EE 782 : Advanced Machine Learning - Assignment 1

# HYPER PARAMETER TUNING

T Sanjev Vishnu (180110090) and Abeen Bhattacharya (170110013)

## Introduction

Analysing loss functions and regularizers is one of the most important aspects in Deep Neural Networks. It is of interest since they decide how fast and steadily a network converges. Hence the motivation behind choosing this problem statements is to get to know the most efficient loss functions w.r.t to the dataset.

## Experiment Design

There were primarily three different kinds of experiments.

Changing the split of dataset : Experiments were conducted with different loss functions (Binary Cross Entropy, BCE + Dice and Focal loss). This was done to analyse how quickly the loss functions converge and to choose the best split to conduct further experiments.

Adding L1 and L2 penalty separately : Tweaked the valuers of hyperparameters for the regulalizers to get an idea about the performance of L1 and L2

## Understanding the Plots

Model chosen is a simple UNet architecture which has 1,179,121 Parameters.

Code : https://github.com/VishnuSanjevThulasiraman/Hyper-Parameter-Tuning

The optimizer chosen is Adam.

Since the loss functions and the dice coefficients (range 0 to 1) are plotted in the same plots.

For better understanding, have been linear scaled down by a factor of maximum(loss, valid_loss)

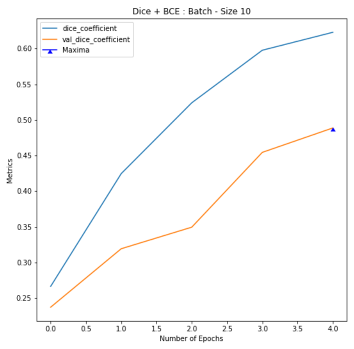
**loss function - blue**, **validation loss - orange**, **training dice - green**, **validation dice - red**

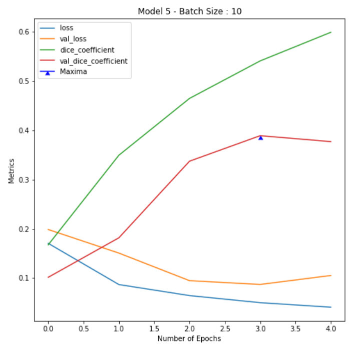
Since the each experiment is data hungry and requires large number of computations, the experiments have been done for a small number of epochs assuming the maxima of dice coefficient reached to be 90% of the maximum dice coefficient attainable even for a large number of epochs. This is a reasonable assumption given that the dice coefficients generally start to flatten out after 10 epochs (clearly visible from the plots)

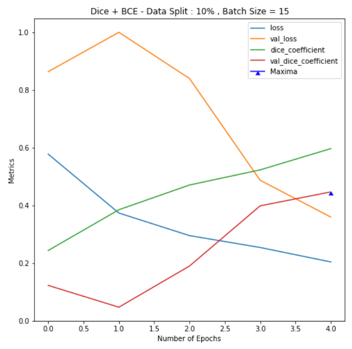
## Choosing Optimal Batch Size within the limitations

Initially, two experiments were conducted on different batch sizes to conclude the best possible batch size which will maximise results in the restrictions (number of epochs and datasize).

Here, the validation dataset and the training dataset is the same. Batchsize = 15 gives more reliable results ( probably since each patient has approximately 15 images to be trained with) than Batch Size = 10 (which can also become unpredictable with variable data splits). This can be confirmed  by analysing the decay of loss functions.







Henceforth, all the plots and results are experiments with Batch Size set as 15.

## Plots and Inferences

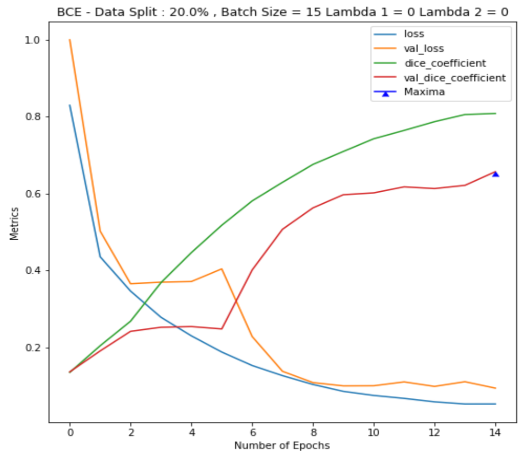
## Part I : A Comparative Study of Different Loss Functions in Image Segmentation

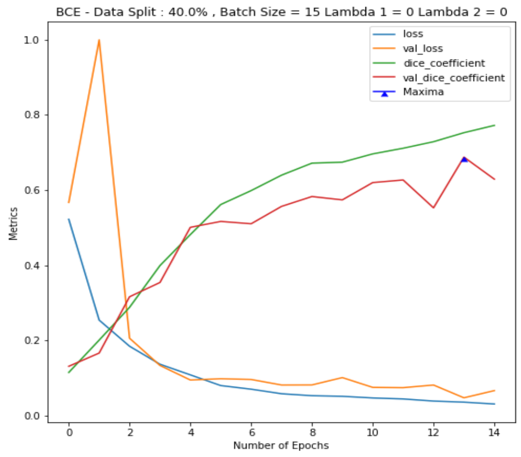
Experiments with different kinds of loss functions are conducted against different data splits. A data split - 20% indicates that 20% of the Data is used as the validation dataset.

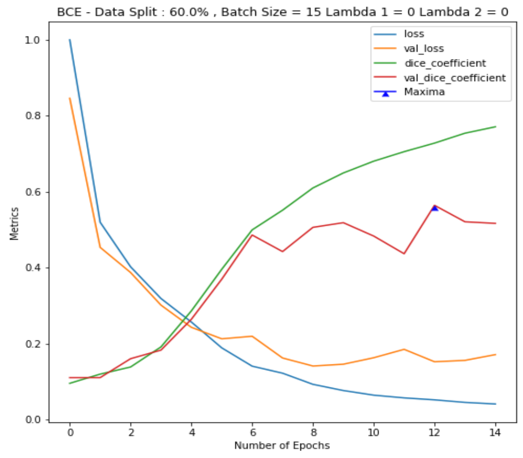
## Binary Cross Entropy

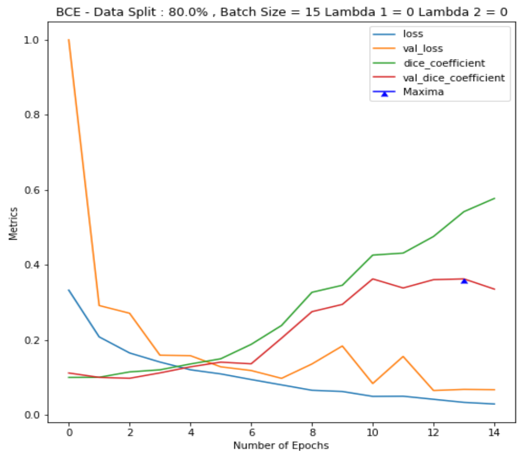
Data Splits of 20% and 40% perform significantly well compared to other splits. Lesser the size of training dataset, more fluctuations in the loss function. The loss functions for splits 60% and 80% ( bottom two figures ) seem to have reached their minimas. This will only result in poorer dice coefficients with high variances. It is to be noted that once the loss function converges to a particular limit the dice coefficients saturate too.

Binary Cross Entropy gives better dice coefficients and can be chosen when the dataset is huge and for a split which has around 70% training data validated against 30%. For lower splits Binary Cross Entropy becomes unreliable as there is instability.





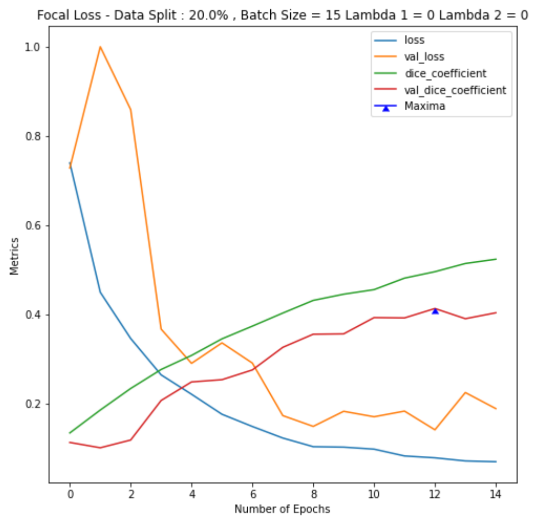


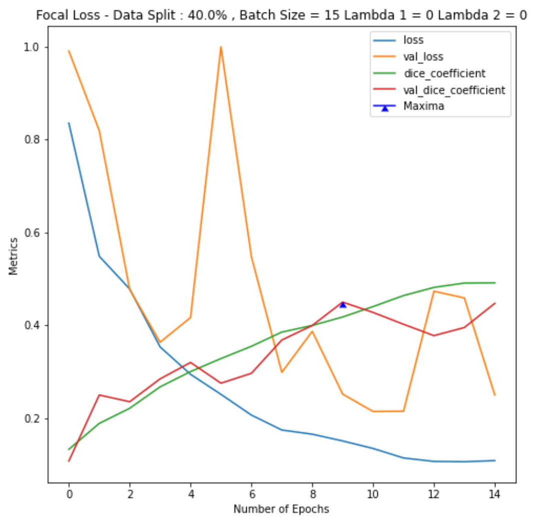


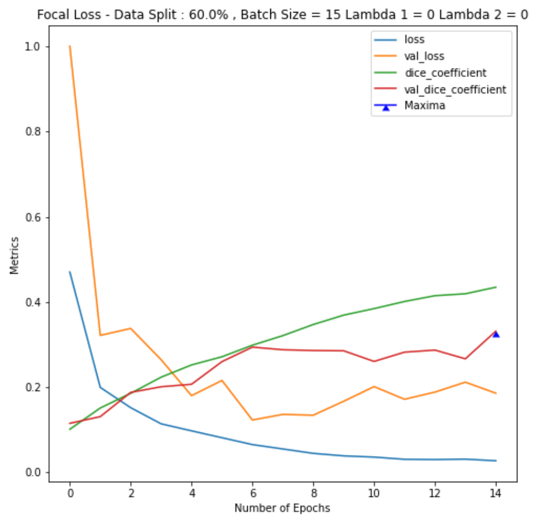
## Focal Loss

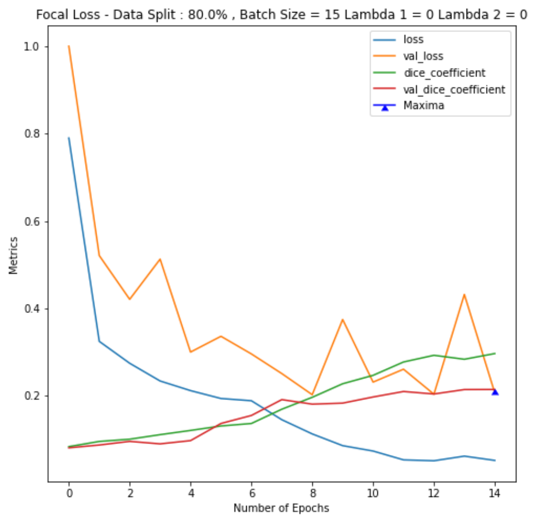
More steady increase in dice coefficients (even for different train test splits) compared to the other loss functions. Though the focal loss is stable for different data splits it is clear that the dice coefficients seem to saturate for a much lesser value compared to the other two losses.

Decay of training loss is fairly the same across different splits whereas there are significant variations in validation loss. This probably caused by the much lesser dice coefficient values compared to using other loss functions and lesser dice coefficients lead to instability in decay.



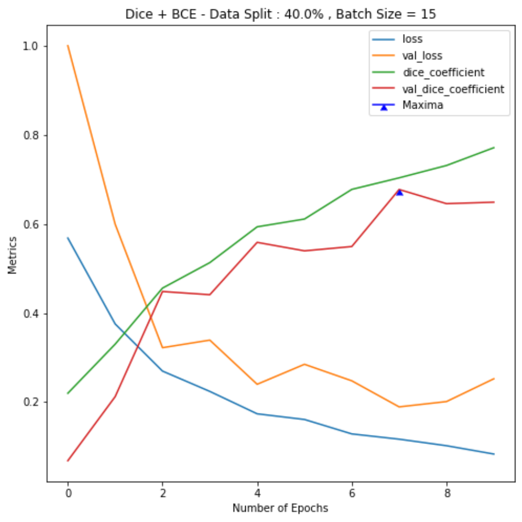


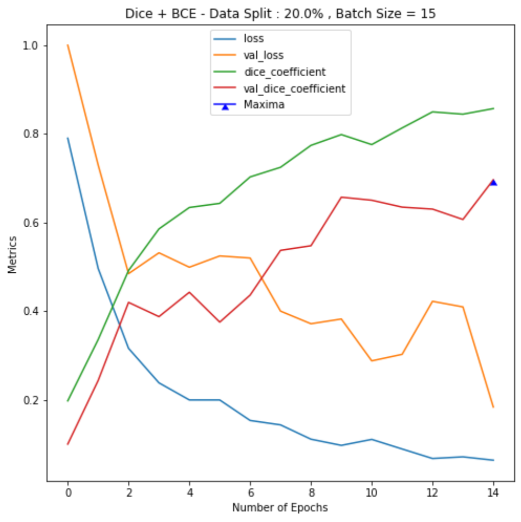


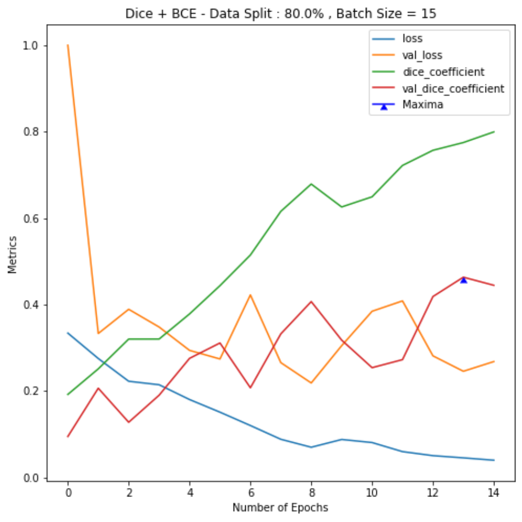


**Binary Cross Entropy + Dice**

This clearly outperforms the other two loss functions. Even for having 80% of test data the model gives better results and growth in dice coefficients can be observed. For a datsaplit of 80 - train, 20 - test, the dice coefficient of training is steadily increasing and nears 0.85 after just 15 epochs whereas validation dice is far less. Whereas the data split of 60, 40 is of interest to us as the gap between validation and training dice coefficients is less which also smoothens the validation loss function.





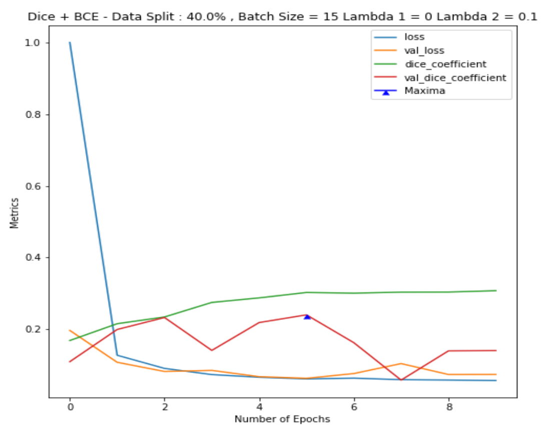


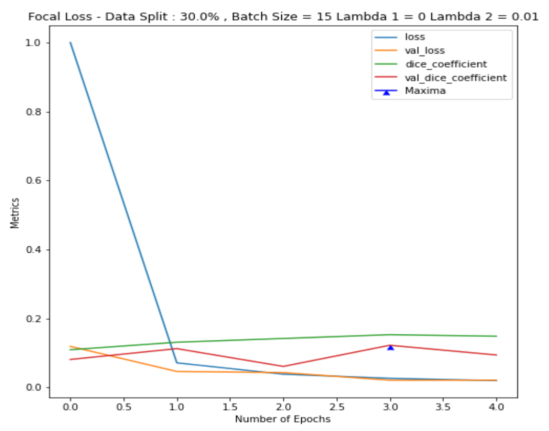
This indicates that there is a tendency to overfit the training data since the first 80% images belong to patients whose heart images are not being evaluated against.. Either regularization penalties or a good mix of training and test data is to be ensured while choosing this function.

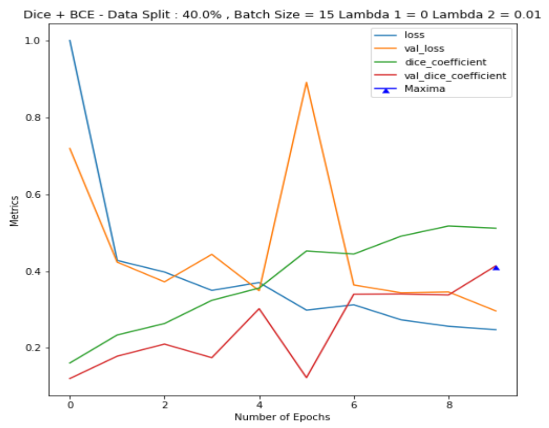
## Part II : Choosing hyperparameters of Lambda in Regularizations

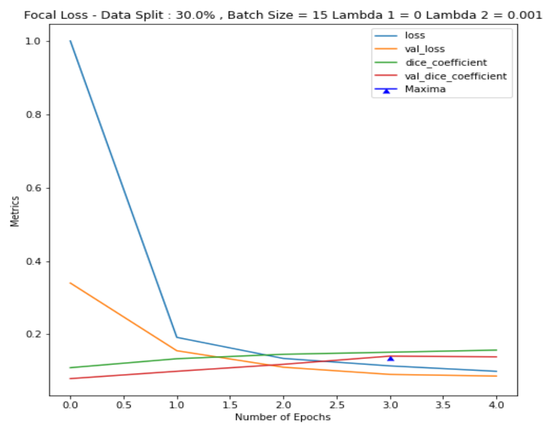
Lambda 1 and Lambda 2 are hyperparameters of L1 and L2 respectively. Regularizations have been applied to layers of Convolution and Deconvolution for all the trainable weights.

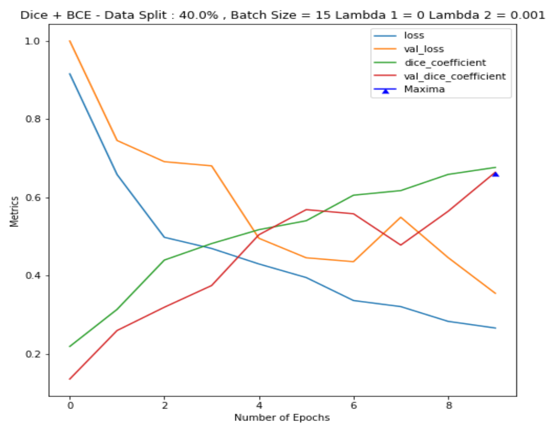
Different values of Lambda 2 ( L2 penalty) are analysed for Dice + BCE and Focal losses.

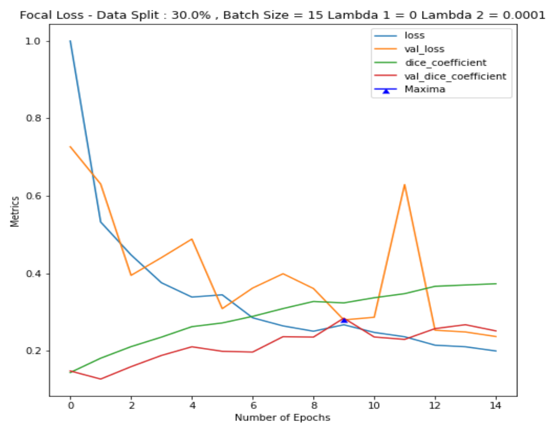


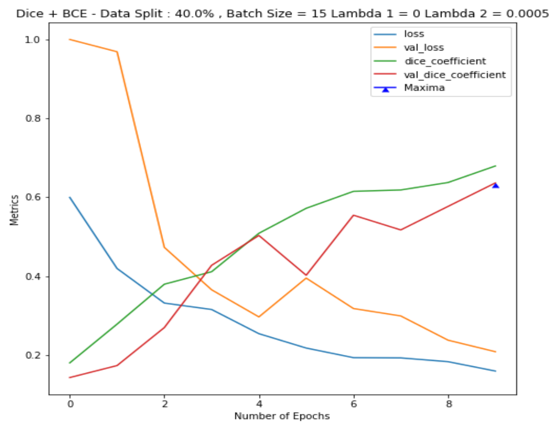


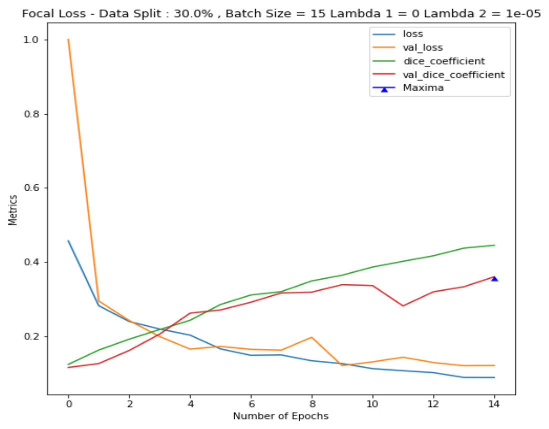


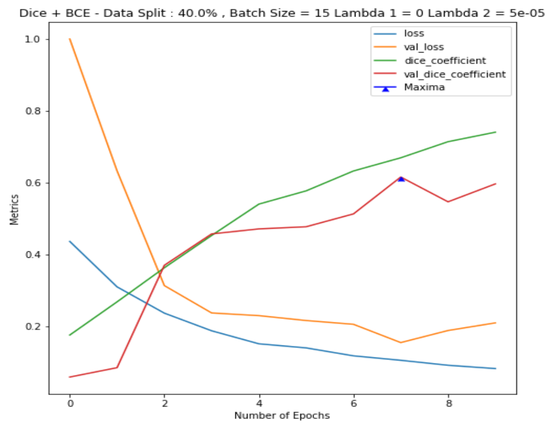








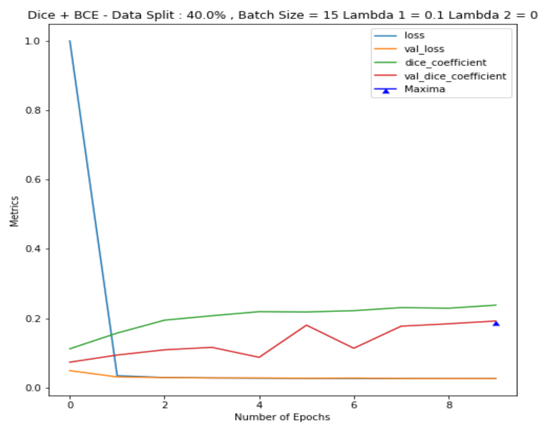


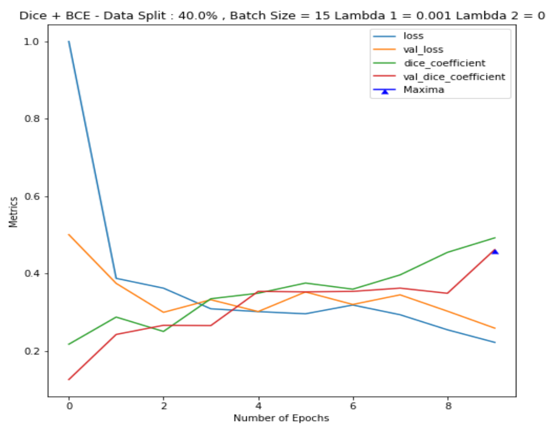


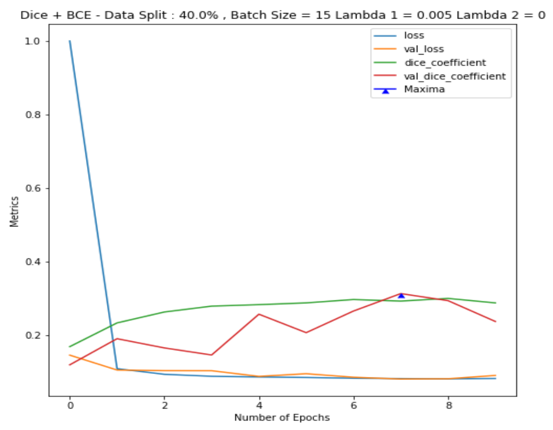
It is clear that values greater than or equal to 0.001 makes the loss function reduce rapidly. This indicates the weights are shrunk massively. Once loss functions decay below a threshold, the gradient doesn't flow back to try different values of weights as it will again be limited. For values of Lambda between 0.0005 and 0.001, the dice coefficients reach higher values quicker than their counterpart experiments without regularization. When lambda 2 is below 0.0005, the results are not far from the original experiments. Therefore, these low values should be avoided and not of interest.

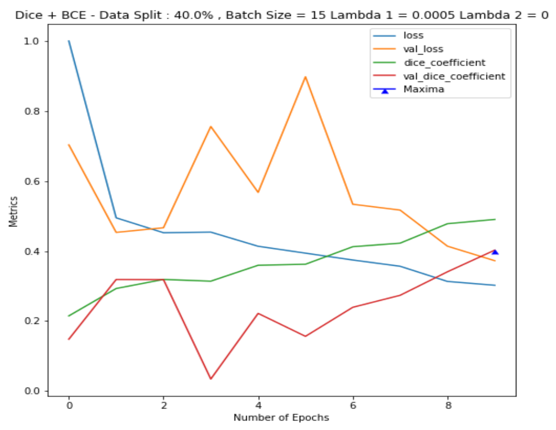
These results indicate that Lambda 2 must be chosen such a way that the loss and the L2-norm * Lambda2 are of the same magnitude. Assuming random initialisation follows uniform distribution (0, 1), it is safe to assume L2-norm lies around 577 (running simulations). So lambda belonging to (0.0005 , 0.001) gives penalty range as (0.288, 0.577) which is comparable to the loss (0, 1). A good way of choosing hyperparameters is to run simulations to find L2 penalty. Now use this value to find hyperparameters as described above.

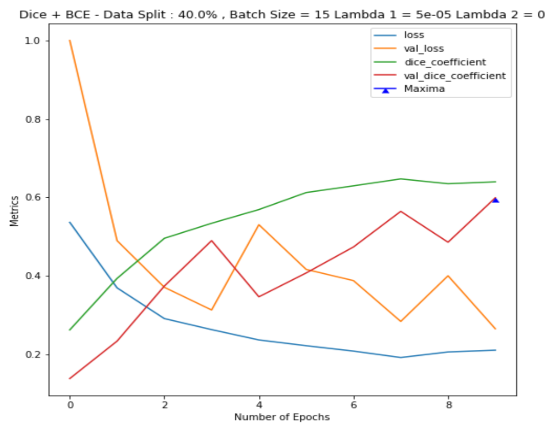
Analysing L1 Regularisation,











We can see high values of Lambda 1 cause the loss functions to decay really quickly. Whereas Lambda 1 which brings the penalty in magnitudes comparable to loss enhances the dice coefficients.

Similar method used in the previous section for L2 is to be used here. For 10E6 parameters, average L1 is 5E5, so Lambda 1 around 5E-7, will help the dice coefficients grow quicker.

## Conclusion

Different ways of choosing loss functions and tuning hyperparameters of regularizers is given above in detail with experimental evidence.

Due to multiple constraints such as GPU and time taken for each experiment the number of epochs used to train were small. With enough time, longer experiments could have been done with more conclusive results. A number of experiments and results were not meaningful and have not been included here. Intelligent experiments had to be devised such that useful inferences could be drawn. There was a lot left for interpretation and with enough time, the number of assumptions could have been reduced. The experiments also demanded huge data and the dataset provided is not good enough to conduct bigger experiments as the model might overfit anyway. More time would have allowed for multiple fold cross validation among the data. Different mix of training and test data will give more insights and help in understanding how to tune the parameters better and the accuracy which can be achieved.