Brain Tumor Segmentation and Survival Prediction using Deep Neural Networks

Shalabh Gupta **Vrinda Jindal**

June 28, 2020

Abstract

In this project, we approach the problem of segmenting MRI images, i.e. classifying tumor and non tumor parts of brain, and using this information , carry out survival prediction of patients undergoing treatment. Our work on segmentation is based on [1], [2], and [3] while work on survival prediction is inspired from [4]. Our implementation, in keras module of tensorflow framework in python, can be found at this Github Repo.

1 Introduction

Quantitative assessment of brain tumors provides valuable information and therefore constitutes an essential part of diagnostic procedures. While manual segmentation is tedious, time consuming and subjective, automatic segmentation is attractive in this context as it allows for faster, more objective and potentially more accurate description of relevant tumor parameters of the brain.
The goal of brain tumor segmentation is to detect the location and extension of the tumor regions, namely active tumorous tissue (vascularized or not), necrotic tissue, and edema (swelling near the tumor).
Brain tumor segmentation results provide the volume, shape, and localization of brain tumors, which are crucial for brain tumor diagnosis and monitoring. Therefore we also attempt at predicting survival rate of patients given their MRI images.

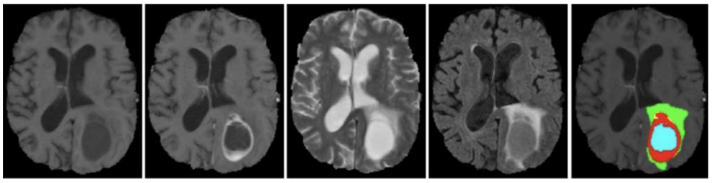


Figure 1: Segmented Ground Truth along with the 4 modalities

2 Dataset

The Brain Tumor Segmentation (BraTS) challenge held annually is aimed at developing new and improved solutions to the problem. We use BraTS 2018 data which consists of 210 HGG(High Grade Glioma) images and 75 LGG(Low Grade Glioma) along with survival dataset for 163 patients. We use only HGG images.
Since glioblastomas are infiltrative tumors, their borders are often fuzzy and hard to distinguish from healthy tissues. As a solution, more than one MRI modality is often employed. The 4 are: T1 (spin-lattice relaxation) , T1-contrasted (T1C) , T2 (spin-spin relaxation), Fluid attenuation inversion recovery (FLAIR). Each modality is a 3D image with size (240,240,155).
Survival dataset is a csv file containing patient ids, age and days survived. The Ground Truth voxels belong to one of the following 4 categories:

- Healthy voxels (label 0)
- Necrosis and non-enhancing tumor (label 1)
- Edema (label 2)
- active/enhancing tumor (label 4)

3 Class Imbalance Problem

One of the key features of the BraTS dataset is that it suffers from the Problem of extreme class imbalance.

Classes	**Occurrence**
healthy voxel (0)	0.9874861111
Necrosis and non-enhancing tumor (1)	0.003045362903
Edema (2)	0.005941980287
active / enhancing tumor (4)	0.003526545699

Table 1: Fraction of Occurrence of Each Class

3.1 Why class imbalance is a problem?

This immensely skewed class distribution leads to several problems, prominently trivial solutions (labelling all the voxels as 0 and achieving a categorical-cross-entropy accuracy of 98%)

3.2 Solutions

- Using Multi-Class Soft-Dice loss (1)
- Class Weights Utility : Different weights to penalize wrong labels. Sample re-weighting where foreground regions are given more importance than background ones during learning.
- Training 2D models on only those slices which have pixels belonging to all the classes.

Links to datasets:

- BraTS 2017.
- BraTS 2018

4 Architectures and Results

4.1 UNet 3D

UNet is one of the most popular architectures designed for BioMedical Image Segmentation. It consists of an ENCODER(contraction) and a DECODER(expansion) network. Most part of the architecture is inspired from [1].

4.1.1 Architecture

In the contraction pathway of our architecture , we design a convolutional block at each level followed by a MaxPooling operation with stride = 2 to decrease the resolution by 2.

The conv block consists of two successive $3\times3\times3$, each followed by Batch Normalization and 'relu' non linearity. Number of filters are constant at a particular level while being double of what they were at the immediate upper level.

The expansion pathway consists of transposed convolutions to up-sample layers and at the same time, learn parameters to produce a high resolution image from a low one. Also feature maps from contraction pathway at the same level are concatenated with feature maps of expansion pathway. The network takes input blocks of size $128 \times 128 \times 128 \times 4$ and outputs softmax $128 \times 128 \times 128 \times 4$. We crop the middle $128 \times 128 \times 128$ sized block from all the modalities.

4.1.2 Explanation

Encoder networks capture increasingly abstract and high level features associated with the context of the MRI images by doubling the feature maps after each level. The number of feature maps in the Decoder network are kept enough to carry the sufficient context information back to high resolution segmentations.

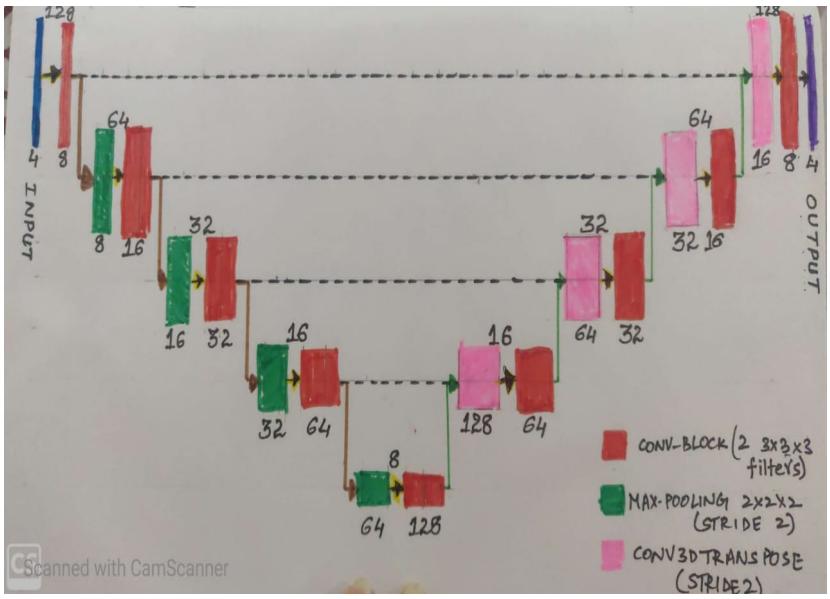


Figure 2: Our UNet architecture

Using Batch Normalization helps in stabilizing inputs and control vanishing and exploding gradient, hence helping the network converge.

The concatenation of feature maps from encoder to decoder at the same level is done in order to *not forget* important information encoded by the ENCODER,helping in precise localization and accurate boundaries.

4.1.3 Training Details

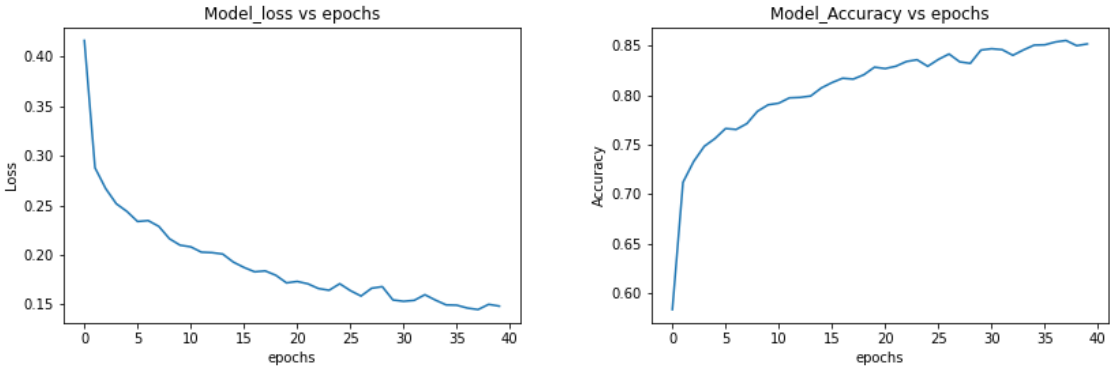
The network was trained with dropout rate set to 0.2 using Adam Optimiser and multi class dice loss as loss function. Dice loss is represented as:

$$DiceLoss = 1 - \frac{2}{|K|} \sum_{k \in K} \frac{\sum_i u_{i,k} v_{i,k}}{\sum_i u_{i,k}^2 + \sum_i v_{i,k}^2} \tag{1}$$

where u is the softmax output of the network and v is a one hot encoding of the ground truth segmentation map. Both u and v have shape I by K with $i \in I$ being the voxels in the training patch and $k \in K$ being the classes. $u_{i,k}$ and $v_{i,k}$ denote the softmax output and ground truth for class k at voxel i, respectively.

Segmentation task is often associated with using Categorical CrossEntropy as loss function. But with a very imbalanced dataset(Label 0 dominating the ground truth) , dice loss is preferred as it averages over all 4 classes. Also different penalizing weights have been given to each class based on their frequency in our implementation to take care of the imbalance problem. The 2D models have been trained on only those slices which have at least one pixel of all 4 labels.

Accuracy metric is set to be Dice Coefficient(or Dice score) which is 1− dice loss. With learning rate set to 0.001 and decay to 1e-7, our network was trained on 45 epochs on Kaggle GPU.



(a) Loss vs epochs (b) Accuracy vs epochs

Figure 3: Training plots

4.1.4 Evaluation/Prediction and Results

We tested our model on the 30 HGG images we had separated, achieving a dice score of 0.735.

Classes	Sensitivity	Specificity
0	0.99	0.79
1	0.45	0.99
2	0.58	0.99
4	0.78	0.99

Table 2: Metrics

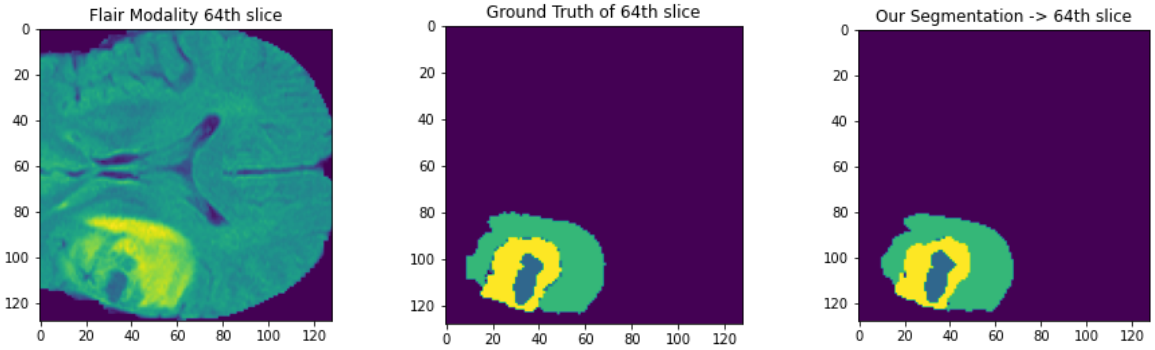


Figure 4: UNet 3D Results on one of the test images

4.2 VNet 3D

VNet is another approach that uses Volumetric Convolutions instead to processing 2D slices from the volumetric modalities. This Vnet architecture has been adopted from [2].

4.2.1 Architecture

This model architecture is quite similar to Unet with some modifications that are aimed at improving the performances of the 3D UNet. Each convolutional block in the compression and expansion path learns a residual function due to element wise addition of layer 0 input with layer 2 output of the convolutional block.

MaxPooling Operations in the contraction path have been replaced by 3D Convolutions of size $2 \times 2 \times 2$ and stride 2. Convolutions in Convolutional block composed of kernel size $5 \times 5 \times 5$ to increase the receptive field being captured. Number of filters are doubled everytime we go down in the downsampling convolution operation.

The remaining architecture is similar to that of the UNet. The network takes input blocks of size $128 \times 128 \times 128 \times 4$ and outputs softmax $128 \times 128 \times 128 \times 4$.

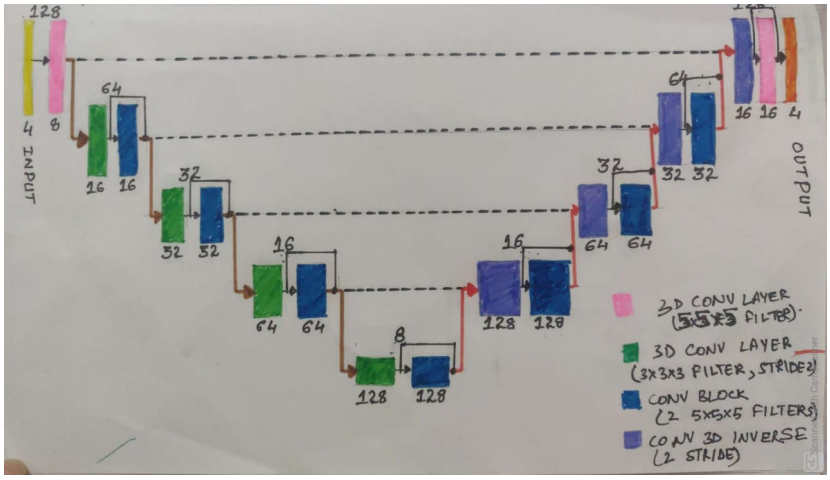


Figure 5: Our VNet architecture

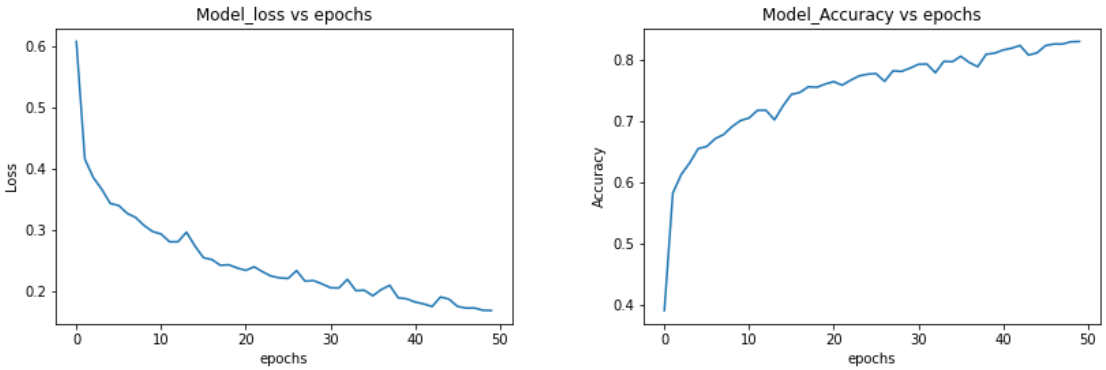
4.2.2 Explanation

Residual networks are famous for learning an identity mapping and mitigating the vanishing gradient problem due to smooth flow of gradients through skip connections. Thus they help train the network faster while also permitting features from previous layers to be carried forward efficiently.

Some recent works on CNNs discourage usage of Pooling operations based on the fact that the the spatial dimensionality reduction performed by pooling which makes covering larger parts of the input in higher layers possible can be replaced by Convolutions with strides to do the same while learning parameters as well.

4.2.3 Training Details

The network was trained with dropout rate set to 0.15 using Adam Optimiser and multi class dice loss as loss function. Accuracy metric is set to be Dice Coefficient(or Dice score) which is 1− dice loss. With learning rate set to 0.00095 and decay to 2e-7, our network was trained on 55 epochs on Kaggle GPU.



(a) Loss vs epochs

(b) Dice Score vs epochs

Figure 6: Training plots

4.2.4 Evaluation/Prediction and Results

We tested our model on the 30 HGG images we had separated, achieving a dice score of 0.68.

Classes	Sensitivity	Specificity
0	0.99	0.76
1	0.32	0.99
2	0.63	0.99
4	0.71	0.99

Table 3: VNet3D Metrics

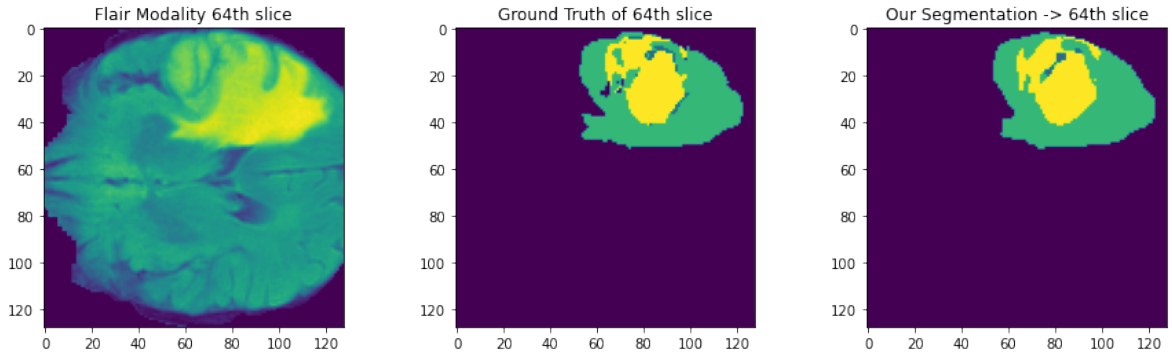


Figure 7: VNet 3D Results on one of the test images

4.3 2D UNets Integration along 3 axes

While features in 3D modalities are more natural and easier for 3D architectures like the ones described above to capture , 2D architectures are preferred as 3D CNNs have large memory and training time requirements. This approach explores methods to segment the images by combining features along the 3 axes . This idea of integrating FCNs from the 3 views has been adopted from [3]. In our implementation, we use Unets instead of FCNs (that do patch-based segmentation).

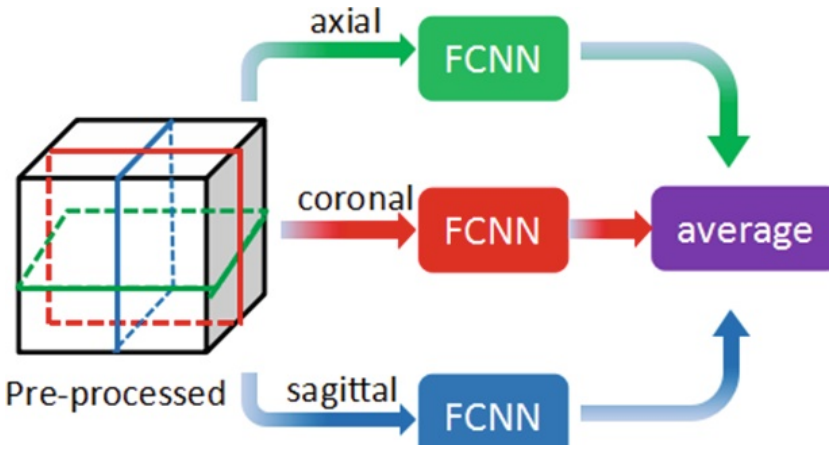


Figure 8: Combination of 2D FCNs

4.3.1 Architecture

To improve the segmentation efficiency, multiple UNet models, each of which is trained to segment slices in different views, can be combined by fusing their segmentation results. In this way, each UNet model can still be tested by 2D slices, guaranteeing the testing efficiency.

2D UNet architecture for slices of size 240×240 is symmetric as shown below.It take input blocks of size $240 \times 240 \times 4$ and output softmax activations of size $240 \times 240 \times 4$. For slices of size 240×155 , we design a different UNet architecture ourselves which has

many asymmetric convolutions along the 2 dimensions.This model takes input blocks of size $240 \times 155 \times 4$ and outputs softmax activations of the same size.

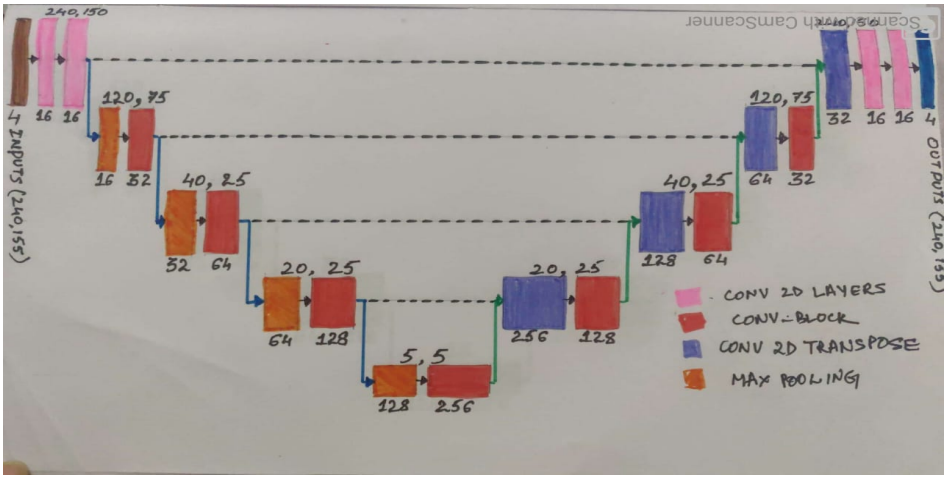


Figure 9: Our 2D-UNet architecture for segmentation along axis 1,2 (4 class)

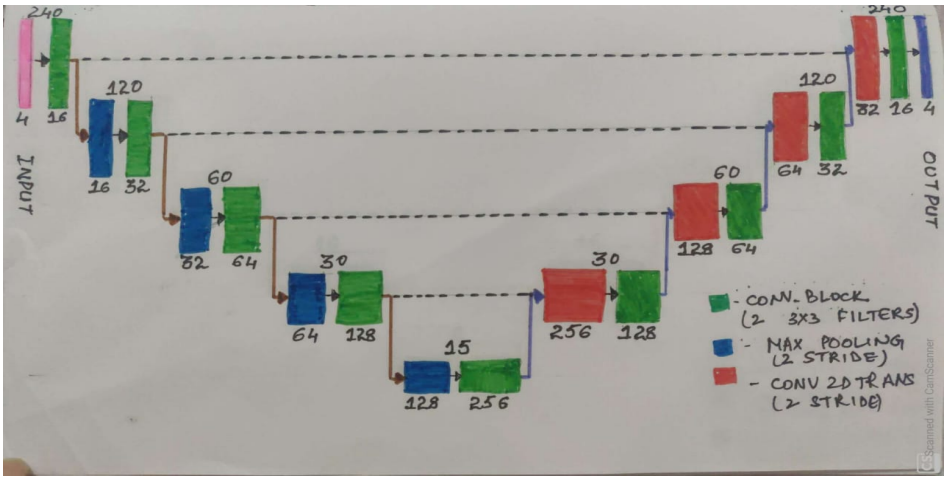


Figure 10: Our 2D-UNet architecture for segmentation along axis 3 (4 class)

We perform both 2 class segmentation(Whole tumor and no tumor region) and 4 class segmentation(original problem statement) using this approach.

4.3.2 Explanation

To Fuse the segmentation results in 3 different views, we use the following technique :-

Let P_a, P_b, P_c denote the outputs of the 3 UNets respectively along their corresponding view. $P_a = P_a^0$, P_a^1 , P_a^2 , P_a^4 , $P_b = P_b^0$, P_b^1 , P_b^2 , P_b^4 , $P_c = P_c^0$, P_c^1 , P_c^2 , P_c^4 . The intensity value of each voxel in P_a^u , P_b^u , and P_c^u denotes the predicted probability of assigning label u to this voxel. We fuse these 3 segmentation results by averaging, that

is $P = (P_a + P_b + P_c)/3$. Now, we can find the max of P^0 , P^1 , P^2 , P^4 for each voxel to assign it a label. This is how we predict finally.

4.3.3 Training Details

Adam Optimiser , learning rate = 0.00095 , learning rate decay = 2e-7 , Soft Dice Loss , Dropout = 0.15 , epochs = 50 for all three models. 2 class dice loss in case of 2 class segmentation and 4 class dice loss in case of 4 class segmentation. Accuracy metric is Dice Coefficient again.

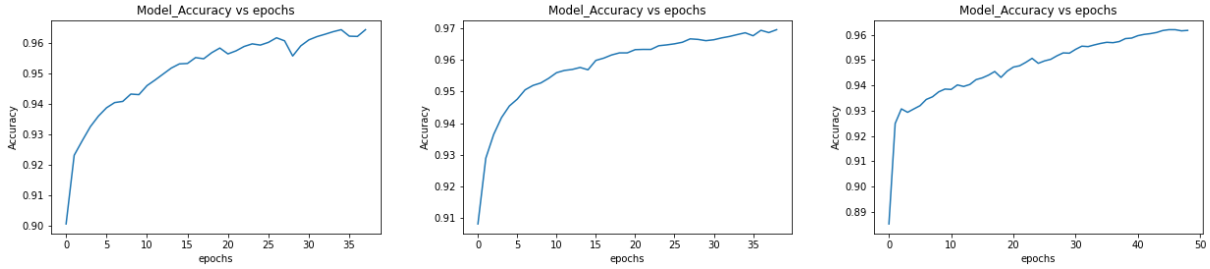


Figure 11: Dice Score(accuracy) vs epochs for all three axes (2 class segmentation)

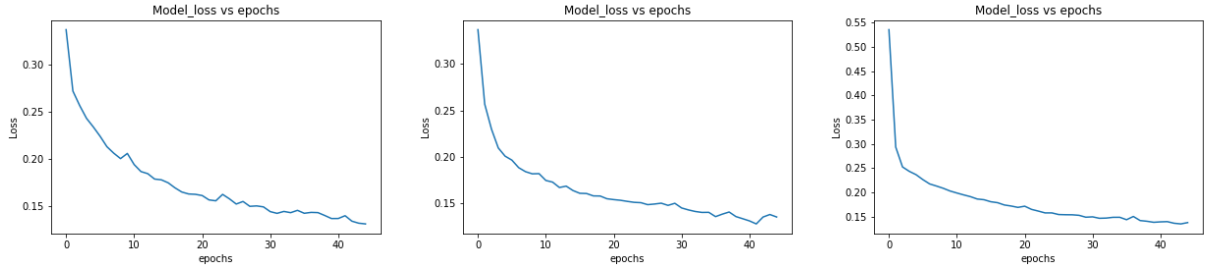


Figure 12: Dice Loss(loss) vs epochs for all three axes (4 class segmentation)

4.3.4 Evaluation/Prediction and Results

Models/Metrics	Dice Coefficient(Accuracy)	Dice Loss
2D Axis 1	0.50	0.33
2D Axis 2	0.49	0.35
2D Axis 3	0.71	0.29
Axis Integrate	0.756	0.244

Table 4: 4 class Models

Models/Metrics	Dice Coefficient(Accuracy)
2D Axis 1 (Sagittal)	0.66
2D Axis 2 (Axial)	0.62
2D Axis 3 (Coronal)	0.67
Axis Integrate	0.713

Table 5: 4 class Models by Zhao Et Al. (without CRF models) [3]

Classes	Sensitivity	Specificity
0	0.99	0.80
1	0.37	0.99
2	0.63	0.99
4	0.84	0.99

Table 6: Metrics for 4class Model

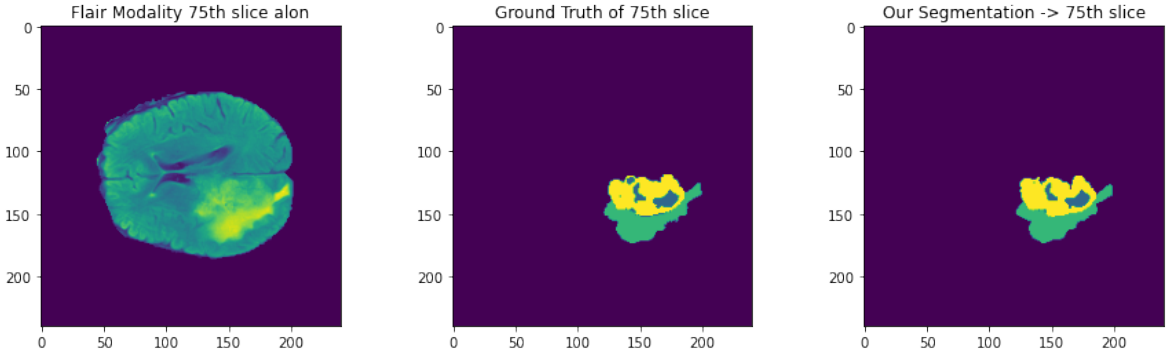


Figure 13: 2D axes integration results for 4 classes on one of the test images

Models/Metrics	Dice Coefficient(Accuracy)	Dice Loss
2D Axis 1	0.71	0.14
2D Axis 2	0.71	0.15
2D Axis 3	0.85	0.14
Axis Integrate	0.937	0.07

Table 7: 2 class Models

4.4 Survival Prediction Network

While most of the Survival prediction attempts make use of clinical and radiomic features related to brain tumor, our attempt is to try and predict survival without any prior knowledge of the patient except his/her age. We try to build a deep learning architecture that makes use of our segmentation model.

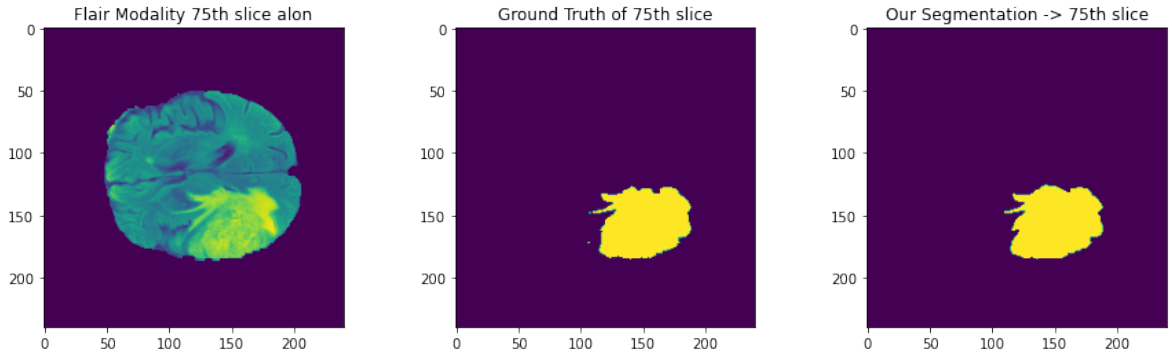


Figure 14: 2D axes integration results for 2 classes on one of the test images

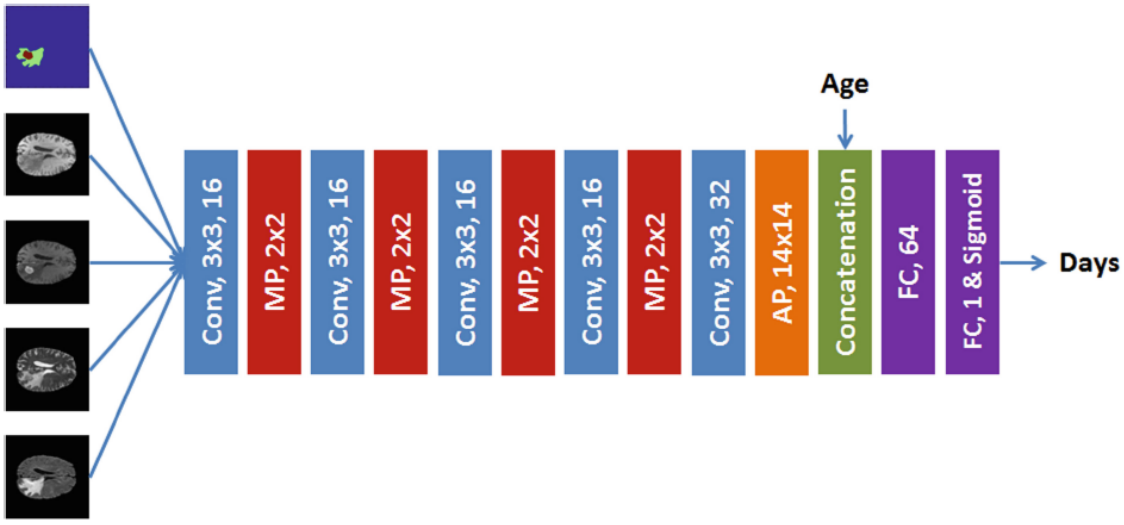


Figure 15: Survival Prediction architecture from the paper

4.4.1 Architecture

Inspired from [4] , our architecture comprises of convolutional blocks that extract survival related features from the MRI images concatenated with their corresponding segmented image(as a channel map), concatenates age of the patient with these feature maps and finally Fully Connected layers are applied. The final prediction has been distributed into 2 models : A classification model that classifies the patient to lie in one of the following 3 categories :-

- Survival days lying in range 0 - 300
- Survival days lying in range 300 - 450
- Survival days lying in range 450+

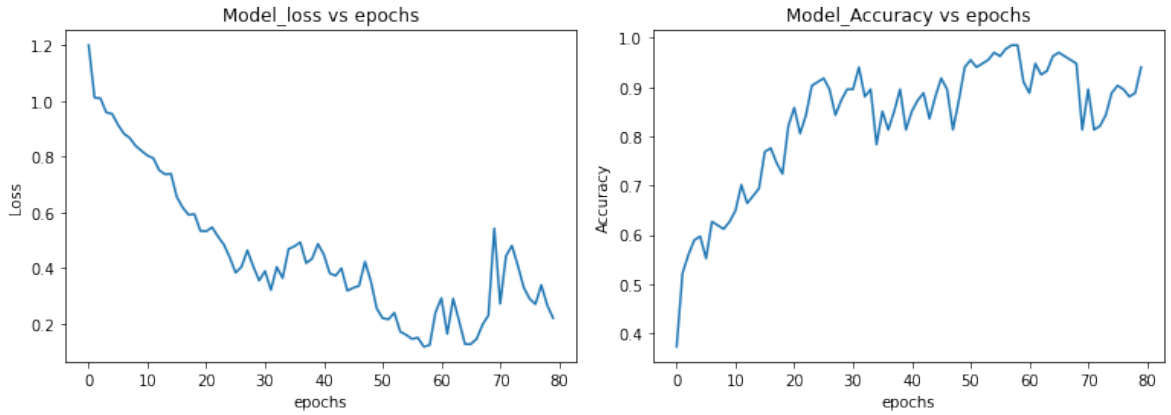
This is what BraTS 2017 challenge had asked for. The other model we try is regression in the number of days. Here the final prediction is sigmoid unlike softmax in the

classification model. For this , we follow the procedure from [4] and divide all ground truth days by the maximum survival days encountered. This way, each ground truth lies in range 0 to 1. So does our sigmoid output. To finally predict, we multiply our output by maxdays.

The classification model takes input of size $240 \times 240 \times 5$ and outputs softmax probabilities for the 3 classes. The regression model takes input of size $240 \times 240 \times 5$ and outputs a sigmoid probability.

4.4.2 Training Details

We have total data for 163 patients. We split it into 134 and 29 for training and testing respectively. The classification model is trained on 134 patients with Adam optimiser, Categorical Cross Entropy as loss function and accuracy as metric. With learning rate set to 1e-2, we train for 80 epochs with batch size 64.



(a) Loss vs epochs

(b) Accuracy vs epochs

Figure 16: Training plots for classification model

The regression model is trained with Adam optimiser, Mean Squared Loss as loss function and MSE as metric. With learning rate 1e-2 , we train for 90 epochs with batch size 64.

4.4.3 Evaluation/Prediction and Results

Our classification model produced an accuracy of 0.5172 on the 29 test patients.

Our regression model produced a mean squared error of 0.2107 on the 29 test images.

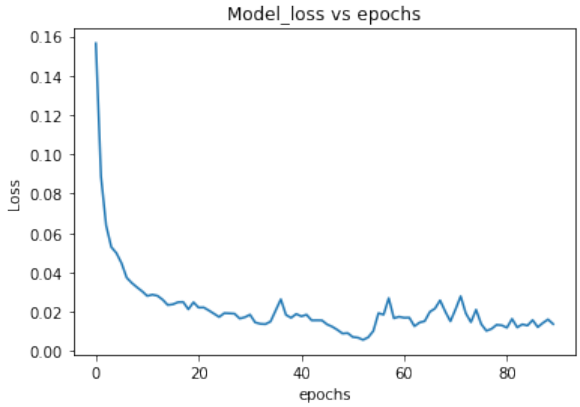


Figure 17: Training Loss plot for regression model

5 Comparison with State of the art methods

5.1 Segmentation Model

Architectures	Dice Scores
Zhao Et Al. (with CRF) [3]	0.88
Isensee Et Al. [1]	0.85
Zhao Et Al. (without CRF) [3]	0.71
UNet 3D	**0.73**
2D Axes Integration	**0.756**

Table 8: Our Results along with some SOTA

5.2 Survival Prediction

Architectures	Accuracy
Isensee Et Al. [1]	0.52
SurvPred	**0.51**
Li Et Al. [4]	0.55

Table 9: Our Results along with some SOTA

6 Limitations and Improvements that can be made

Having limited GPU and memory at hand, we were unable to try many architectures. While there are different ways our 2D models can be improved, one good idea is to combine connected component analysis during post processing of predicted image through

MRFs.

For Survival Prediction , we could not train a 3D model for feature extraction due to memory limitations. It may produce better results. Also many SOTA methods combine clinical and radiomics features with the features learnt through the segmentation model to predict survival. We were focused on a complete neural network based approach and did not touch on the handpicked nature of features.

7 Metrics and their Formulae

Let TP denote True positives(i^{th} label correctly classified as i), TN denote True Negatives(j^{th} label correctly classified as not being i) , FP denote False positives(ith label incorrectly classified as j) and FN denote False negatives(jth label incorrectly classified as i), then for label i:

$$Accuracy = \frac{TP + TN}{TP + FP + TN + FN}$$

$$Sensitivity = \frac{TP}{TP + FN}$$

$$Specificity = \frac{TN}{FP + TN}$$

References

[1] Fabian Isensee, Philipp Kickingereder, Wolfgang Wick, Martin Bendszus, and Klaus H. Maier-Hein. Brain tumor segmentation and radiomics survival prediction: Contribution to the BRATS 2017 challenge. *CoRR*, abs/1802.10508, 2018.

[2] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. *CoRR*, abs/1606.04797, 2016.

[3] Xiaomei Zhao, Yihong Wu, Guidong Song, Zhenye Li, Yazhuo Zhang, and Yong Fan. 3d brain tumor segmentation through integrating multiple 2d fcnns. In Alessandro Crimi, Spyridon Bakas, Hugo Kuijf, Bjoern Menze, and Mauricio Reyes, editors, *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries*, pages 191–203, Cham, 2018. Springer International Publishing.

[4] Yuexiang Li and Linlin Shen. Deep learning based multimodal brain tumor diagnosis. In Alessandro Crimi, Spyridon Bakas, Hugo Kuijf, Bjoern Menze, and Mauricio Reyes, editors, *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries*, pages 149–158, Cham, 2018. Springer International Publishing.