UNets for Semantic Segmentation

17 November 2020 18:40

Anmol Biswas (194076019)

Vineet Kotariya (16D110009)

Problem description : Provided images of the heart, it is required to segment the right ventricle. The input data is provided in the form of dcm images and their corresponding ground truth segmentation contours are provided in the form of text files containing the list of contour points. There is a great variability in the shapes and sizes of the contours and there is very little annotated data available (comparatively speaking).

Choice of experiments : Given the problem statement, we chose to concentrate on Loss function and Regularization as the hyperparameters which could impact the results. We also played around with a few types of data augmentation. Regularization was expected choice because of the lack of sufficient labeled data. Loss function was the second choice as it can strongly affect training and has to try and has to try to account for the wide variety in the shapes and sizes of the target

Experimental Setup : The first step is a pre-processing of the given ground truth contours into binary images. OpenCV image processing (fillPoly) is used to convert the given contour points into a polygon and then fill the area of the polygon with 1s while the background is 0s. This effectively generates our ground truth images. For simplicity, the corresponding files for input image and ground truth images (we generated i and o counters separately) are kept in different folders with the same name. This makes loading the dataset simply a matter of reading those files into memory in order.

The network architecture we are using is a standard U-Net with skip connections. However, transposed convolutions are replaced by nearest neighbour upsampling, followed by a concatenation of the skip connection and a convolution layer. The basic network architecture is defined as follows :

Input -> conv2d1 (5x5 16 filters)-> max_pool -> conv2d2 (5x5 32 filters)-> max_pool -> conv2d3 (5x5 64 filters)-> max_pool -> conv2d4 (5x5 128 filters) -> upsample -> concat(conv2d3) -> conv2d5(3x3 32 filters) -> upsample -> concat(conv2d2) -> conv2d6(3x3 16 filters) -> upsample -> concat(conv2d1) -> conv2d7(3x3 2 filters) -> Output

Data augmentation is performed on the input image pairs to increase the number of training examples. Two main types of operations are performed :

1. Horizontal and Vertical flips
2. Image rotations through small angles (-15 degrees to 15 degrees)

Among the Loss experiments, Exp 1-5 are performed with the Horizontal and Vertical flips only while Exp 6 onwards are performed with Image rotations only. In the image rotation augmentations, the images with small area of ground truth are over sampled

Testing : We have tested the setups in two different ways. One is by using the entire training set to train and using samples in Test1Set to test the model. A second approach is by splitting the Training set into Training and Validation sets in a roughly 3:1 ratio. In the Loss Experiments, Exp 1-6 are validated against the Test1Set and Exp 7 and 8 are validated against unseen images taken from Training set itself.

Validation details :

1. Main comparison metric = IOU (Intersection over Union)
2. output image is thresholded by 0.5 to generate binary maps - which are used to calculate the IOU with the ground truth

Two sets of experiments are performed independently on the network configuration. Loss Experiments and Regularization Experiments. We report the outcomes of the said experiments next and then end it by making final conclusions

Loss experiments :

Losses :

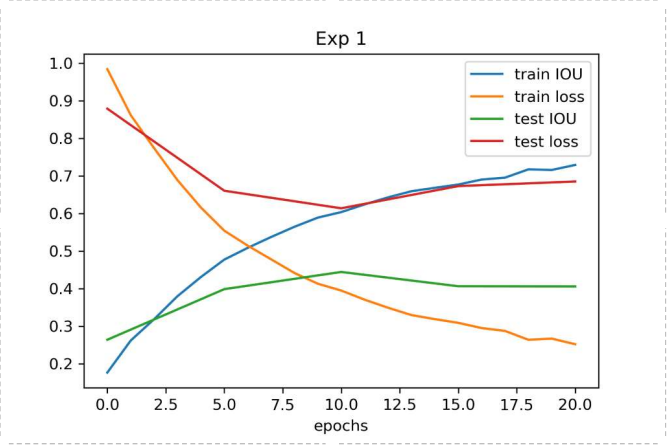
1. Binary cross entropy Loss (L_ce)
2. Dice Loss (L_dice)
3. Inverse Dice Loss (L_inv_dice)

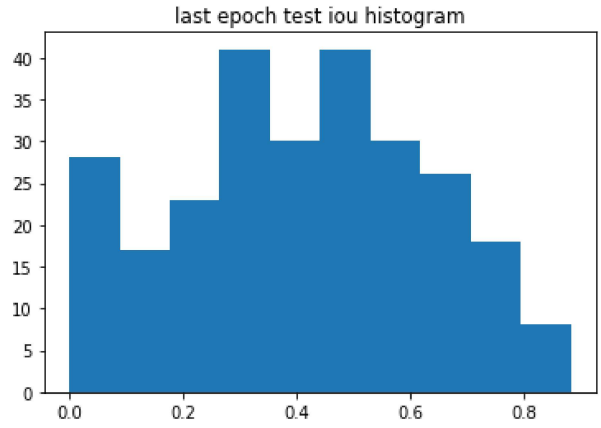
Combination loss = a_ce*L_ce + a_dice*L_dice + a_inv_dice*L_inv_dice

Switching Loss = a_ce*L_ce + a_dice*L_dice + (1-a_dice)*L_inv_dice (where a_dice switches to give different weightage to dice loss vs inverse dice loss based on the area covered by the ground truth)

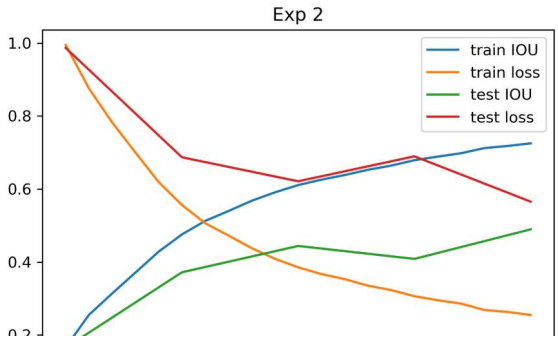
We perform a set of experiments to figure out the optimal loss to train the network. The experiment takes the form of different combinations of the weights of the loss functions. The main plot of comparison is the "**last epoch test/val/ iou histogram**" which plots the histogram distribution of the IOU values and the plots of "**IOU vs ground truth size**"

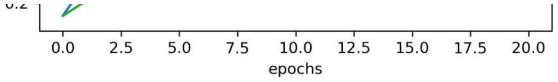
Exp1 : (a_ce, a_dice, a_inv_dice) = (1, 1, 1)

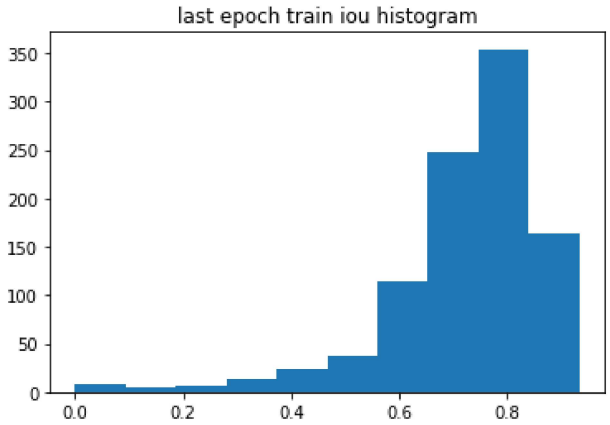


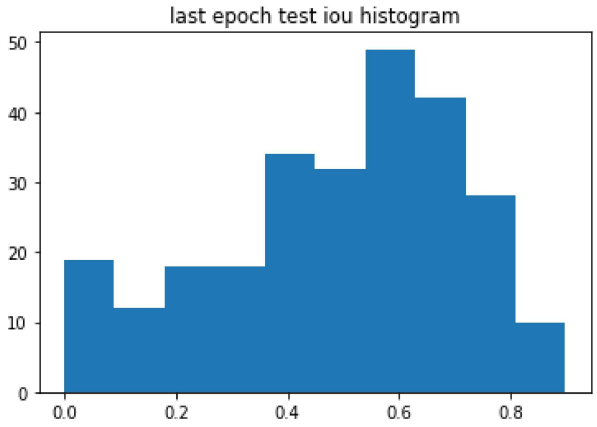


Exp2 : (a_ce, a_dice, a_inv_dice) = (1, 0.5, 0.5)

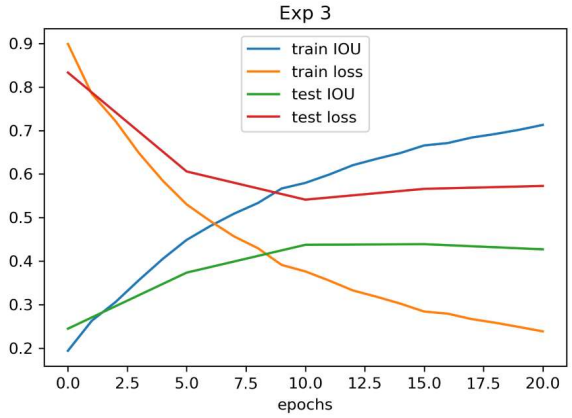


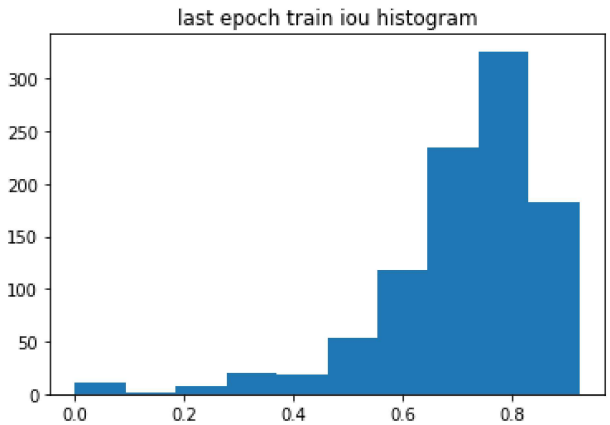


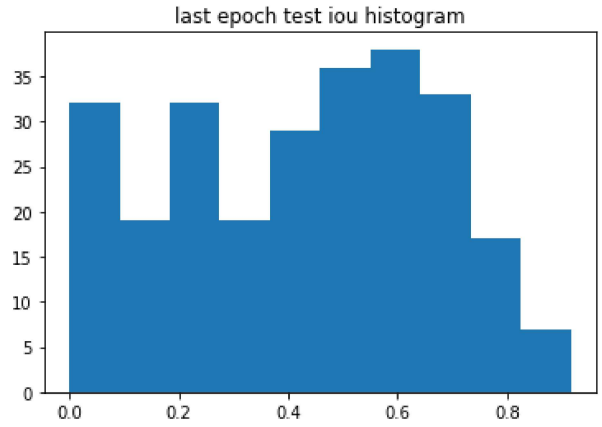




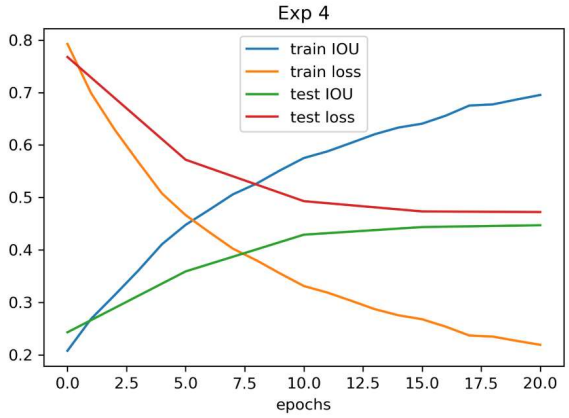
Exp3 : (a_ce, a_dice, a_inv_dice) = (0.5, 1, 1)

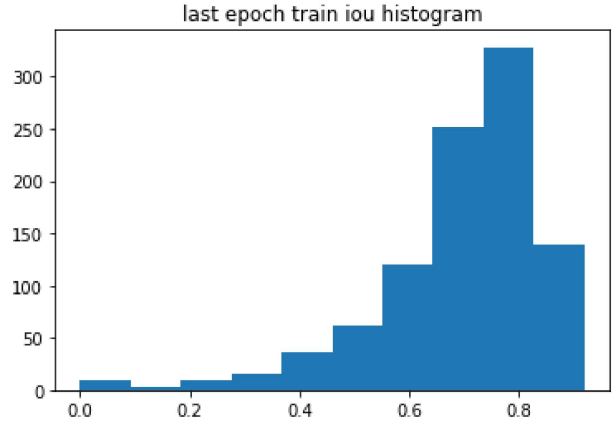


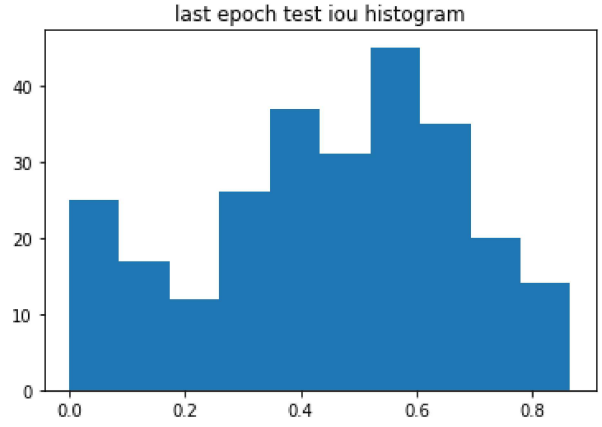




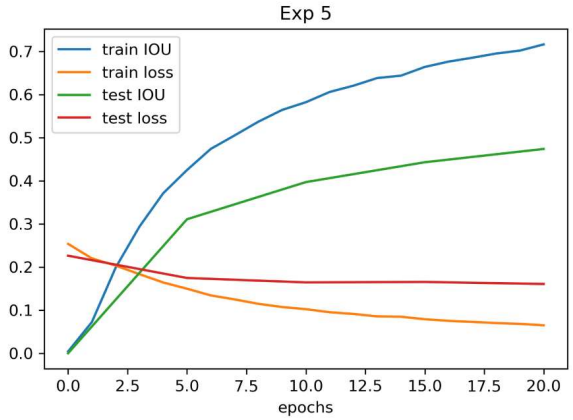
Exp4 : (a_ce, a_dice, a_inv_dice) = (0.1, 1, 1)

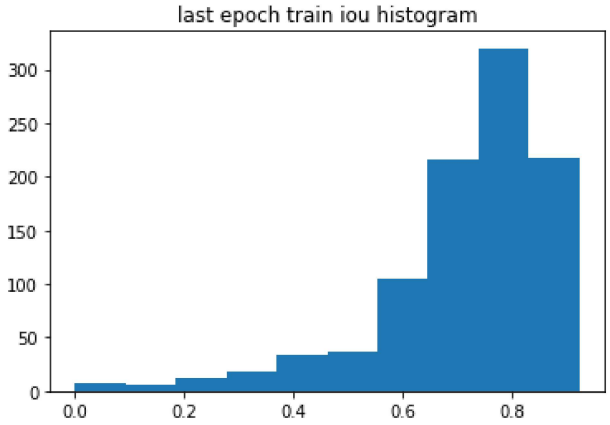


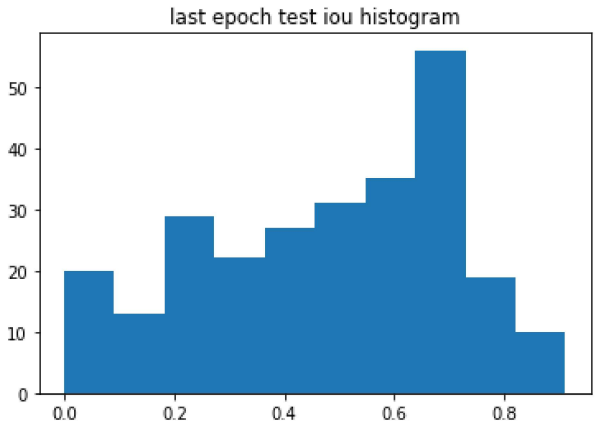




Exp5 : (a_ce, a_dice, a_inv_dice) = (1, 0.1, 0.1)

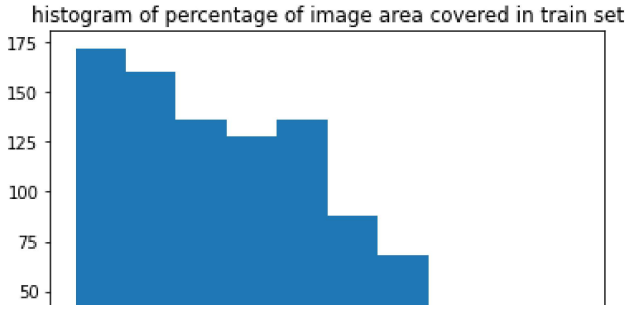


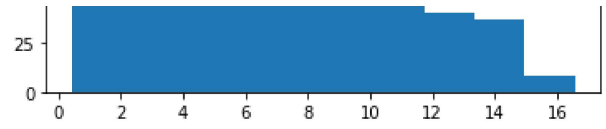


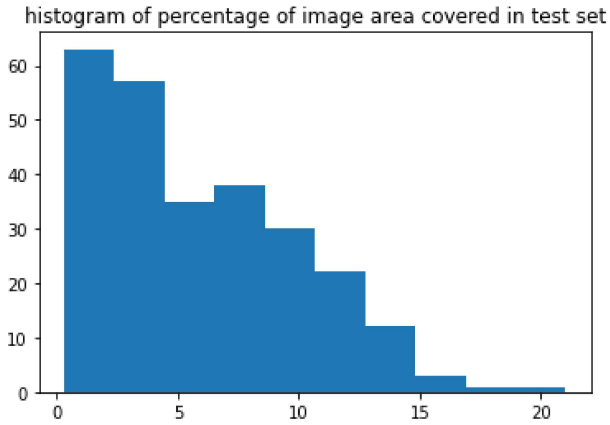


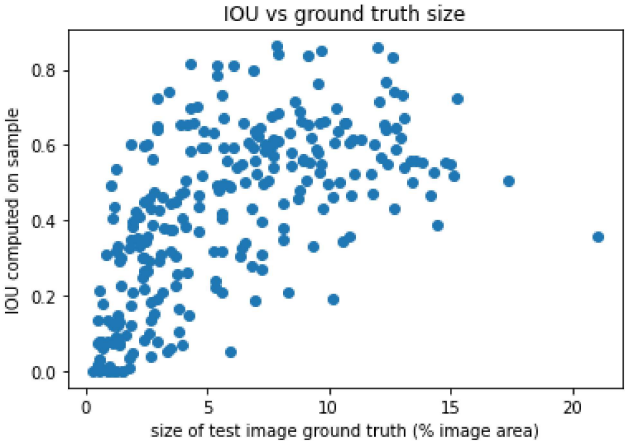
A common observation with all of the above experiments is that the training doesn't seem to generalize well on the test set

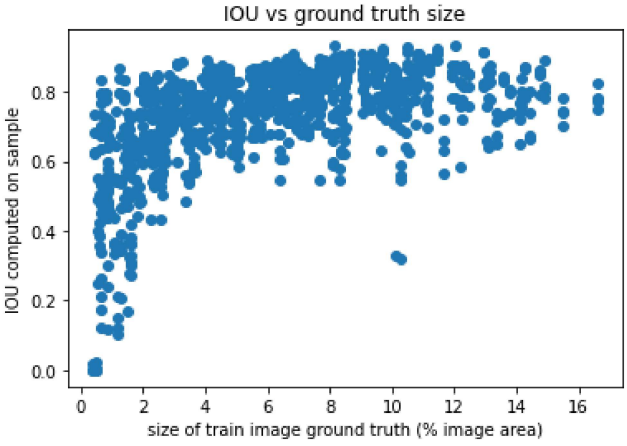
Comparison of ground truth size statistics in train set vs test set





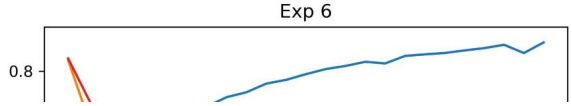


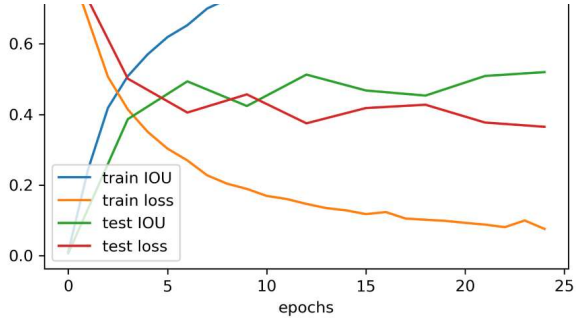


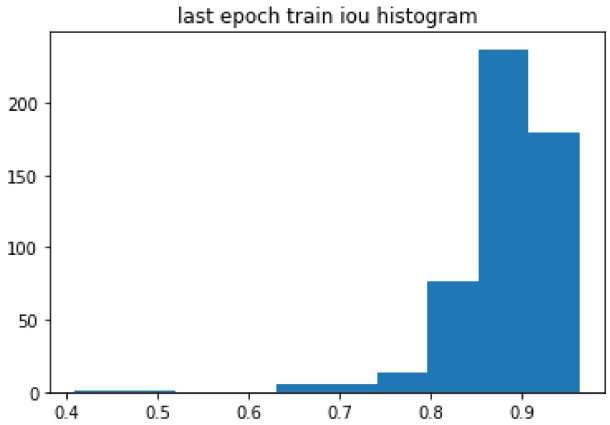


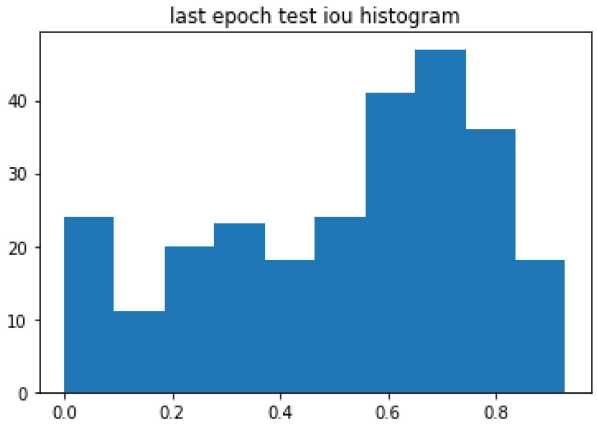
It is clear that there is a significant difference in the area statistics of train and test sets and extremely small ground truth images are more frequent in the test set. It can also be seen that the performance of the network suffers especially when there is a small ground truth. This is quite clearly evident in the plot of IOU vs ground truth for both training and test sets and especially true for the test set

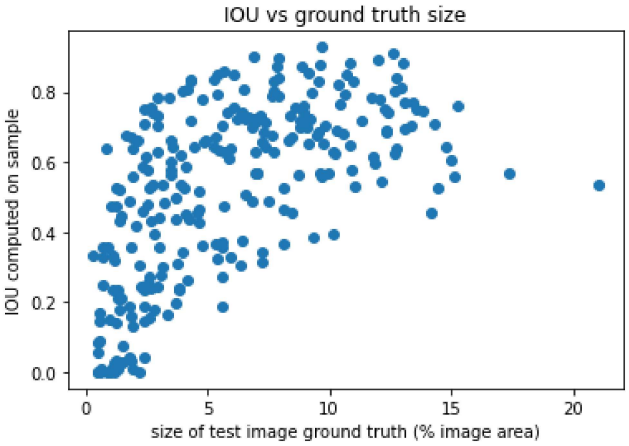
Exp 6 : (a_ce, a_dice, a_inv_dice) = (1, 0.25, 0.75) + new data augmentation using only rotations + emphasizing small area samples

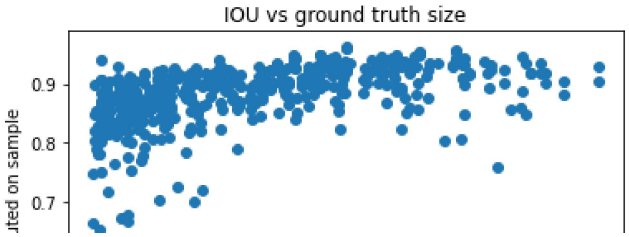


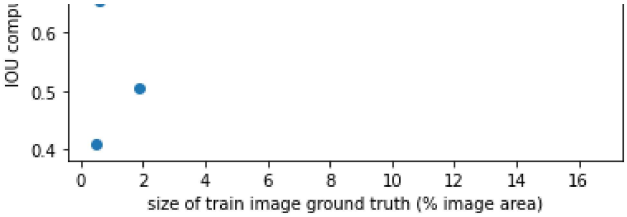

train IOU
train loss
test IOU
test loss
epochs


last epoch train iou histogram

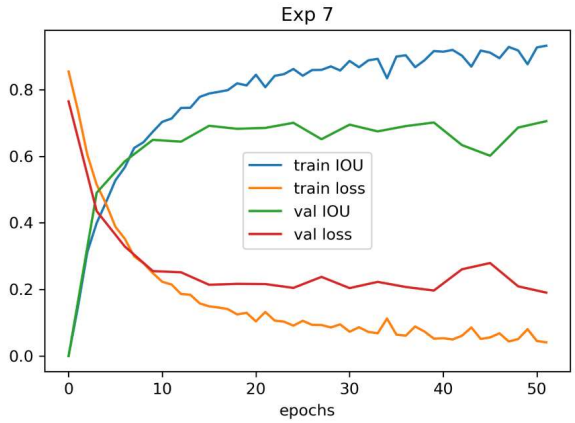

last epoch test iou histogram

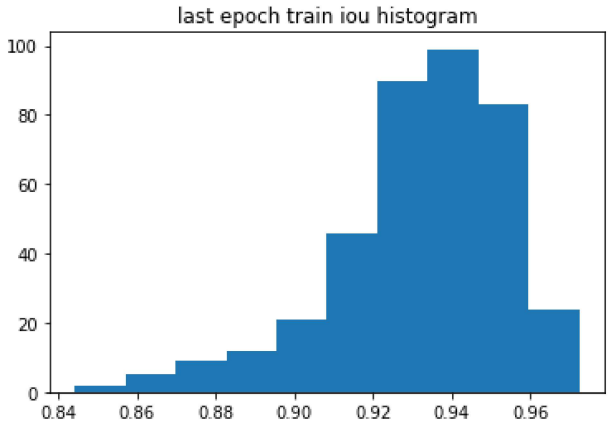

IOU vs ground truth size
IOU computed on sample
size of test image ground truth (% image area)

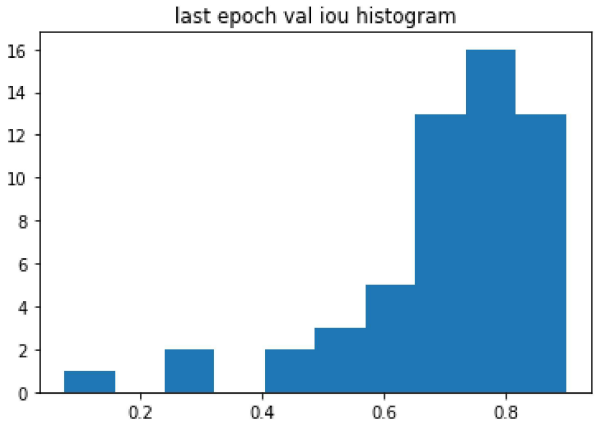

IOU vs ground truth size

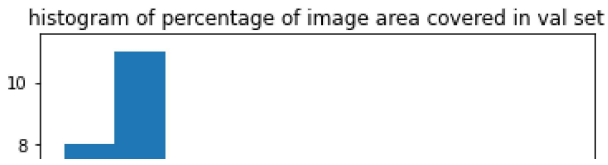


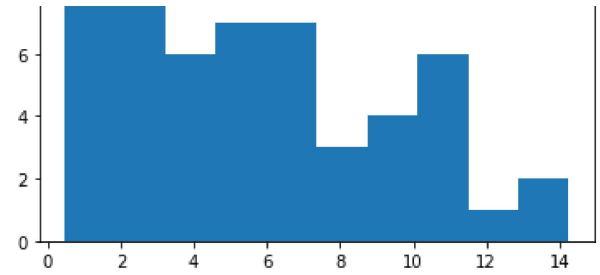
Exp 7 : Repeat with more epochs (a_ce, a_dice, a_inv_dice) = (1, 0.25, 0.75) + new data augmentation using only rotations + emphasizing small area samples, **but validating on a validation set taken out of the training set itself**. Original Training Data split in 3:1 ratio to generate train set and validation set

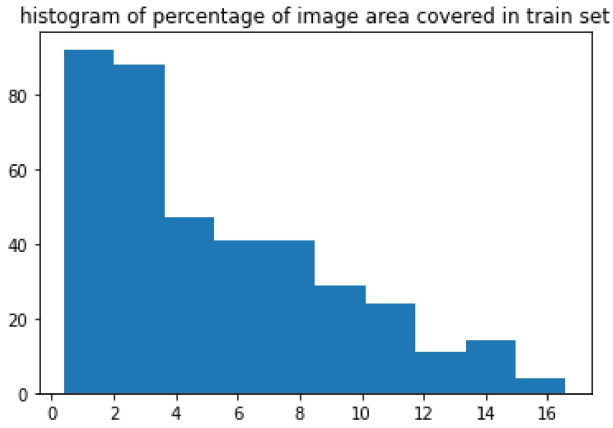


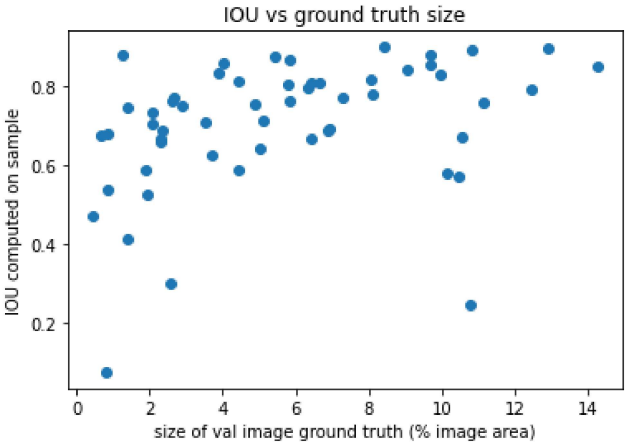


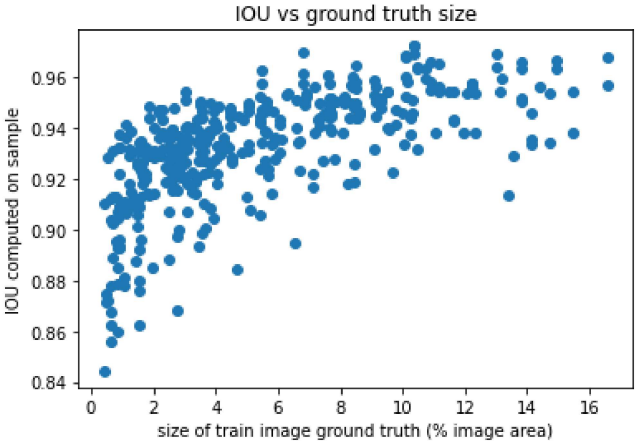




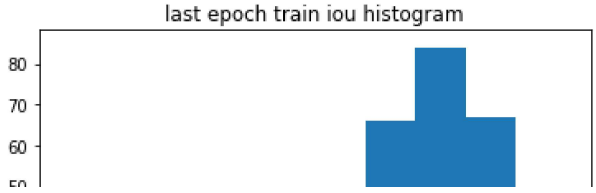


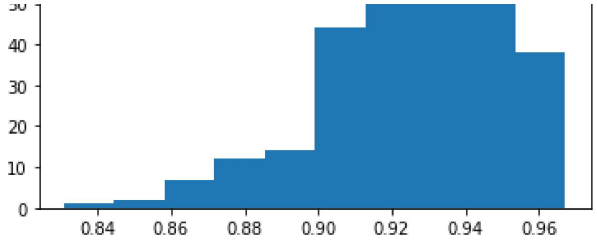


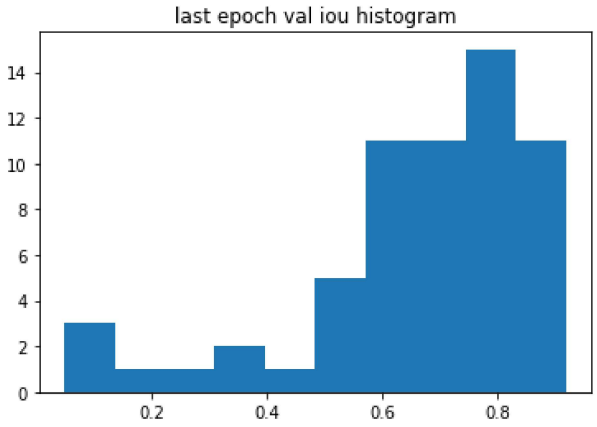


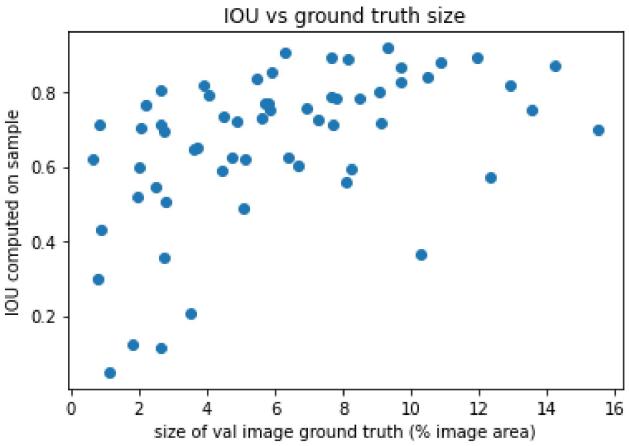


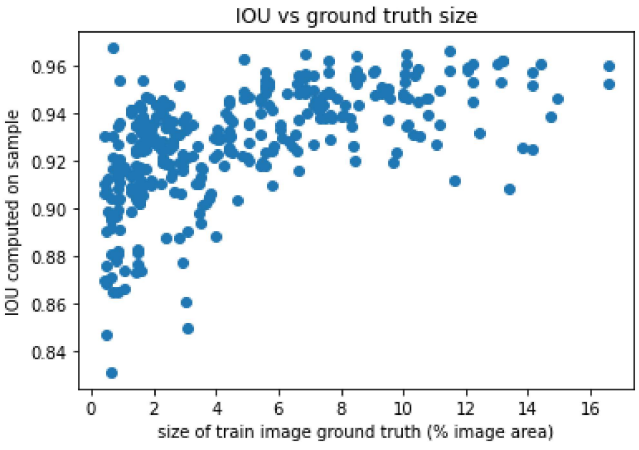
Exp 8 : Switching loss (validating on validation set drawn from train set)











Regularization Experiments :

For 100% training data:

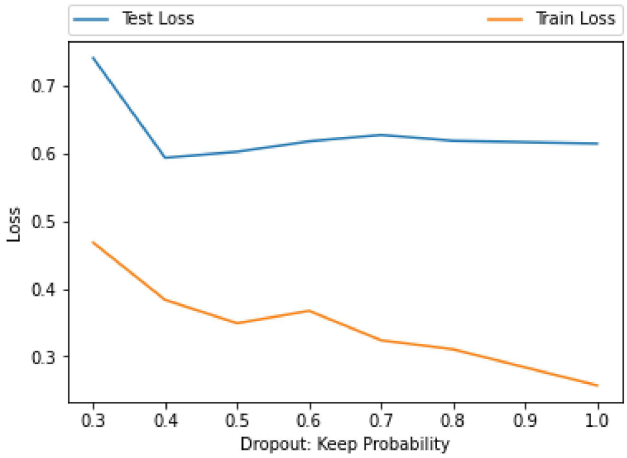
DROPOUT:

Dropout after conv layers 3 and 4 having equal 'keep probabilities'

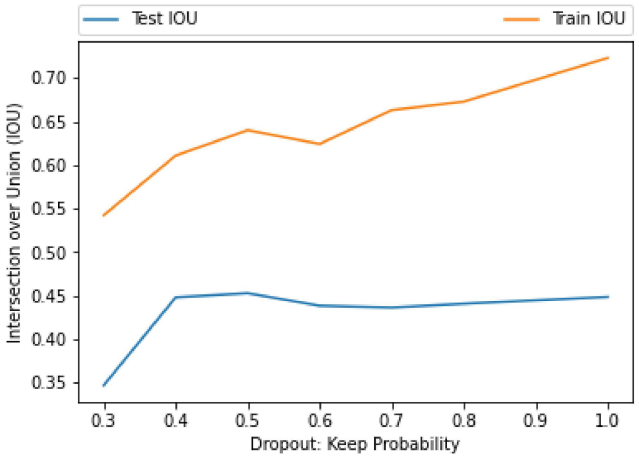
Loss used: Comb Loss (1,1,1)

No regularizer

Plots:

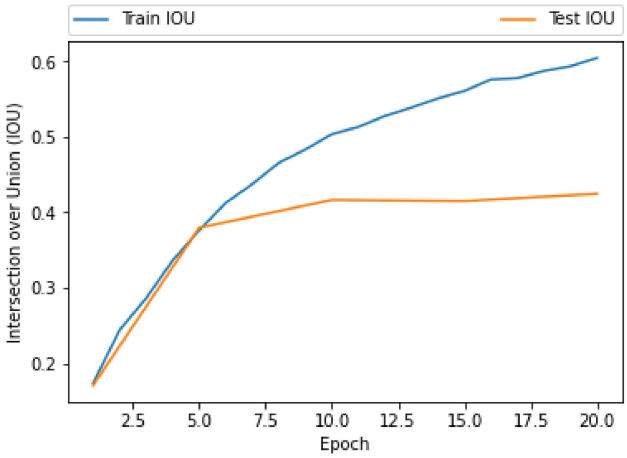


When keep Prob is too low, the losses are high. Test loss has a peak at p = 0.7, Train loss keeps on decreasing till the no-dropout case of p = 1

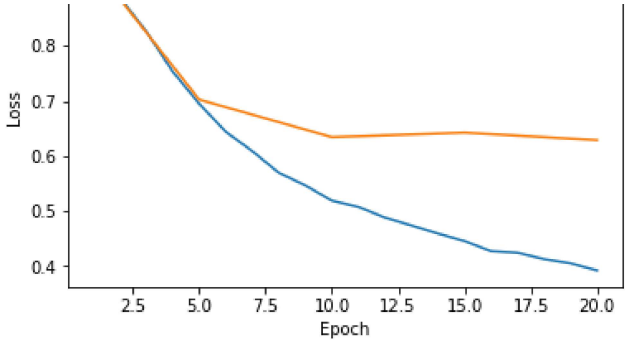


Similarly, IOU value keeps on increasing for training data, but saturate at around p=0.5 for testing data

IOU v/s epoch number for p=0.5:



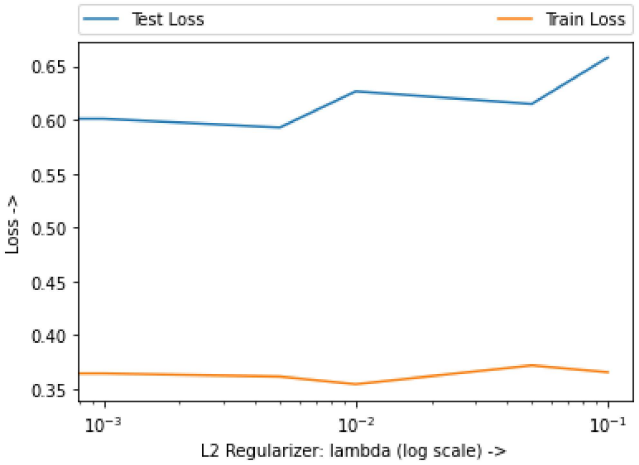
Loss v/s epoch number for p=0.5:



p=0.5

Comb loss = (1,1,1)

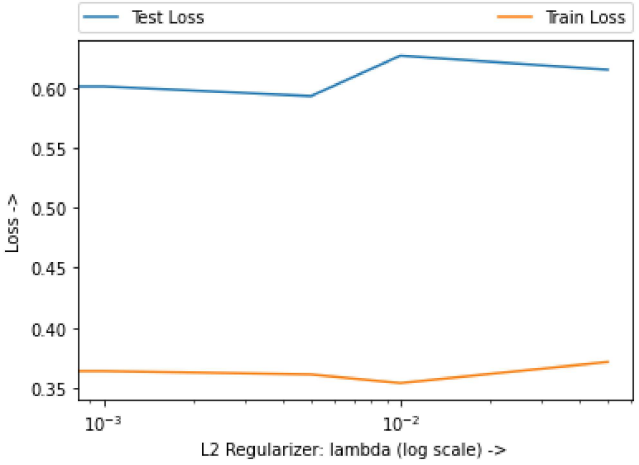
Regularizer: L2

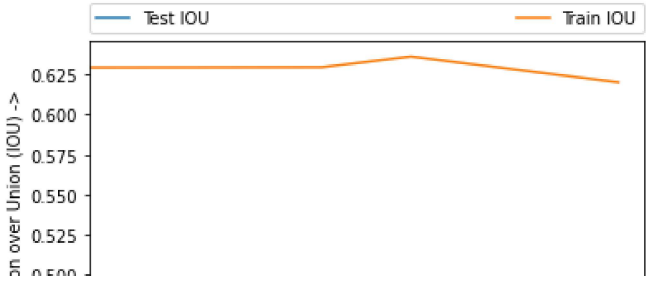


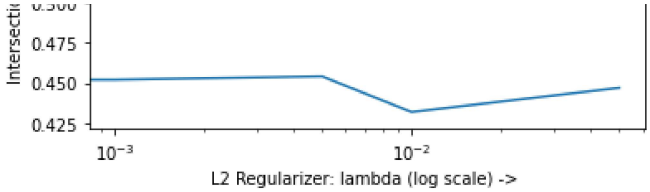
The training loss minimizes around lambda = 0.01 and then increases.

Testing loss minimizes around lambda = 0.05 then increases.

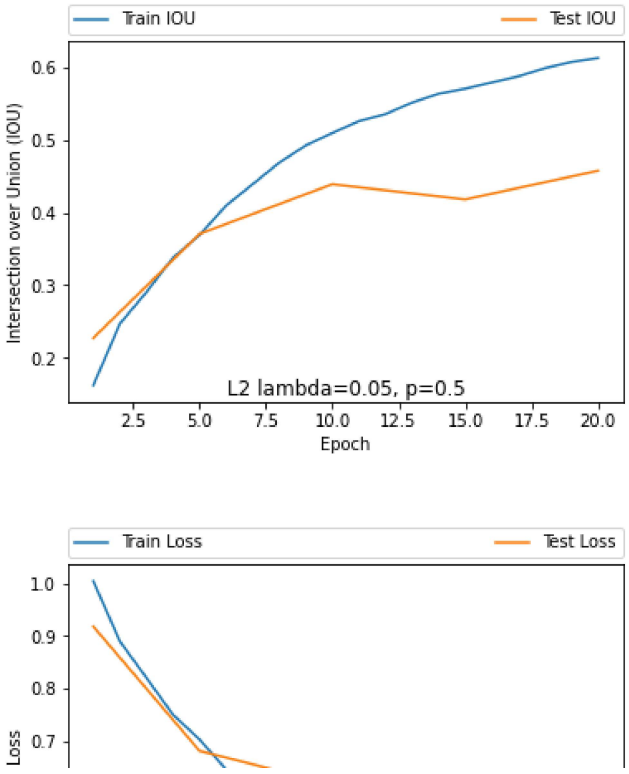
This is because either points away from a particular 'optimal' range of lambda. The model is either overfitting or underfitting

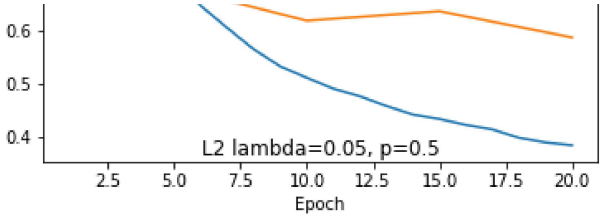




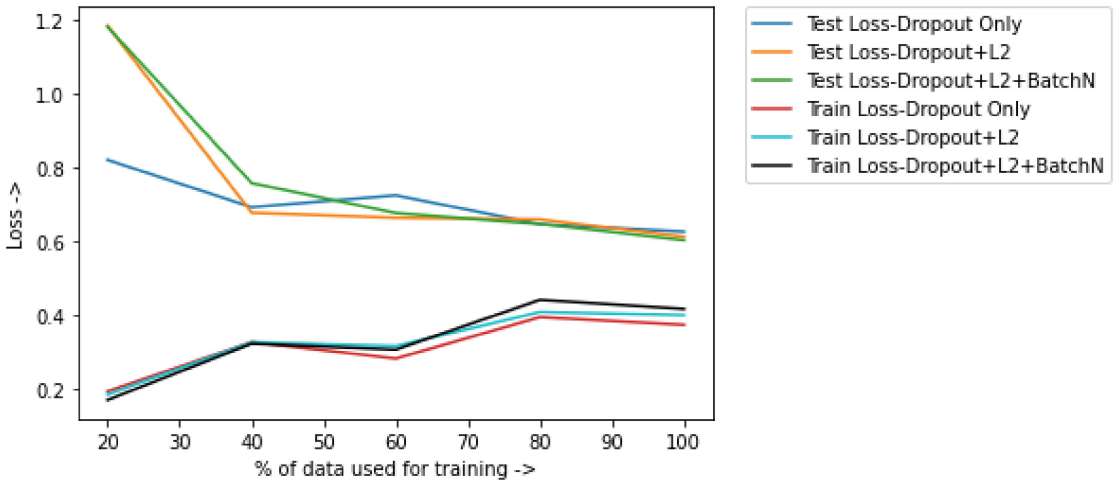


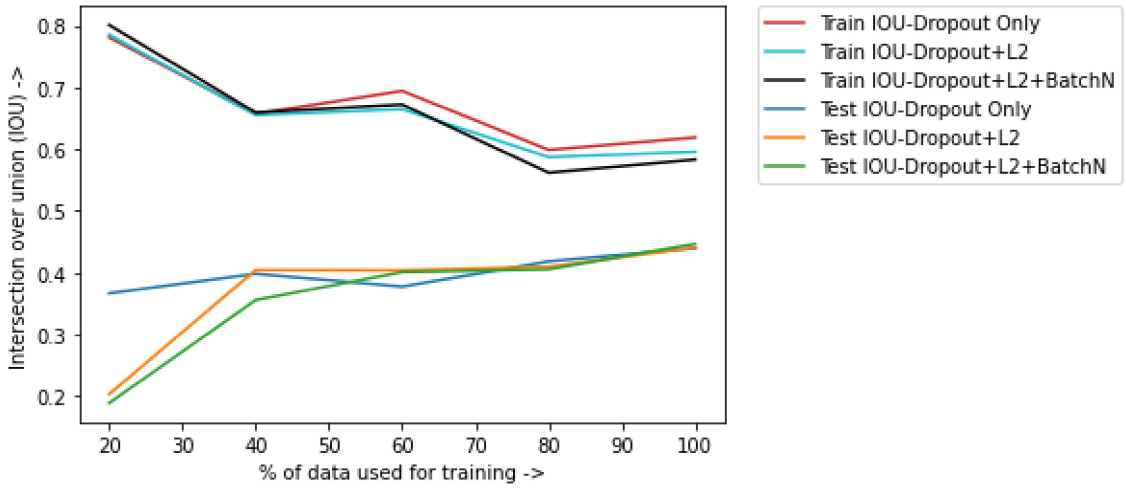
Dropout p=0.5 + L2 lambda=0.05 + Batch Norm:





Comparison of regularization techniques for different fractions of training dataset:





Approx. Best config:

L2: lambda= 0.04

Batch Normalization

P = 0.55 for both

Conclusions :

One of the biggest observations was that the Test loss/IOU values saturate very quickly and the models seem to perform bad whenever the area to be segmented is small. This is evident in all the "**IOU vs ground truth size**" plots (for test/val sets) above in the Loss Experiments. Comparing the **test IOU histogram** of Exp 6 with all of the others before, it appears that the loss weights combination of (a_ce, a_dice, a_inv_dice) = (1, 0.25, 0.75) is the superior one as It has the peak at the highest bin compared to the previous Experiments. Comparing Exp 7 with Exp 8, it is evident that the configuration : (a_ce, a_dice, a_inv_dice) = (1, 0.25, 0.75) is superior to switching loss as well. Moreover, comparing the **IOU vs ground truth size** plots (for the training set specifically) we can see that they are significantly improved Exp 6 onwards. This leads us to believe that the method of data augmentation by random image rotation and oversampling small area samples is superior to horizontal and vertical flips even with fewer augmented samples (in all of our augmentation experiments, the training sets augmented with

image rotations were smaller than the training set augmented with horizontal and vertical flips). This makes intuitive sense in the given problem because there is positional information in the task of segmenting the **right** ventricle which is lost in case of some flips, but is preserved in image rotations, especially for small angles.

We also plotted the ground truth area distribution of the Training and Test sets and found that they were quite different. Finally, we ran some Loss Experiments (Exp 7 and Exp 8) with a Validation set drawn from the Training set itself (the network does not learn on these validation images) and we found that, while it is still saturating, it saturates at a much higher value of IOU compared to the Test Set (compare Exp 6 with Exp 7)

Another set of experiments were performed over various regularization methods. Most notably Batch Norm, L2 Regularization and Dropout in the inner conv2d layers. There is some marginal improvement in the test loss and IOU values at a low value of L2 regularization (0.04) and a keep_prob of around 0.55 but the improvements are not significant.

Further experiments :

The lack of sufficient data and variation within the data seems to be the major challenge. So if we had more time we would look to pre-train the network (at least the encoder part) with self supervised learning on all the unlabeled images that are also present in the data set. Or try transfer learning so that the number of weights to be trained on the small dataset is reduced

Note : We have relied almost entirely upon IOU values to compare the performance. Dice coefficient is another possible comparison metric, however Dice coefficient turns out to be a more lenient metric and we found out that Dice coefficient value of any output-ground truth pair is generally about 0.1+/-0.02 higher than the corresponding IOU value