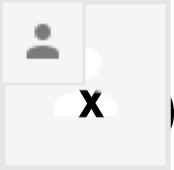


Summer Intern Report

Raziuddin Mahmood

UC Berkeley



Xoran project

- Goal:
 - Survey and select relevant annotation tooling to achieve 3D anatomical segmentation in cone beam CT studies
- Main results
 - Completed survey and prepared a presentation on annotation tooling for medical images
 - Selected ITKSnap as an appropriate tools for 3D anatomical segmentation
 - Generated 3D segmented volumes for selected anatomical structures (left eye, right eye, etc.)
 - Trained Xoran staff on the annotation tool which resulted in 17 studies annotated for about 9 regional volumes. However, not all regions were properly annotated
 - Experimented with a 2d U-net to learn and predict anatomical regions using annotated volumes for training data
 - Observations:
 - 2D U-net works well to interpolate within a volume. I.e. it can predict segmentation of regions within a volume
 - 2D U-net does not work well to predict regions across volumes, i.e if it is trained with raw unaligned CT studies, it cannot predict well in new studies continuous volumes.
 - Solutions to try in future:
 - Align volumes and normalize slice thickness so that there is a good correspondence between the training data CT studies. Right now, we are simply normalizing on the image resolution (i.e. 128 x 128). Need to normalize along the number of slices as well
 - Try 3D U-net with 4D data as input, i.e. each volume is treated in entirety rather than as a list of slices
- Conclusions:
 - 3D anatomy segmentation in cone beam CT volumes is a challenging problem that requires dedicated development and expertise of experienced medical imaging researchers



Weekly Progress

- Week 1: Choose a platform ([e.g.](#) 3D Slicer) and get familiar
 - ✓ Chose ITKSnap after researching several tools
 - ✓ Detailed report presentation on annotation tools provided
- Week 2-4: Prototype a simple labelling tool
 - ✓ User loads a Xoran Sinus CT image volume (can be DICOM)
 - ✓ User paints on various slices to generate binary DICOM mask
 - ✓ User saves mask image with associated label (e.g. eyeballs, or maxillary sinuses)
 - ✓ Label 10 or more datasets (e.g. the eyeballs, max. sinuses)
 - ✓ Trained on recognizing 4 different regions in 5 head CT DICOM studies provided.
 - ✓ Sinuses (maxillary sinus), deviated septum, left and right eyes in 1 full study
 - ✓ Left and right eye in all 4 CT studies
 - ✓ Generated 3d volumes from the annotation.
- Week 5-6: Train CNN using label masks
 - ✓ Focusing on training to recognize eye balls only
 - ✓ Implemented a basic 2D U-net architecture (classic)
 - ✓ Trained and tested on left eye ball currently using 2D slices within a volume for training
- Week 7: Demonstrate CNN can identify (e.g. eyeballs, max. sinuses) from labelled data.
 - ✓ Improve performance and expand to regions
 - ✓ Writing up a summary of experiments conducted
 - ✓ Final report

Progress report outline

- Annotation labeling tools survey
- Instructions on ITKSnap labeling
- Region segmentation U-net
- Issues encountered for training continuous volumes

ML Annotation for medical imaging

- Modalities
 - 2D imaging
 - 3D imaging
 - 4D imaging
- Annotation types
 - Whole image labeling
 - Anatomy regional labeling
 - 2D labeling
 - 3D labeling
 - Multi-d labeling
 - Anomaly regional labeling
 - 2D labeling
 - 3D labeling
 - Multi-d labeling
- Labels
 - Single level labels
 - Hierarchical labels

Annotation features

- Load
 - From local files
 - From server files
- Labels
 - Give options to create labels
 - Give options to edit labels (add, delete, save)
 - Give options to import labels (custom labels)
 - Allow labels to be described hierarchically (advanced)
- Regional annotations
 - Draw
 - Bounding box labels
 - Polygons
 - Continuous contours
 - Undo, clear features
 - Region fill
 - Auto-filling of regions once outlined
 - Undo feature
 - Multi-frame annotation in 3D
 - Interpolate using various methods
 - Works mostly for normal anatomy annotation
 - Abnormalities require advanced interpolation methods
- Rendering
 - Show in the same window
 - Show multiple objects
 - Show CT in multiple views
- Save formats
 - Nifti is easiest for processing Segmentation masks or rendered volumes in vtk format
 - Nifti to PNG and JPG can be done for ML training

Questions about Xoran scenario (Will determine what systems will be useful)

- Who are the annotators? Mostly Xoran employees?
 - Clinicians actually using your scanners or additional experts dedicated for annotation tasks?
 - If actual users, will they have the time to annotate data in routine care?
 - Will they require integration into their PACS so they don't have to switch systems?
 - How computer savvy are they?
 - Do they need training
 - Can software be installed on their machines? (permission, space, compute capabilities)
- Where will the data be stored? On desktop, user interface
 - On servers
 - On box
 - Can users access files and deposit completed files back?
- Is annotation management needed?
 - Should we keep track of division of work or status of annotations (who is done, how many more to go per annotator)?
 - Should it be linked to payment systems or some way to keep track for payment purposes? Keep track of payments themselves (who got paid)
- Cost of tools
 - Free tools or licensed software?
 - Pay as you go tools?

Essential consideration for annotation tools

- Desktop

- Pros:

- Easier to install and manage by one person
 - Display of 3D is local and fast
 - Local files can be used

- Cons:

- Data is copied multiple times
 - Data is copied onto client desktops (if the annotators are separate from the clinician users, this requires PHI and other issues to be addressed)
 - Stand alone tool if not integrated into PACS (picture archiving system), extra work for the clinician unless integrated into PACS
 - No way to manage annotators and their annotation status (advanced)
 - Active learning harder to do as the model training code requires larger infrastructure than is possible in client desktops (Advanced)

- Web-based

- Pros:

- Easier for collaborative work and management of the annotators
 - More secure as the data doesn't have to leave the server
 - Easy integration into a model-building environment
 - Payment tracking can be done
 - Easier to integrate an active learning model to reduce annotation effort, transparent to user

- Cons:

- Harder to display and render 3D imaging and their volumes extracted
 - Need separate browser-based access and internet connection
 - Upload and download delays can slow down operation

Summary of Annotation tools and their features

- [TrainingData.io](#): TrainingData.io is a medical image annotation tool for data labeling. It supports DICOM image format for radiology AI.
- [Lionbridge AI](#): Lionbridge AI has deep experience in all aspects of the medical devices vertical. We have 500,000 qualified contributors who can provide image annotation services quickly, with high precision. In addition, Lionbridge's team can help you manage your project timeline, budget, and quality control.
- [ImageJ](#): ImageJ is a Java-based image processing program developed at the National Institutes of Health and the Laboratory for Optical and Computational Instrumentation.
- [OsiriX Viewer](#): OsiriX is an image processing application for Mac dedicated to DICOM images produced by equipment. OsiriX is complementary to existing viewers, in particular to nuclear medicine viewers. It can also read many other file formats: TIFF (8,16, 32 bits), JPEG, PDF, AVI, MPEG and QuickTime.
- [ITK-SNAP](#): ITK-SNAP is a free-software and cross-platform tool that provides semi-automatic segmentation using active contour methods, as well as manual delineation and image navigation.
- [Cogito](#): Cogito provides machine learning training data. The services offered include image annotation, content moderation, sentiment analysis, chatbot training, and more.
- [LabelBox](#): Labelbox is a platform for data labeling, data management, and data science. Its features include image annotation, bounding boxes, text classification, and more.
- [Daturks](#): Daturks is a data annotation outsourcing company that offers many data annotation capabilities, ranging from image segmentation to named entity recognition (NER) tagging in documents.
- [ePAD](#): ePAD is a freely available quantitative imaging informatics platform, developed by the Rubin Lab at Stanford Medicine Radiology at Stanford University. Thanks to its plug-in architecture, ePAD can be used to support a wide range of imaging-based projects.
- [3D Slicer](#): 3D Slicer is an open source software platform for medical image informatics, image processing, and three-dimensional visualization. Built over two decades through support from the National Institutes of Health and a worldwide developer community, Slicer brings free, powerful cross-platform processing tools to physicians, researchers, and the general public.
- [Edgecase](#): Edgecase is a data factory providing synthetic data and data labeling services. With connections to universities and industry experts, Edgecase provides data annotation and complex datasets to AI companies in retail, agriculture, medicine, security and more.
- [Parallel Dots](#): Parallel Dots provides text, image, and video analysis API such as sentiment analysis and face recognition.
- [Ratsnake](#): Ratsnake is a software tool for efficient, semantically-aware graphic annotation of images. It aims to aid the collection of knowledge regarding image content for pattern recognition, image mining and related applications. Ratsnake uses a novel semi-automatic image annotation framework based on a customizable active contour model that takes into account quick user annotations.
- [Superannotate.com](#): The tool offers both vector annotations (boxes, polygons, lines, ellipses, key-points with templates and cuboids) and pixel-wise annotation with a brush. It provides both image and video annotations.
- [Supervise.ly](#): A web-based tool is likely not what your team wants to develop for medical images, but it gives you a feel for that alternative. It does not have all the features, for

Experiences with 3D Slicer

- Takes a long time to launch
- Annotation with the pen tool not very easy. Expects we are using an actual mouse
 - Seems to give annular region at the time of drawing (based on the pen size)
 - Frame number of the sequence not easily visible in the scroll bar
 - The pen tool's drawing point is outside the pen and non-intuitive
 - Correct the contour is also cumbersome having to draw around the region to erase rather than simply backspace and undo
 - While drawing, there is no clear option to do a mass undo (None is the non-obvious clear option)
 - The image region disappeared

References for advanced interpolation and ML-driven annotation

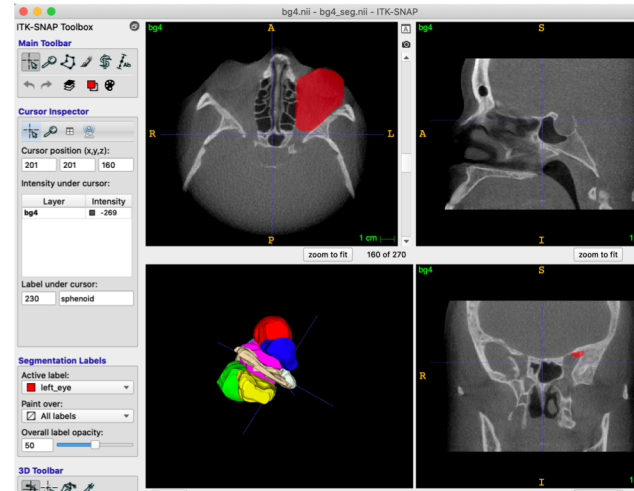
1. [Fast anatomy segmentation by combining low resolution multi-atlas label fusion with high resolution corrective learning: An experimental study](#)
Wang, Hongzhi and Prasanna, Prasanth and Syeda-Mahmood, Tanveer
Biomedical Imaging (ISBI 2017), 2017 IEEE 14th International Symposium on, pp. 223—226
2. [An experimental study on combining the auto-context model with corrective learning for canine LEG muscle segmentation](#)
Wang, Hongzhi and Cao, Yu and Syeda-Mahmood, Tanveer
Biomedical Imaging (ISBI), 2015 IEEE 12th International Symposium on, pp. 1106—1109
3. [Multi-atlas label fusion with augmented atlases for fast and accurate segmentation of cardiac MR images](#)
Xie, Long and Sedai, Suman and Liang, Xi and Compas, Colin B and Wang, Hongzhi and Yushkevich, Paul A and Syeda-Mahmood, Tanveer
Biomedical Imaging (ISBI), 2015 IEEE 12th International Symposium on, pp. 376—379
4. Rapid annotation of 3D medical imaging datasets using registration-based interpolation and adaptive slice selection
H Wang, P Prasanna, T Syeda-Mahmood, 2018 IEEE 15th International Symposium on Biomedical Imaging (ISBI 2018)
5. A Multi-atlas Approach to Region of Interest Detection for Medical Image Classification, H Wang, M Moradi, Y Gur, P Prasanna, T Syeda-Mahmood, International Conference on Medical Image Computing and Computer-Assisted Interaction (MICCAI), 2015
6. Fast anatomy segmentation by combining coarse scale multi-atlas label fusion with fine scale corrective learning
H Wang, D Kakrania, H Tang, P Prasanna, T Syeda-Mahmood, *Computerized Medical Imaging and Graphics* 68, 16-24, 2018
7. Atlas propagation through template selection, H. Wang, R. Zhang, MICCAI 2018
8. Integrating Deformable Modeling with 3D Deep Neural Network Segmentation, H Tang, M Moradi, KCL Wong, H Wang, A El Harouni, T Syeda-Mahmood, *Deep Learning in Medical Image Analysis*, 2018

Conclusions from Annotation Tooling Survey

- Based on the requirement analysis of Xoran's needs, down-selected to two tools:
 - 3D Slicer
 - ITKSnap
- After comparative evaluation of features and ease of use for the intended tasks suggested to adopt ITKSnap for annotation

Using the ITKSnap tool

- Assembled instructions for use of ITKSnap tool
- [Link to the tutorial document recorded](#)
- Performed annotations of left and right eye in 6 CT volumes



Annotations in ITKSnap - Tutorial

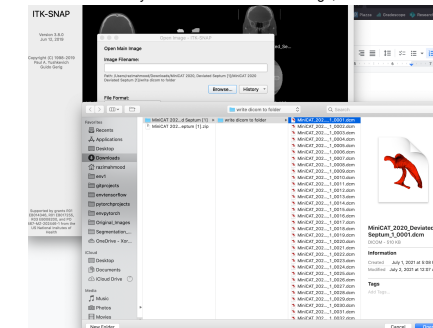
In this tutorial, I will outline some basic annotation operations using the ITKSnap tool which makes it easy to do volume annotations. The instructions are a bit tailored for use in downstream AI model building as well. The basic operations are:

1. Loading and saving DICOM files as Nifti format files.
2. Performing basic region annotation
3. Interpolation to create 3D volume and saving the segmentation map
4. Choosing label name and intensity for rendering
Repeat with more than one region annotated within the same segmentation map

1. Loading and saving DICOM files as Nifti format files.

Nifti's gz compressed format takes up a small space. All DICOMDIR single DICOM files can be bundled into one Nifti (nii.gz) file for ease of use later in AI model building, where we will need to load the original volumes and their segmentation masks.

1. Export all .xstd, .xpla, etc file formats into a .dcm format folder
 - a. Containing the Dicomdir file within the folder might be a good idea
2. Open ITKSnap > Open Image > Browse > go to the folder with the .dcm images
 - a. Select the first .dcm since ITK will automatically load all the images
 - b. Verify that it's a DICOM Series Image, click Next and Finish



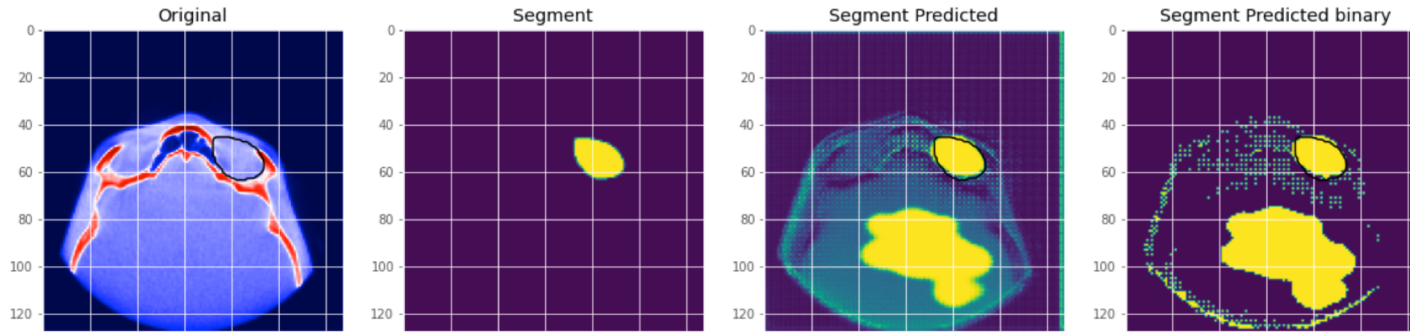
3. Once you have the core image, save the core image as a NIFTI format image

Segmentation experiments

- Using a 2D U-net implementation
 - Customized a 2D U-net to work with 128 x 128 size images (to fit into my machine memory)
 - Scaled all incoming images to be of size 128 x 128 since every volume was of somewhat different size images
 - Inputs were full 3D volumes organized as consecutive slices
 - Outputs were masked images corresponding to marked region(s)
- Experiment-1:
 - Train a U-net model for a single region, e.g. left or right eye
 - Train-validate partition were 90%-10%
 - Validation accuracy =0.998
 - A network trained on a volume was able to predict the slices containing the masked region in its own volume => U-net was OK for interpolation within a volume
 - However, prediction on a field test study showed choppy prediction

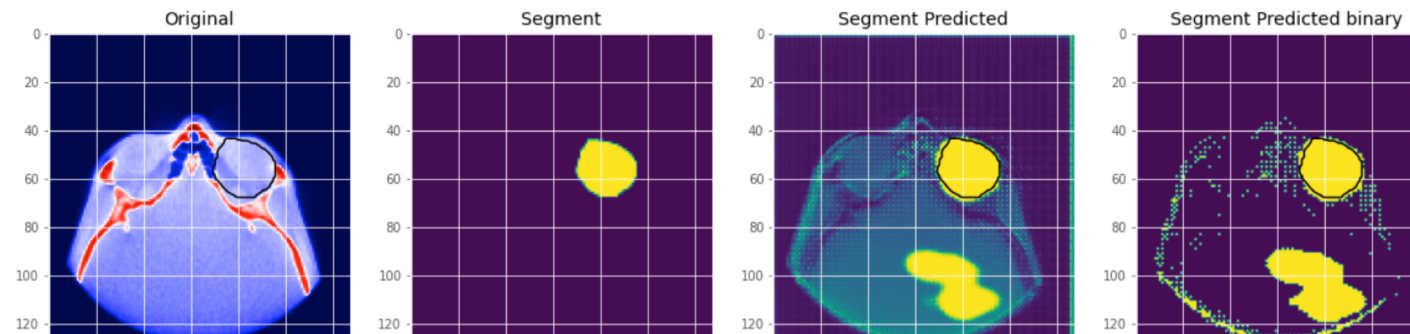
Results on Sinus CT Scan

```
: 1 plot_sample(X_valid, y_valid, preds_val, preds_val_t, ix=0)
```

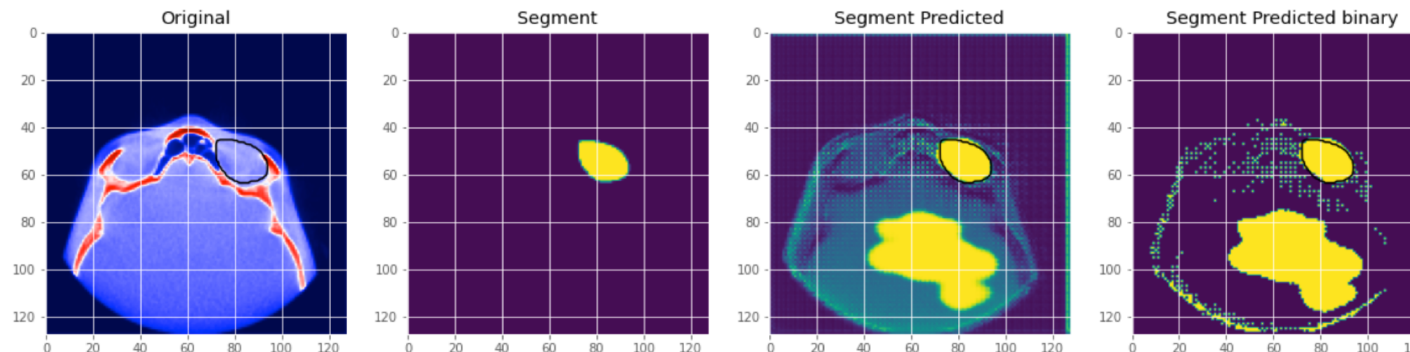


After 10 epochs
(insufficient training)

```
] : 1 plot_sample(X_valid, y_valid, preds_val, preds_val_t, ix=1)
```



```
: 1 plot_sample(X_valid, y_valid, preds_val, preds_val_t, ix=3)
```

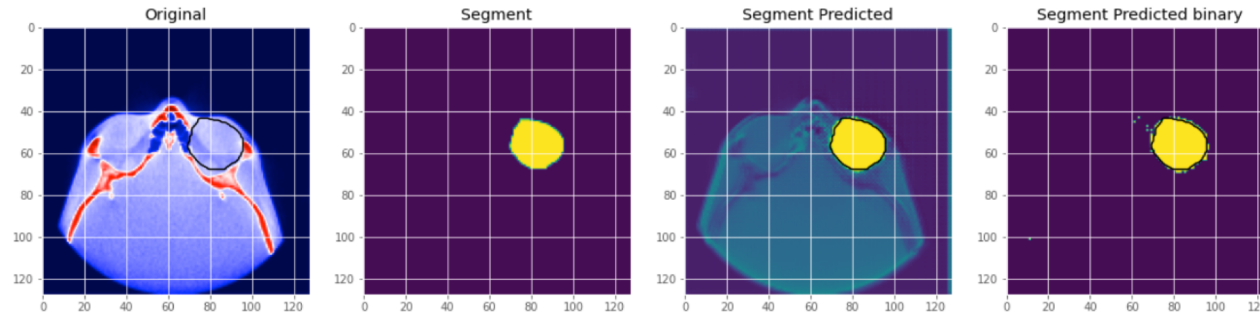


Result on Sinus CT Scan

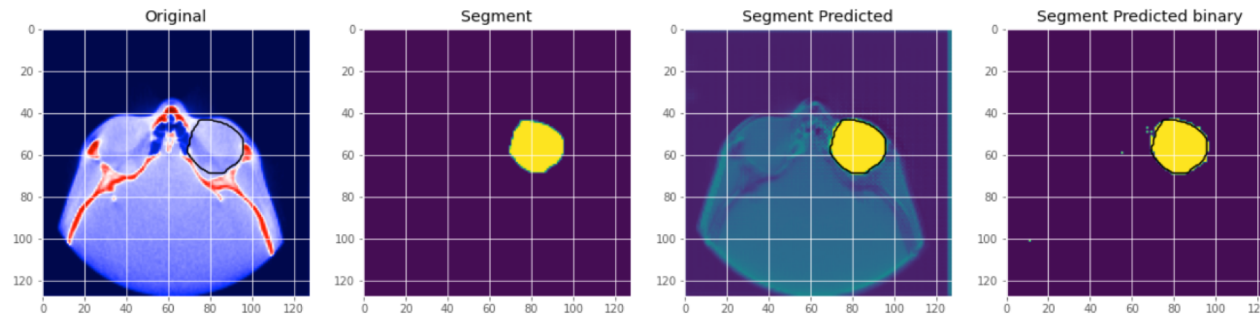
After 17 more
epochs

Result on left eye segmentation
prediction

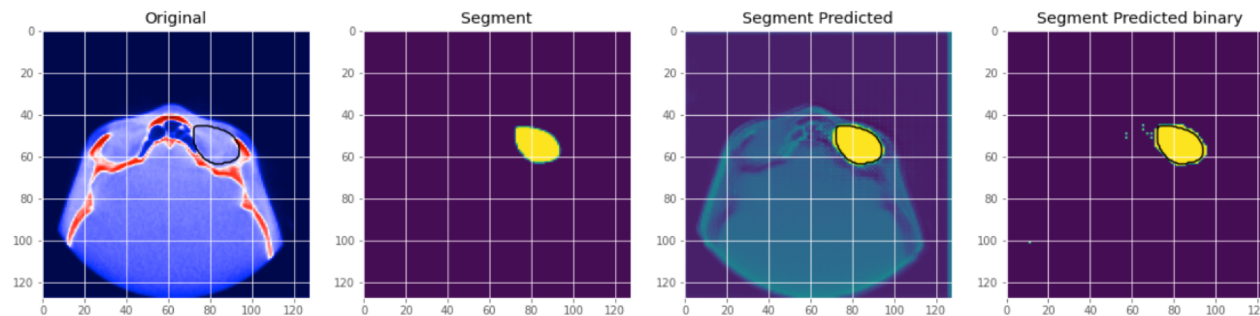
```
In [199]: 1 plot_sample(X_valid, y_valid, preds_val, preds_val_t, ix=1)
```



```
In [200]: 1 plot_sample(X_valid, y_valid, preds_val, preds_val_t, ix=2)
```

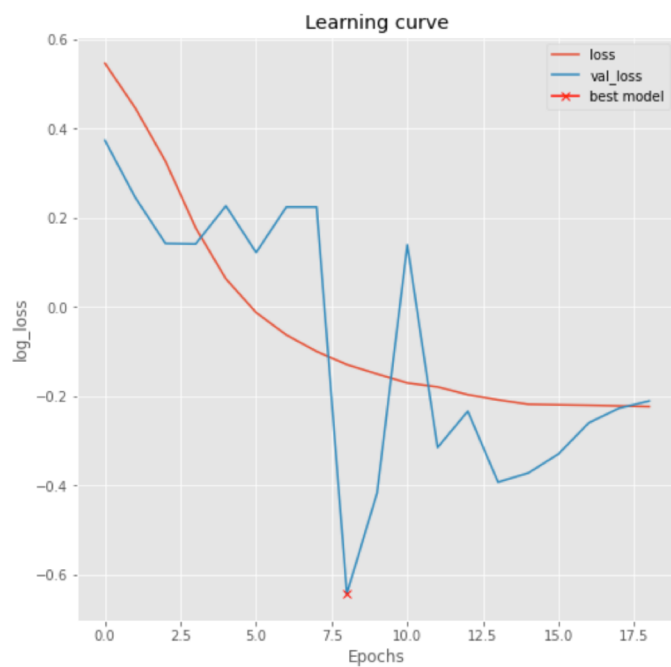


```
In [201]: 1 plot_sample(X_valid, y_valid, preds_val, preds_val_t, ix=3)
```

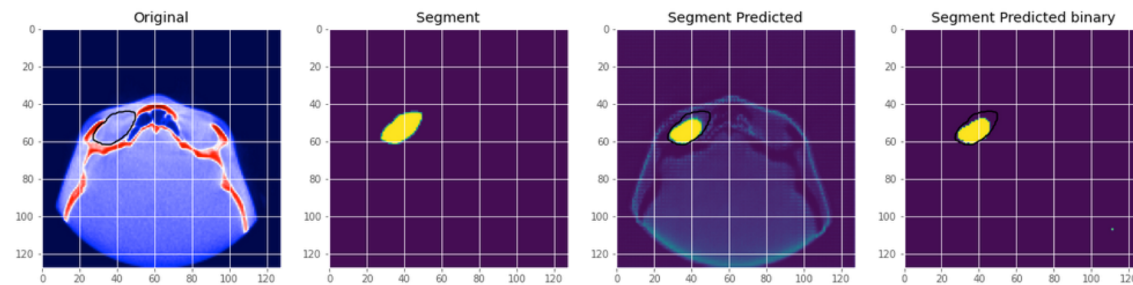


Result on right eye prediction

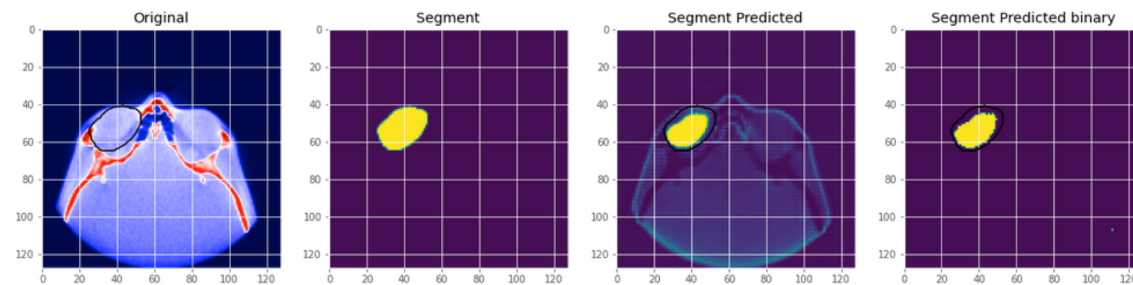
Stopped after 19 epochs



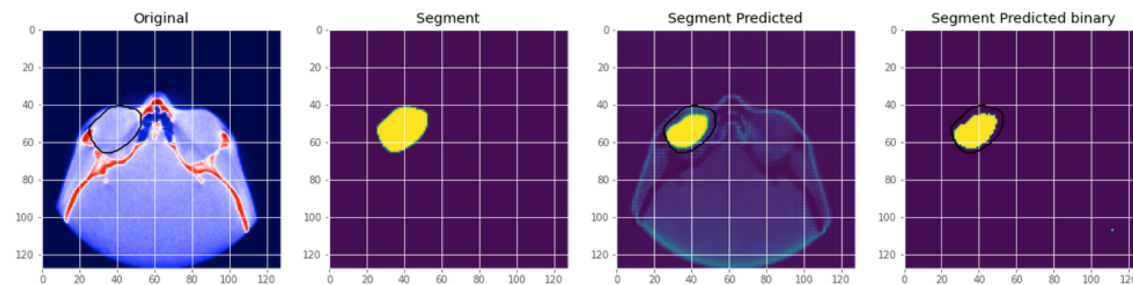
```
In [228]: 1 plot_sample(X_valid, y_valid, preds_val, preds_val_t, ix=0)
```



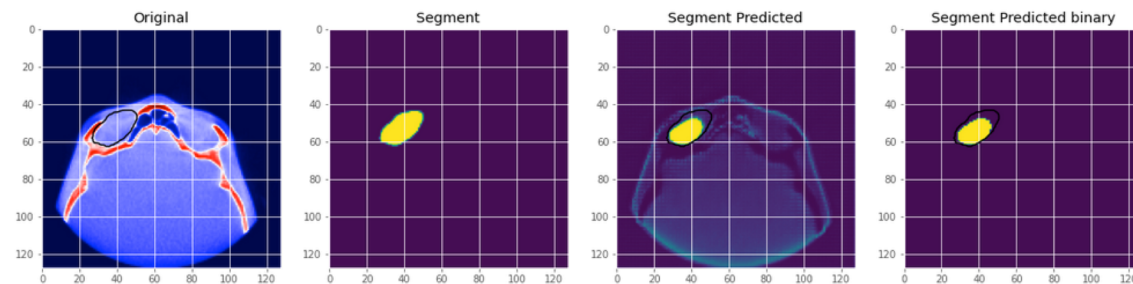
```
In [229]: 1 plot_sample(X_valid, y_valid, preds_val, preds_val_t, ix=1)
```



```
In [230]: 1 plot_sample(X_valid, y_valid, preds_val, preds_val_t, ix=2)
```



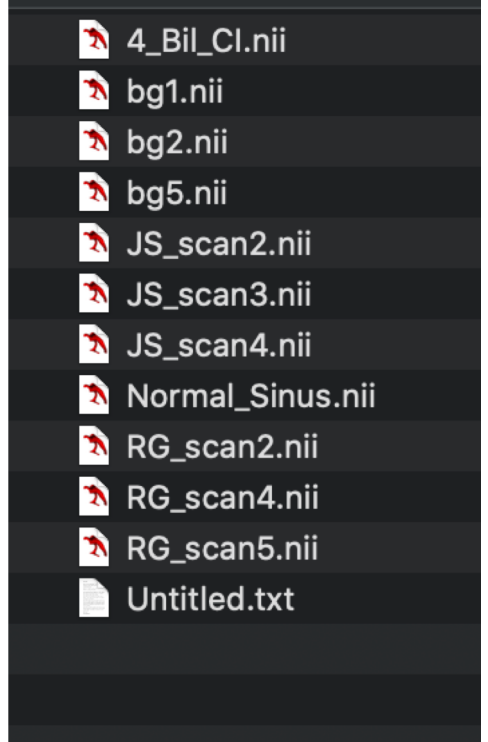
```
In [231]: 1 plot_sample(X_valid, y_valid, preds_val, preds_val_t, ix=3)
```



Segmenting multiple regions

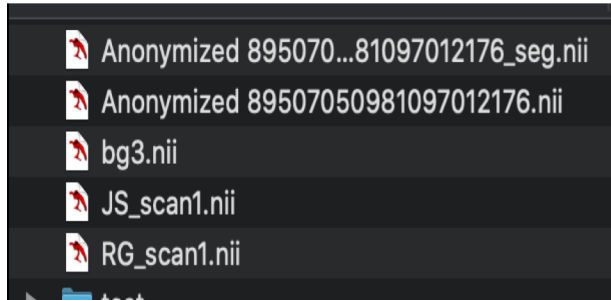
- For simultaneous multi-region segmentation
 - One U-net trained for all regions
 - Trained U-net worked well for prediction within the training volumes
 - Prediction on a new volume was choppy
 - Region distinction was lost due to mask image being a binary image
 - 11 studies for training, and 3 studies for prediction
 - Other studies annotated had problems with one region annotation or another
 - Problematic studies:
 - Training across volumes
 - All slices across volumes intermixed and used for train-test splits
 - This did not yield good results
 - Training of the model, one volume at a time, and each volume split into train-test partition
 - Model initialized for the first volume
 - Subsequent volumes start from a pre-trained model

Training volumes used



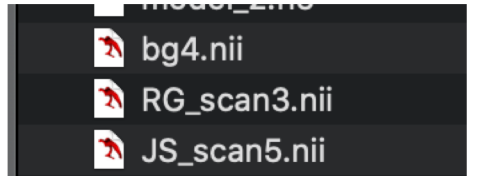
A screenshot of a file explorer window showing a list of training volumes. The files are: 4_Bil_Cl.nii, bg1.nii, bg2.nii, bg5.nii, JS_scan2.nii, JS_scan3.nii, JS_scan4.nii, Normal_Sinus.nii, RG_scan2.nii, RG_scan4.nii, RG_scan5.nii, and Untitled.txt. Each file has a small icon to its left.

Excluded volumes due to annotation issues



A screenshot of a file explorer window showing a list of excluded volumes. The files are: Anonymized 895070...81097012176_seg.nii, Anonymized 89507050981097012176.nii, bg3.nii, JS_scan1.nii, and RG_scan1.nii. Each file has a small icon to its left.

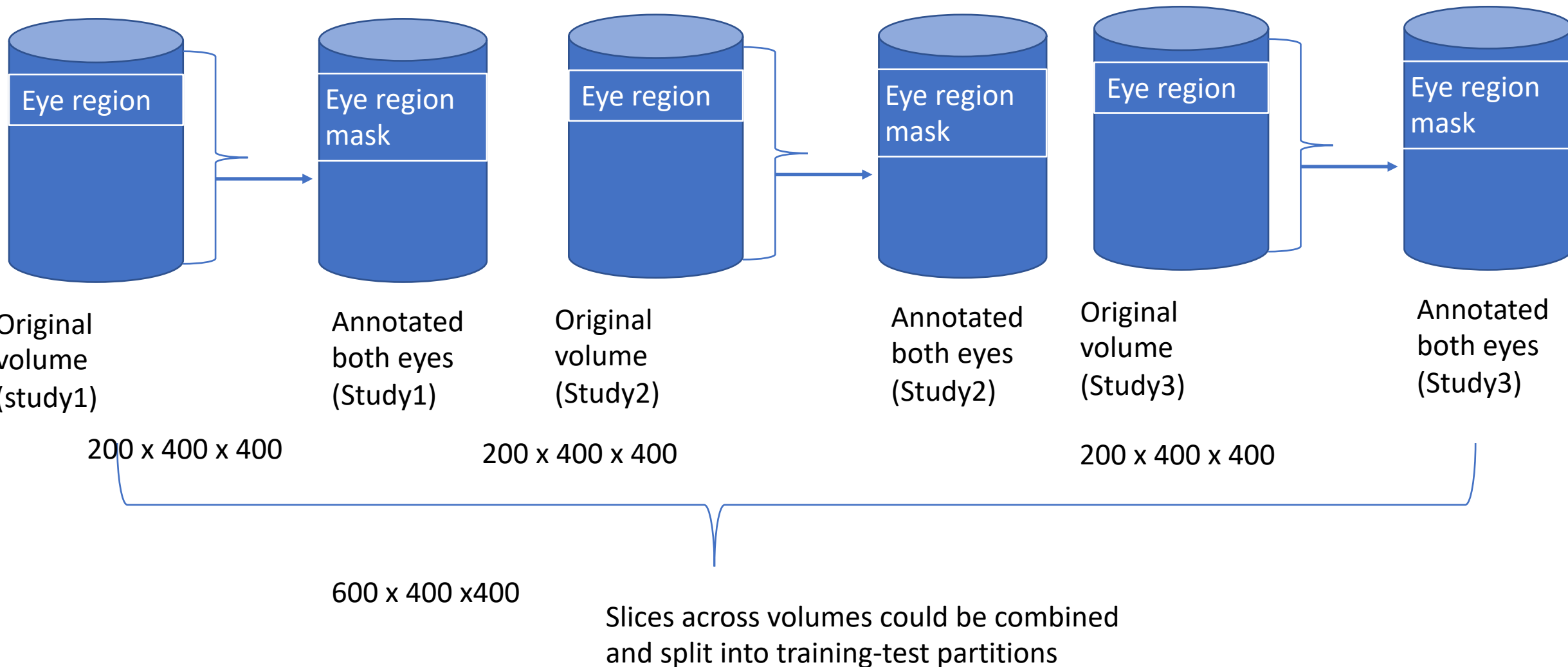
Prediction volumes used



A screenshot of a file explorer window showing a list of prediction volumes. The files are: bg4.nii, RG_scan3.nii, and JS_scan5.nii. Each file has a small icon to its left.

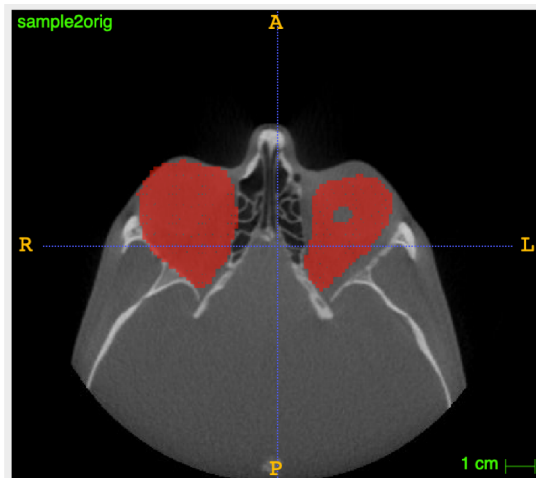
Training from multiple CT studies

Mixing and matching of slices across volumes and using a 2D U-net. -> This did not yield good results

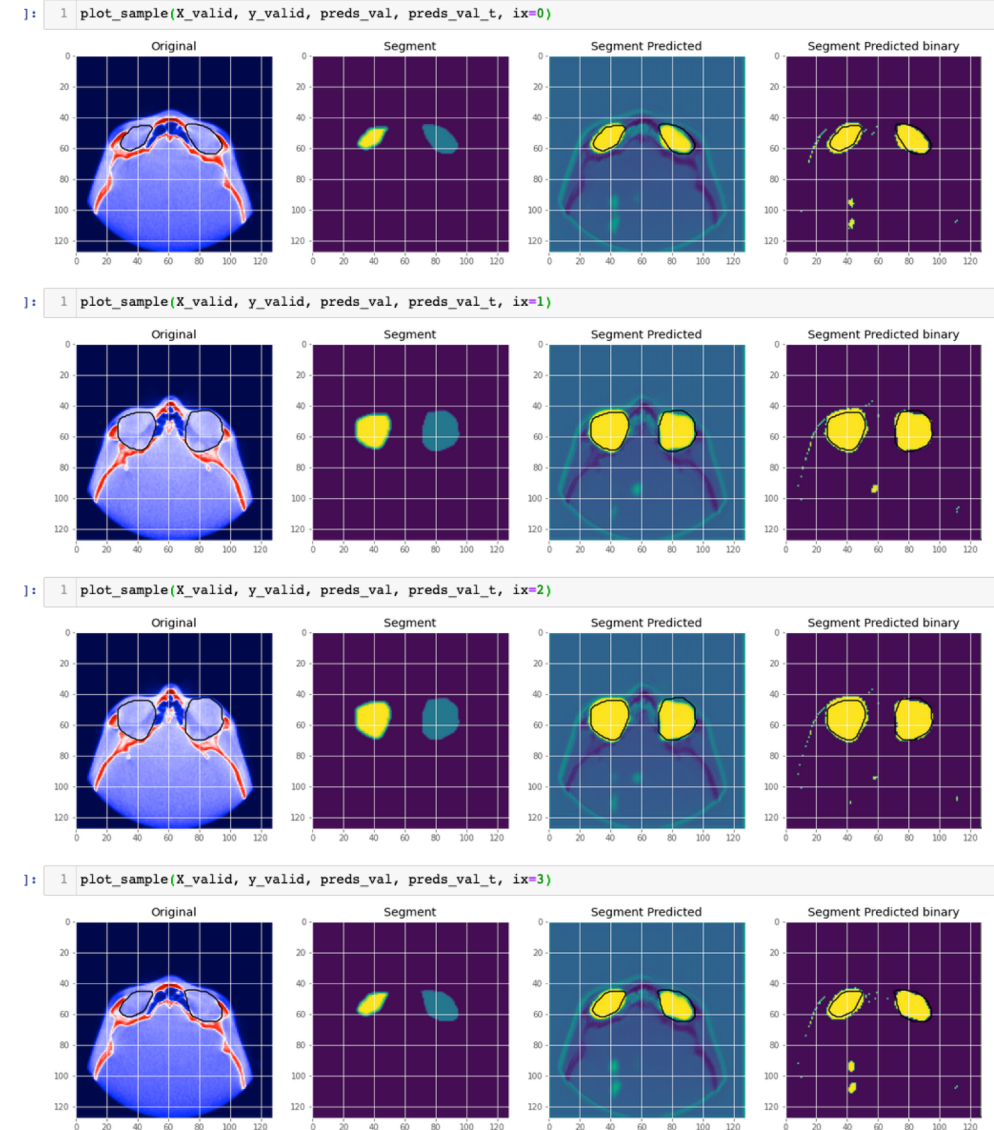


Predicting left and right eye using a single model

- Single model can recognize both regions
- But region label distinction lost after prediction
 - For example, both eyes have red color here in this figure



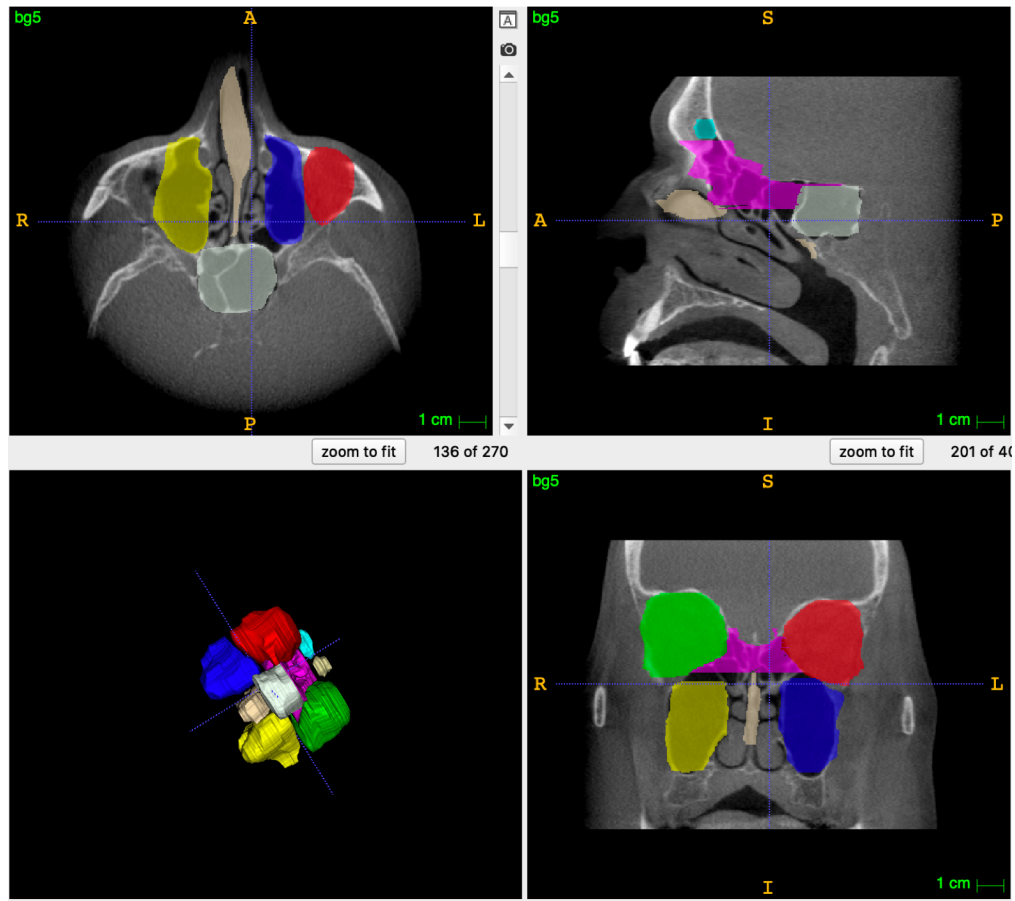
Predicted regions in validation study slices



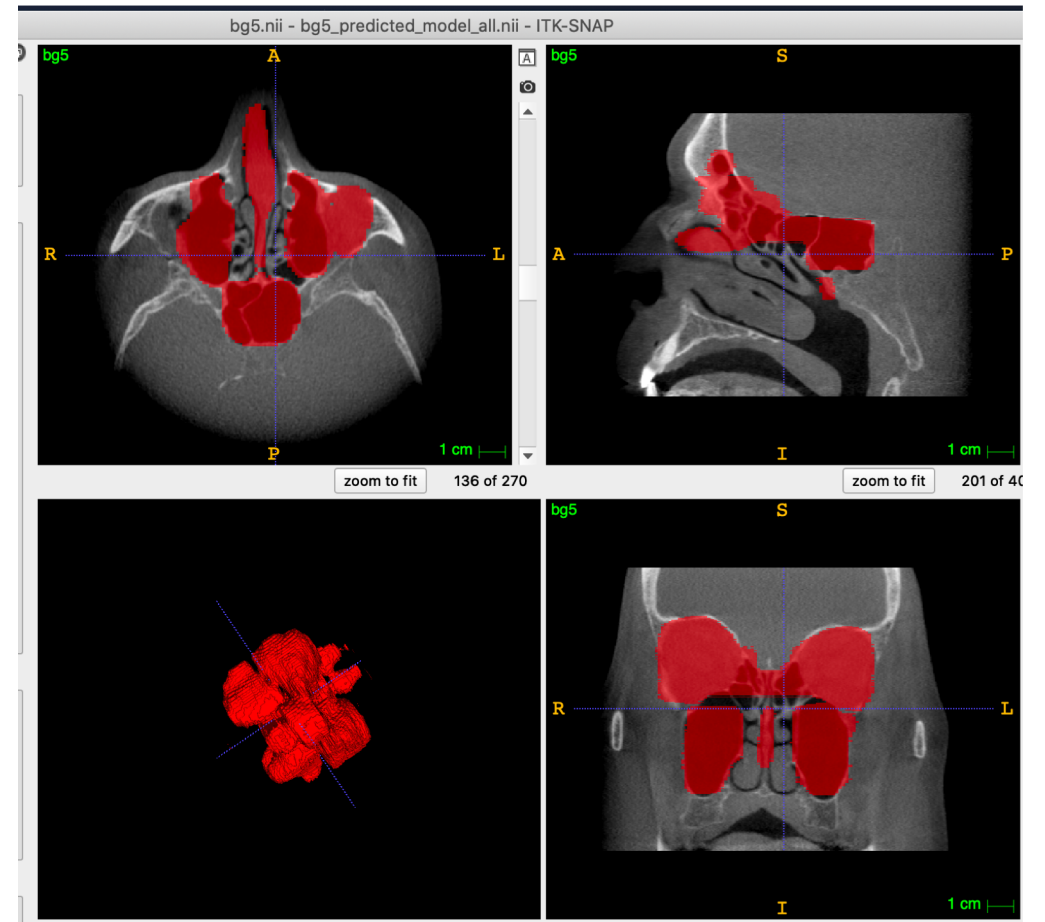
Prediction within and across volumes

- Using a single model for multiple regions prediction within the training volumes after learning with the 2D U-net

Ground truth segmentation – bg5.nii



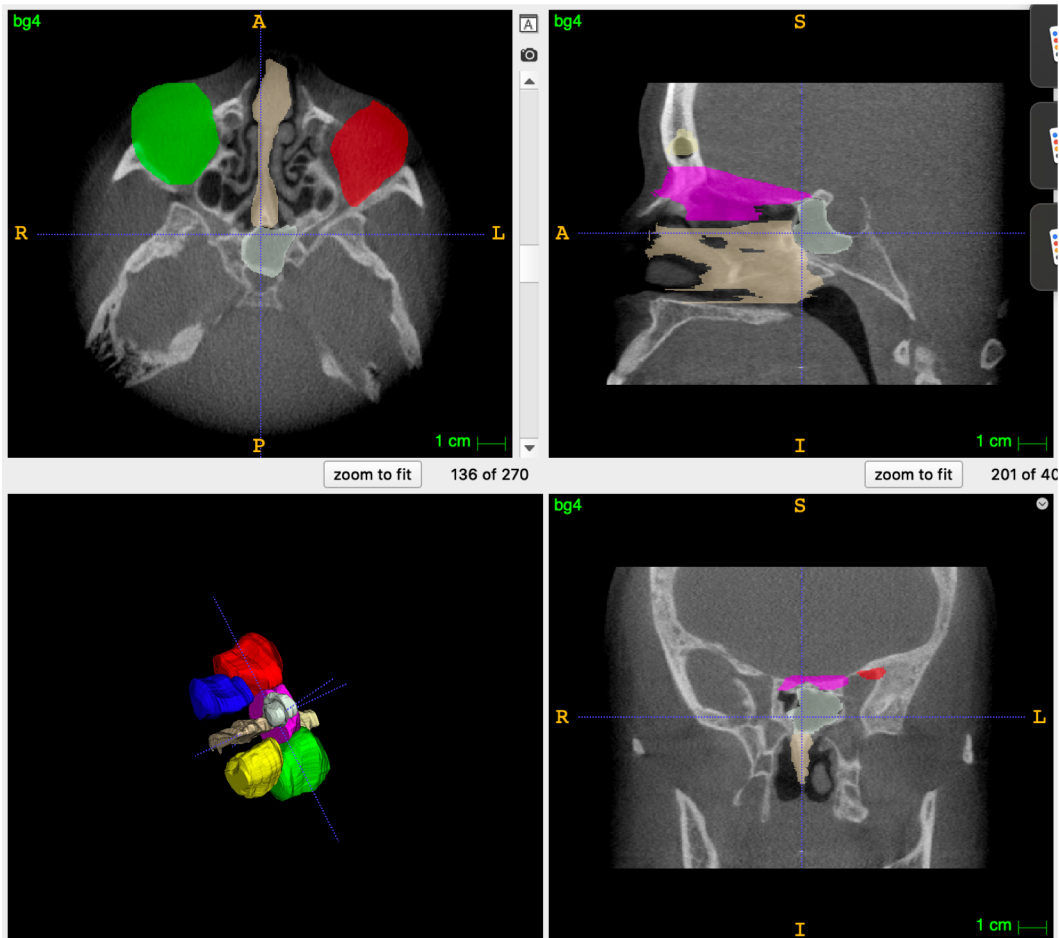
Predicted segmentation – bg5_predicted.nii



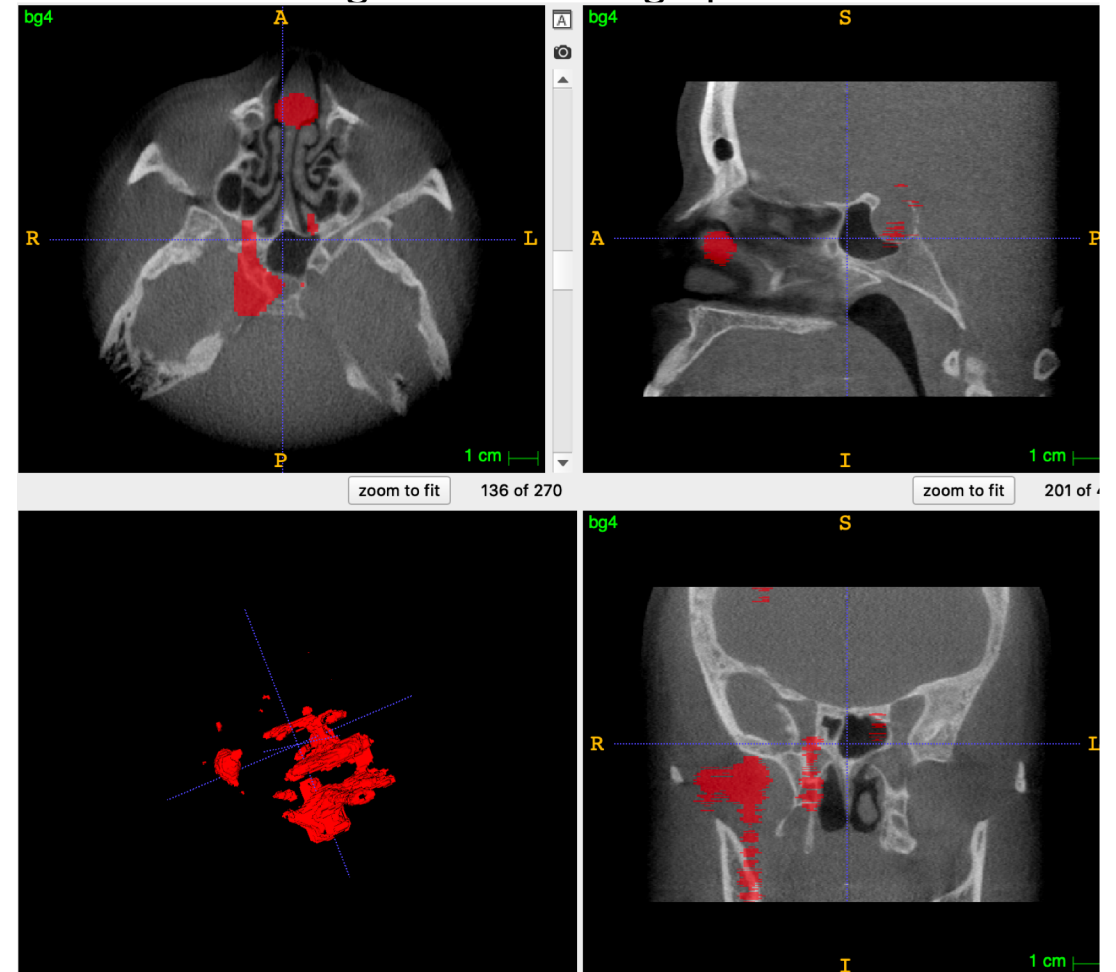
Prediction of multiple regions using single U-net model

- For prediction in a test volume – the results are choppy

Ground truth segmentation – bg4.nii



Predicted segmentation – bg4-pred.nii



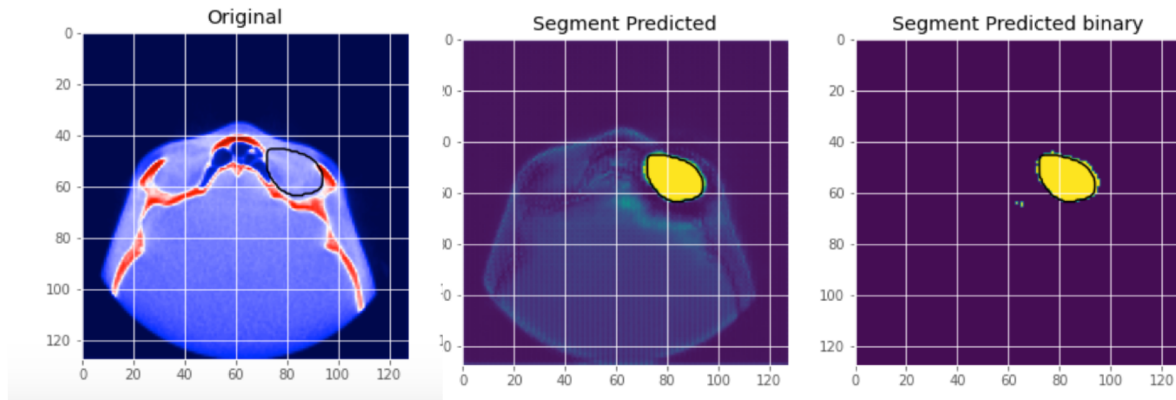
Prediction of regions using multiple U-net models

- Train 1 U-net model per anatomical region to be segmented
- Blend the results of all predicted segmentations
- 11 studies used for training and 3 studies for testing the predictions
- Trained 9 models for the 9 regions
- Prediction results on test volumes were spotty but captured some portions of the volumes well
 - Results not sufficient for producing Dice coefficient-based accuracy

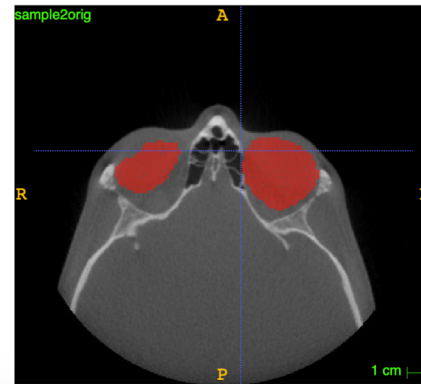
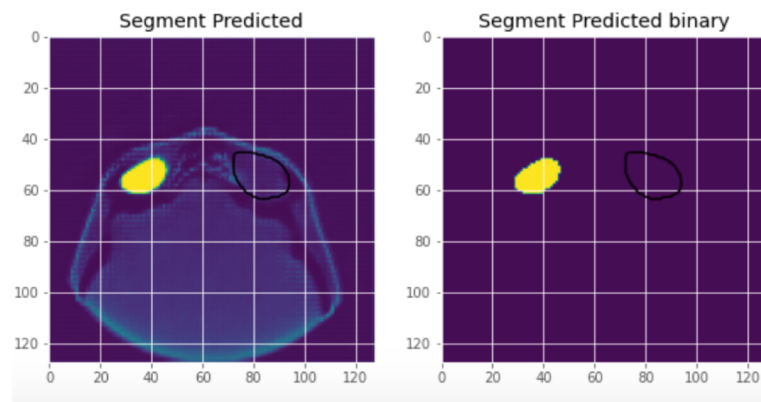
0	Clear Label
1	Label 1
20	ethmoid
50	left_eye
80	right_eye
110	left_maxillary
140	right_maxillary
170	left_frontal
200	right_frontal
230	sphenoid
250	nasal_septum

Predicting both left and right eye from separate models

```
: 1 plot_sample(X_valid, y_vali
```



```
=3)
```



Predicting left and right eye using respective models in the same test image

2-model solution works!

Combined the binary predictions into a single image. It is possible to retain the color distinction of regions later.

Demo

- Showing reconstruction in ItKsnap by overlaying on the original image
- Training data:
 - Original image + manual segmentation
 - sample2orig.nii
 - botheyeseg.nii (manual segmentation)
- Predictions using 2 separate models (one for left and one for right eye)
 - Original image + predicted segmentation (2models combined result)
 - sample2orig.nii
 - predictedSeg2.nii
- Predictions using 1 single model (for both left and right eye together)
 - Original image + predicted segmentation (1 model output)
 - sample2orig.nii
 - predictedbothSeg2.nii
- Prediction using 2 separate models (one for left and right eye) on a field dataset (i.e. not part of training or validation or testing)
 - Different Original image + predicted segmentation (2models combined result)
 - study2orig.nii
 - predictedstudy2.nii

Conclusions and next steps

- Segmentation of complex 3D structures is a difficult problem
 - Simple 2D U-net approach is insufficient to get full 3D segmentation volumes reconstructed
 - Results not so good due to 2 reasons:
 - Alignment between CT volumes for training
 - Each CT volume has different number of slices and begins in somewhat different areas of the body
 - Intensity differences need to be normalized as well
 - Registration software may need to be used for registering between all training and test volumes before training is done
 - Advanced registration algorithms are available with medical imaging researchers (please consult them)
 - The learning may need to be on 3D volumes directly instead of slices. A 3D version of U-net may need to be used
 - Current implementations of 3D U-net need some advanced libraries and GPU support to run in reasonable time
 - Since I didn't have access to GPUs, this could not be implemented in time.
 - Current dataset of 11 training volumes is insufficient for training a 3D U-net
- Next Steps (to be carried out by Xoran staff)
 - Create more training data and retrain model
 - Record Dice coefficient for the 3D volume reconstructed
 - Optimize ML parameters
 - Run cross-validation experiments
 - Annotate more structures