Assignment 3

Goal:

In previous assignments, you experimented with downloading clinical data of a substantial size, exploring the data, and pre-processing, resampling, and filtering the images. In this assignment you will apply deep learning (DL) to analyze the images that you have already explored.

You will devise a segmentation approach for delineation of clinically significant prostate cancer (csPCa) in the PIC-AI challenge dataset. For this assignment, you are using all 5 folds (1500 samples) of the dataset. You will build either a 2D U-Net, nnU-Net, or other variations for this purpose.

1) **Re-slicing**: The dimension of each pixel should be the same in whole dataset to preserve the scale in our deep model. Use the resampling code you implemented in Assignment 1 part 6 to re-slice the data volumes (T2W, ADC, HBV), prostate mask (anatomical_delineations/AI) and lesion mask (csPCa_lesion_delineations/AI) to have the spacing of `(0.5, 0.5, 3.0)`, for all cases. To make sure that you do this step correctly, please double check your re-slicing code with the function provided in the preprocessing repository of the challenge:

   https://github.com/DIAGNijmegen/picai_prep/blob/376ccc697c3c1a60078006237a4c0 d6b0e8beae3/src/picai_prep/preprocessing.py#L71

2) **Cropping**: In addition to spacing, the dimension of images (number of pixels) among all patients should be the same for deep model training. Crop a `(300, 300, 16)` pixel volume from the MRI volumes and masks that you re-slice in part 1, symmetrically around the center voxel of each volume. At the end of this part, you should have 7500 volumes (MRI sequences and masks), each of size `(300, 300, 16)` and spacing of `(0.5, 0.5, 3.0)`.

3) **2D slices:** Since we want to train a slice-based 2D deep model, we'll slice each MRI volume in z axis to convert it to 16 MRI images of 300x300 pixels. Do this for all MRI volumes and masks of all patients. You will use the MRI images of different sequences (T2W, ADC, HBV) as the input, and the mask images (prostate gland, lesions) as the output for building your deep model.

4) **Augmentation**: Data augmentation is used in DL to increase the sample size and improve model training. You can simply double the number of your samples by flipping the images horizontally (left-right flip). Note that the masks should also be flipped for these images as well. Double your data size using this approach.

5) **Normalization:** As a preprocessing step, use z-score to normalize each MRI image (**NOT** the masks) in your dataset, which means the intensity of pixels in each image will have a mean of 0 and variance of 1.

6) **Stratification**: Your dataset is already stratified by patients into 5 folds (double check the Data Split portion of the challenge webpage). Use folds (1, 2, 4) as training, fold 3 as validation, and fold 0 as test. Optional: you can use different combinations of folds and run a cross-validation study if you have enough computational power.

7) **Model architecture**: U-Net is the deep structure that you will use for the segmentation task. You can develop your own version, or modify different variations that already provided in the challenge website as baselines:

   https://github.com/DIAGNijmegen/picai_baseline

   Whichever model you choose to use, provide details about the structure in your report and include the model summary. Consider at least four levels of contraction and expansion for your model. Note that if you take code from public repositories, the input size needs to be adjusted to match the size of your images. Also, ensure you have properly cited/commented the source of your architecture.

8) **Loss function:** You should use Dice Score (DSC) for optimization of weights and monitoring of the performance during training. Note that while DSC is used as your metric, one-minus-DSC should be used as your loss.

9) **Training**: Train 3 models with the following input / output pairs:

   a) *T2W MRI images / prostate gland*

   b) *ADC MRI images / cancer lesions*

   c) *HBV MRI images / cancer lesions*

   Train the model with the training data. Monitor the performance of the model at the end of each epoch by plotting the losses of training and validation data. Use early stopping based on the performance of validation data (validation loss) to avoid overfitting. After training, report the average performance of your model on segmenting the test data. Do the same for validation and training sets as well.

10) **Ablation**: You should consider a small ablation study to optimize some of your model and training parameters. Do not go overboard with experiments but attempt it for a few parameters such as the learning rate, batch size, number of layers. Select the models with best performance and for the case you visualized in Assignment 1 (`10522_1000532`), show the true masks (prostate gland and lesions) as well as the prediction of your 3 best models for comparison

What to Upload:
- Include all the screenshots, figures, and responses to the questions in your report and save it as *student-id_last-name.pdf.* For each part, provide an explanation about your approach and results, if applicable.

- Save your codes with proper structure and comments as one Python or notebook file with similar naming.
- Compress the above files as *student-id_last-name.zip* and upload it to OnQ by the deadline.
- Create a public repository on GitHub and upload all your codes there. Include the link to the repository in your report.

Good luck