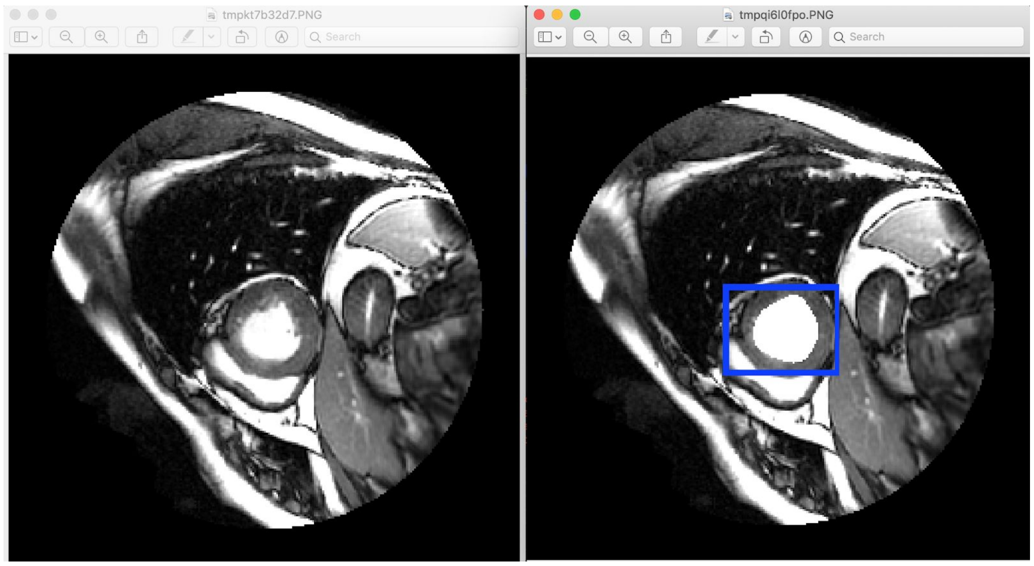
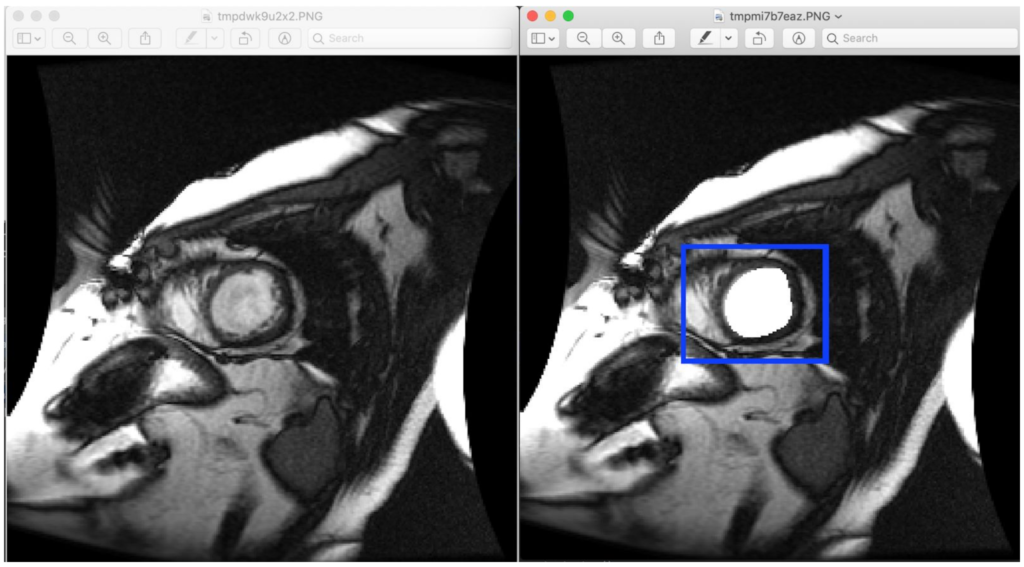
**Part 1:**

- How did you verify that you are parsing the contours correctly?

  With the help of unit tests. Moreover, I also visualized a few DICOM-contour pairs to ensure the linking is done right. In the two figures below, the left image is the original DICOM image and the right one shows the contour applied to the original image in white color. Notice the contour is applied correctly to the left ventricular blood pool region.





Besides, logging also provides some useful insights such as the number of images loaded and any errors while parsing. Below are some logs for reference.

```
2019-03-08 19:58:57,514 src.data_loader MainThread INFO Finding contours of dicom files. links_filename:../final_data/link.csv, dicom_path:../final_data/dicoms/, contour_path:../final_data/contourfiles/

2019-03-08 19:58:57,541 src.data_loader MainThread INFO Found contour files for 96 dicoms

2019-03-08 19:58:57,541 src.data_loader MainThread INFO Parsing dicom and contour files

2019-03-08 19:58:58,107 __main__ MainThread INFO loading batches of data for input to deep learning model. epochs:2, batch_size:8, buffer_size:10

2019-03-08 19:59:51,999 __main__ MainThread INFO done
```

- What changes did you make to the code, if any, in order to integrate it into our production code base?

  Among other things,

  1. test cases,
  2. structuring of the project (modules, classes, documentation, etc.), and
  3. logging

  were introduced.

  **Anything specific expected here that I missed?**

**Part 2:**

- Did you change anything from the pipelines built in Parts 1 to better streamline the pipeline built in Part 2? If so, what? If not, is there anything that you can imagine changing in the future?
  Given the pipeline you have built, can you see any deficiencies that you would change if you had more time? If not, can you think of any improvements/enhancements to the pipeline that you could build in?

  I introduced a new class for loading data (part 1) for clarity and easy scaling in the future. Since both parts are essentially pre-processing (data transformation) steps, I believe there is some more room for generalization. For example, all pre-processing steps could be turned into discrete ‘transformer’ steps and fed into sci-kit’s Pipeline. There might also be some utilities in TensorFlow that I could explore.

- How do you/did you verify that the pipeline was working correctly?

  With the help of unit tests. I used a standard library in part 2.

- List any other improvements that you think could be made?

  1. Mapping DICOM files to their corresponding contour files is a bit messy right now. I have to assume that the *contour file name* = IM-0001-<*DICOM filename*>-icontour-manual.txt. Perhaps in a production environment, we can organize the images and their contours in a way that their mapping becomes trivial.
  2. Software packaging can be improved by for e.g., making use of setup.py and other utilities.
  3. The results of part 1 can be persisted (that could be a part of the pipeline).
  4. Testing can be improved by increasing the code coverage and using mocks.
  5. Logging can be improved if necessary to add more details.