Cardiac MRI pipeline

Omar Dekhil

09/05/2019

Project scope:

This project consists of 2 main parts:

1- Parse the DICOM images and contour files
2- Create model training pipeline

In the first part, the required steps are:

1- Map the patient ID to the contour files folder name using the provided linker sheet
2- Create pairs of DICOM- contour files
3- Parse the DICOM files
4- Parse the contour files
5- Create image with overlay of the mask over the DICOM image and save it for each DICOM.

In the second part the required steps are:

1- Randomly shuffle the data
2- Generate batches for training of 2D images to be used as input for a training model

Figure 1 shows the proposed pipeline steps:

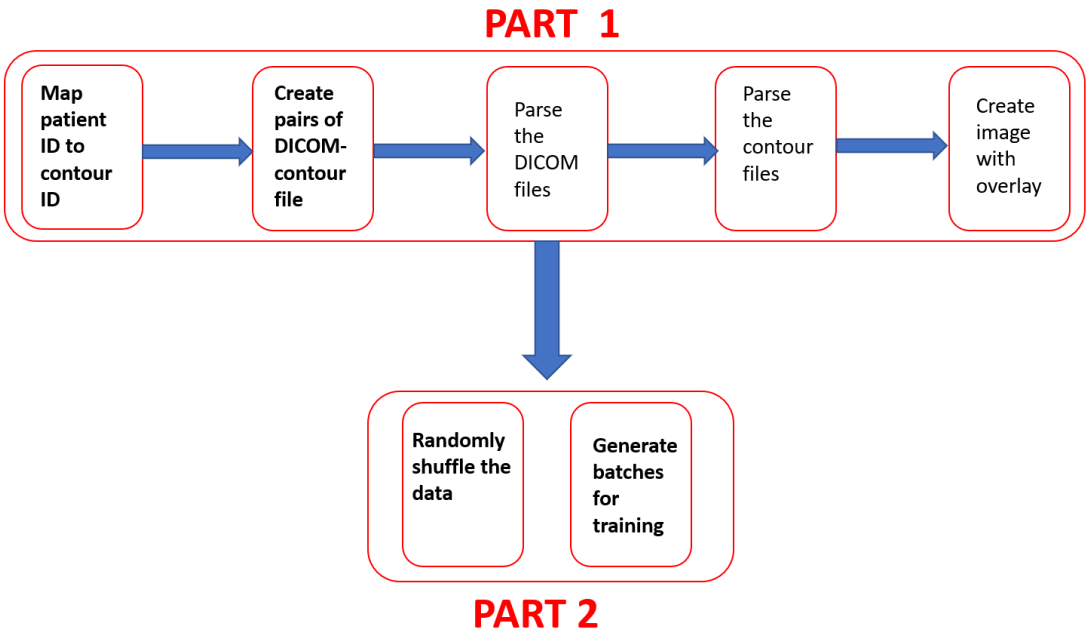

Figure 1: The proposed steps to implement part 1 and part 2 of the pipleine

Part 1

Implementation Details:

The pipeline consists of a main function which should be executed first. This main call set of functions in separate files under the function's directory:

1- **ReadLinkerSheet.py:** To read the sheet that links patient IDs to counter Folder names
2- **ParseLinkerSheet.py:** To parse the sheet that links patient IDs to counter Folder names and store them in dictionary
3- **DicomContourPair.py**: This file creates a dictionary that conations the DICOM path as key and the corresponding contour file as value
4- **Parsing.py:** This file parse both DICOM and mask files and return them as NumPy arrays
5- **SaveImages**.py: This file extracts the outer boundary of the mask image, overlay on the DICOM extracted image and save the overlay image
6- **GenerateBatches**.py: Randomly shuffle the data and divide them into batches. The final output is a list of tuples for images batches and masks batches. To be used in training later.

Dataflow in the pipeline:

The file names read from the DICOM directory and masks directory are stored in a nested dictionary as shown in Figure 2. In this way, the pipeline keeps tracking of the patient ID and its DICOM-MASK pairs to be able to build the output directory correctly. These nested dictionary structures are then used to read DICOM images, mask data, create the overlay and save it.

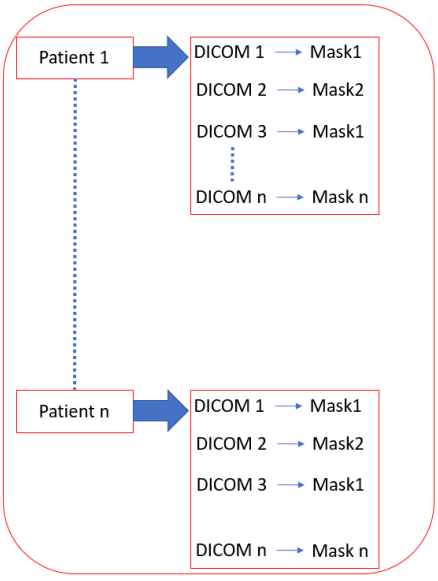

Figure 2: The nested dictionaries used in the pipeline dataflow

Creating images overlays:

To create image overlays the following steps are applied:

1- Normalize the DICOM values to be in range (0,255)

2- Convert the DICOM image to RGB
3- Extract the edges of the contour image to keep the outer boundary only
4- Loop over the all the pixels and check if the pixel is on the outer contour boundary to mark it with red on DICOM image
5- Save the image to the selected output path

Figure 3 shows some examples of the output overlay images.

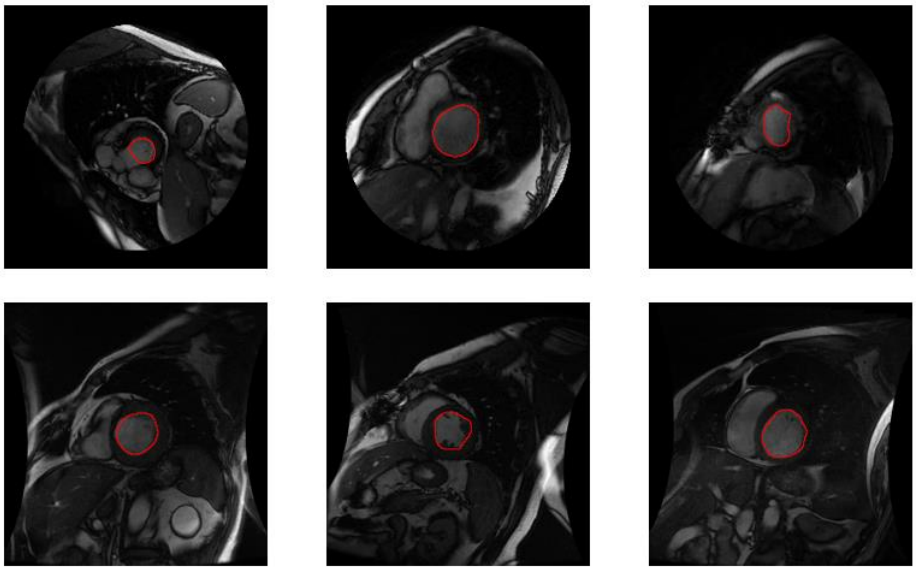

Figure 3: Examples of the created overlays of outer contour over the DICOM image

Part 2

Implementation Details:

The output of part 1 was used to construct two 3D-arrays, one for DICOM images and the other for mask. The size of each array is W x L X n, where W and L are the width and height of the images respectively, and n is the total number of images having masks. To generate batches, the following steps were applied:

1- Generate a random sequence of range (0,n), this sequence is used as index for both DICOM and masks arrays.
2- Shuffle the 2 arrays with the same random index.
3- Loop over the images to extract batches. If the number of images is not divisible by the batch size, the last batch will be of smaller size.
4- Return an array of tuples with 2 elements per tuple, batch of images and batch of masks

These batches are to be used in the future learning steps.

There was no need to change any thing from part 1 to feed the output to part 2.

Testing and validation

In Order to make sure that the pipeline is working correctly, the following steps were applied:

1- Unit test for each developed module. Each module was tested with set of valid and invalid inputs to make sure that is working properly and handles the invalid inputs and exceptions correctly. Driver codes and some test cases are provided at each module
2- Visual inspections for the created overlays. The main purpose to create the overlay images is to visually check that the contour correctly bounds the left ventricle.
3- Inspection of the generated batches to make sure that they are randomly shuffled each time and the batches are correctly created with the desired sizes and the they cover the whole dataset.
4- Creating log file to track the start time and finish time of each step by adding the time stamp of start and finish time of each step.

Limitations and future work

There are some limitations and some points to be considered in the future work:

1- Adding a configuration file for the paths and the batch size and parse it is better than coding these parameters in the main function.
2- This pipeline does not handle the case of having images of different sizes. In case we have multiple sizes another step of resizing all images to a standard size will be required.
3- The generated images are rotated so they appear different from the original DICOM file when opened with DICOM viewer. This should not cause a problem in the future steps of the learning process.
4- The linker sheet column name are not obtained dynamically in this pipeline. This may be an improvement also in the future work.

Libraries needed and environment used

This pipeline is implemented using python 2.7 running on windows 10 and using pycharm IDE. All the needed libraries are open source and publicly available. The used libraries are:

1- Numpy
2- Pillow
3- Pydicom
4- Pandas
5- OS
6- Datetime