wild"* deliverable).
- A **"discussion"** section where you **discuss the obtained results**, providing a rationale for the differences in performance and other relevant aspects.
- Important notes:
    - **The "results" and "discussion" subsections are the ones that will mostly impact the mark of this deliverable**, so be particularly careful about them.
    - Be brief and to the point. Make **use of tables and bullet points** as much as possible throughout.

Video showcase (10%)

Provide **a screen capture video with audio commentary** (recorded by you) in which you showcase the training code implementation and test the implemented models.

- The video must **not exceed 8 minutes**.
- In the first 4 minutes, you must briefly show and describe the code you have implemented for **loading the data, training the models and measuring their performance**. No code needs to be run necessarily during this part.
- In the remaining 4 minutes, you must briefly show and describe your **implementation for the** `EmotionRecognition` **function**. You must also **execute this function live** using your test_functions.ipynb Google Colab notebook file to show respectively **the results produced by the different trained models on the provided test set and on your personal dataset** (more on this in the relative sections).
- To record it, the suggested solution is to start a Zoom or Teams call with just yourself and record the meeting. Both software conveniently provide compressed video recordings right after the meeting has ended.
- Important notes:
    - Failure to submit the video showcase will result in a fail mark.
    - The contents of the report and video showcase will **also be used to mark the coding deliverables** (see next sections).

Facial Emotion Recognition in the provided dataset (50%)

Implement a pipeline for FER by training and testing different machine learning models using a provided dataset.

- **Access the dataset** following the info in the *Implementation details, Data access* section. The dataset consists of a training and a testing set of cropped faces labelled based on the expressed emotion.
- (35%) **Implement a series of image classification models to perform FER** using the provided dataset:
    - You need to implement **at least 3 models** using different combinations of feature descriptors (e.g., SIFT, HOG) and classifiers (e.g., SVMs, MLPs, CNNs). For instance, HOG+SVM and HOG+MLP would count as 2 models.
    - The implementation of **a CNN model will provide substantially more points** than more basic classifiers.
    - The choice of the combinations, the implementation details, the hyper-parameter tuning and the overall training set-up are up to you.
    - **Compare the different models quantitatively** on the provided test set using performance metrics of your choice (e.g., accuracy, speed, model size) and assess which model works best.
- (15%) Implement a function called `EmotionRecognition` that **allows to visualise the predictions obtained by the different models** on the test set images:
    - The function should accept the syntax `EmotionRecognition(path_to_testset, model_type)`, where `path_to_testset` is a string pointing at the test set of the provided dataset and `model_type` is a string used to select a given model (for this, provide a few comments in your code listing the available entries).
    - Each time the function is called, **4 random images must be loaded from the test set** (don't load the whole test set each time!) **and displayed together with the model's predictions and the ground-truth labels** (as image titles).

Facial Emotion Recognition on images "in the wild" (20%)

Test the `EmotionRecognition` function on a dataset of images that you have created.

- **Download or create a small dataset** of test images of people expressing different emotions (more on this in the following sections) and provide ground-truth labels for each of them based on your subjective evaluation.
- Make sure that the `EmotionRecognition` function can run on this dataset too by simply changing the string `path_to_testset`. All the functionality requested in the previous deliverable should be produced on this one too (i.e., loading and showing 4 random images from the dataset and displaying the model's predictions together with the ground-truth labels).

Implementation details

Data access

To get access to the dataset to use for training and testing your pipeline, **you must fill out this online form** (https://forms.gle/P7jK9TThhUjvUuvi9) providing your name and your City email. The link for download will be available right after submission.

Carefully **read and accept the terms and conditions** relative to using the data: in particular, you must "agree not to reproduce, duplicate, copy, sell, trade, resell or exploit for any commercial purposes, any portion of the images". **Failure to submit the form with your details might lead to a fail mark**. Also, don't share the link to the dataset to anyone else, including other students enrolled in the module.

The provided dataset consists of respectively 12271 training and 3068 testing images of human faces exhibiting one of 7 emotions (namely: surprise, fear, disgust, happiness, sadness, anger, neutral). Labels are provided for each image (as an integer from 1 to 7) in accompanying txt files (**look at the *README.txt* file in the dataset folder** for more details).

General implementation

- Note that **no explicit evaluation set is provided**. Perform a sensible split from the available data to ensure proper validation and reliable testing.
- Do not panic if **some features-classifier combinations do not provide extremely accurate results**! This is expected for some of them. The aim of the coursework is to assess your ability to properly implement and evaluate the methods presented in class, not to achieve high accuracy in all cases.
- Different types of classifiers (e.g., different kernel settings for SVM, different CNN architectures) won't count as separate models (but will be rewarded with some additional points if the different implemented choices are properly discussed).
- For CNN models, **make sure that your implementation can run on CPU at test time** (code to achieve this will have been presented in a lab tutorial).

Testing on the personal dataset

- If you want to download images from the web, it is recommended to look for those with **Creative Commons or Public Domain licensing**: the CC Search website provides filters for these types of content. Alternatively, you can use your own photos (including selfies!).
- Make sure that the images you selected constitute a good test set for your models, with faces showing the 7 different emotions in different settings (e.g. lighting conditions). The images don't have to be many (an average of 3-4 images per emotion will be enough).
- Note that you can **either manually crop the faces** in the images to be put in your test folder or **include a face recognition algorithm** in your `EmotionRecognition` function to detect faces automatically (only when executed on the personal dataset). The latter implementation is more complex and will be rewarded with a higher mark. If you decide to go this route, feel free to use a pre-trained model for face recognition. You might also have `EmotionRecognition` display the whole images together with bounding boxes for the recognised faces and the predictions and labels as text captions (see Figure 1):

Figure 1. Example of a FER result in an image with overlaid bounding box and label.

Submission details

You will **need to submit some of the produced materials on Moodle as well as share a Google Drive folder** to allow quickly testing of your functions. The latter part will ensure consistency between what you and the marker see as output from your code.

Moodle submission

You must submit on Moodle the following files:

- **Report** (PDF format): **make sure to add the link to your *CW_Folder_UG* in Drive at the beginning** of the report (more details in the *Submission details, Google Drive submission* section).
- **Video showcase** (mp4, or anything readable by the VLC player).
- **All the produced code** (single PDF): copy-paste **all the code you have implemented in a single text file and export to PDF**. This is a requirement for auditing purposes but will also allow running TurnItIn on your code. Failure to submit this file will lead to a fail mark. Write the name of the different files and location in your directories (e.g., "Code/train_SVM.py") at the top of each copied file to allow recreating your folders if need be.
- Files need to be **submitted separately and not as a single zip.** Failure to do so will lead to a fail mark.

Google Drive submission

You must also create and share a folder on Google Drive named *CW_Folder_UG* to allow testing of your functions. For this, you **should use the provided template** available on Moodle. In it you must upload:

- A **Google Colab notebook** called **test_function.ipynb** that has at least two cells, both calling your `EmotionRecognition` function and displaying the results on each of the two test sets (provided and personal). In the notebook, you must add a few lines explaining the syntax needed to use the different models with your function.
- **All the code and data needed to run test_functions.ipynb**. Specifically:

```
Your_CW_Folder
├── Code
│   ├── All the code you produced
├── CW_Dataset
│   ├── The provided dataset (at least the test set)
├── Personal_Dataset
│   ├── The images you have downloaded/created
├── Models
│   ├── The different trained models you implemented
└── test_functions.ipynb
```

- Important notes:
  - You need to upload in the *Code* folder **all the implemented code**, but **only test_functions.ipynb will be explicitly executed**, if need be.
  - The trained models should be loaded using test_functions.ipynb, **not re-trained from scratch**
- To share your folder, follow this procedure:
  - Open Google Drive in the browser and right-click on *Your_CW_Folder*
  - Select "Get link"
  - **Select the option: "Anyone with the link"** (see Figure 2)
  - Copy the link and paste it clearly at the start of your report

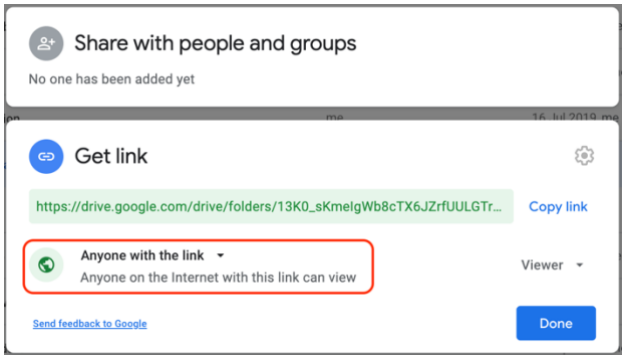

Figure 2. How to create a link for sharing Your_CW_Folder from Drive.

- Make sure **not to edit this folder in any way after the deadline**. The folder is time-stamped: any changes after the deadline will lead to a fail mark.

Good reporting practices

There are some *dos* and *donts* when it comes to writing a good project report:

- **Make sure to add your name and ID to all the submitted files**:
  Markers will sort through ~100 submissions: having to deal with "report.pdf" files adds unnecessary confusion and could result in your mark going to someone else.
- **Never add portions of code as screenshots:**
  Doing this lowers incredibly the quality of your report's layout. Respect your datatypes: if you are dealing with text, treat it as text (and use a different font to signal the difference from normal paragraphs). For what concerns this report, it's extremely unlikely that adding code will be useful/meaningful.
- **Pay attention to your layout:**
  Avoid common mistakes like poor formatting, images too small or too close to the page edges, lack of captions, lack of references in the text to figures and tables.
- **Avoid trivial grammar mistakes and typos:**
  Please, please read your report before submitting it. In the past we've often seen very bad grammar and poorly spelled methods' names.
- **Keep a formal and scientific tone:**
  This is a technical report, not your diary: avoid sentences like "I tried method A, but it didn't work very well, so I moved to method B, people said it was better!, but then I tested it and it didn't work either, so then…" and replace with "methods A and B were implemented and tested. Unfortunately, none of them led to good results (i.e., in both cases evaluation accuracy was lower than 20%), arguably due to…". To improve your style, have a look at some of the scientific papers referenced in the lecture slides.