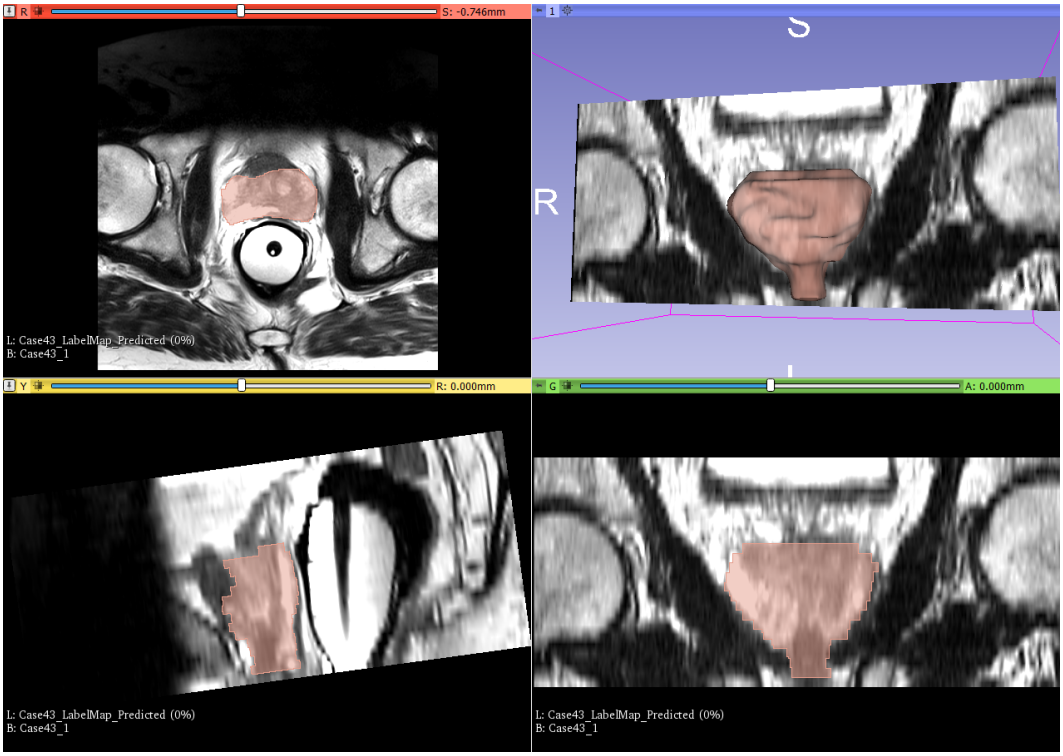
Assignment 3
CISC 881

Fraser Raney
fraser.raney@queensu.ca
10027693

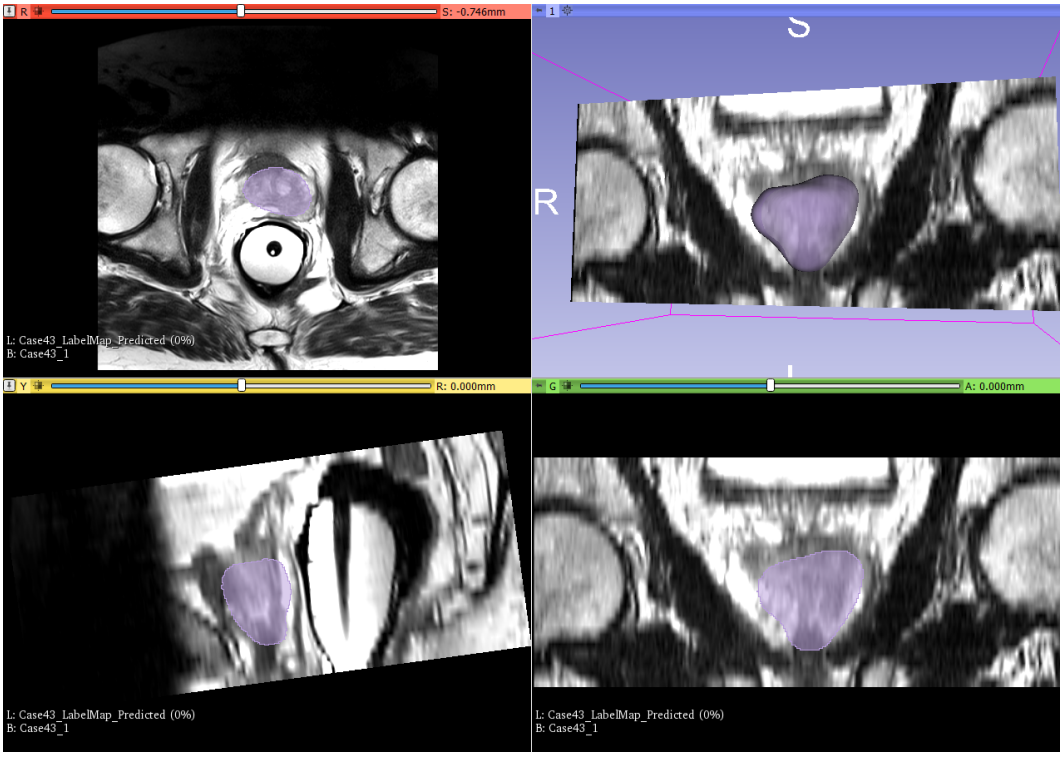
March 28, 2020

Part A

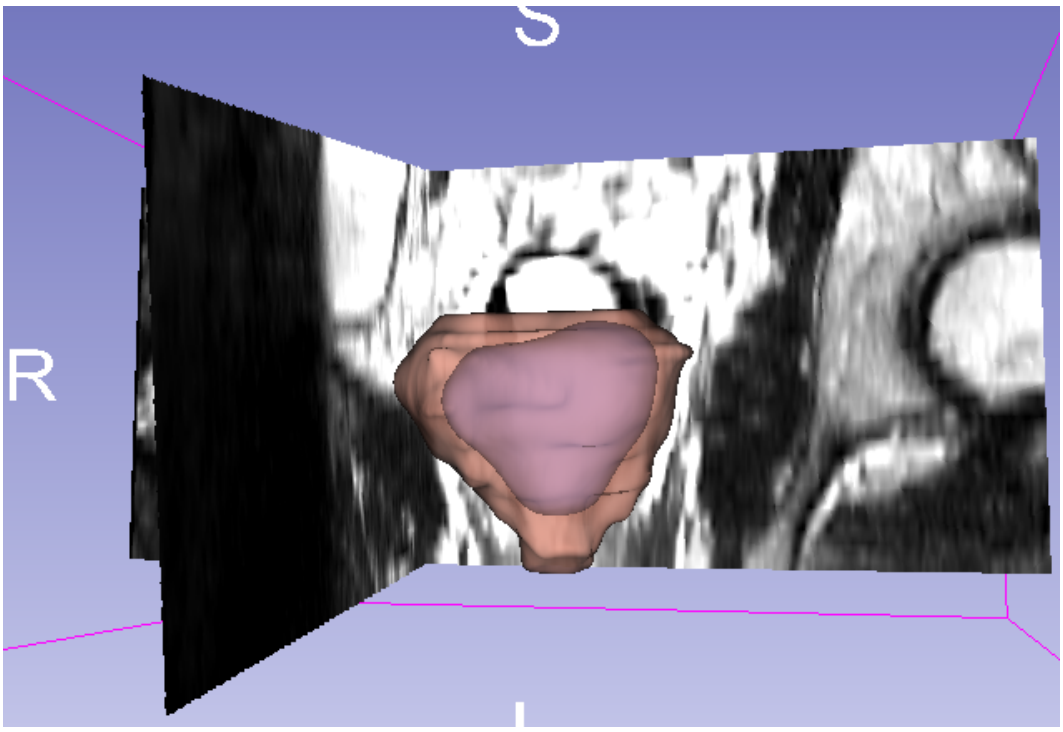
a)



b)


S: -0.746mm
S
R
L: Case43_LabelMap_Predicted (0%)
B: Case43_1
R: 0.000mm
A: 0.000mm
L: Case43_LabelMap_Predicted (0%)
B: Case43_1
L: Case43_LabelMap_Predicted (0%)
B: Case43_1

c)


S
R

Part B

f)

The data was stratified into train, validation and test datasets using the Scikit-Learn library method `train_test_split` with a list of the case identifiers, e.g. Case00. The slice filenames were fetched with a method that matched all Case identifiers in the set lists to directory contents.

```
def fetchNpyNames(caseNos):
    return np.array(list(filter(lambda filename: filename.split('.')[-1] \
                                == 'npy' and filename.split('_')[0] in caseNos,
                                os.listdir(imagesTr))))
```

g)

The deep learning model was implemented in PyTorch. The UNet model from `github.com/milesial/Pytorch-UNet` was used. The output layer was modified to include a sigmoid activation funciton.

$$f(x) = \frac{1}{1 + e^{-x}}$$

This model used four levels of contraction and expansion. Skip connections concatenated output from the respective downsampling level to the input to the upsampling levels. Each level of the encoder consisted of maxpooling to follwed by two sequences of ReLU activation, convoluton and batchnormalization, and followed by another maxpooling operation. The decoder used bilinear upsampling followed by the double convolutional sequence used in the encoder. The model summary is shown below. The input dimensions were adjusted for the Promise12 data.

Layer (type)	Output Shape	Param #
Conv2d-1	[-1, 64, 224, 224]	640
BatchNorm2d-2	[-1, 64, 224, 224]	128
ReLU-3	[-1, 64, 224, 224]	0
Conv2d-4	[-1, 64, 224, 224]	36,928
BatchNorm2d-5	[-1, 64, 224, 224]	128
ReLU-6	[-1, 64, 224, 224]	0
DoubleConv-7	[-1, 64, 224, 224]	0
MaxPool2d-8	[-1, 64, 112, 112]	0
Conv2d-9	[-1, 128, 112, 112]	73,856
BatchNorm2d-10	[-1, 128, 112, 112]	256
ReLU-11	[-1, 128, 112, 112]	0
Conv2d-12	[-1, 128, 112, 112]	147,584
BatchNorm2d-13	[-1, 128, 112, 112]	256
ReLU-14	[-1, 128, 112, 112]	0
DoubleConv-15	[-1, 128, 112, 112]	0
Down-16	[-1, 128, 112, 112]	0
MaxPool2d-17	[-1, 128, 56, 56]	0
Conv2d-18	[-1, 256, 56, 56]	295,168
BatchNorm2d-19	[-1, 256, 56, 56]	512
ReLU-20	[-1, 256, 56, 56]	0
Conv2d-21	[-1, 256, 56, 56]	590,080
BatchNorm2d-22	[-1, 256, 56, 56]	512
ReLU-23	[-1, 256, 56, 56]	0
DoubleConv-24	[-1, 256, 56, 56]	0
Down-25	[-1, 256, 56, 56]	0
MaxPool2d-26	[-1, 256, 28, 28]	0
Conv2d-27	[-1, 512, 28, 28]	1,180,160
BatchNorm2d-28	[-1, 512, 28, 28]	1,024
ReLU-29	[-1, 512, 28, 28]	0
Conv2d-30	[-1, 512, 28, 28]	2,359,808
BatchNorm2d-31	[-1, 512, 28, 28]	1,024

ReLU−32	[−1, 512, 28, 28]	0
DoubleConv−33	[−1, 512, 28, 28]	0
Down−34	[−1, 512, 28, 28]	0
MaxPool2d−35	[−1, 512, 14, 14]	0
Conv2d−36	[−1, 512, 14, 14]	2,359,808
BatchNorm2d−37	[−1, 512, 14, 14]	1,024
ReLU−38	[−1, 512, 14, 14]	0
Conv2d−39	[−1, 512, 14, 14]	2,359,808
BatchNorm2d−40	[−1, 512, 14, 14]	1,024
ReLU−41	[−1, 512, 14, 14]	0
DoubleConv−42	[−1, 512, 14, 14]	0
Down−43	[−1, 512, 14, 14]	0
Upsample−44	[−1, 512, 28, 28]	0
Conv2d−45	[−1, 256, 28, 28]	2,359,552
BatchNorm2d−46	[−1, 256, 28, 28]	512
ReLU−47	[−1, 256, 28, 28]	0
Conv2d−48	[−1, 256, 28, 28]	590,080
BatchNorm2d−49	[−1, 256, 28, 28]	512
ReLU−50	[−1, 256, 28, 28]	0
DoubleConv−51	[−1, 256, 28, 28]	0
Up−52	[−1, 256, 28, 28]	0
Upsample−53	[−1, 256, 56, 56]	0
Conv2d−54	[−1, 128, 56, 56]	589,952
BatchNorm2d−55	[−1, 128, 56, 56]	256
ReLU−56	[−1, 128, 56, 56]	0
Conv2d−57	[−1, 128, 56, 56]	147,584
BatchNorm2d−58	[−1, 128, 56, 56]	256
ReLU−59	[−1, 128, 56, 56]	0
DoubleConv−60	[−1, 128, 56, 56]	0
Up−61	[−1, 128, 56, 56]	0
Upsample−62	[−1, 128, 112, 112]	0
Conv2d−63	[−1, 64, 112, 112]	147,520
BatchNorm2d−64	[−1, 64, 112, 112]	128
ReLU−65	[−1, 64, 112, 112]	0
Conv2d−66	[−1, 64, 112, 112]	36,928
BatchNorm2d−67	[−1, 64, 112, 112]	128
ReLU−68	[−1, 64, 112, 112]	0
DoubleConv−69	[−1, 64, 112, 112]	0
Up−70	[−1, 64, 112, 112]	0
Upsample−71	[−1, 64, 224, 224]	0
Conv2d−72	[−1, 64, 224, 224]	73,792
BatchNorm2d−73	[−1, 64, 224, 224]	128
ReLU−74	[−1, 64, 224, 224]	0
Conv2d−75	[−1, 64, 224, 224]	36,928
BatchNorm2d−76	[−1, 64, 224, 224]	128
ReLU−77	[−1, 64, 224, 224]	0
DoubleConv−78	[−1, 64, 224, 224]	0
Up−79	[−1, 64, 224, 224]	0
Conv2d−80	[−1, 1, 224, 224]	65
OutConv−81	[−1, 1, 224, 224]	0

Total params: 13,394,177
Trainable params: 13,394,177
Non−trainable params: 0

Input size (MB): 0.19
Forward/backward pass size (MB): 689.06
Params size (MB): 51.09
Estimated Total Size (MB): 740.35

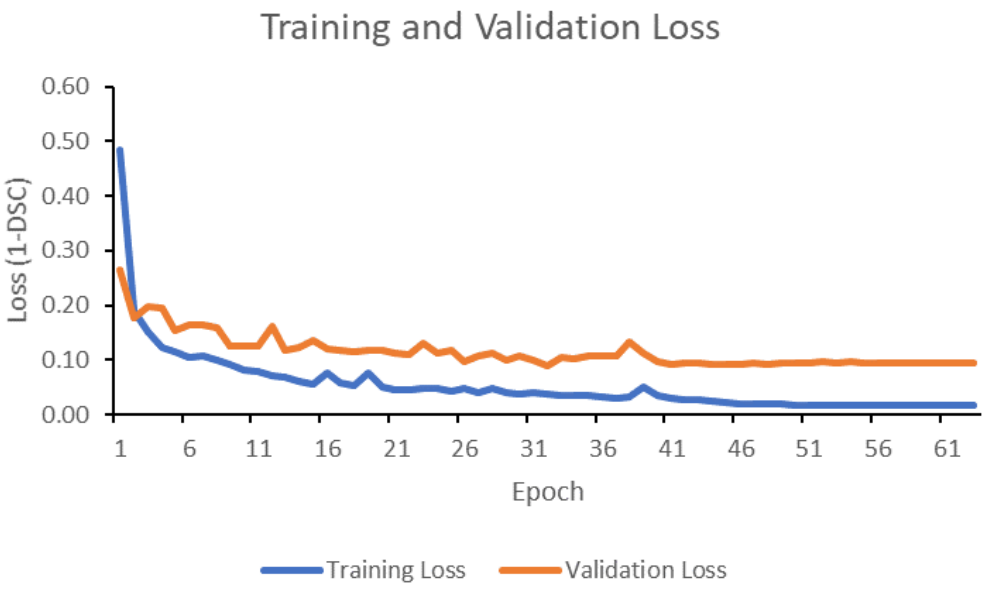
h)

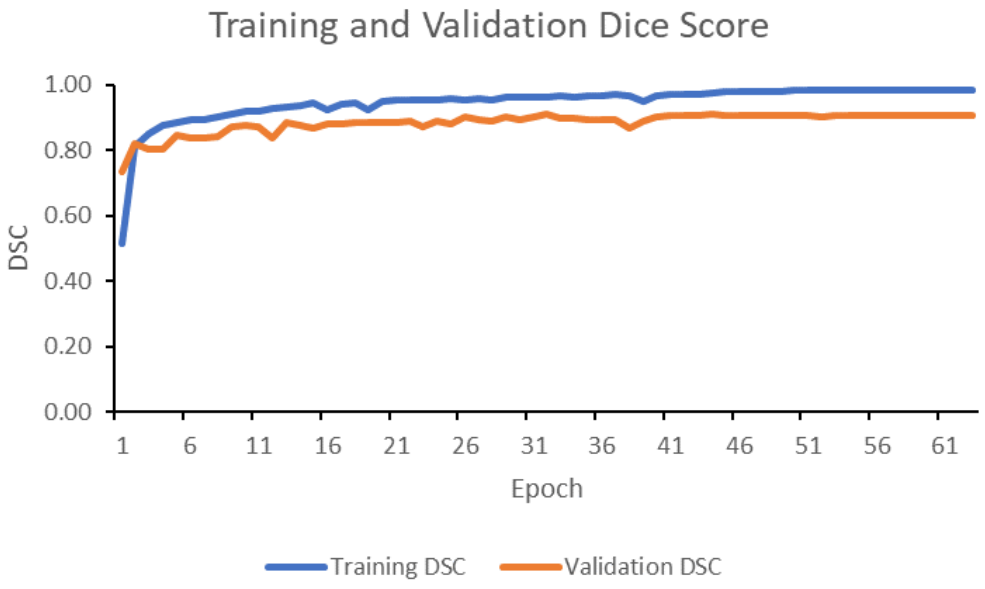
The Dice Score (DSC) loss described in `github.com/pytorch/pytorch/issues/1249` was used as the model's objective function.

$$DSC = \frac{2 \cdot | X \cap Y | + C}{| X | + | Y | + C}$$

where $C = 1$ is a smoothing constant. The corresponding DSC was used as a metric.

i)





j)

Average performance of the model on train, validation and test data is shown in table 1.

Table 1: Average Performance

Set	Train	Validation	Test
μ DSC	0.984	0.904	0.909

k)