Image Segmentation

– Semantic Segmentation on Brain MRI (Magnetic Resonance) Images

Abstract

The task is to use deep learning technology to complete segmentation on brain MRI (magnetic resonance) images, that is, for a processed MRI image, the lesion area will be segmented out.

The data set used in this case is the "Brain MRI Images for Brain Tumor Detection" from Kaggle[1]. It contains MRI data of 110 patients. Each patient corresponds to multiple images(in .tif format) with a channel number of 3, and the corresponding segmentation result is a single-channel black and white image (white represents the diseased area).

The U-Net[2] network is provided within the code framework. We tested how it behaves with different loss functions and optimizers. We also look into U-Net++ and TransU-Net structures. Due to the time scope limitation, we have to focus only on U-Net structure. With the same optimizer and the same loss function, U-Net++ has a slightly better performance(0.2%-0.3% higher DSC scores), but not statistically significant. The best DSC score with U-Net is 0.915472, and the best with U-Net++ is , both with Dice loss and Adam optimizer.

Both the U-Net and U-Net++ architectures have comparatively good performances on segmentation and edge maintenance, but still fail to detect accurate shapes especially when the target area is small or when the shape has edges with many turns. To improve this, we could test TransU-Net, or try to combine with some adaptive methods to detect these poor-behaving areas and make them tokens with higher importance or semantic meaning.

1 Introduction

Medical image segmentation is used to extract regions of interest (ROIs) from medical images and videos. Segmentation in medical imaging is a powerful way of identifying objects, segmenting pixels, grouping them, and using this approach to labeling to train computer vision models[3].

For segmentation, we use a deep learning architecture known as U-Net, a convolutional neural network. It is particularly well-suited for tasks that require precise segmentation of small or complex structures, such as cells or organs. The reason being: the semantics of medical images are relatively simple and the image structure under a certain task is often fixed. All features in medical images are important in segmentation, so both low-level features and high-level semantic features are important, and in that case, the U-shaped skip connection structure (feature splicing) is more useful[4].

In our experiments, we compared some common-used loss functions and optimizers and how they affect the U-Net behaviors under our task. Dice similarity coefficient is used to statistically analyze the learning results. The highest DSC score 0.9155 with U-Net structure is with Dice loss and Adam optimizer.

2 Programming Modules

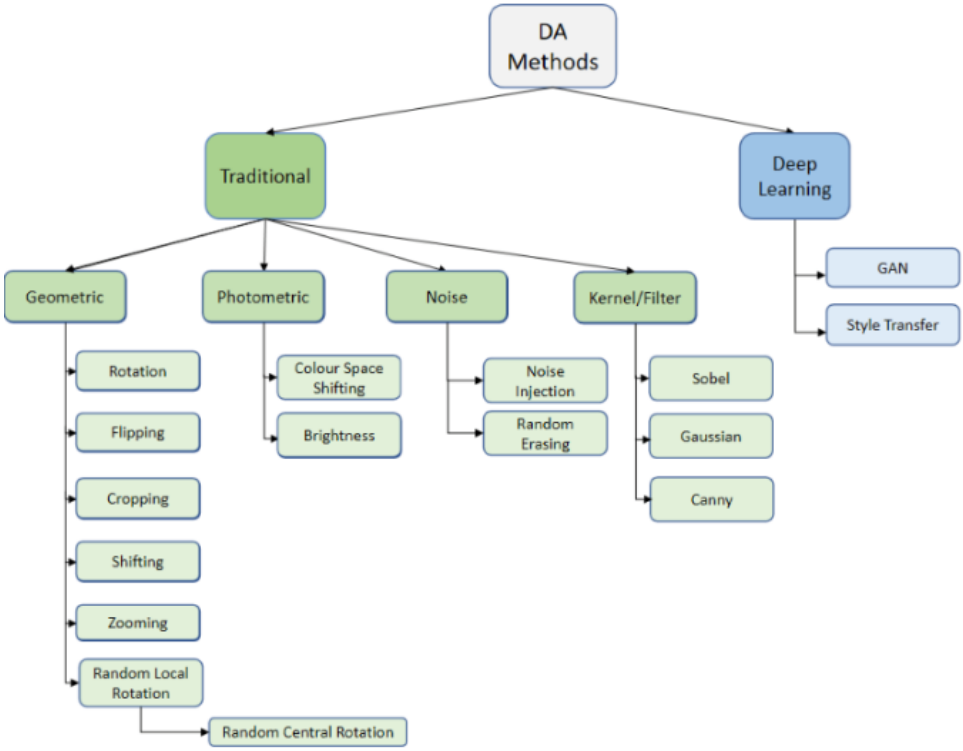
The network is built based on the PyTorch module and mathematical calculation relies on both PyTorch and Numpy. Matplotlib is used for generating diagrams for model behaviors. To log the training and validation process out, we also invoke Tensorboard to track and visualize related metrics. To load and process medical images, Medpy is used. All training runs on GPU with torch CUDA to accelerate the calculation.

3 Data Load and Data Argumentation

Code for this section is provided with the assignment resources. After reading in the images, data argumentation is applied.

Data augmentation is a technique in machine learning used to reduce overfitting when training a machine learning model, by training models on several slightly-modified copies of existing data[5]. In some fields like medical imaging, the availability of the huge amount of data is not possible, as it takes a good amount of effort to collect the data and then labeling it requires domain expertise. To pursue better learning results, we use data augmentation to increase the amount of training data by making some reasonable modifications or transformations in the existing data[6].

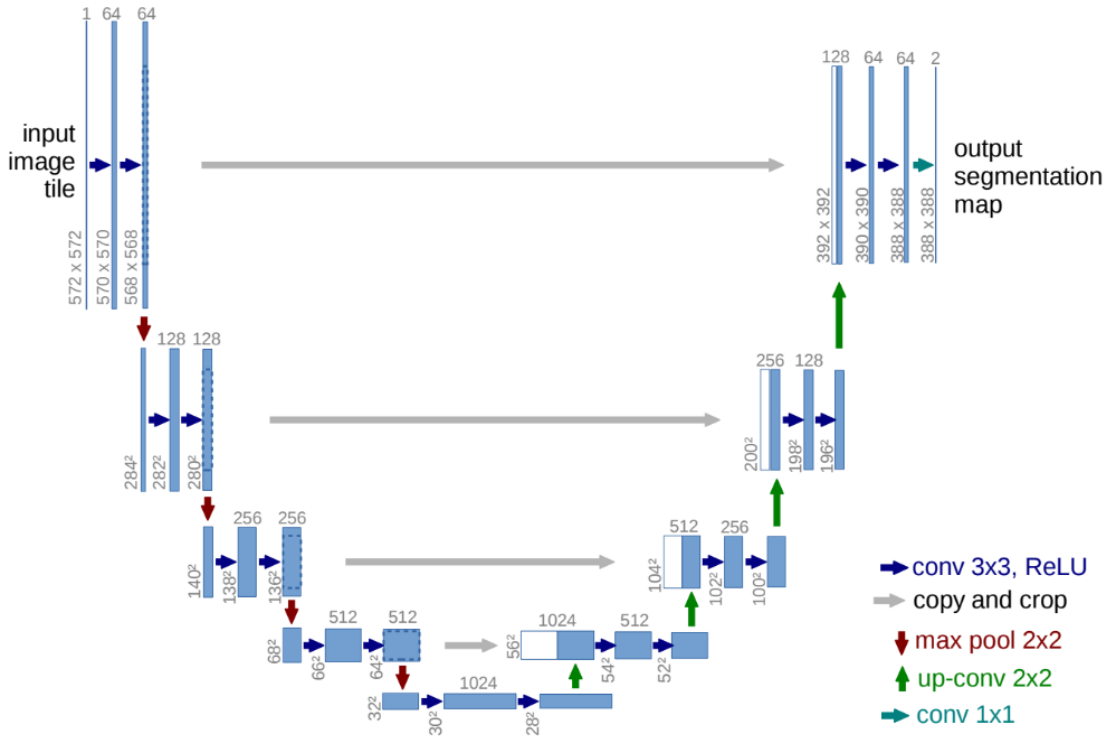
Argumentation methods usually vary based on the tasks and the data. Below listed some popular argumentation methods for image segmentation[7].



In our framework, data read-in and data argumentation happens in transform.py and dataset.py. The argumentation preprocessing contains: cropping, padding, resizing and normalization.

4 U-Net Architecture and Implementation

U-Net is a convolutional neural network that was developed for biomedical image segmentation at the Computer Science Department of the University of Freiburg. The network is based on a fully convolutional neural network, whose architecture was modified and extended to work with fewer training images and to yield more precise segmentation. It has the training strategy that highly relies on the strong use of data augmentation to use the available annotated samples more efficiently[2]. The implementation of this network is in the unet.py.



5 Common Loss Functions for Image Segmentation

Inspired by an online article “3 Common Loss Functions for Image Segmentation”[8], we will mainly compare and contrast Dice loss, Binary Cross-Entropy loss and Shape-Aware loss. We also implement Log-Cosh Dice loss proposed in a paper “A survey of loss functions for semantic segmentation”[9]. Hausdorff Distance Loss also has its advantage at maintaining more precise edges of segmented shapes. However, due to its non-convex nature, it is not widely used as a loss function.

Type	Loss Function
Distribution-based Loss	Binary Cross-Entropy Weighted Cross-Entropy Balanced Cross-Entropy Focal Loss Distance map derived loss penalty term
Region-based Loss	Dice Loss Sensitivity-Specificity Loss Tversky Loss Focal Tversky Loss **Log-Cosh Dice Loss**(ours)
Boundary-based Loss	Hausdorff Distance loss Shape aware loss
Compounded Loss	Combo Loss Exponential Logarithmic Loss

The choice of loss function for the task of segmentation is key in determining both the speed at which a Machine-Learning model converges, as well to some extent, the accuracy of the model[8].

5.1 Dice Loss

Dice coefficient is a similarity metric commonly used in image segmentation, natural language processing, and other fields where there is a need to measure the similarity between two sets[10].

It is particularly useful for imbalanced datasets, where one set may be much larger than the other. It is a better choice for image segmentation tasks, as it is more sensitive to overlap between the predicted and ground truth masks.

5.2 Binary Cross-Entropy Loss(BCE)

Cross-Entropy loss is a widely used loss function used when adjusting model weights during training. The aim is to minimize the loss. As the predicted probability diverges from the actual label, the loss will increase.

We use Binary Cross-Entropy Loss instead of Cross-Entropy Loss here, since Cross-Entropy Loss is mainly used for multi-class classification, and BCE is for binary classification. Although it is doable to use Cross-Entropy Loss for binary classification, Softmax is a more suitable method for this loss to get probability output. For binary classification when there are only 2 values, the output from softmax is always going to be something like [0.1%, 99.9%] or [99.9%, 0.1%] based on its formula. Eg. softmax([-2,34, 3,45])=[0.3%, 99.7%]. It does not represent meaningful probability[11].

5.3 Shape-Aware Loss

Shape-aware loss as the name suggests takes shape into account. Generally, all loss functions work at pixel level, however, Shape-aware loss calculates the average point to curve Euclidean distance among points around the curve of predicted segmentation to the ground truth and use it as coefficient to cross-entropy loss function.

5.4 Log-Cosh Dice Loss

Proposed in “A survey of loss functions for semantic segmentation”[9]. It is a variant of Dice Loss and inspired regression log-cosh approach for smoothing variants on skewed data.

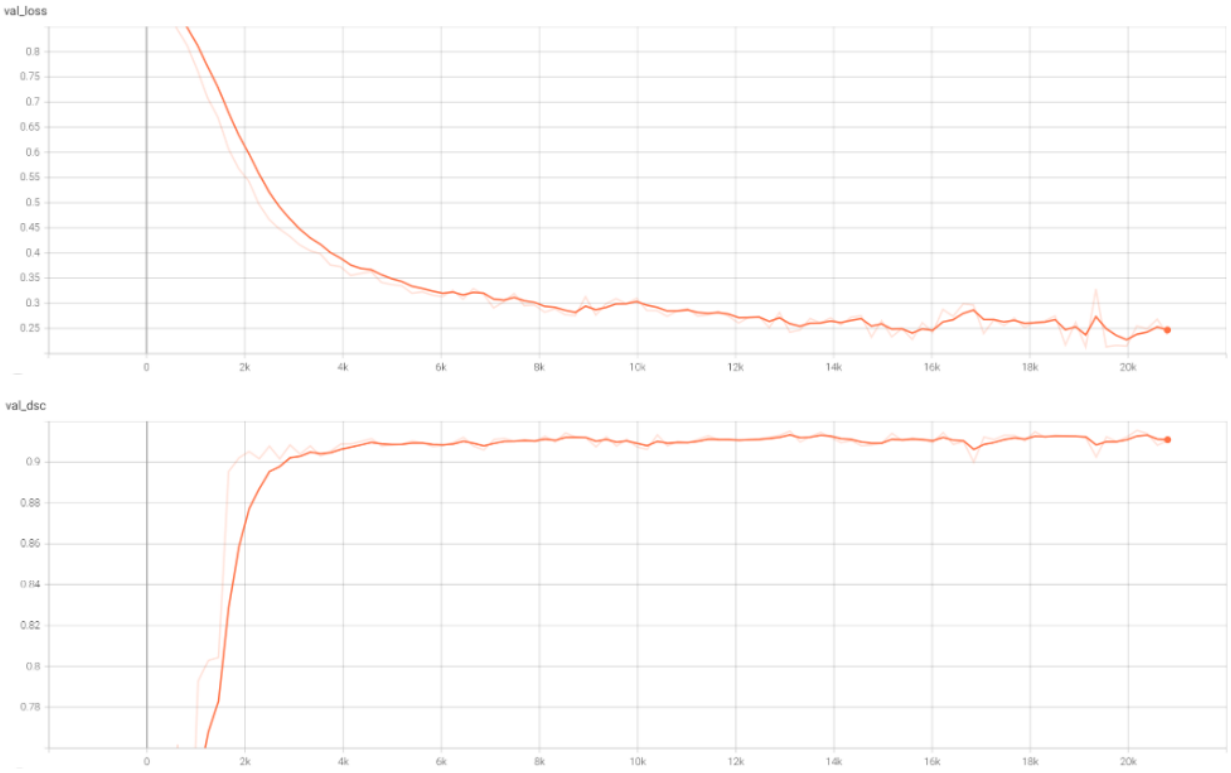
$$coshx = \frac{e^x + e^{-x}}{2}$$

$$L_{lc-dce} = log(cosh(DiceLoss))$$

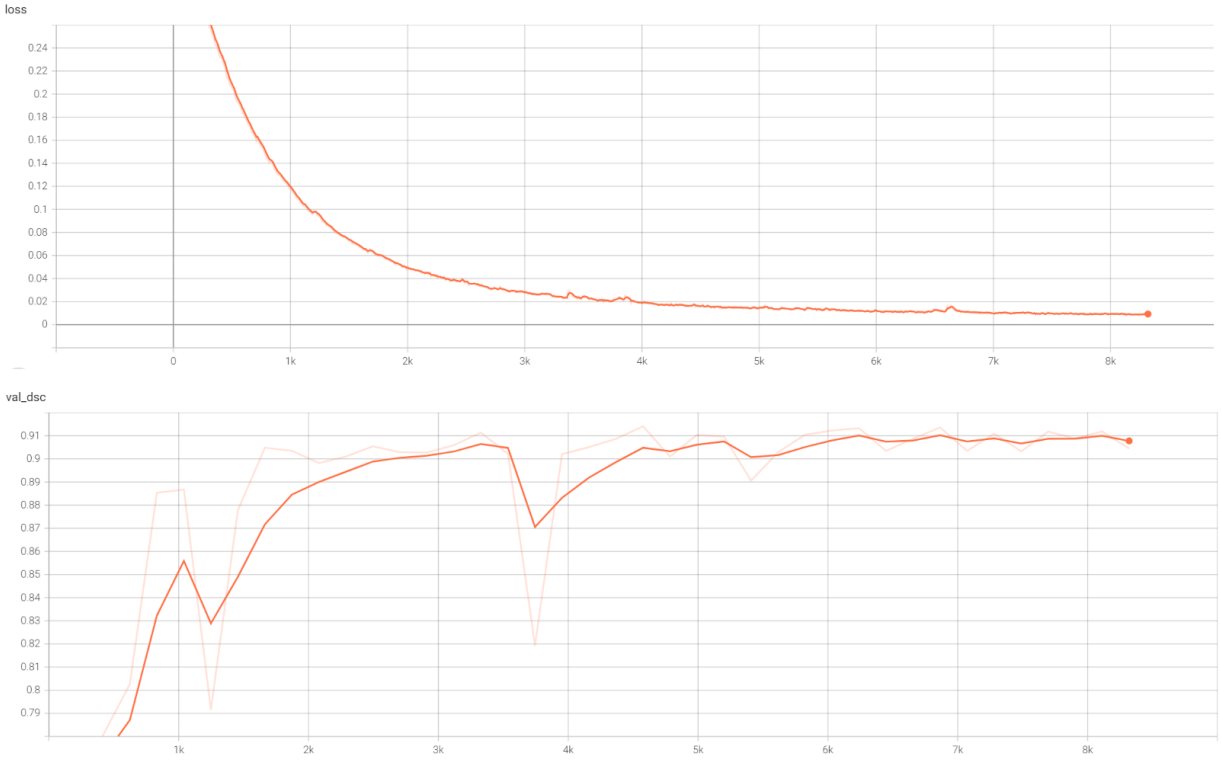
6 Experiment: U-Net with Adam Optimizer

In this section, we compare how U-Net performs with different losses under Adam optimizer. To cut down the training budget and the report length, we only show the hyperparameter combinations that shout out comparatively good learning results under the same loss method. Therefore, many parameter tuning experiments are omitted.

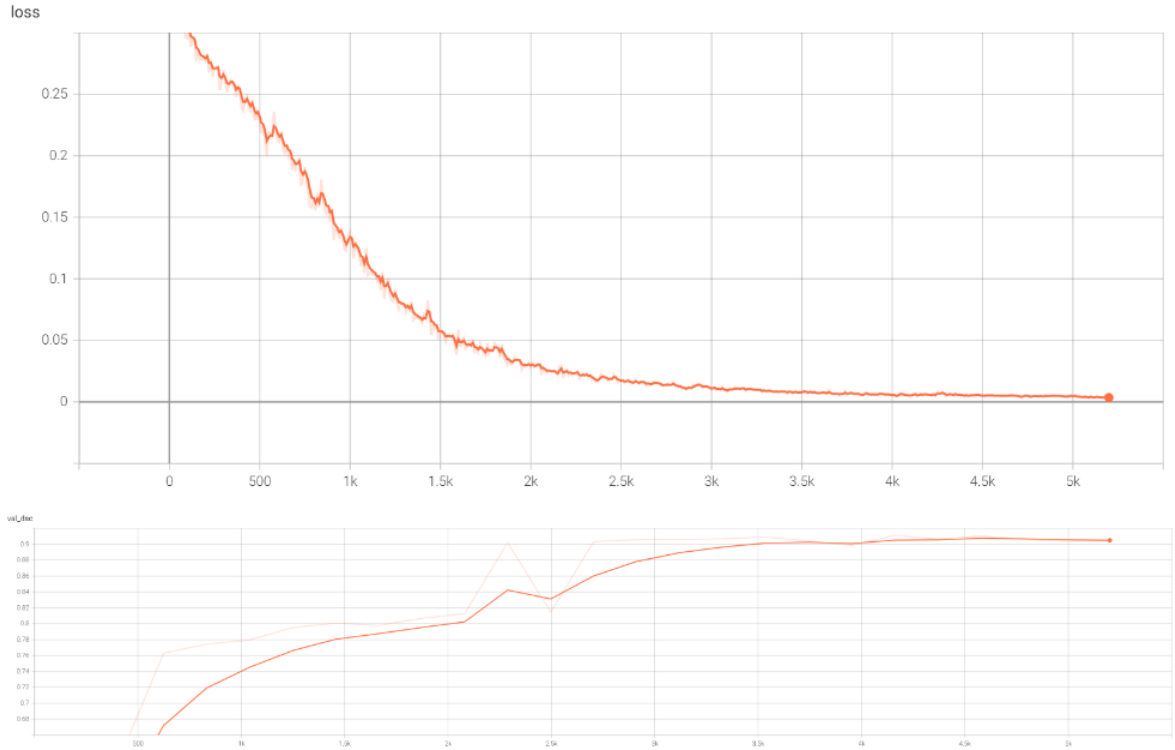
6.1 U-Net with Adam Optimizer and Dice Loss



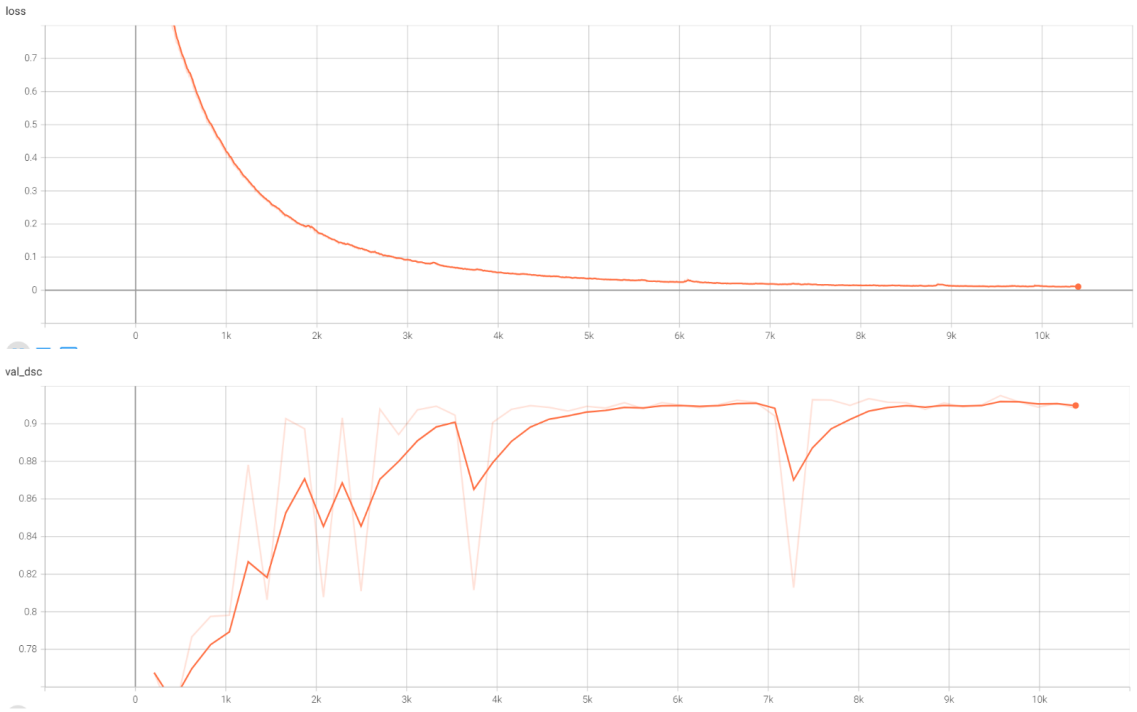
6.2 U-Net with Adam Optimizer and Binary Cross Entropy Loss



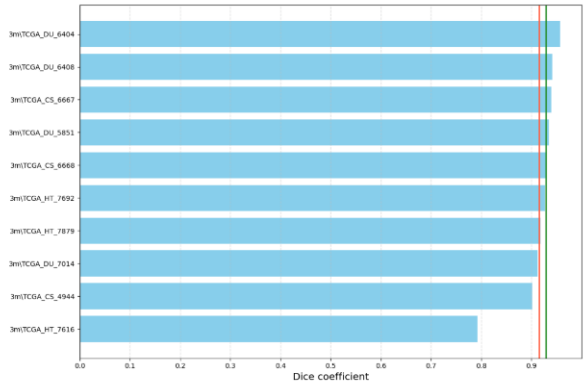
6.3 U-Net with Adam Optimizer and Log Cosh Dice Loss



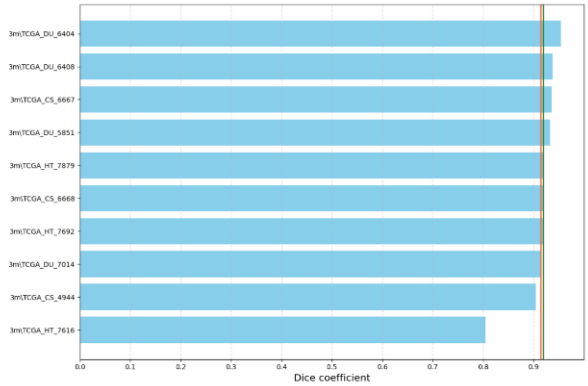
6.4 U-Net with Adam Optimizer and Shape-Aware Loss



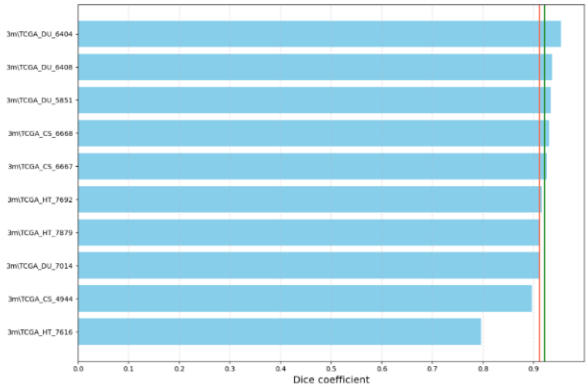
6.5 Inference Dice Similarity Coefficient



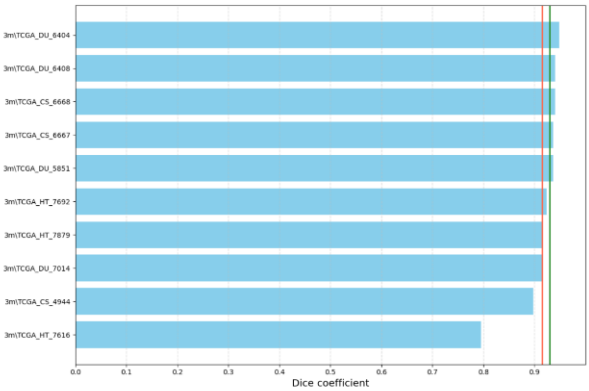
U-Net with Adam Optimizer and Dice Loss



U-Net with Adam Optimizer and Binary Cross-Entropy Loss

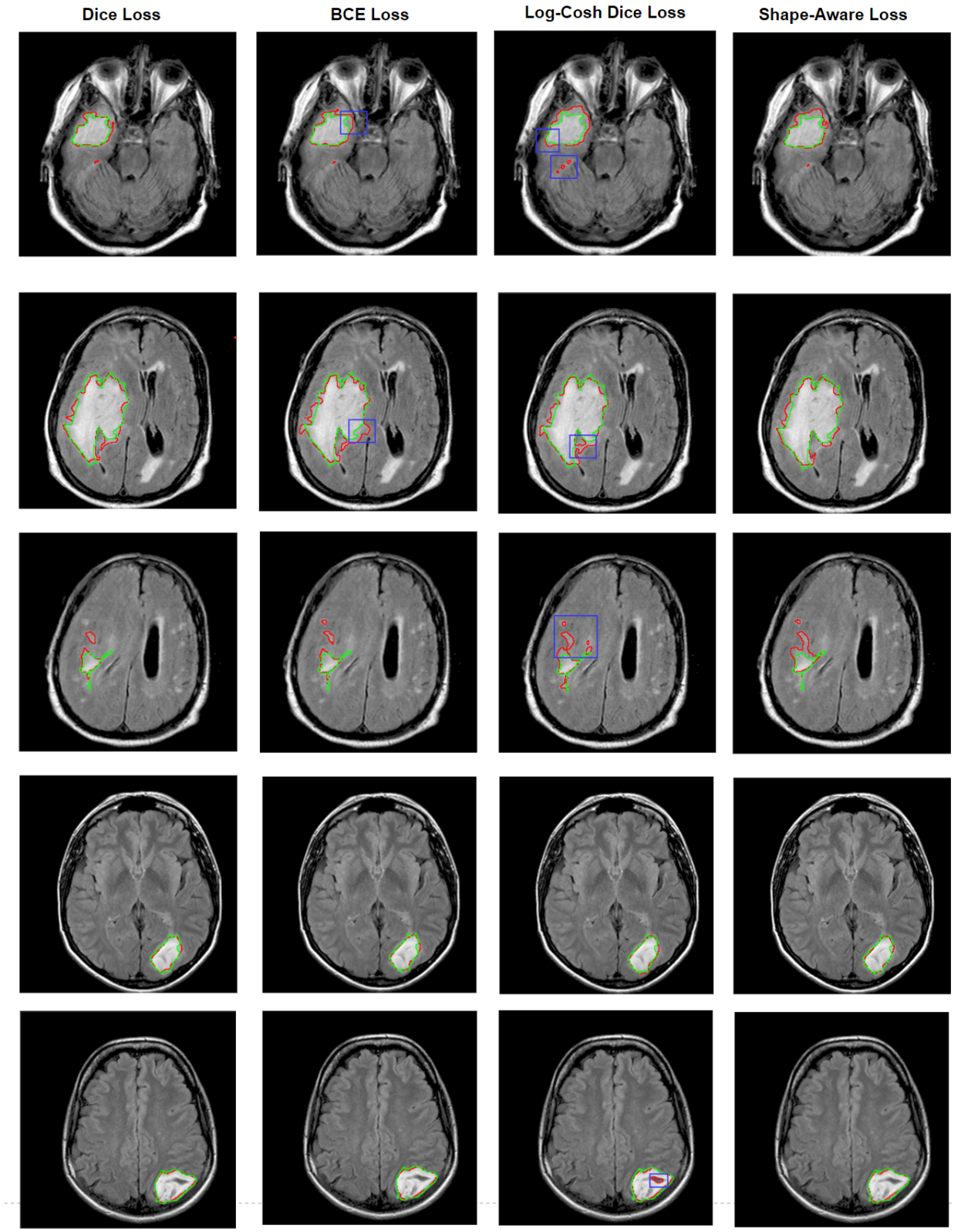


U-Net with Adam Optimizer and Log Cosh Dice Loss



U-Net with Adam Optimizer and Shape-Aware Loss

6.6 Segmentation Examples



6.7 Training Data Comparison

Loss Function	Learning Rate	Max Epoch	Batch Size	Pipeline Configuration	Approximate Converge Time (in minute)	Traing Time (in minute)	Average Traing Time Per Epoch (in minute)	Best validation mean DSC
Dice Loss	0.0001	100	16	U-Net Optimizer: Adam Image Szie: 256 Argumentation Aug Scale: 0.05 Aug Angle: 15	97	248	2.48	0.915472
Binary Cross-Entropy Loss	0.0001	40	16		69	82	2.05	0.914088
Log Cosh Dice Loss	0.0001	25	20		44	52	2.08	0.911516
Shape-Aware Loss	0.0001	50	20		78	99	1.98	0.914978

6.8 Analysis

1. With U-Net architecture and Adam optimizer, our model has the best performance(mean DSC on validation set) with Dice Loss and Shape-Aware Loss Dice loss is generally useful for region-based classification, and Shape-Aware loss works better with boundary-based classification.
2. Binary Cross-Entropy(refer as BCE in the report) and Log-Cosh Dice loss have comparatively mediocre performances. BCE usually works better with equal data-distribution amongst classes. That is, image masks with very heavy class imbalance (such as in finding very small, rare tumors from X-ray images) may not be adequately evaluated by BCE[8].
3. Based on the analysis in <2>, to improve model behaviors with cross-entropy related loss, we should introduce Weighted Cross-Entropy or Balanced Cross-Entropy, since they are widely used with unbalanced data distribution. Focal loss also seems to be a good direction to investigate into.
4. Log-Cosh Dice loss has the worst performances here. However, theoretically, it should provide good learning results as it can smooth variants for skewed dataset[9]. Also, see from its loss diagram, the loss curve drops slowly at the first 1/6 epochs and then follows a common elbow shape This means that our hyperparameter setting is not suitable. Perhaps the learning rate is too small that the network has to spend comparatively long time to reach a point that could actually learn the classification rules. If time applies, we will try to tune parameters for this network.
5. With BSE, Log-Cosh Dice and Shape-Aware loss, all validation DSC curves fluctuate a lot along the training process. Among which Shape Aware loss has the Dice Similarity Coefficient curve with the most fluctuations. However, its loss curve is quite smooth. DSC is based on hard classification; if the probability scores are hovering near the threshold, then the classifications may be flopping a lot, leading to unstable DSC scores[12]. We should also introduce IoU score(Jaccard index) to observe the area of the intersection over union of the predicted segmentation and the ground truth. However, choosing dice coefficient over IoU or vice versa is based on specific use cases of the task[13].

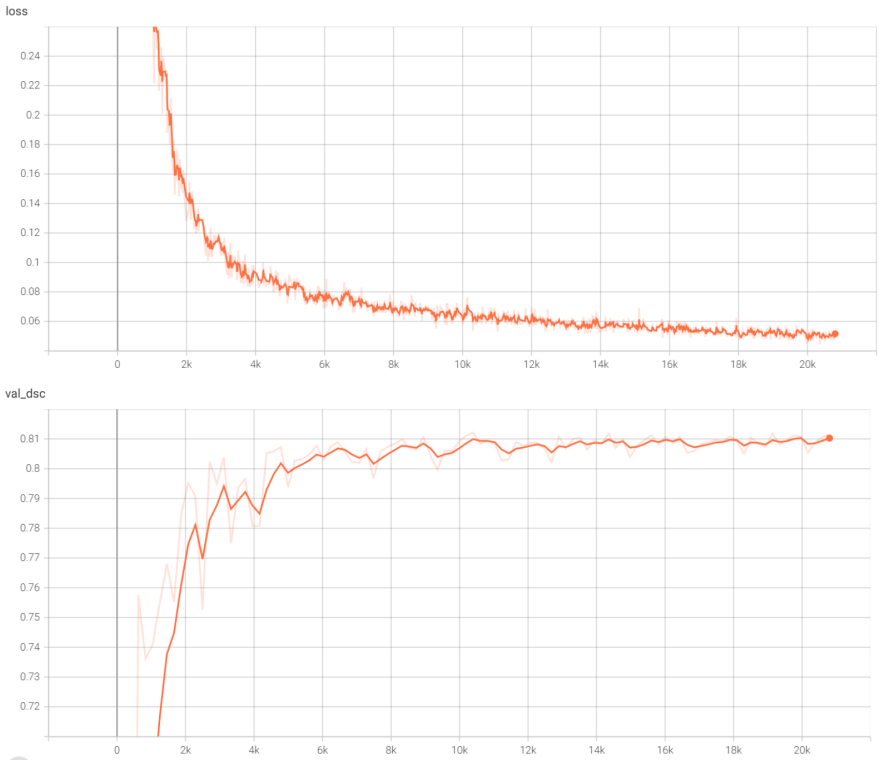
6. The green vertical line in the dice coefficient diagram shows the median of dice coefficients of all volumes (tumor types/labels). The red one specifies the mean value of dice coefficients of all volumes. Mean is more often closer to the true center of describing a bunch of values than the median. The median can sometimes be inaccurate because it does not use very much of the information available in the data. However, under skewed data, the median is more useful because the mean will be distorted by outliers[14].

7. As mentioned in <6>, since our data set is unbalanced, we will focus more on the median values of dice similarity coefficient, rather than the mean. For all four diagrams, mean and median are close. This suggests that there are no extreme observations. The gap inbetween for Dice loss and Shape-Aware loss is slightly larger, and the median under these two loss functions are slightly higher than the other two losses. A high Dice coefficient value indicates a high level of similarity between the predicted and ground truth masks, meaning that the segmentation model or algorithm is performing well[15].
8. In the "Segmentation Examples" section, green is the edge of the ground truth, red is the prediction edge, and blue circles out areas with clear poor segmentation. We can see that Dice loss and Shape-Aware loss are better at detecting irregular shapes and maintaining borders with turns. and maintaining borders with turns.

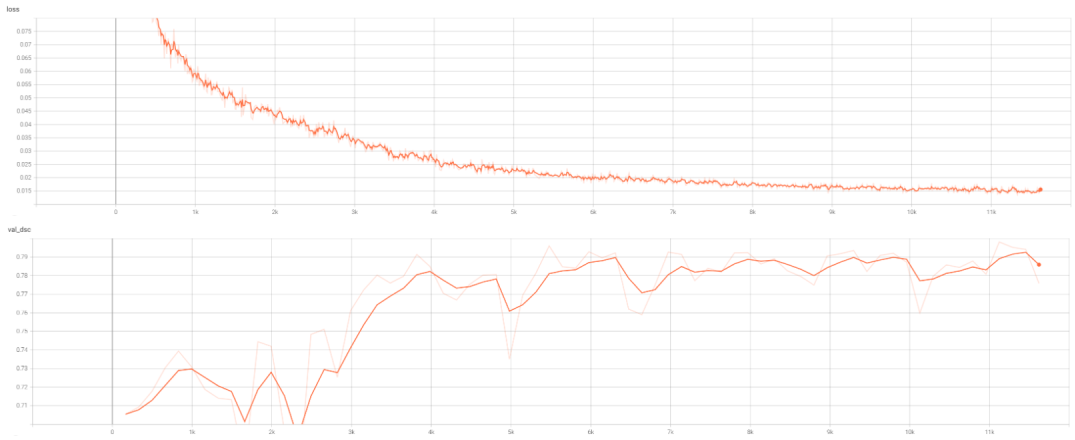
7 Experiment: U-Net with SGD Optimizer

We use the gradient descent optimizer with momentum and weight decay(a regularization technique by adding a small penalty, usually the L2 norm of the weights (all the weights of the model), to the loss function). Weight decay is known to prevent overfitting and keep weight small to avoid exploding gradients.

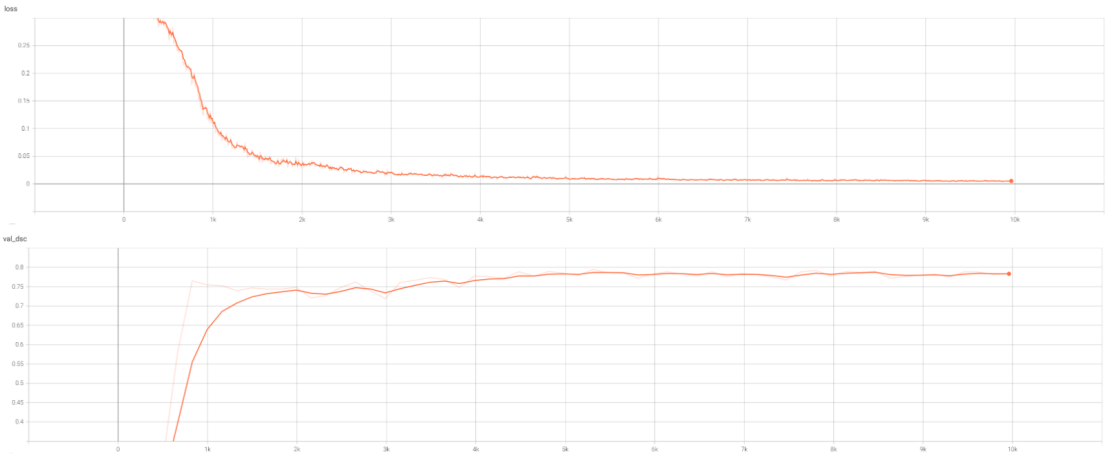
7.1 U-Net with SGD Optimizer and Dice Loss



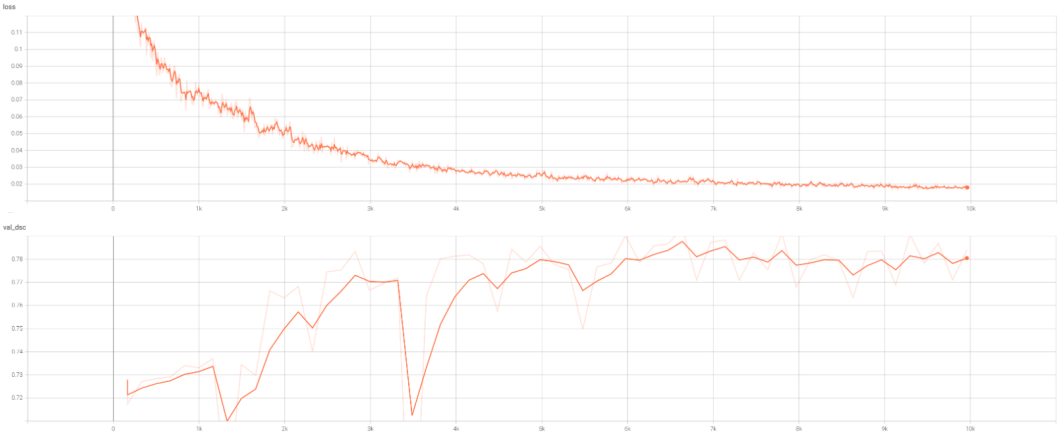
7.2 U-Net with SGD Optimizer and Binary Cross Entropy Loss



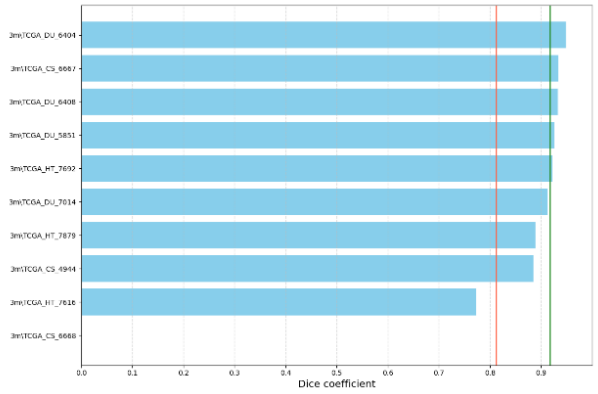
7.3 U-Net with SGD Optimizer and Log-Cosh Dice Loss



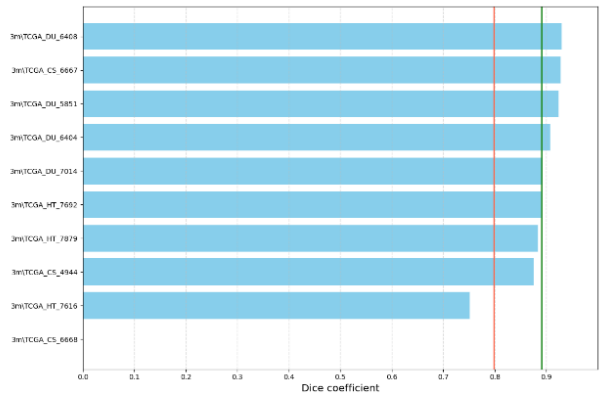
7.4 U-Net with SGD Optimizer and Shape-AwareLoss



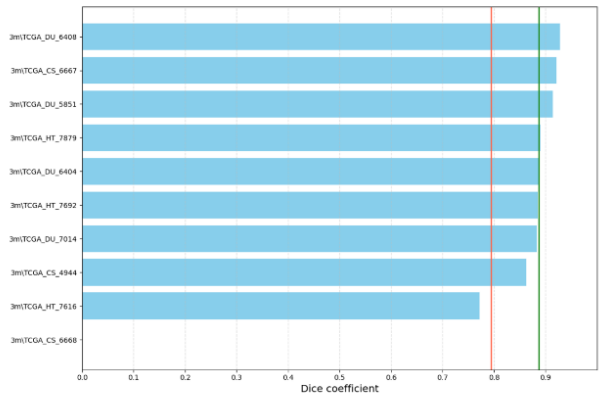
7.5 Inference Dice Similarity Coefficient



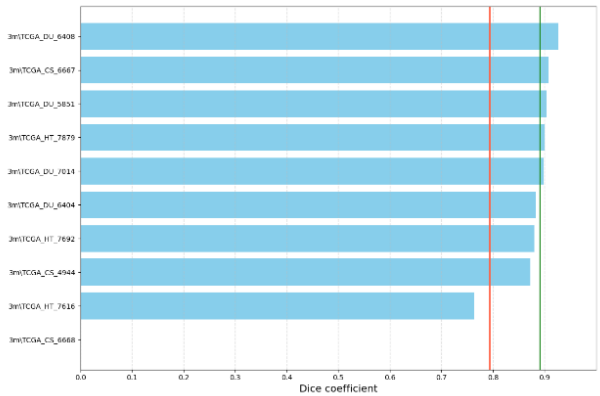
U-Net with SGD Optimizer and Dice Loss



U-Net with SGD Optimizer and BCE Loss

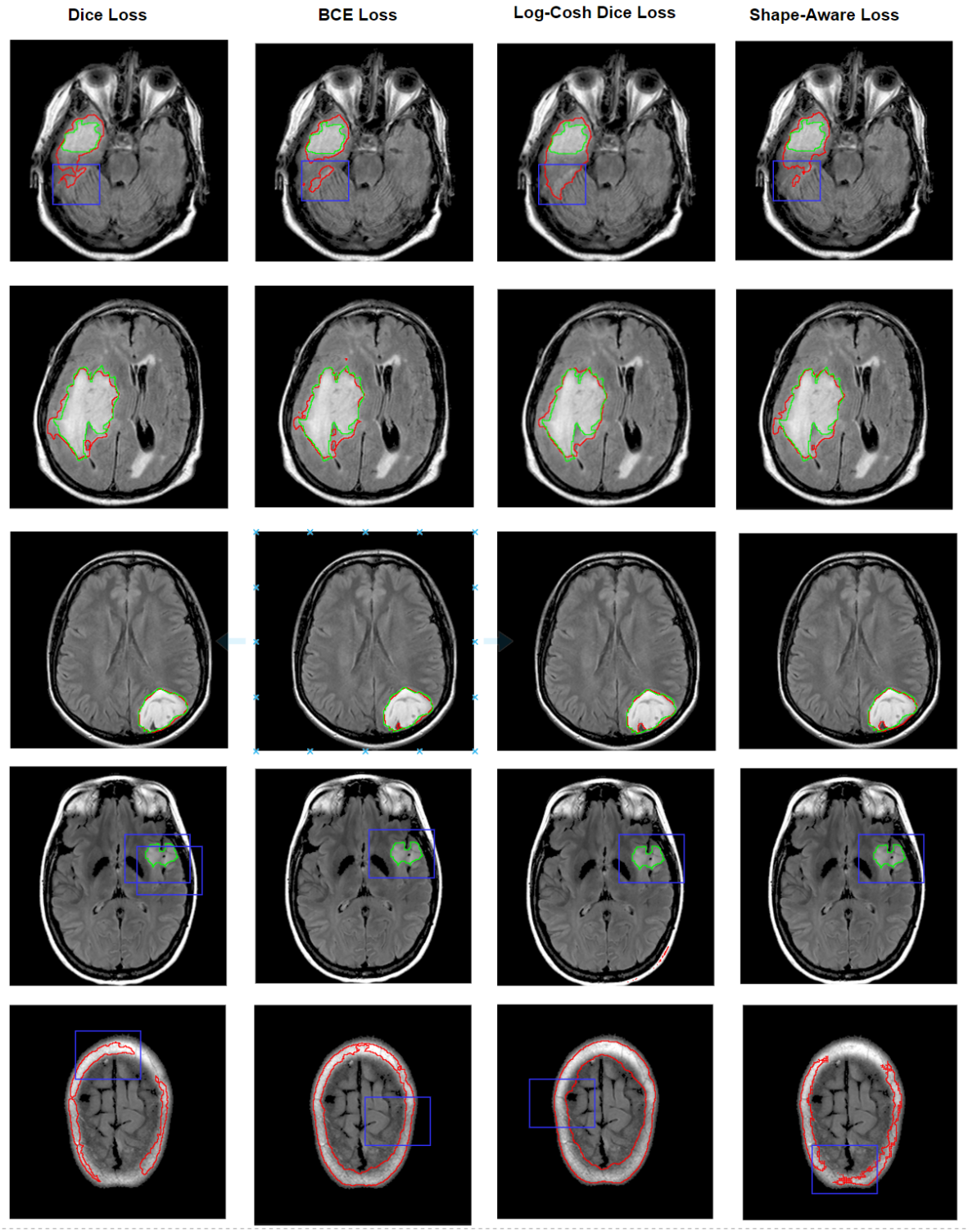


U-Net with SGD Optimizer and Log-Cosh Dice Loss



U-Net with SGD Optimizer and Shape-Aware Loss

7.6 Segmentation Examples



7.7 Training Data Comparison

Loss Function	Learning Rate	Max Epoch	Batch Size	Pipeline Configuration	Approximate Converge Time (in minute)	Traing Time (in minute)	Average Traing Time Per Epoch (in minute)	Best validation mean DSC
Dice Loss	0.001	100	20	U-Net Optimizer: SGD Momentum: 0.9 Weight Decay: 1e-4 Image Szie: 256 Argumentation Aug Scale: 0.05 Aug Angle: 15	187	235	2.35	0.812136
Binary Cross-Entropy Loss	0.001	70	20		129	164	2.34	0.798099
Log Cosh Dice Loss	0.001	60	20		105	133	2.08	0.794323
Shape-Aware Loss	0.001	60	20		113	121	2.01	0.793439

7.8 Analysis

1. Same as the U-Net models with Adam optimizer, Dice loss has provided the best segmentation results. The high DSC scores for Dice loss could also benefit from the iteration amount(greatest max epoch value among the experiments we run).
2. Judging from the validation loss curves and validation DSC curves, not much overfitting happens during the training process.
3. Learning rate for U-Net with SGD optimizer is much larger than with Adam optimizer(SGD: 0.001, Adam: 0.0001). SGD generally converges slower than Ad m, therefore without increasing the learning rate, it is hard for the network to converge within a reasonable amount of time.
4. BCE loss has the second best DSC scores when the optimizer is set to SGD. However, based on analysis in the previous section(U-Net with Adam) BCE by theory should not have such good performances especially when our data is quite unbalanced. This could be because the hyperparameter settings for our tests with SGD optimizer are not reasonable enough.
5. Most of the U-Net models with SGD optimizer only converge after an hour of training, whilst the Adam optimizer has a much faster convergence speed.
6. Adam optimizer has better segmentation results(around 10% higher on DSC scores) than SGD optimizer. We could also get a glimpse of this through the Dice Coefficient diagram for inference. Unlike Adam optimizer, with which the mean and median DSC scores for different volumes are close, these two statistical measure scores have comparatively large gaps in between with SGD optimizers. This suggests that there are some outlying observations or predictions happening with SGD optimizer. Detection for TCGA_CS_6688 and TCGA_HT_7616 are evidently much worse than the detection of other volumes.
7. For large or more-regular shape detection, we can see from the segmentation examples here, networks with SGD still have a promising result . However,
   It can also be noticed that many misclassifications(not detecting the lesion or false detecting normal areas as tui ors) happens. It seems that when the optimizer is set to SGD, the network tends to classify pixels more by the color/grayscale/single-channel contrast. This results in misbehavior when part of the image is too bright or too dark. The detected shape finds it hard to spot the correct area of a class or difficult to maintain shape details(edges). Overall, Adam seems to behave better than SGD in our case.

8 Conclusion for U-Net Network

8.1 Optimizers

Optimizers have a huge impact over the performance of a model. We say an optimizer is good if it is fast and efficient, and can generalize well(less overfitting). In our experiment, we discuss the debate on whether Adam or SGD is better under the given task with our dataset. It is generally difficult to select or design an optimizer that provides fast convergence speed and learning results. One interesting and dominant argument about optimizers is that SGD better generalizes than Adam[16]. However, it does not appear to be the case among the experiments we run. One potential explanation is the hyperparameter settings. Recent papers also argue that Adam could be better with proper hyperparameters.

From Section 6 and 7, we noticed that U-Net with Adam has higher DSC scores, regardless which loss function is used. Models with Adam optimizer also converge much faster than with SGD.

8.2 Loss Functions

A loss function is a mathematical equation that a deep learning architecture tries to minimize or optimize[17]. The choice of loss functions depend heavily on the nature of tha dataset and the goal of the task.

We compare Dice loss, BCE loss, Log-Cosh Dice loss and Shape-Aware loss in this report. With both optimizers, Dice loss and Shape-Aware loss have provided the best segmentation results with the highest DSC scores. As analyzed previously, with unbalanced data(tumor MRI etc.), BCE loss usually has weak performances.

For further investigation, we should test Weighted BCE and IoU loss since these two are also popular loss function choices for semantic segmentation.

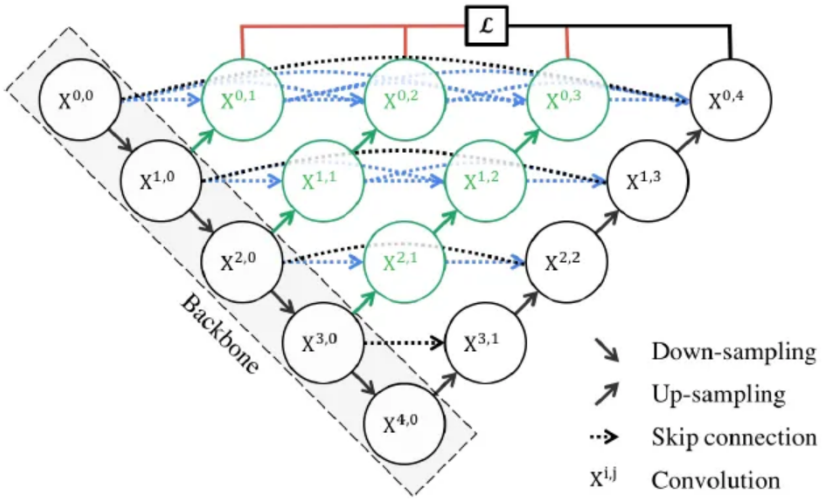
9 Entend: U-Net++(Nested U-Net)

For extended experiments, we also try to implement Nested U-Net and TransU-Net architecture. Due to the limited time, we mainly focus on testing Nested U-Net.

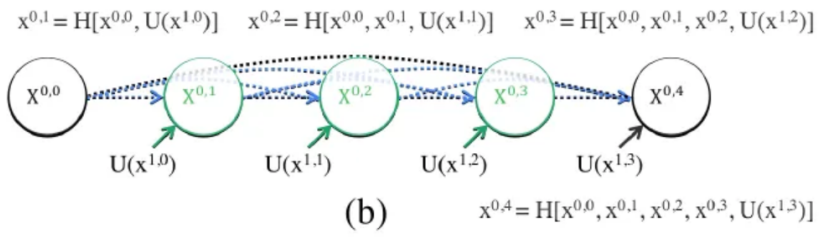
UNet++ uses the Dense block ideas from DenseNet to improve U-Net. It mainly has below differences compared to U-Net[18]:

1. Having convolution layers on skip pathways, which bridges the semantic gap between encoder and decoder feature maps.
2. Having dense skip connections on skip pathways, which improves gradient flow.
3. Having deep supervision, which enables model pruning and improves or in the worst case achieves comparable performance to using only one loss layer.

The main idea for Nested U-Net is to bridge the semantic gap between the feature maps of the encoder and decoder prior to fusion[18].

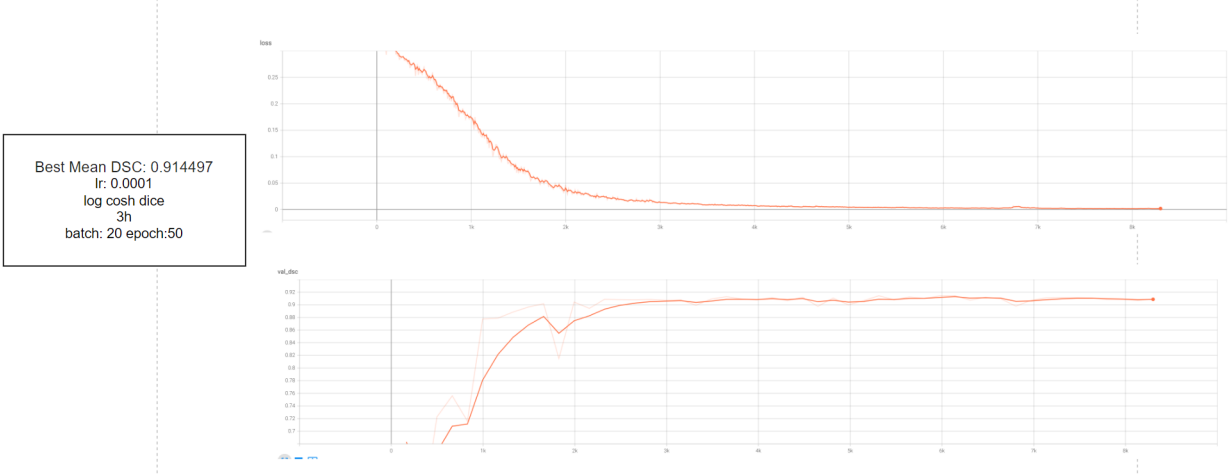


UNet++ Architecture



Re-designed Skip Pathways

Our experiment uses fast mode rather than deep supervision mode, due to the limited time budget. Below are diagrams for some of the experiments we run.



Architecture	Loss Function	Learning Rate	Max Epoch	Batch Size	Approximate Converge Time (in minute)	Traing Time (in minute)	Average Traing Time Per Epoch (in minute)	Best validation mean DSC
U-Net	Dice Loss	0.0001	100	16	97	248	1.48	0.915472
U-Net++	Dice Loss	0.0001	50	20	96	117	2.34	0.913730
U-Net++	Log Cosh Dice Loss	0.0001	70	16	92	175	2.5	0.914266
U-Net++	Log Cosh Dice Loss	0.0001	60	20	167	181	3.01	0.914497

The equipment is parallel running other large calculations. This is perhaps the reason why the training time is taking too much unreasonably

With the same loss function, there is improvement in U-Net++ learning results compared to U-Net, but not too much(about 0.2% to 0.3%). With more parameter tuning, we can assume that the segmentation can be further improved, however, not statistically significant.

In conclusion, the performance of a network architecture can be affected by hyperparameter settings and the dataset. Besides U-Net and Nested U-Net, there are other many U-Net structures and networks that aim at semantic segmentation, and some may have good training results and some may not, depending on the data and the task itself.

10 Summary

In our experiments, we compared some common-used loss functions and optimizers and how they affect the U-Net behaviors under our task. Dice similarity coefficient is used to statistically analyze the learning results. The highest DSC score 0.9155 with U-Net structure is with Dice loss and Adam optimizer.

With U-Net architecture, Dice loss and Log-Cosh Dice loss have proved their advantages at region detection and boundary detection. Throughout the experiments, Adam optimizer also provided faster convergence speed and better segmentation results than SGD optimizer. However, a conclusion of which optimizer is better still could not be made since we only run limited experiments, and the behaviors of an optimizer could be hugely affected by the hyperparameter settings. For further investigation, we should do more experiments, perhaps with GridSearch and so on to find the best hyperparameter settings for each optimizer and then compare the segmentation results accordingly. It would also be interesting if we could introduce ways to quantify and compare the stability of a model with different optimizers.

With the same loss function, there is improvement in U-Net++ learning results compared to U-Net, but not too much(about 0.2% to 0.3%). With more parameter tuning, we can assume that the segmentation can be further improved, however, not statistically significant.

In conclusion, the performance of a network architecture can be affected by hyperparameter settings and the dataset. Besides U-Net and Nested U-Net, there are other many U-Net structures and networks that aim at semantic segmentation, and some may have good training results and some may not, depending on the data and the task itself.

Both the U-Net and U-Net++ architectures have comparatively good performances on segmentation and edge maintenance, but still fail to detect accurate shapes especially when the target area is small or when the shape has edges with many turns. To improve this, we could test TransU-Net, or try to combine with some adaptive methods to detect these poor-behaving areas and make them tokens with higher importance or semantic meaning.

11 References

[1] Brain MRI segmentation – Brain MRI images together with manual FLAIR abnormality segmentation masks https://www.kaggle.com/datasets/mateuszbuda/lgg-mri-segmentation

[2] Ronneberger, O., Fischer, P. and Brox, T. (2015) 'U-Net: Convolutional Networks for Biomedical Image Segmentation', Lecture Notes in Computer Science, pp. 234–241. doi:10.1007/978-3-319-24574-4_28.

[3] Medical Image Segmentation: A Complete Guide https://encord.com/blog/medical-image-segmentation/#:~:text=Medical%20image%20segmentation%20involves%20the,Resonance%20Imaging%20(MRI)%20files.

[4] Detailed Explanation of Unet Theory With Code https://www.zhihu.com/question/269914775

[5] Data augmentation https://en.wikipedia.org/wiki/Data_augmentation

[6] Data Augmentation for Semantic Segmentation — Deep Learning — Idiot Developer https://nikhilroxtomar.medium.com/data-augmentation-for-semantic-segmentation-deep-learning-idiot-developer-e2b58ef5232f

[7] Alomar, K., Aysel, H.I. and Cai, X. (2023) 'Data Augmentation in classification and segmentation: A survey and new strategies', Journal of Imaging, 9(2), p. 46. doi:10.3390/jimaging9020046.

[8] 3 Common Loss Functions for Image Segmentation https://dev.to/_aadidev/3-common-loss-functions-for-image-segmentation-545o

[9] Jadon, S. (2020) 'A survey of loss functions for semantic segmentation', 2020 IEEE Conference on Computational Intelligence in Bioinformatics and Computational Biology (CIBCB) [Preprint]. doi:10.1109/cibcb48159.2020.9277638.

[10] Dice Coefficient! What is it? https://medium.com/@lathashreeh/dice-coefficient-what-is-it-ff090ec97bda

[11] Learning Day 57/Practical 5: Loss function — CrossEntropyLoss vs BCELoss in Pytorch; Softmax vs sigmoid; Loss calculation https://medium.com/dejunhuang/learning-day-57-practical-5-loss-function-crossentropyloss-vs-bceloss-in-pytorch-softmax-vs-bd866c8a0d23

[12] issue with early-stopping on f1 score with imbalanced data https://datascience.stackexchange.com/questions/41340/issue-with-early-stopping-on-f1-score-with-imbalanced-data

[13] Understanding Evaluation Metrics in Medical Image Segmentation https://medium.com/mlearning-ai/understanding-evaluation-metrics-in-medical-image-segmentation-d289a373a3f

[14] Mean or median? Choose based on the decision, not the distribution
https://towardsdatascience.com/mean-or-median-choose-based-on-the-decision-not-the-distribution-f951215c1376

[15] Dice Coefficient! What is it?
https://medium.com/@lathashreeh/dice-coefficient-what-is-it-ff090ec97bda#:~:text=A%20high%20Dice%20coefficient%20value,value%20indicates%20p%20oor%20segmentation%20performance.

[16] A 2021 Guide to improving CNNs-Optimizers: Adam vs SGD
https://medium.com/geekculture/a-2021-guide-to-improving-cnns-optimizers-adam-vs-sgd-495848ac6008

[17] Role of Choosing Correct Loss Function
https://medium.com/aiguys/role-of-choosing-correct-loss-function-b58c5ad28e02

[18] Review: UNet++ — A Nested U-Net Architecture (Biomedical Image Segmentation)
https://sh-tsang.medium.com/review-unet-a-nested-u-net-architecture-biomedical-image-segmentation-57be56859b20#:~:text=Outperforms%20U%2DNet%20and%20Wide%20U%2DNet&text=UNet%2B%2B%20differs%20from%20the,pathways%2C%20which%20improves%20gradient%20flow.