Grading

1. [18pts] **Model Performance on TTE and TEE**: This grade is determined by the following:

[5pts] DICE score higher than baseline code for TTE.

[5pts] Model tested and working on TEE images.

[8pts] Placement and final dice score depending on how you are doing compared to your Classmates (TTE 4pts, TEE 2pts).

2. [12pts] **Preprocessing** Data

In order to improve the performance of the model you can try a number of pre-processing steps:

[2pt] Rotate the TEE data by 180 degrees in order to make it more similar to TTE data

[2pt] Convert the data to iso-tropic pixel size

[2pt] Try and report Dice score on various input resolutions of your data

[2pt] Use both the ED and ES timepoints from the TTE data for training purposes

[2pt] Report the effect of image smoothing on the Dice score (Gaussian or bilateral smoothing are 2 possible options)

[2pt] Surprise us with an idea you have

3. [20pts] **Presentation, approach and documentation**: See section 4 for more information.

Introduction

In this project we will go through a typical medical imaging workflow often seen in real world scenarios. The main task in the project will be segmentation of structures of interest in ultrasound images of the heart (see figure). Two modalities will be considered transthoracic ultrasound (with ground truth segmentations available) and trans esophageal ultrasound (expert validated test set is available).

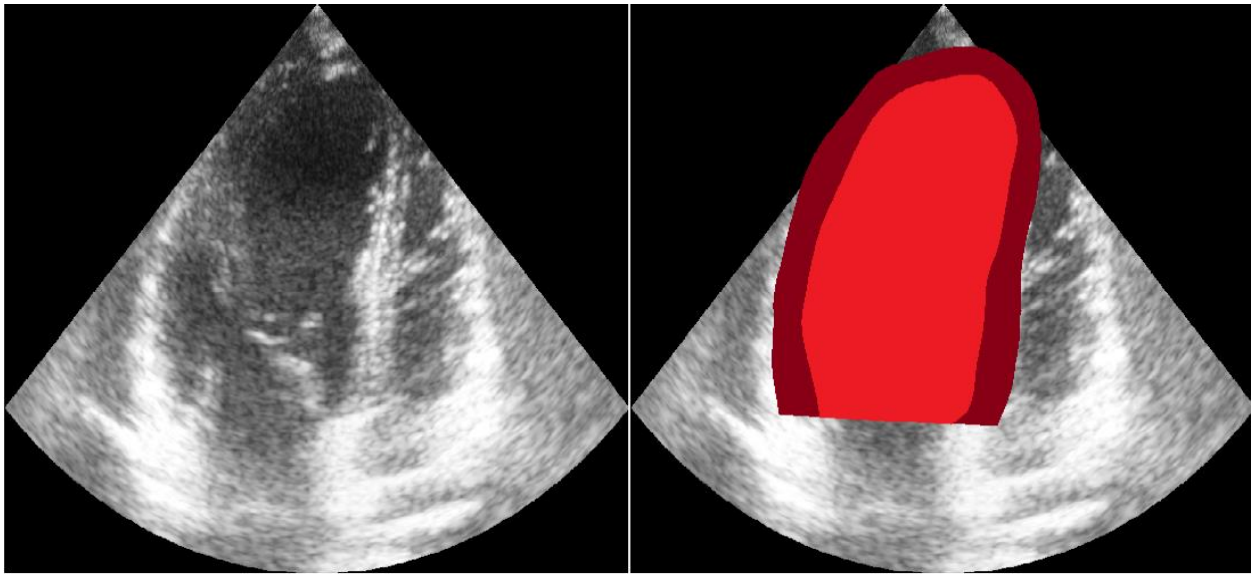

Figure 1Typical TTE recording of the left ventricle, bright red is the inside of the ventricle, dark red is the myocardium, the muscle responsible for the contraction of the chamber

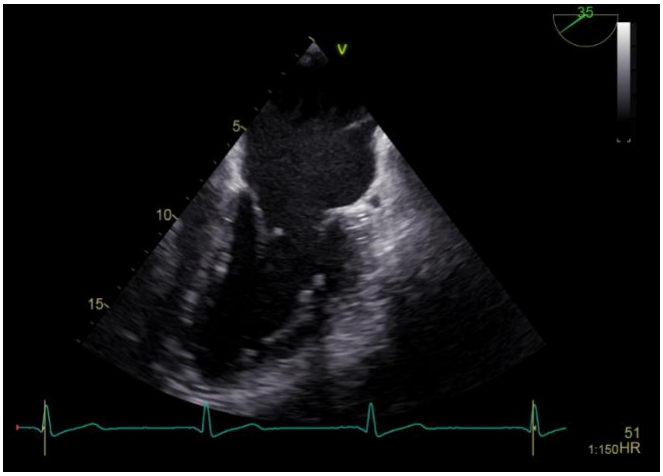

Figure 2Typical TEE ultrasound image, which is a 180-degree rotated TTE image due to acquisition differences

To achieve a proper segmentation, we need to solve several tasks:

1. Developing and building your model. Your starting point will be an existing Unet2D model available here: https://github.com/gg1977/Unet2D.git

and your task is to improve your model for both TTE and TEE data. See section 1 for more information.

2. Preprocess TTE and TEE data. To achieve the best possible performance on a custom dataset, you will need to do some preprocessing tasks test out what is working best on the data, then fine-tune the model on the dataset. However, for TEE data there is no ground truth so you will need to use the TTE data for optimization.

3. Evaluating your model. The last step is to evaluate the model on test data. This will be done on the TTE (with known round truth) and TEE data without known ground truth (but a set of 10 images with corresponding segmentation will be made available). You will report the Dice score on the test dataset and score your model.

4. Documenting your approach. In a real-world scenario documenting your approach is important. Either you're going to hand-on the project to someone else, or reporting about the amazing work you have done to your boss. For this project, you will deliver a short report and give a presentation about the project. See section 4 for more information.

Competition Rules

To achieve a fair competition (and grading), we ask you to follow these rules for all submissions and report your best Dice score in your presentation which will be used to compute your position in the competition.

1. The code should be developed by yourself (except what is given on https://github.com/gg1977/Unet2D.git).

Of course, finding inspiration from open-source repositories is allowed. If you take code from anywhere else, please attribute the original authors in your source code and write a notice in your report. Therefore, it is not allowed to use open source repositories "out of the box". Also all methods should be a form of Unet2D, but data pre-processing and post-processing is allowed.

2. Annotating the test data by yourself and using it to train/validate your model is NOT allowed. If you choose to annotate TEE data and use that to improve the model performance on TEE you should state it clearly in the presentation and report.

3. It is not allowed to train your model on any other data except the data provided by us. However, it is allowed to use backbone networks pre-trained on the ImageNet dataset, but you are not allowed to perform this training yourself. This is to prevent that the winning solution is the one who use the most amount of data/compute.

4. Training your model should not take more than 12 hours on the tdt4265.idi.ntnu GPUs (or the cybele/tulipan computers) per dataset. In total you can train your model for 24 hours. Note that you are not allowed to train for more than 12 hours for a specific dataset. Password for login is given in section 1

If you are unsure if something is allowed, please post on piazza or contact the teaching assistant through email: hakon.hukkelas@ntnu.no. If we believe that you've broken any of these rules, we will retrain your model following the steps in your report, and validate that we achieve a similar results. Breaking rules will be considered cheating and we will subtract a large amount of your grade on the project.

Section 1: Building the Model

In this section we will describe how to get started on building your model, including datasets, and some recommendations on what to do to improve your initial model.

Datasets

We recommend the following two datasets in this task: the TTE dataset and our custom TEE dataset. They are available both online on the cluster:

/work/datasets/medical_project/TTE

/work/datasets/medical_project/TEE

And for offline access here:

https://folk.ntnu.no/kiss/AIproject/

The link contains the CAMUS_resized.zip file that is the data required to start with the git code and all the TEE data.

The entire CAMUS dataset (TTE data) is released officially here, this will be required for part of the exercise:

https://www.creatis.insa-lyon.fr/Challenge/camus/

The TTE data is annotated while the TEE data is not. In the start of the project, there will be very little (or none) annotated TEE training data. Therefore, develop your model on the TTE dataset initially.

Model development

In previous assignments we've told you quite explicitly what you should do. For this project, however, we will give you the freedom to use whatever means necessary to achieve a high dice score for your segmentation model (as long as you follow the competition rules).

Here are a couple of recommendations to get you started:

- Split the code into train and inference. Write loader and savers for the trained models
- Implement proper testing based on Dice score
- Reserve some of the data for testing data, split the rest into train/validation
- Look at various types of data augmentation to improve training (translation, rotation and random crop are some alternatives)
- Extend the unet so that it can segment more classes (myocard in addition to the left ventricle)
- Customize the Unet, by adding more layers to it

Section 2: Preprocess Data

- Customize your model to use another image size then the default
- Data preprocessing on the initial images like isotropic pixel size, image filtering

Section 3: Evaluating the Model

To evaluate your model, we will provide two sets of images for testing purposes with known ground truth. You should report the Dice scores for every group, TTE and TEE and this will be compared against your colleagues.

Furthermore, points will be given if you beat the original code which should be quite trivial.

Show some of the best/worst cases (images) in the presentation.

Section 4: Reporting your approach

Documenting and reporting your approach is an important part of any deep learning project. This will consist of a presentation and a short and concise report.

Presentation

Students working alone will have 9 minutes to present, groups of two will have 10, and groups of three or more will have 11 minutes to present. The presentation will most likely be online. More information on blackboard will be published after easter.

Topics you should cover in the presentation should be:

1. Initial model: Shortly describe the model you started with from github.

2. Development: The approach you decided on and what steps you did to improve the model. It should clearly describe the reasons to the changes you did, and what kind of improvements you noticed from these changes. Remember from previous assignments, your reasoning should be supported by either theoretical arguments, previous experiences made from reading the curriculum, or empirical experiments. An example of this could be:

ResNet is known to improve gradient flow and diminish the problem of the degradation problem, therefore, we decided to use ResNet as the backbone. By replacing the backbone we notice a significant improvement, which you can see from the loss curve which we are currently pointing to on the our powerpoint slide.

Of course, doing this for every improvement will take a lot of time, but please do it for the major improvements you made.

3. Final model: Describe your final model. This might be short if a lot of it is covered in the ”Development” section. Interesting things to include is:

-A brief overview of the final architecture.

-How you trained it, for example how did you pre-train it? What kind of data augmentation did you use? What kind of data pre-processing did you do?

-Amount of time required to train it.

4. Results: Show results of your model. Include quantitative and qualitative results. Quantitative results are for example loss curves, dice score, and average precision for specific classes. Qualitative analysis could be testing your model on different images in the dataset and see what kind of objects it’s good at detecting, and what it struggles with.

5. Discussion Discuss your approach for solving this task and the final results you achieved. Examples of questions you might want to answer is:

Did you test something that did not work?

Was there any unexpected results?

Looking back at the task, is there anything you would want to do differently?

If you did not have the limitation of the rules in this assignment, what would you do?

Is there any further work you would like to do? (for example, things you didn’t get time for)

6. Group member contribution: In the end of the presentation, shortly describe every group members contribution to the project and what they were responsible for.

The final presentation will be a significant part of your grade. I recommend you to start early and prepare yourself well for the actual presentation. For general guidelines for presenting a project, Professor Charles Elkan has some good advice.

With these guidelines we are trying to help you to show your knowledge about the curriculum in the course, and prevent you from waste time on nitty, gritty details. Often students struggle with managing the time during the presentation and spend all their time on describing the initial model, architecture etc. This leads to a very rushed discussion and result section, making it hard for the evaluators to understand the work gone into the project and the student’s understanding of the underlying concepts. Also, we are not interested in what learning rate you used, what batch

size you used or any kind of hyperparameters you chose; we can read up on this ourself if we are interested.

Documentation of Details

The report should be short and concise. What we expect you to include in this is anything "boring" and note that we will not read this document except if we are interested in technical details of your model, or we want to re-run/validate your experiments. Note that anything included in the presentation should not be included in the report. What we expect is that the report includes anything required to re-produce your results. Such as:

Hyperparameters. This can be referred to as: "We used the config file our_amazing_model.yml and all hyperparameters are there". Nothing else is required.

How to train your model. Assume that we want to re-run all your experiments. Document clearly how we should be able to do this. An example of this could be "To setup your environment, install the additional packages "some-package" (Not required if you used the default environment used in the assignments). Then, you can train the model on cityscapes by running the file "some train.py". Furthermore, fine-tune the model on TDT4265 dataset by running "some train2.py. Finally, run the evaluation script.

Specific details of your model architecture. Examples of this can be the tables with models given in previous assignments.

Any additional results that you did not have place for in the report. However, we do not want any discussion of this result in the report.

The reason we want such a short report is that we do not have enough staff resources to read through everything. Even though I truly enjoy reading some of your assignments and reports, it would take me way too much time getting through all of your reports!