Brain Tumor Image Analysis using a Deep Learning Framework Pipeline Built on Monai

Team 8: Anirudh Suri and Varsha Srinivasan

Abstract

A significant problem that scientists encounter when working with biological datasets has to do with the complexities of labeling the data. Since the intricate data arises from a complex biological phenomena, it is quite difficult to generate features and to label them. In most cases, human intervention is required to assign labels. Moreover, annotating these datasets requires the insight of an expert to decipher and distinguish between these features. This becomes a rate-limiting step when it comes to computational research and discovery of solutions to biological problems. Another issue that arises when deploying machine learning models for biological datasets is that the pre-trained models become highly redundant due to the extremely high degree of variation in biological samples. Thus, building one model for either sample won't be effective due to the large amount of generalization required to accommodate both samples. One novel trend that has been identified as a solution to this problem is self-supervised learning. The complex neural network model identifies features on its own accord rather than calculating weights for downstream analysis. This study seeks to explore a solution to this problem by analyzing MRI scans of brain tumor patients (brain tumor segmentation - gliomas) coupled with context restoration. The data was obtained from the Medical Segmentation Decathlon webpage and the Monai DL framework was used. A segmentation model was created and evaluated using the Dice coefficient and deep learning was used to train this model.

Introduction

Generating features and labelling the data in biological datasets can be a challenging task due to the complex nature of biological phenomena. The accuracy and the output of any machine learning model essentially boils down to the quantity and quality of the data used for training. Acquiring large datasets is usually not a problem, so the quantity part is not something to worry about. However, making sure that the data is labelled accurately and has perfect features is a challenging task. It also does not help that in general, the amount of unlabeled data for biological datasets is significantly more than the labeled data.

In most cases, researchers are required to manually assign labels, which is close to impossible in massive datasets. This, owing to human nature, can result in the features being faulty. Differentiating between features is also a huge task when it comes to annotating biological

datasets. This step can significantly slow down the entire study because further analyses cannot be made without performing this step.

Another problem when it comes to deploying machine learning models is that pre-trained models have the tendency to become highly redundant due to the vast variation in biological samples. For example, two people may have the same immune profile but lack any disease. Thus, one model cannot be effective for both samples. This would require a lot of generalization which can change the model into something else.

This has been a problem computational biologists have been trying to overcome for a long time. In thi study, we explore a possible solution to this problem by analyzing the MRI scans of brain tumor patients (brain tumor segmentation - gliomas).

There are four main steps involved in this process:

1. Preprocessing - Load the dataset, crop the brain region, normalize the intensity range, and resample to a consistent voxel spacing
2. Dataset preparation - Create training, validation, and test datasets from the preprocessed images and labels
3. Model training - Define and train a 3D U-Net model using the training dataset and a dice loss function
4. Inference - Use the trained model to perform segmentation on new unseen images from the test dataset.

Changes: We realised we would need a lot of compute in order for context restoring the images. Additionally we found it it difficult to corrupt the images (all the images). We still wanted to stay relevant to the to the topic and work with medical image samples. As a result of which we decided to build an end to end workflow using the Monai package. This project aims at loading, pre-processing, and training a 3D segmentation model and then evaluating the model.

Related Work

A recent novel trend that has been gaining popularity and has been identified as a potential solution to this issue is self-supervised learning. The convolutional neural network model identifies features of its own accord instead of calculating weights for downstream analysis (classification, segmentation, etc.).

This study was inspired by the Multimodal Brain Tumour (BRaTS) Segmentation Challenge conducted by the University of Pennsylvania's Medical School in collaboration with the Computer Science Dept. The outcomes enable 1) Segmentation of intrinsically heterogeneous (shape, histology) MRI scans and 2) prediction of the patient's OS. Most of our referencing and

knowledge stems from the contributors of this organization/ initiative, along with other resources such as research and review papers published in this domain.

Data

The data required was sourced from the Medical Segmentation Decathlon webpage and the Monai DL framework was used. The data obtained was in hdf5 format. We also got a json file that describes and defines the training and testing datasets.

This Decathlon dataset contains the BraTS data that was downloaded from Google Drive using the link provided in the webpage. To reduce high memory usage, the cache rate was set to 0. It is also to be noted that it is sufficient to download the data just once for training and not for validation.

In this dataset, brain tumor labels had to be converted into multi-label segmentations. In the BraTS dataset, there are four different tumor classes:

1. Edema (Label #1)
2. Non-enhancing tumor core (Label #2)
3. Enhancing tumor core (Label #3)
4. Necrosis (Label #4)

For imaging, these medical classes were converted into distinct channels in order to increase accuracy, diagnosis, treatment and prognosis. These channels were:

1. Tumor core (Labels #2 and #3)
2. Whole tumor (Labels #1, #2 and #3)
3. Enhancing tumor (Label #2)

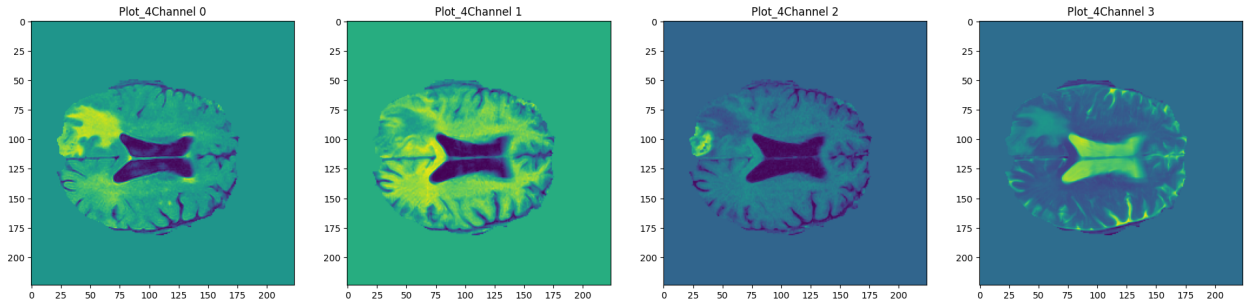
In summary, the transform takes in the original label data containing four classes and creates three new binary channels representing tumor core, whole tumor and enhancing tumor.

For training and validation, voxels or pixels can have one or more channels that represent different features of the data. The channels had to be organized as the first dimension of the image tensor, followed by spatial dimensions such as height, width and depth. Multi-class conversion was also performed. Right anterior superior (RAS) orientation ensures that all the image data has the same orientation irrespective of the orientation they were originally captured in. This makes the model easier to assess and more accurate.

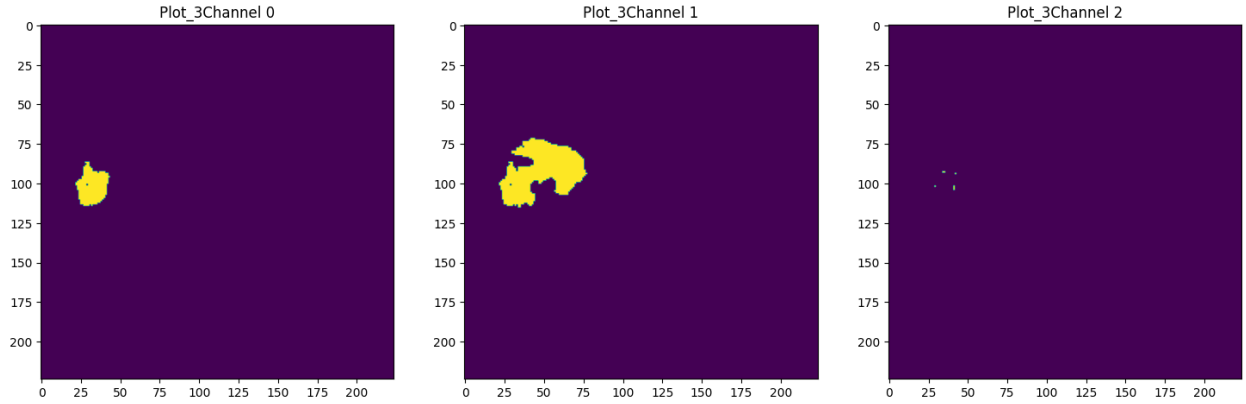
Following this transformation, the data was trained as follows:

1. Rescaling pixel spacing
2. Random cropping of images
3. Random flipping of images

4. Intensity normalization (scaling and shifting)



Sample plot of random image data and labels from the training set



Three channels for the same random sample

Method

The model was first created using a Pytorch by incorporating the Dice loss function and the Adam optimizer. The model also defined an evaluation metric for the Dice co-efficient which measures the overlap between the predicted and ground truth segmentation masks. The model architecture has four downsampling stages and and three upsampling stages.

The Dice loss function was used to calculate the loss between the predicted and ground truth segmentation masks. The model was optimized using the Adam optimizer and the learning rate was scheduled using CosineAnnealingLR. It reduced the learning rate according to the cosine annealing schedule with a maximum number of epochs of 20.

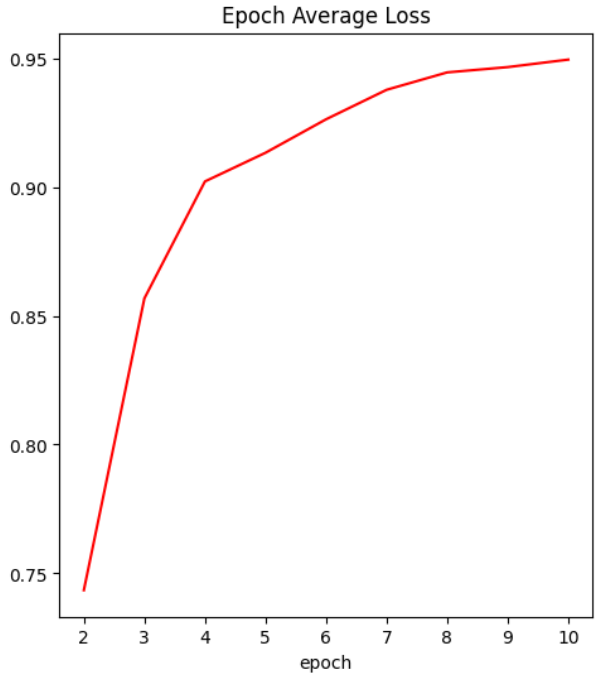
A post-processing transform was designed by applying the sigmoid activation function to the output. The sliding window inference method was used for inference. Finally, gradient scaling was deployed to prevent overflow and underflow of tensor values.

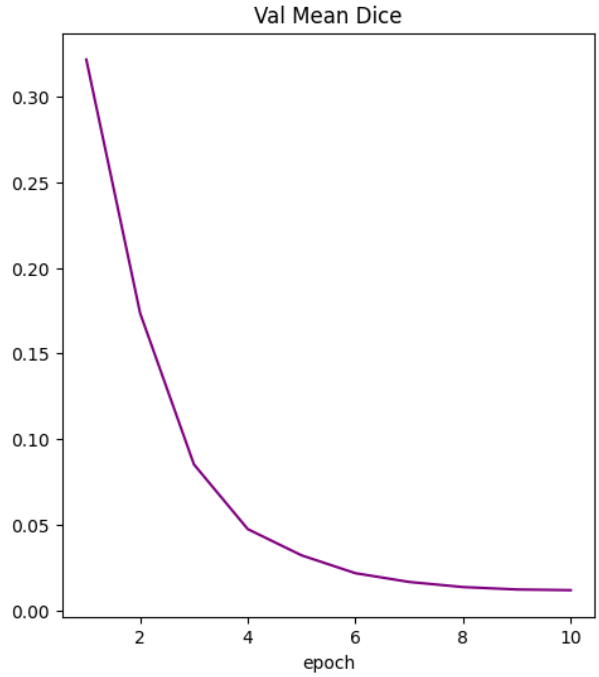
Deep learning was deployed using Pytorch in order to train this model. We also evaluated its performance on a validation set at regular intervals, and saved the model weights if the metric improves.
The model was iterated over a number of epochs and trained. For each batch, we sent it to the device and reset the optimizer. Following this, forward propagation, loss calculation and backward propagation were performed using the input data and labels to obtain the outputs and loss. AMP was used to improve the speed and efficiency of training by using lower precision for some calculations without sacrificing accuracy and the epoch loss was computed.

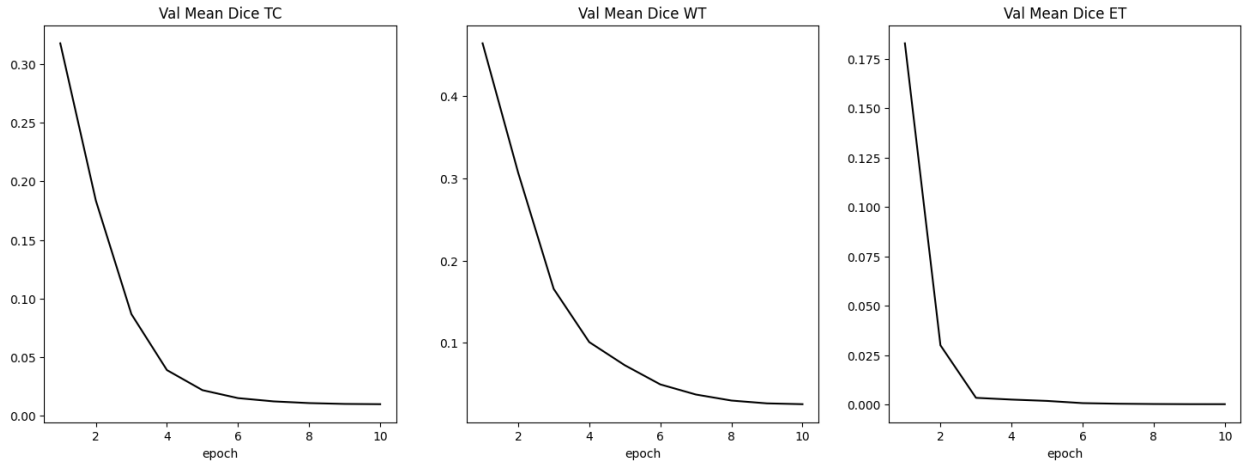
Experiments

The Dice coefficient was used as an evaluation metric and graphs were plotted to test the same

Plot #1- Delta loss value (y-axis) over epochs (x-axis), the trend line depicts the average loss per epoch.
Plot #2- Mean Dice Coefficient, measures the similarity between the predicted and the ground truth for the segmentation task for the validation set over the epochs defined. The trend line represents the mean Dice coefficient over the epochs
Plot #3, #4, #5- These plots depicts the the mean Dice coefficient for the various tumour stages, i.e Tumor Core (tc), Whole Tumor (wt), and enhancing tumor (et) over the epochs



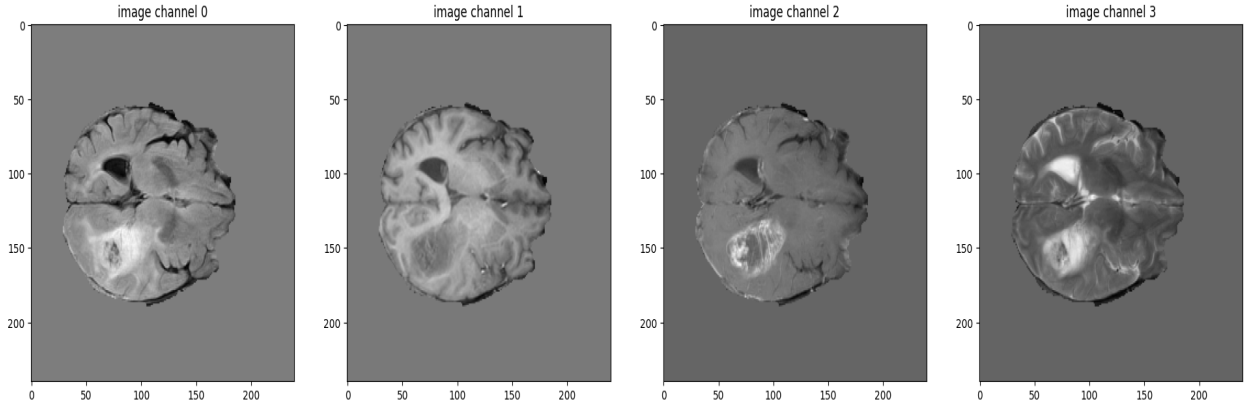




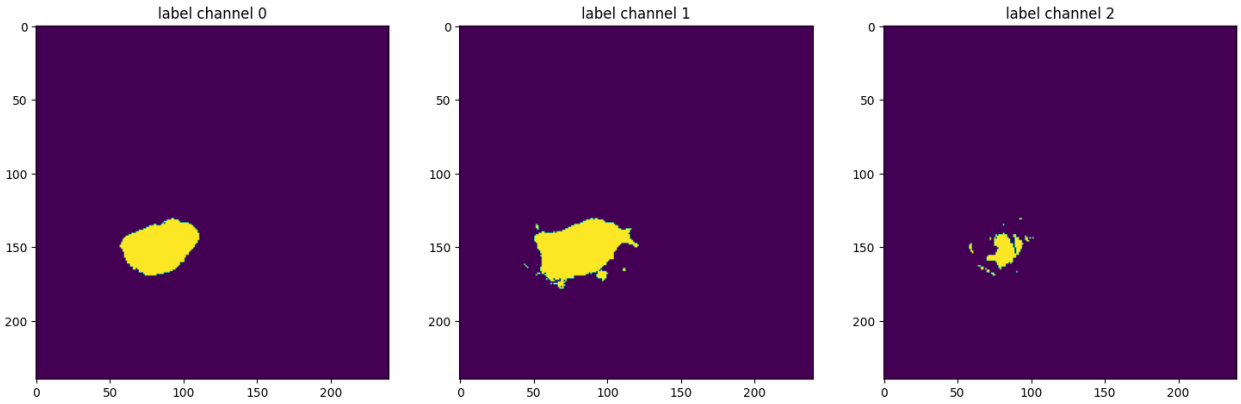
These plots are significant as they help us to evaluate the model's performance during training and even adjust the hyperparameters of the model to increase the efficiency as well as accuracy. The trend depicted by the Dice mean coefficient helps us to evaluate the accuracy of segmentation. Loss value plots helps us to analyse the optimization of the model.

Conclusion and Discussion

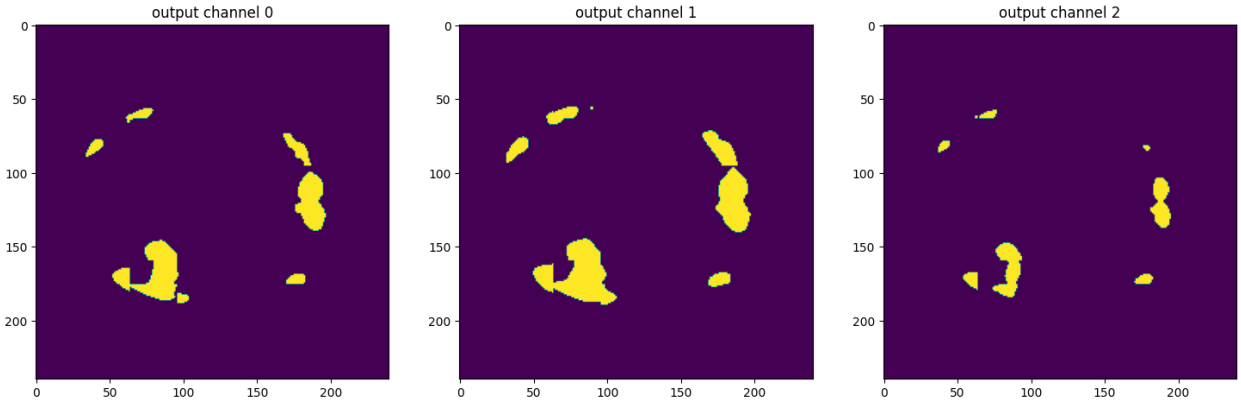
We also used the trained model applied to a random sample from the validation set to generate 3 types of outputs.
#1- Original 4 channel image



#2- 3 channel converted representing the ground truth of the sample. Each channel representing tc, wt, and et.



#3- Model output: 3 channel of the predicted segmentation of the sample sample



The output data generated helps us validate the model and its prediction using the dice coefficient metric.

The output:

1. metric_tc: 0.5578
2. metric_wt: 0.7104
3. metric_et: 0.3184

This implies that the model is able to segment 52.88% of the samples accurately as the dice coefficient for the entire validation set is 0.5288. The model also evaluated across 3 different tumour regions defined by the channels- enhancing tumor (ET), tumor core (TC), and whole tumor (WT). The model is best suited for segmenting whole tumour samples.

By tuning the hyperparameters and training the model over a higher number of epochs can increase the accuracy. But owing to computing constraints we decided to focus on implemented a pipeline that included segmentation rather than focusing solely on the segmentation.

This pipeline was put together from several resources and documentation, and can be scaled up for longer epochs and a greater sample size.

Contribution

This project consists of equal contribution by both members. Each task was tackled together.

The notebook where this project was done can be found:
https://colab.research.google.com/drive/1HH5-fbcaA--hFsmG0k7Ivd8iRplvc5Ux#scrollTo=-r423dy8H3wa