TFG Initial Research and thoughts

Problem:
The main problem is that there is a lack of integration of external algorithms in the clinical workflow. For instance, in the case of segmentation, the process is very time consuming and the current software doesn't allow much freedom when segmenting images.

Objective:
Make the deployment of AI algorithms in a clinical environment, more specifically, the TotalSegmentator open-source algorithm in the Hospital de Sant Pau. To do so, we need to follow these steps:

- Get the desired images we want to process from the PACS server in the hospital (DICOM format). The hospital uses the Orthanc DICOM server.
- Forward those images into the TotalSegmentator and get the desired segmented images.
- Once the images have been treated and segmented, store them in the Orthanc server again.
- Visualize those images in the OHIF visualization tool.

What are the functionalities that the developed product has to have?

- MONAI Label: extension 3D slicer
  The doctor's task is to open the MONAI Label extension, select the corresponding model, and click a button to run the segmentation algorithms.
  Includes the possibility to manually correct them using the segment editor and submit the corrected masks as new labeled samples used for an instantly model re-train.

- **MONAI SDK: python based application**
  Segmentation algorithms to be run in multiple environments while processing multiple studies at a time.

It would make sense to deploy the model for a 3D slicer app, because clinicians are used to it and the extension is already done. However the idea of making a custom app offers some freedom and additional functionalities that the first method doesn't. This is why it is important to **enhance those additional functionalities** and **search for more new ones by talking to the end users and their needs**.

Functionalities:
Main:

- Python based application
  - **Fast**, minimize the time clinicians spend on image segmentation
  - **Easy to use and set up**, clinicians may not necessarily possess the same level of expertise in setting up complex segmentation tools
  - **Understandable,** intuitive interface or commands to run the algorithm.

    - **Satisfy medical needs,** search for actual medical needs and problems and try to give the solutions in the final product.
    - **Safe (privacy and security)**. Which are the requirements regarding security and data privacy that the app has to follow.
    - **Prevent errors that come from the algorithm** (DICOM images do not match the engine rules, execution errors in the algorithm or lack of findings).
    - **Feedback to the users,** give information about the status of the segmentation to the users
    - Textual or numerical results **showing** the **output of the algorithm**

  Technical
    - Boolean flag indicating **whether the results can be sent to PACS or not**
    - **Reference list of:** images that should be **displayed** and the ones that should be **sent to PACS** (this list may be the same)
- **User guide** showing all the steps to set up and use the product.

Optional:

1. Give some **metrics** about the received outputs
2. Presenting AI results with minimal friction, but enabling the radiologist to **accept, reject or request rework of the results**.
3. **Add a free text to send a message** or annotation for the radiologist to the image post-processing lab staff if they want to point out something about the algorithm results.
4. **Record** rejection or accepting of results with the **time and user.**
5. **Privacy and security standards**
6. Do we need to have a **control and record of the people that use the algorithm** and how? We could store the **motive of rejection** that could later be **reviewed by an AI scientist.**
7. **User interface** (adapted to pacs, hand-made interface?)
8. **Display demographic data metrics**
9. **Package to install the environment**, all the dependencies and libraries.
10. Its **automatic execution** when a **CCT study is stored in the hospital's PACS serve**r, through a **trigger application** which is constantly listening to the specified address, would provide a **never stopping segmentation algorithm requiring zero human intervention**.
11. **Flexibility. Multiple segmentation models and algorithms** (if we implement correctly the TotalSegmentator we could squeeze the software and make it flexible to have the opportunity to run multiple algorithms for an image. As more and more AI algorithms are developed, a single chest abdomen pelvis exam could hypothetically have results for kidney volume, liver volume, detection of early-stage pancreatic cancer, body composition, etc)
12. Active learning (input the new segmented images as part of the training dataset).
13. The possibility of **appending a decision-making support module to the CAD pipeline**, which, using all the results automatically obtained through the pipeline, **helps the clinicians to identify the severity of CAD** that the patient has.

Tasks:

- How to **connect all the different parts** (PACS, Orthanc, TotalSegmentator, OHIF)
- Which are the **involved teams** in using the algorithm
- Which should be the **inputs and outputs** (data format, name…) For the whole project and for each phase in specific.
- Do some **early testing** to find out problems at the beginning and solve them ASAP.
- **How to write in TFG style**. Do I have to search for papers, web sites are also allowed, can I paraphrase papers, how to quote…?
- Find out the **computing power of the hospital**, how to **access** it and **control** it (queuing algorithms, using different components for different tasks, parallelization…).
- Do the images need to be a **copy** when segmented or **update the original**?
- TotalSegmentator has a **fastmode** flag which makes the segmentations in low resolution. Look for information and consider the possibility of adding a **preview option** and then the full hd segmentation option.
- Should they be **processed by CPU** and **reserve GPU's** when needed?
- Which should be the **name of the output file** so that the radiologist knows **which are the changes** that **have been made in the original file.**
- Do some research about **MONAI MetaTensor objects**. A data structure that combines both image data and its metadata (I could create this data structure from scratch to manage the images and its information packed).
- Search information about the **sliding window method**, provided by MONAI.
- **Weight dictionary** of the model as a **TorchScript**
- Information about the **meshlib** 2.1.79 python library to generate the 3D meshes, taking into account image metadata, which are then saved as **STL files**.
- It used **Docker10** through the NVIDIA Container Toolkit11. This resulted in a Docker image containing the necessary segmentation models and all the required libraries installed, so that it can be distributed to any environment compatible with Docker for its execution.
- Nifti and DICOM format diferences. The images were fed in NIfTI format, so that they can be fed to the segmentation algorithms. To do so, python library **dicom2nifti** 2.4.013 was employed.
- **Materialise Mimics Innovation Suite14** clinical viewing software.
- **Pydicom17** python library, which allows working with DICOM data in an easy pythonic way.
- **Sequence diagram** for representing the pipeline of the final application?

Bernat's TFG:

Structure of the python application:

- Configuration: Label names and indices, network architecture, image resolution and ROI (region of interest) size, flag to train from scratch or finetune a pretrained model.

- Inference transforms: transformations applied to input data when a model is making predictions or inferences. Used to prepare the input data in a format that is compatible with the model's requirements

- Trainers: data augmentation, custom validation split

The models are saved in a subfolder that must contain the corresponding weight dictionary of the model as a TorchScript.

What Bernat did was to use the MONAI Label extension to make the segmentations available on its compatible medical image viewers (OHIF).
If I don't use MONAI do I have to create a kind of OHIF extension?

With this data, it creates a MONAI Dataset applying the defined pre-process transforms to the input images. Then, the corresponding segmentation inferences for each of them are carried out using the inputted model to apply the **sliding window method**, provided by MONAI.

Other:

Also finding an integration with the hospital's data infrastructure is an area of improvement of high interest to **smoothen the pipeline as much as possible**. Its **automatic execution** when a **CCT study is stored in the hospital's PACS serve**r, through a **trigger application** which is constantly listening to the specified address, would provide a **never stopping segmentation algorithm requiring zero human intervention**.

The metrics that could be obtained **in addition to the demographic data** of the patients, coming from the PACS server, would open the possibility of **appending a decision-making support module to the CAD pipeline**, which, using all the results automatically obtained through the pipeline, **helps the clinicians to identify the severity of CAD** that the patient has.

Do I need to implement some kind of mask to the images before doing segmentation with the model? (I suppose not but just in case). Example:

> KeepPericardium: The goal of this operator is to transform the original input images so that they contain only the pericardium, removing all the unnecessary information for segmenting the coronary arteries. To achieve so, given the original images and the pericardium segmentation masks, it sets the minimum intensity value to those voxels of the original images that match with voxels where the segmentation mask value equals 0.

Problems with totalSegmentator v2:

- the start of the ribs close to the spine is always missing a small part
- the end of the ribs at the costal cartilages is not properly defined
- the GT segmentation of the colon/small_bowel is sometimes bad because the colon is so messy it is not possible to disentangle colon and small_bowel
- there might be other problems I am not aware of

ROCKET (Records of Computed Knowledge Expressed by neural nets):
The UI has 4 major components:

1. display of the display series
2. corresponding text report

3. “Report Navigator” listing all available results in the event of multiple AI results for a given Accession
4. “Accept” and “Reject” controls

Definitions:

DICOM:
A standardized format for storing and sharing medical images and associated data, such as patient information and imaging parameters. It was specifically designed for the healthcare industry to ensure interoperability and the consistent exchange of medical images and information between different imaging devices and healthcare systems.

PACS:
Picture archiving communication system. A system that stores medical images.

The medical device makes an association request to the PACS server to send some data, if that request is accepted the data is sent with a specific transfer syntax. The transfer syntax allows different Application Entities to negotiate common encoding techniques they both support.

DICOM Tag:
Unique identifier for an element of information composed of an ordered pair of numbers which is used to identify the Attributes and corresponding Data Elements.

Nifti:
NIfTI is a simpler and more specialized format primarily used in neuroimaging. It is designed to store brain imaging data, particularly data from functional MRI (fMRI) and structural MRI (sMRI) scans. NIfTI aims to simplify the storage and exchange of neuroimaging data.

Differences between DICOM and NIfTI:

- Scope: DICOM is a comprehensive, widely-used format for all medical images and their associated information, whereas NIfTI is a more specialized format focused on neuroimaging data.
- Complexity: DICOM files are often more complex and can include extensive metadata, whereas NIfTI files are simpler, containing mainly image data and basic metadata.
- Interoperability: DICOM is crucial for interoperability in the healthcare industry, allowing medical images to be shared and viewed across different systems. NIfTI is more specialized for research in neuroimaging.
- Use Cases: DICOM is used in clinical settings for various medical imaging modalities, while NIfTI is primarily used in neuroscientific research, particularly for MRI data.

https://imaging.nci.nih.gov/ncia/

FUMPE (figshare.com)

arboiscodemedia/Dicom (github.com)

DICOM Library - Anonymize, Share, View DICOM files ONLINE

Working with NIfTI images — Neural Data Science in Python

FHIR (standard for interoperability in medicine): ¿Qué es HL7 FHIR? | TIBCO Software