Final Project – 50% of the final Grade

Dear Students,

as discussed, this homework covers topics across all the material taught in the class. It should be solved in teams of ideally three people.
The results of the group work are to be presented by all team members during the acceptance/technical discussion meeting end of January! This includes practical demonstration of developed software.

Not all tasks of this homework require coding. For the practical parts you can freely choose the programming language (MATLAB or Python) you prefer. It is up to you as data scientists to choose the most appropriate tool.

Code of Conduct: You can use any resources (incl. LLMs) but are **not** allowed to ask other humans outside your team for help. By your signature you certify that you agree with this code of conduct. Please upload a scan of this first page with your team filled and signed on Moodle in the Teams forming forum before **December 5th**.

The written answers to the exercises + the source code of the practical parts **have to be submitted via Moodle until January 23th**. A submission system for this in Moodle will be opened.

Good Luck!

Team

Name	Matricula	Signature

Task 1(10 pts):

The Fourier transformation $f(x, y) \multimap F(u, v)$ of a greyscale image $f(x,y)$ results in a band-limited signal in the spatial frequency range with maximum frequencies f_{umax} and f_{vmax}. For representation in the computer, the (partial) image is sampled in x direction with 20 sampling points per *mm* and in y direction with 10 sampling points per *mm*.

1. What is the theoretical maximum value of f_{umax} and f_{vmax} if error-free image reconstruction from the digital image should be possible (not using any compressive-sensing techniques)? (6pts)

2. What is the minimum memory requirement for the color image $f_F(x, y)$ when stored in a conventional computer system, if 1024 values are to be distinguished per color channel. Describe the image format to be used. (2pts)

3. How many colors could be represented with the quantization chosen in sub-task 3? (2pts)

Task 2 (10pts):

For the subjective enhancement of a greyscale image $G = g(x, y)$, a transformation T_G is performed as a so-called gamma correction in the form $T_G : g \rightarrow f$ with $f(x, y) = c\, g^{\gamma}(x, y)$ where $g, f \in [0, 255]$.

1. Sketch the transformation curve T_G for $\gamma_1 = 0.5$ and $\gamma_2 = 2$. (2pts)

2. How is the coefficient c typically determined? (2pts)

3. In which respect and for which type of input images G do the two gamma values γ_1 , γ_2 lead to an image enhancement respectively? (2pts)

4. What should be the minimum slope of the transform function?
    1. for a grey value spread (2pts)
    2. for a grey value compression (2pts)

Task 3 (Coding, 20pts):

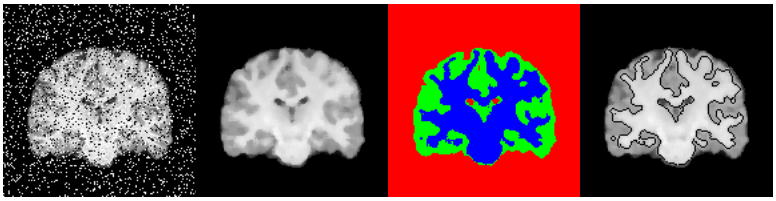

Figure 1: From left to right: The input image, after denoising, a threshold-based segmentation into background (red), grey matter (green), and white matter (blue), boundary between grey and white matter.

In this task you will need to perform threshold-based image analysis:

1. Read the greyscale image brain.png, which is provided on the lecture homepage. Reduce the salt and pepper noise in the image using a median filter. (3pts)

2. Otsu thresholding is a histogram-based method for image segmentation. Use it to find an intensity threshold to segment brain pixels from background. Use Otsu thresholding again to find the threshold only over the brain pixels to segment brain's grey matter from the white matter. Using the two thresholds create three binary masks brain-bg.png, brain-gm.png, brain-wm.png, which should be white in regions of background, grey matter, and white matter, respectively, and black elsewhere. (4pts)

3. Plot a log-scaled histogram of the image, which should show how frequently different intensity values occur in the image. How could you roughly estimate the two thresholds you found in the previous task just by looking at the histogram? (3pts)

4. Combine the three masks into a single colour image so that background, grey matter, and white matter are mapped to red, green and blue, respectively. (3pts)

5. Use erosion (or any other morphological) filter to produce a border between the grey and white matter. Overlay that border on the denoised input image. (3pts)

6. Use bilinear interpolation to up-sample the image by a factor of four along each axis. Apply the same thresholds as in 2) to obtain a segmentation into background, grey matter, and white matter. Up-sample the masks from 2) in the same way and compare the up-sampled masks to the masks from the up-sampled image. Can you see a difference? Why? Repeat the same procedure using nearest neighbour interpolation. Can you see a difference now? (4pts)

For this task, please submit the python code and the images. Please state for the images to which sub-task they correspond by naming them accordingly.

Task 4 (Coding, 50pts):

The goal of this project is to implement an automatic glioma (a common type of tumor originating in the brain) segmentation pipeline using the BraTS dataset and the U-Net architecture using the pytorch framework in Python. The model should be able to take an MRI scan as input and segment 4 classes, namely “no tumor or backbackground”, “Edema”, “Growing Region”, “Necrotic Core”.

Hint: You do not need to reimplement the U-Net architecture in pytorch. Usually, common archtictures and methods are available in well-implemented libraries. Please have a look at one of the MONAI implementations of the U-Net architecture.

1. Data Handling and Preprocessing (10 Points)
    a. You can focus for now on loading the T1-weighted images and the matching labels.
    b. Create a dataloader for the data using PyTorch’s Dataloader (or Monai’s Dataloader class)
    c. Create suitable augmentations for the task to solve. **Please note:** If you apply transformations to the input data, you should think about if you need to apply any transformation to the label of the image as well.
2. Data Splitting (5 Points)
    a. Think about an appropriate split of the dataset into train, validation and test split. Briefly explain why it is good scientific practice to have separate Training, Validation and Test dataset?
    b. Please define the patient ids for each split in a separate file, like test.txt or validation.yml
3. Model Selection and Training (20 Points)
    a. Implement a training pipeline to train the U-Net model of MONAI (or your own implementation if you want). Make use of the dataloader you created.
    b. At the end of each epoch, you should run a validation of the models performance.
    c. As we do not want to waste unnecessary energie and time, please implement an early stopping condition, that stops the training when the performance of the model does not improve for a few epochs on the validation set.
    d. Which loss function do you think suitable to train your model for this segmentation task?
    e. Please save the weights of the model that you trained. You can either save the weights after each epoch seperatly or you can override the previously saved model with the new best performing one after each validation.
4. Testing and Performance Evaluation (15 Points)
    a. Think about a suitable metric to evaluate the performance of your model on the test set. You can also use more than one metric to capture different aspects of the models performance.

    b. Briefly explain, which model should you evaluate? You get a trained model after each epoch? Should you evaluate all of them and pick the best performing one?
    c. Code a test pipeline to evaluate your model's performance.
5. Extension to new modalities (5 Bonus):
    a. Extent the model to include multiple modalities of your choice as input (T2, T1ce, …). Justify your choice.
    b. Train the model with the same data split and asses the performance of the model using your test pipeline.
    c. Why and how does the performance change now that you use multiple modalities compared to using a single T1-weighted image as input?
    d. Your test pipeline should include a method that minimizes the probability of your models performance relying too much on the split you chose (e.g. cross validation, …).

You should be able to justify each choice that you made. It is up to you to structure your code, however, to please the computer science gods it is forbidden to use python notebooks (like ipynb) and your code should be structure into multiple files. The content of each file should be dedicated to a specific functionality only. It does not need to be perfect, but it should follow some structure. Look at the example below:

- src
    - test.py
    - train.py
    - dataloader.py
    - create_split.py
    - splits
        - split1
            - test.txt
            - train.txt
            - validation.txt

NOTE: For the coding exercises please submit the code as a zip file. Each exercise should be one subfolder.