# MPHY0041: Machine Learning in Medical Imaging
# Assessed Coursework 2 - 2019-20

Available on 19th March 2021
Submission before 23:55 – 9th May 2021, at UCL Moodle submission section

## WHAT NEEDS TO BE SUBMITTED

### Code (20%)

✓ Python code: all python code used to complete the tasks [10 marks for completeness[1]].
✓ "Instruction.pdf": a text document that outlines the steps to reproduce the results [10 marks for reproducibility].

### Scientific Report (80%)

✓ "report.pdf": This coursework requires to summarise the completed tasks (in the Tasks section) in a scientific report format. The Lecture Notes in Computer Science template is recommended[2] and please use the font sizes and page margin specified in the template without modifications. The report should not exceed 4 pages in addition to maximum 2 pages of references, submitted in PDF format. The report should include the following sections to *cohesively* address what are specified in the Tasks section.

- Introduction
- Methods
- Experiments
- Results
- *Discussion can be included but will not be assessed.
- Conclusion
- *Optionally, selected coursework submissions may be invited a full version to submit the work for further academic publications.

### Background and Objectives - Medical ultrasound classification and segmentation

Transrectal B-mode ultrasound images are clinically used to guide different urologic procedures, such as ablation therapy and needle biopsies. Real-time segmenting prostate gland from the 2D ultrasound images can help surgeons to localise the relevant anatomical structures and subsequently help targeting regions of interest. However, identifying the boundaries of the prostate gland capsules is a challenging task even for experienced urologists.

This coursework is to develop an automatic 2D image segmentation tool based on deep neural networks. The objective is to develop and compare different model development strategies with available data set and labels from multiple observers.

### Data Set

IMPORTANT: The conda-based *marking environment*, specified as below, will be used for marking the submitted Python code and for downloading the provided data set, by the following steps.

```
conda create -n cw2 numpy scipy matplotlib h5py tensorflow pytorch torchvision
```

[1] Square bracket contains the marks in each task.
[2] https://www.springer.com/gp/computer-science/lncs/conference-proceedings-guidelines

1. Pull the latest module repository.
2. Change directory to the "cw2" subfolder under the module repo.
3. Run the provided script "data.py" to download the data set.

The script also provides an example to read the relevant data groups in the downloaded h5 file.

- Labelled image data are acquired from 200 cases.
- For each case, multiple ultrasound image frames are sampled.
- For each frame, three *segmentation labels* (three binary masks indicating region of prostate gland) are annotated from three different observers.
- For example, data group "/frame_0004_003" indicates the No.3 image frame from the No.4 case, with segmentation labels from three observers found in data groups, "/label_0004_003_00", "/label_0004_003_01", "/label_0004_003_02".
- All case-, frame- and label numbers are zero-indexed.

## Tasks

1. Build a U-Net-like encoder-decoder neural network, accepting a single 2D image and predicting a class probability map of the same size. From the predicted probability map, a threshold of 0.5 can be applied to obtain a binary segmentation of the prostate gland from the given image. The network should contain the following key architectural components. Describe the motivation and implementation in the report [10].
    a. Residual blocks that contain convolutions, nonlinear activations (e.g. relu) and residual shortcuts.
    b. One of the batch/instance/group normalisation methods.
    c. Down-sampling, up-sampling and skip layers.
2. Develop a network training procedure that contains the following key strategies. Describe the motivation and implementation in the report [10].
    a. A training loop that optimises a 2D Dice-based loss function with a selected optimiser.
    b. Training monitoring methods that regularly output the loss value and one type of validation metric value.
3. Implement the following regularisation methods to improve the network training. Describe the motivation and implementation in the report [10].
    a. One data augmentation strategy.
    b. One ensemble method.
4. Implement a data loader that samples a minibatch of image-label pairs at each iteration step, to facilitate the following sampling approaches. Describe the motivation and implementation in the report [10].
    a. Each epoch should sample all the cases once without replacement, i.e. equal occurrence between cases.
    b. In each case, all frames should have equal chance to be sampled, i.e. frames from different cases could have different sampling frequencies.
    c. *Segmentation label sampling 1*: For each image frame, random sample one label from the available three labels.
    d. *Segmentation label sampling 2*: For each image frame, sample the consensus label for each image frame. The consensus label should be constructed based on majority voting at the pixel level among the three available segmentation labels.
5. Design an experiment with relevant metrics to answer the following research question. Describe the experiments and summarise your results in the report, including i) summary numerical results in a table, ii) comparison using a Bland-Altman plot and iii) example images with overlaid segmentation prediction and ground-truth [10].

   a. Based on the consensus segmentation labels as ground-truth, using an independent holdout test set, compare the two (training) segmentation label sampling methods.

6. Choose one of the built-in 2D image classification networks, among VGG, DenseNet and MobileNet. Adapt the multi-class classification network to a binary classifier that predicts whether a given image frame contains prostate (true) or not (false). Summarise the implementation of such a built-in network in the report [10].
   - TensorFlow: https://www.tensorflow.org/api_docs/python/tf/keras/applications
   - PyTorch: https://pytorch.org/vision/stable/models.html
7. Implement a classification training procedure, using a consensus image-level, binary *classification label* for each image frame. The consensus classification label should be constructed by majority voting between the three classification labels at image level, i.e. each "true" vote should have two or more none-zero segmentation labels, whilst a "false" classification label is a result of two or more all-zero segmentation labels. Summarise the implementation in the report [10].
8. Design an experiment to answer the following two research questions. Describe the experiments and summarise your results in the report, including i) summary numerical results in tables and ii) comparison using Bland-Altman plots, and iii) other illustration to support the conclusions. [10].
   a. Apply the trained classification network prior to a trained segmentation network, and compare the resulting segmentation results, i.e., between the segmentation results with- and without the "pre-screening" classification of the image frames. The comparison should be based on the consensus segmentation labels as ground-truth, using an independent holdout test set.
   b. Find an optimum classification threshold on the classification-network-predicted class probability [0, 1], with which the highest segmentation accuracy can be achieved.

# OTHER IMPORTANT INFORMATION

## Python with TensorFlow/PyTorch Development Environment

Python is required in this coursework with either TensoFlow2 or PyTorch, as in the module tutorials. It is recommended to use a conda-based development environment, although it is neither mandatory nor assessed. The coursework marking environment is specified in the Data Set section.

- This also means NO other Python packages or libraries can be used for completing this coursework.
- Only *.py files need to be submitted to avoid large files, without unnecessary system-dependent configurations.
- Please do not submit the downloaded data set.
- It is the student's responsibility to ensure that the submitted code can be successfully run using the marking environment. It is recommended to have the provided **data.py file** in the same unzipped folder with other ***.py files** and the **Instruction.pdf** document.
- For technical support for work with such an environment, please refer to the documents available in the module repository.

## Computing Resource

It is recommended to use CMIC HPC Cluster hosted at CS Department and other publicly available parallel computing facilities, such as Google Colab. The use of the parallel computing and other high-performance computing is not assessed in this coursework. With proper configurations, each network training in this coursework is expected to take no more than several hours on a modern Nvidia GPU card. Training on CPU may take a significantly longer time but is largely dependent on the hardware.

However, it is feasible to develop, debug and run the inference/testing procedures on a personal computer.

## Patient Data

Although the image data this project has access to are anonymised and with patient consents for research purposes. These are still sensitive clinical data and need to be treated with highest ethical standards and caution. Care must be taken when storing, transferring and processing the images and labels and, **under no circumstances**, these can be used for any other purposes outside this coursework (even for other research purposes) or be distributed further. Upon the completion of the project, all the copies of the data need to be removed completely from any computer or cluster storage.

## THE CHECKLIST

This is a list of things that help you to check before submission.

- ✓ All the code are zipped in a single file, e.g. code.zip
- ✓ The zip file does not contain files other than *.py Python code and Instruction.pdf.
- ✓ The zip file does not contain downloaded data.
- ✓ All the scripts run in the unzipped folder, within the marking environment without requiring additional libraries or packages.
- ✓ A single PDF report file is submitted, e.g. report.pdf
- ✓ The report has the sections described above and appropriate references.

## Academic Integrity

UCL Guides for Academic Integrity includes guides to references, citations and avoiding plagiarism.