

AML ASSIGNMENT - 1

Submitted by - Samarth Nigam(19307R002)
Sachin Doifode(193079033)
Varsha S (193079005)

1. MOTIVATION

The assignment provided a great learning experience to work on a specific deep learning problem. We chose a few basic hyperparameters which include learning rate, optimizers and loss functions.

The opportunity to apply the theoretical knowledge of various optimizers, learning rate schedulers, conv layers and conv transpose layers and see their effect on actual data practically was a big motivation.

We chose loss functions as a hyperparameter as they give a good understanding on how the model is training for the specific semantic segmentation problem. The first hand experience of working with pytorch and scikit library was very insightful.

2. EXPERIMENT STRATEGY

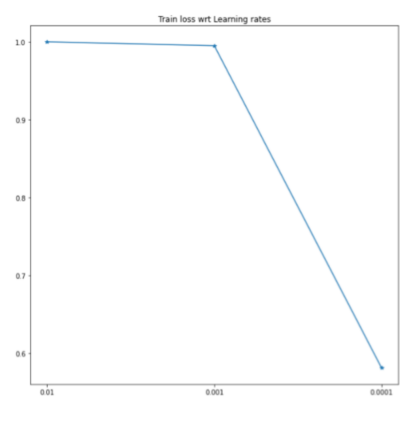
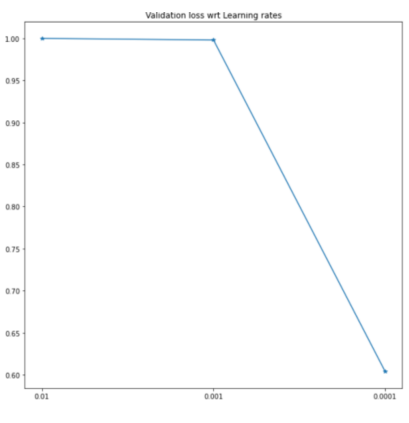
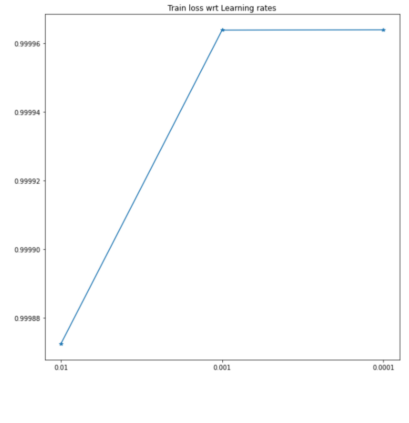
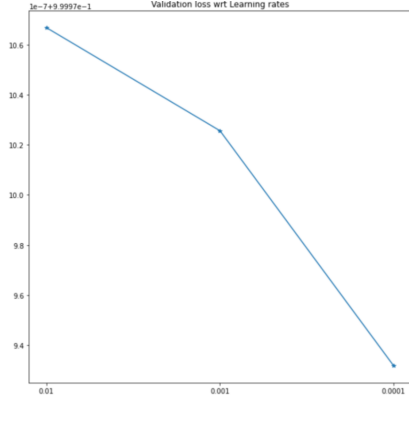
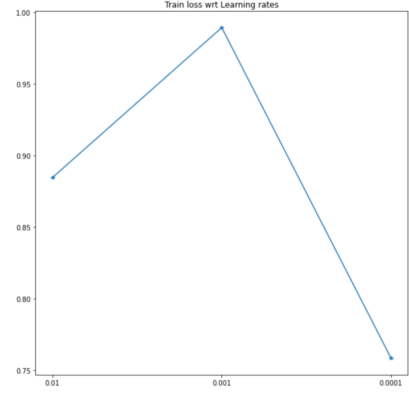
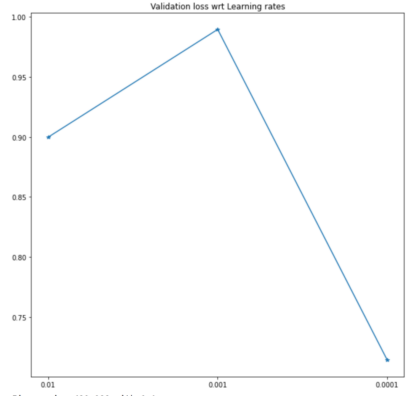
- Model was coded on Pytorch.
- The initial make was a simple BaseLine model for which we used the Unet model given in the paper - "Pixel-wise Segmentation of Right Ventricle of Heart"
- The division of work
 - Data preprocessing and contour maps generation
 - Developing the Model
 - Training and validation
 - Testing the model and interpretation of results
 - Performing hyperparameter tuning
 - Report preparation
- The hyperparameter tuning task was equally divided amongst the team members.

3. EXPERIMENTS CARRIED OUT

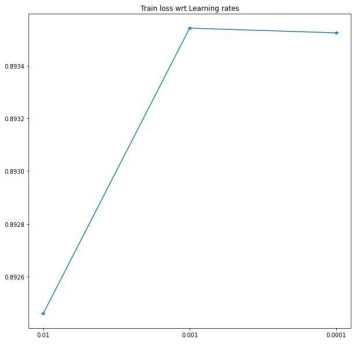
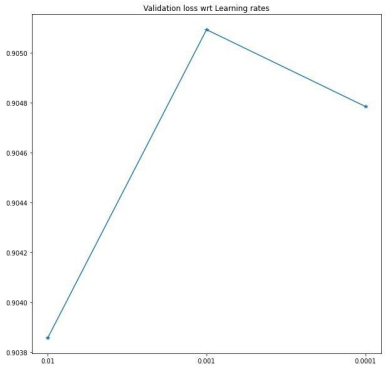
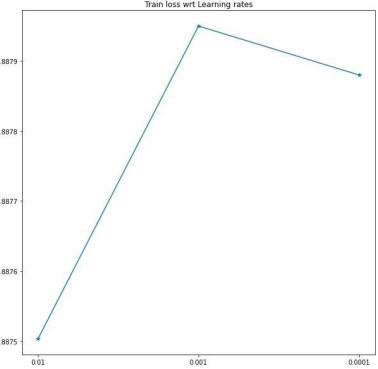
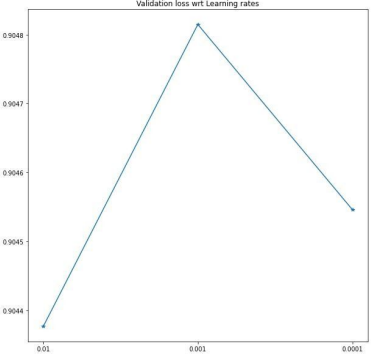
Hyperparameters tweaked are

1) Learning Rate (Optimizer and LR scheduler) - with dice loss

Optimiser	LR rate scheduler	%data used	Train loss wrt LR	Validation loss wrt LR																
Adam	Cosine Annealing	100%	<table border="1"><thead><tr><th>Learning Rate</th><th>Train Loss</th></tr></thead><tbody><tr><td>0.01</td><td>0.885</td></tr><tr><td>0.001</td><td>0.997</td></tr><tr><td>0.0001</td><td>0.994</td></tr></tbody></table>	Learning Rate	Train Loss	0.01	0.885	0.001	0.997	0.0001	0.994	<table border="1"><thead><tr><th>Learning Rate</th><th>Validation Loss</th></tr></thead><tbody><tr><td>0.01</td><td>0.901</td></tr><tr><td>0.001</td><td>0.997</td></tr><tr><td>0.0001</td><td>0.995</td></tr></tbody></table>	Learning Rate	Validation Loss	0.01	0.901	0.001	0.997	0.0001	0.995
Learning Rate	Train Loss																			
0.01	0.885																			
0.001	0.997																			
0.0001	0.994																			
Learning Rate	Validation Loss																			
0.01	0.901																			
0.001	0.997																			
0.0001	0.995																			
Adam	Cosine Annealing	80%	<table border="1"><thead><tr><th>Learning Rate</th><th>Train Loss</th></tr></thead><tbody><tr><td>0.01</td><td>1.00</td></tr><tr><td>0.001</td><td>0.702</td></tr><tr><td>0.0001</td><td>0.615</td></tr></tbody></table>	Learning Rate	Train Loss	0.01	1.00	0.001	0.702	0.0001	0.615	<table border="1"><thead><tr><th>Learning Rate</th><th>Validation Loss</th></tr></thead><tbody><tr><td>0.01</td><td>1.00</td></tr><tr><td>0.001</td><td>0.718</td></tr><tr><td>0.0001</td><td>0.632</td></tr></tbody></table>	Learning Rate	Validation Loss	0.01	1.00	0.001	0.718	0.0001	0.632
Learning Rate	Train Loss																			
0.01	1.00																			
0.001	0.702																			
0.0001	0.615																			
Learning Rate	Validation Loss																			
0.01	1.00																			
0.001	0.718																			
0.0001	0.632																			
Adam	Cosine Annealing	40%	<table border="1"><thead><tr><th>Learning Rate</th><th>Train Loss</th></tr></thead><tbody><tr><td>0.01</td><td>0.885</td></tr><tr><td>0.001</td><td>0.665</td></tr><tr><td>0.0001</td><td>0.712</td></tr></tbody></table>	Learning Rate	Train Loss	0.01	0.885	0.001	0.665	0.0001	0.712	<table border="1"><thead><tr><th>Learning Rate</th><th>Validation Loss</th></tr></thead><tbody><tr><td>0.01</td><td>0.901</td></tr><tr><td>0.001</td><td>0.668</td></tr><tr><td>0.0001</td><td>0.710</td></tr></tbody></table>	Learning Rate	Validation Loss	0.01	0.901	0.001	0.668	0.0001	0.710
Learning Rate	Train Loss																			
0.01	0.885																			
0.001	0.665																			
0.0001	0.712																			
Learning Rate	Validation Loss																			
0.01	0.901																			
0.001	0.668																			
0.0001	0.710																			

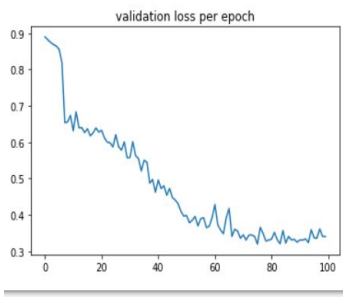
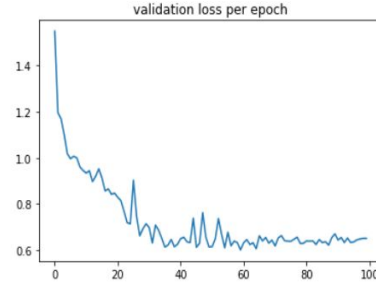
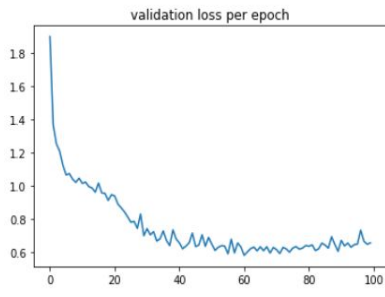
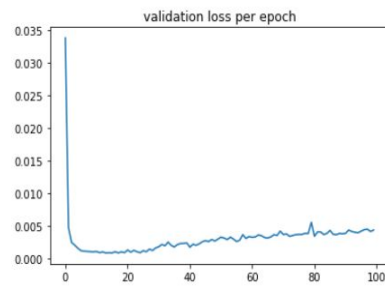
Adam	Step	100%	 <p>Train loss wrt Learning rates</p> <table><tr><th>Learning Rate</th><th>Train Loss</th></tr><tr><td>0.01</td><td>1.0</td></tr><tr><td>0.001</td><td>1.0</td></tr><tr><td>0.0001</td><td>0.58</td></tr></table>	Learning Rate	Train Loss	0.01	1.0	0.001	1.0	0.0001	0.58	 <p>Validation loss wrt Learning rates</p> <table><tr><th>Learning Rate</th><th>Validation Loss</th></tr><tr><td>0.01</td><td>1.0</td></tr><tr><td>0.001</td><td>1.0</td></tr><tr><td>0.0001</td><td>0.61</td></tr></table>	Learning Rate	Validation Loss	0.01	1.0	0.001	1.0	0.0001	0.61
Learning Rate	Train Loss																			
0.01	1.0																			
0.001	1.0																			
0.0001	0.58																			
Learning Rate	Validation Loss																			
0.01	1.0																			
0.001	1.0																			
0.0001	0.61																			
Adam	Step	80%	 <p>Train loss wrt Learning rates</p> <table><tr><th>Learning Rate</th><th>Train Loss</th></tr><tr><td>0.01</td><td>0.9988</td></tr><tr><td>0.001</td><td>0.9996</td></tr><tr><td>0.0001</td><td>0.9996</td></tr></table>	Learning Rate	Train Loss	0.01	0.9988	0.001	0.9996	0.0001	0.9996	 <p>Validation loss wrt Learning rates</p> <table><tr><th>Learning Rate</th><th>Validation Loss</th></tr><tr><td>0.01</td><td>30.6</td></tr><tr><td>0.001</td><td>30.25</td></tr><tr><td>0.0001</td><td>9.35</td></tr></table>	Learning Rate	Validation Loss	0.01	30.6	0.001	30.25	0.0001	9.35
Learning Rate	Train Loss																			
0.01	0.9988																			
0.001	0.9996																			
0.0001	0.9996																			
Learning Rate	Validation Loss																			
0.01	30.6																			
0.001	30.25																			
0.0001	9.35																			
Adam	Step	40%	 <p>Train loss wrt Learning rates</p> <table><tr><th>Learning Rate</th><th>Train Loss</th></tr><tr><td>0.01</td><td>0.88</td></tr><tr><td>0.001</td><td>0.99</td></tr><tr><td>0.0001</td><td>0.76</td></tr></table>	Learning Rate	Train Loss	0.01	0.88	0.001	0.99	0.0001	0.76	 <p>Validation loss wrt Learning rates</p> <table><tr><th>Learning Rate</th><th>Validation Loss</th></tr><tr><td>0.01</td><td>0.90</td></tr><tr><td>0.001</td><td>0.99</td></tr><tr><td>0.0001</td><td>0.71</td></tr></table>	Learning Rate	Validation Loss	0.01	0.90	0.001	0.99	0.0001	0.71
Learning Rate	Train Loss																			
0.01	0.88																			
0.001	0.99																			
0.0001	0.76																			
Learning Rate	Validation Loss																			
0.01	0.90																			
0.001	0.99																			
0.0001	0.71																			

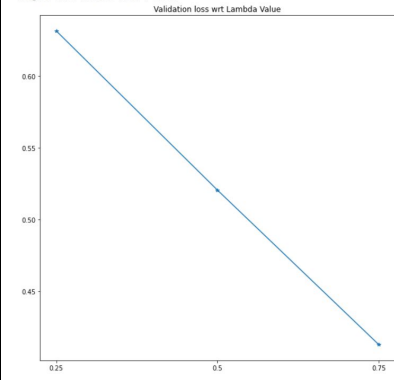
SGD	Cosine Annealing	100%	<p>Train loss wrt Learning rates</p> <table border="1"><thead><tr><th>Learning Rate</th><th>Train Loss</th></tr></thead><tbody><tr><td>0.01</td><td>0.884</td></tr><tr><td>0.001</td><td>0.890</td></tr><tr><td>0.0001</td><td>0.889</td></tr></tbody></table>	Learning Rate	Train Loss	0.01	0.884	0.001	0.890	0.0001	0.889	<p>Validation loss wrt Learning rates</p> <table border="1"><thead><tr><th>Learning Rate</th><th>Validation Loss</th></tr></thead><tbody><tr><td>0.01</td><td>0.889</td></tr><tr><td>0.001</td><td>0.895</td></tr><tr><td>0.0001</td><td>0.894</td></tr></tbody></table>	Learning Rate	Validation Loss	0.01	0.889	0.001	0.895	0.0001	0.894
Learning Rate	Train Loss																			
0.01	0.884																			
0.001	0.890																			
0.0001	0.889																			
Learning Rate	Validation Loss																			
0.01	0.889																			
0.001	0.895																			
0.0001	0.894																			
SGD	Cosine Annealing	70%	<p>Train loss wrt Learning rates</p> <table border="1"><thead><tr><th>Learning Rate</th><th>Train Loss</th></tr></thead><tbody><tr><td>0.01</td><td>0.8910</td></tr><tr><td>0.001</td><td>0.8945</td></tr><tr><td>0.0001</td><td>0.8940</td></tr></tbody></table>	Learning Rate	Train Loss	0.01	0.8910	0.001	0.8945	0.0001	0.8940	<p>Validation loss wrt Learning rates</p> <table border="1"><thead><tr><th>Learning Rate</th><th>Validation Loss</th></tr></thead><tbody><tr><td>0.01</td><td>0.8910</td></tr><tr><td>0.001</td><td>0.8950</td></tr><tr><td>0.0001</td><td>0.8940</td></tr></tbody></table>	Learning Rate	Validation Loss	0.01	0.8910	0.001	0.8950	0.0001	0.8940
Learning Rate	Train Loss																			
0.01	0.8910																			
0.001	0.8945																			
0.0001	0.8940																			
Learning Rate	Validation Loss																			
0.01	0.8910																			
0.001	0.8950																			
0.0001	0.8940																			
SGD	Cosine Annealing	40%	<p>Train loss wrt Learning rates</p> <table border="1"><thead><tr><th>Learning Rate</th><th>Train Loss</th></tr></thead><tbody><tr><td>0.01</td><td>0.8845</td></tr><tr><td>0.001</td><td>0.8870</td></tr><tr><td>0.0001</td><td>0.8845</td></tr></tbody></table>	Learning Rate	Train Loss	0.01	0.8845	0.001	0.8870	0.0001	0.8845	<p>Validation loss wrt Learning rates</p> <table border="1"><thead><tr><th>Learning Rate</th><th>Validation Loss</th></tr></thead><tbody><tr><td>0.01</td><td>0.89300</td></tr><tr><td>0.001</td><td>0.89475</td></tr><tr><td>0.0001</td><td>0.89450</td></tr></tbody></table>	Learning Rate	Validation Loss	0.01	0.89300	0.001	0.89475	0.0001	0.89450
Learning Rate	Train Loss																			
0.01	0.8845																			
0.001	0.8870																			
0.0001	0.8845																			
Learning Rate	Validation Loss																			
0.01	0.89300																			
0.001	0.89475																			
0.0001	0.89450																			
SGD	Step	100%	<p>Train loss wrt Learning rates</p> <table border="1"><thead><tr><th>Learning Rate</th><th>Train Loss</th></tr></thead><tbody><tr><td>0.01</td><td>0.8894</td></tr><tr><td>0.001</td><td>0.8902</td></tr><tr><td>0.0001</td><td>0.8896</td></tr></tbody></table>	Learning Rate	Train Loss	0.01	0.8894	0.001	0.8902	0.0001	0.8896	<p>Validation loss wrt Learning rates</p> <table border="1"><thead><tr><th>Learning Rate</th><th>Validation Loss</th></tr></thead><tbody><tr><td>0.01</td><td>0.8938</td></tr><tr><td>0.001</td><td>0.8948</td></tr><tr><td>0.0001</td><td>0.8941</td></tr></tbody></table>	Learning Rate	Validation Loss	0.01	0.8938	0.001	0.8948	0.0001	0.8941
Learning Rate	Train Loss																			
0.01	0.8894																			
0.001	0.8902																			
0.0001	0.8896																			
Learning Rate	Validation Loss																			
0.01	0.8938																			
0.001	0.8948																			
0.0001	0.8941																			

SGD	Step	70%	 <table><caption>Train loss wrt Learning rates (70% Step)</caption><thead><tr><th>Learning Rate</th><th>Train Loss</th></tr></thead><tbody><tr><td>0.01</td><td>0.8925</td></tr><tr><td>0.001</td><td>0.8934</td></tr><tr><td>0.0001</td><td>0.8933</td></tr></tbody></table>	Learning Rate	Train Loss	0.01	0.8925	0.001	0.8934	0.0001	0.8933	 <table><caption>Validation loss wrt Learning rates (70% Step)</caption><thead><tr><th>Learning Rate</th><th>Validation Loss</th></tr></thead><tbody><tr><td>0.01</td><td>0.9038</td></tr><tr><td>0.001</td><td>0.9050</td></tr><tr><td>0.0001</td><td>0.9048</td></tr></tbody></table>	Learning Rate	Validation Loss	0.01	0.9038	0.001	0.9050	0.0001	0.9048
Learning Rate	Train Loss																			
0.01	0.8925																			
0.001	0.8934																			
0.0001	0.8933																			
Learning Rate	Validation Loss																			
0.01	0.9038																			
0.001	0.9050																			
0.0001	0.9048																			
SGD	Step	40%	 <table><caption>Train loss wrt Learning rates (40% Step)</caption><thead><tr><th>Learning Rate</th><th>Train Loss</th></tr></thead><tbody><tr><td>0.01</td><td>0.8875</td></tr><tr><td>0.001</td><td>0.8879</td></tr><tr><td>0.0001</td><td>0.8878</td></tr></tbody></table>	Learning Rate	Train Loss	0.01	0.8875	0.001	0.8879	0.0001	0.8878	 <table><caption>Validation loss wrt Learning rates (40% Step)</caption><thead><tr><th>Learning Rate</th><th>Validation Loss</th></tr></thead><tbody><tr><td>0.01</td><td>0.9044</td></tr><tr><td>0.001</td><td>0.9048</td></tr><tr><td>0.0001</td><td>0.9045</td></tr></tbody></table>	Learning Rate	Validation Loss	0.01	0.9044	0.001	0.9048	0.0001	0.9045
Learning Rate	Train Loss																			
0.01	0.8875																			
0.001	0.8879																			
0.0001	0.8878																			
Learning Rate	Validation Loss																			
0.01	0.9044																			
0.001	0.9048																			
0.0001	0.9045																			

2) Loss functions

- We are using the four loss functions provided in the reference paper with hyperparameter tuning in switching loss for the hyperparameter lambda.
- In the final model notebook, all the used losses have been mentioned.
- All the different losses were tested for 100 training epochs with adam optimizer with $lr = 0.0005$.
- Discussed below is brief information about the training and testing observations for the loss functions used
 - Simple Dice Loss
 - Dice Loss combined with Binary Cross Entropy Loss
 - Dice Loss combined with inverse Dice Loss and Cross Entropy Loss
 - Switching loss which is an optimised combination of Dice Loss, Inverse Dice Loss and Cross Entropy Loss

S. No.	Loss Used	Tuned Loss Hyperparameter	Test Dice Score	Figures
1.	Simple Dice Loss	NA	0.2761	 <p>The graph shows validation loss per epoch on the y-axis (ranging from 0.3 to 0.9) against epoch number on the x-axis (ranging from 0 to 100). The loss starts at approximately 0.85 at epoch 0 and decreases steadily with some fluctuations, reaching about 0.35 by epoch 100.</p>
2.	Dice Loss + BCE	NA	0.2668	 <p>The graph shows validation loss per epoch on the y-axis (ranging from 0.6 to 1.4) against epoch number on the x-axis (ranging from 0 to 100). The loss starts at approximately 1.4 at epoch 0 and decreases with significant fluctuations, stabilizing around 0.65 by epoch 100.</p>
3.	Dice Loss + BCE + Inverse Dice Loss	NA	0.2527	 <p>The graph shows validation loss per epoch on the y-axis (ranging from 0.6 to 1.8) against epoch number on the x-axis (ranging from 0 to 100). The loss starts at approximately 1.8 at epoch 0 and decreases with fluctuations, reaching about 0.65 by epoch 100.</p>
4.	Focal Loss	Alpha = 0.25, Gamma=2, (standard values used)	0.3762	 <p>The graph shows validation loss per epoch on the y-axis (ranging from 0.000 to 0.035) against epoch number on the x-axis (ranging from 0 to 100). The loss starts at approximately 0.035 at epoch 0 and drops sharply to near 0.005 by epoch 10, remaining stable thereafter.</p>

5.	Switching Loss	Lambda = 0.25	0.2426	
		Lambda = 0.5		
		Lambda = 0.75		

3) Data Pre-processing approaches

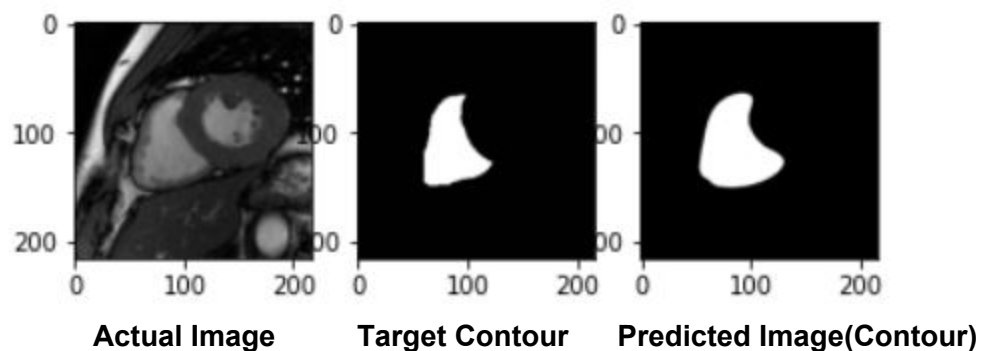
- The dcim images were converted to jpeg images.
- The contours were generated with three different backgrounds - white, grey and black.
- The images were resized to a size of (216,216).
- The Model was tested for both RGB and grayscale images.

4. RESULTS AND PLOTS

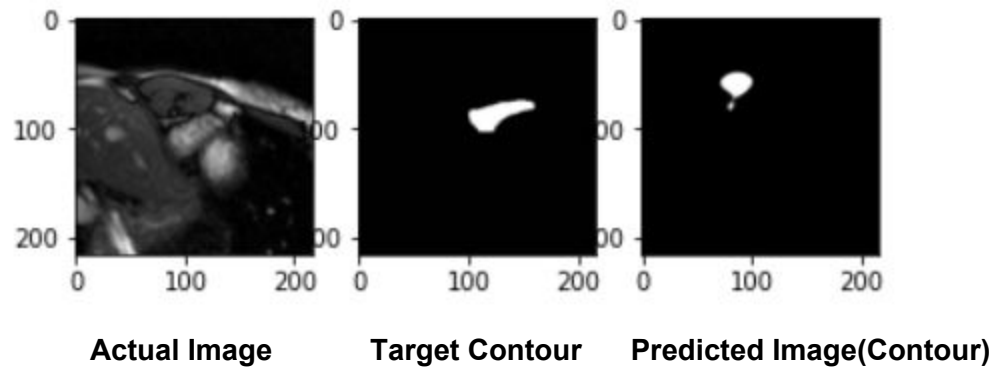
Our baseline model is based on **simple Unet** as shown in the paper. We are training for 100 epochs at a batch size of 16. We are using Dice Loss and Adam Optimizer with a learning rate of 0.0005. For the testing part we are using dice loss to calculate test loss.

- Results for baseline model

Good Result Obtained

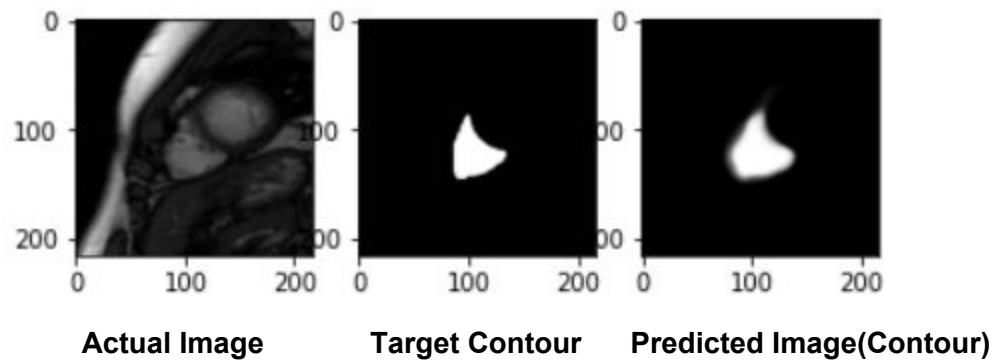


Bad Result Obtained

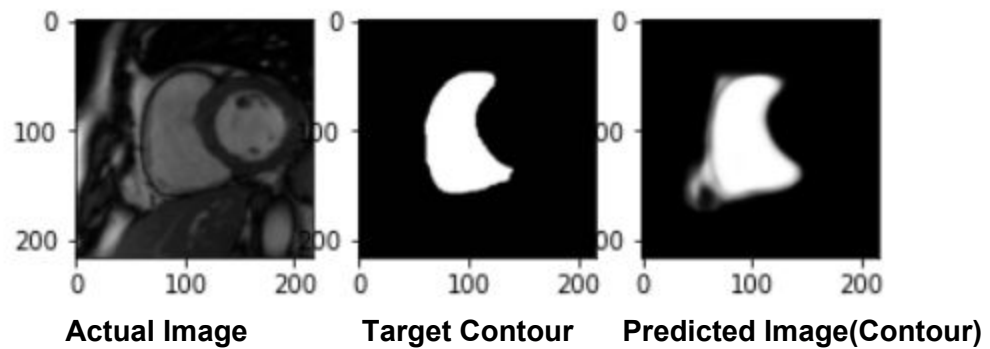


- Results for baseline model with Switching loss in place of dice loss.

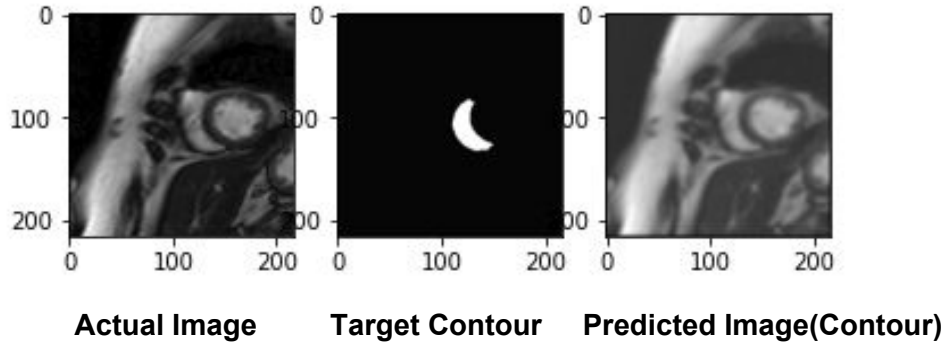
Good Result Obtained



Bad Result Obtained



- Results for baseline model with SGD in place of Adam optimizer



Above we can see that the SGD optimizer is not able to predict the proper output with 90% loss. Means only around 10-11% accuracy. Other ADAM is giving around 70 - 75% accuracy.

- Hyperparameter tuning results are put in the table under experiments performed.

5. CONCLUSION

- Adam with CosineAnnealing rate scheduler performs better with $lr=0.001$ at 100% and 40% train data usage, while $lr = 0.0001$ performs better when 80% of train data is used.
- Adam with the Step rate scheduler performs better with $lr=0.0001$ at 100% and 40% train data usage, while $lr = 0.001$ performs better when 80% of train data is used.
- The model was tested for 3 types of contour maps, one with just the boundary, one with grey backgrounds and finally with black backgrounds. The results gave an intuitive way to think how actually our model is working in terms of pixel values of output we get. We got the best results with black background masks. This lies in correspondence with the fact that black masks provide a zero value for background which leads to better losses.
- The model was tested for both grayscale and RGB Images. Grayscale approach worked better for us. The reason may lie in the fact that our target masks take either 1 value or 0 in black background case. Thus training the model would be easier if we train it for grayscale images.
- The best loss function among all loss functions used was switching loss with $\lambda = 0.75$ as well as threshold factor 0.5. This lies in agreement with the observation reported in the paper. Also the order of behaviour of different loss functions on the basis of avg test dice loss is similar to that reported in the paper. Switching loss performs better than others due to few reasons,

- First, It is a combined loss of Dice and Inverse Dice which ensures that both the foreground and background are well classified.
- Second, we can change the emphasis between foreground and background based on the target mask. With increased lambda, it can be said that the more frequent occurring value in the target was classified better.
- Several recent works have aimed to explain why severely overparameterized models generalize well when trained by Stochastic Gradient Descent (SGD). The emergent consensus explanation has two parts: the first is that there are "no bad local minima", while the second is that SGD performs implicit regularization by having a bias towards low complexity models. In the context of image classification with common deep neural network architectures, there exist bad global minima, i.e., models that fit the training set perfectly, yet have poor generalization. And, given only unlabeled training data, we can easily construct initializations that will cause SGD to quickly converge to such bad global minima. From the results we show that SGD performs worse on the complex model and complex image segmentation or object detection problem.
- Even though we used CosineAnnealing learning rate scheduler or Step learning rate scheduler for training the given U-net model, we're not getting much better results. As SGD converges very slowly and sometimes stuck at bad global minima and not able to converge to true minima. We've trained for 50 epochs just to show results and can be improved if the epochs are increased.

6. IF WE HAD MORE TIME WHAT WOULD WE DO

- We would have checked if the model trained on segmented natural images can predict well on medical images.
- We would have tried various backbones for the Unet.
- We would have checked the performance of RGB inputs too.
- We would have studied the effect of pretrained weights obtained by Transfer learning.
- We would have used even more values for various hyperparameter tuning.
- We would have trained the models for more number of epochs.
- We could have planned and come up with more intuitive ways to do the hyperparameter tuning. For example - we could have compared our best results and would have combined them to get a better loss value.

7. NOTEBOOKS INCLUDED IN SUBMISSION

- Final_BaseModel_Train.ipynb - The training script for baseline model.
- Image_to_Contour_Mapping.ipynb - To map the training images with the contour files. Since there were some contour files missing for some of the training images.

- Baseline_losses_train.ipynb - The training script including the code segments for all loss functions used. This Script currently shows outputs over switching loss.
- Final_BaseModel_Test.ipynb - The test script for our model. The script currently shows output related to the BaseLine model.
- Optimizer_LearningRateSchedulers_Train.ipynb - The notebook used to do hyperparameter tuning.