Eskisehir Technical University

Computer Engineering Departmant

Deep Learning Project Report

Brain Tumor Segmentation with 3D Unet

Authors:
Emirhan Yildiz
Furkan Çakir
Salih Kizilişik

Supervisor:
Asst.Prof.Cahit Perkgöz

Engineering Faculty, Eskisehir Technical University, Eskisehir, Turkey

August 23, 2024

ESKİŞEHİR TEKNİK ÜNİVERSİTESİ
ESKİŞEHİR TECHNICAL UNIVERSITY

Abstract

Early and accurate brain tumor diagnosis is crucial for effective treatment. However, manual analysis of 3D MRI scans, a critical step, is time-consuming, delaying treatment decisions. This necessitates a fully automated system for brain tumor segmentation from 3D MRI data. We propose an approach utilizing 3D version of UNet which one of from Recurrent Neural Networks (RNNs) for this task. We evaluate our method on the BraTS 2021 dataset and discuss the achieved performance in this report.

Keywords: segmentation · BraTS · UNet · CNN · Brain Tumor · deep learning

1 Introduction

Quantitative assessment of brain tumors provides valuable information and therefore constitutes an essential part of diagnostic procedures. Automatic segmentation is attractive in this context, as it allows for faster, more objective and potentially more accurate description of relevant tumor parameters, such as the volume of its subregions. Due to the irregular nature of tumors, deep learning algorithms struggling on understanding nature of tumors , however the algorithms of automatic segmentation developing and remains capable to compute the nature of tumors .

2 Methodology

2.1 Brats

BraTS (BRAin Tumor Segmentation) is a challenge focused on brain tumor segmentation. This challenge is designed for researchers interested in working on the automatic detection and segmentation of brain tumors in MRI images. BraTS provides a platform for evaluating the latest techniques and approaches in this field.

BraTS is usually held annually, and participants come together to complete various tasks over a certain period of time. These tasks typically involve topics such as the segmentation of a specific type of brain tumor or the prediction of a specific biological marker.

Table 1: Table 1: Summary of distribution of BraTS Challenge data across training, validation and test cohorts since the inception of BraTS initiative.

Year	Total Data	Training Data	Validation Data	Testing Data
2013	60	35	NA	25
2014	238	200	NA	38
2015	253	200	NA	53
2016	391	200	NA	191
2017	477	285	46	146
2018	542	285	66	191
2019	626	335	125	166
2020	660	369	125	166
2021	2040	1251	219	570

BraTS is jointly organized by the Radiological Society of North America (RSNA), the American Society of Neuroradiology (ASNR), and the Medical Image Computing and Computer Assisted Interventions (MICCAI) community.

BraTS is a significant event in the field of brain tumor segmentation and plays an important role in advancing research in this field. This challenge provides researchers, clinicians, and other stakeholders with an opportunity to explore and discuss the latest developments and innovations in this field.

The RSNA ASNR MICCAI Brain Tumor Segmentation (BraTS) 2021 challenge utilizes multi-institutional multi-parametric Magnetic Resonance Imaging (mpMRI) scans, to address both the automated tumor sub-region segmentation and the prediction of one of the genetic characteristics of glioblastoma (MGMT promoter methylation status) from pre-operative baseline MRI scans. Specifically, BraTS 2021 focuses on the evaluation of state-of-the-art methods for the accurate segmentation of intrinsically heterogeneous brain glioma sub-regions and on the evaluation of classification methods distinguishing between MGMT methylated (MGMT+) and unmethylated (MGMT-) tumors. [4]

2.2 MRI

The MICCAI BraTS challenge is a competition hosted by the Center for Biomedical Image Computing and Analytics (CBICA) at the University of Pennsylvania. The BraTS challenges identify and showcase state-of-the-

art techniques for brain tumor segmentation. The datasets distributed by the competition organizers consist of real world data in the form of multi-institutional routine MRI scans, manually segmented by multiple board-certified neurologists. The most common MRI sequences are T1-weighted and T2-weighted scans. **T1-weighted images** are produced by using short TE and TR times. The contrast and brightness of the image are predominately determined by T1 properties of tissue. Conversely, **T2-weighted images** are produced by using longer TE and TR times. In these images, the contrast and brightness are predominately determined by the T2 properties of tissue.

In general, T1- and T2-weighted images can be easily differentiated by looking the CSF. *CSF is dark on T1-weighted imaging and bright on T2-weighted imaging.*

A third commonly used sequence is the **Fluid Attenuated Inversion Recovery (Flair)**. The Flair sequence is similar to a T2-weighted image except that the TE and TR times are very long. By doing so, abnormalities remain bright but normal CSF fluid is attenuated and made dark. This sequence is very sensitive to pathology and makes the differentiation between CSF and an abnormality much easier.

	TR(msec)	TE(msec)
T1-Weighted(short TR and TE)	500	14
T2-Weighted(long TR and TE)	4000	90
Flair(very long TR and TE)	9000	114

Table 2: Most common MRI Sequences and their Approximate TR and TE times.

The scans are split into high-grade gliomas (HGG) and low-grade gliomas (LGG) and provided in the T1w, T1ce, T2w, and FLAIR modalities. The individual sequence types make the dataset more robust owing to the different strengths of each MR image modality. T1-weighted (or T1w) sequences display fluid and water-based tissues as mid grey whilst fatty tissue has a high intensity . Contrast agents applied to T1w images produce T1ce images, which enhance the intensity of highly vascular tumours . T2-weighted (T2w) images are visually opposite of T1w scans, as fluids are now the brightest feature, and fat, water based tissues are mid-grey . Finally, FLAIR sequences are a variation of T2w images, where the cerebrospinal fluid (CSF) within

the brain and any tissues with a similar T1 value are suppressed from the scan . A sample of each sequence type taken from the training data used in this study is shown in Figure 14

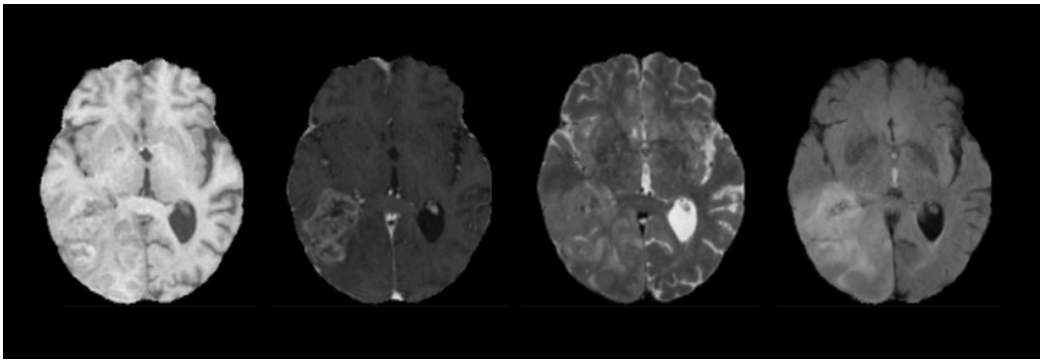


Figure 1: Left to Right: T1, T1ce, T2, and FLAIR slice samples of the same patient from the BraTS 2019 training dataset [10]

2.3 Channels

The BraTS ground truth annotations are composed of three main categories, split into labels 1, 2, and 4 in the ground truths. The BraTS target classes ET, WT, and TC are composed of different combinations of these labels, as shown in 3 . A visual representation of the labels is also shown in Figure

Class	Enhancing Tumor Core (Label 4)	Peritumoral Edema (Label 2)	Non-Enhancing & Necrotic Tumor Core (Label 1)
ET	+		
WC	+	+	+
TC	+		+

Table 3: . Labels and target classes for BraTS 2019[10]

A visual representation of the labels is also shown in Figure 2 [10]

2.4 U-net

U-Net, a deep learning model specifically designed for biomedical image segmentation, exemplifies this. Introduced in 2015 by Olaf Ronneberger's team,

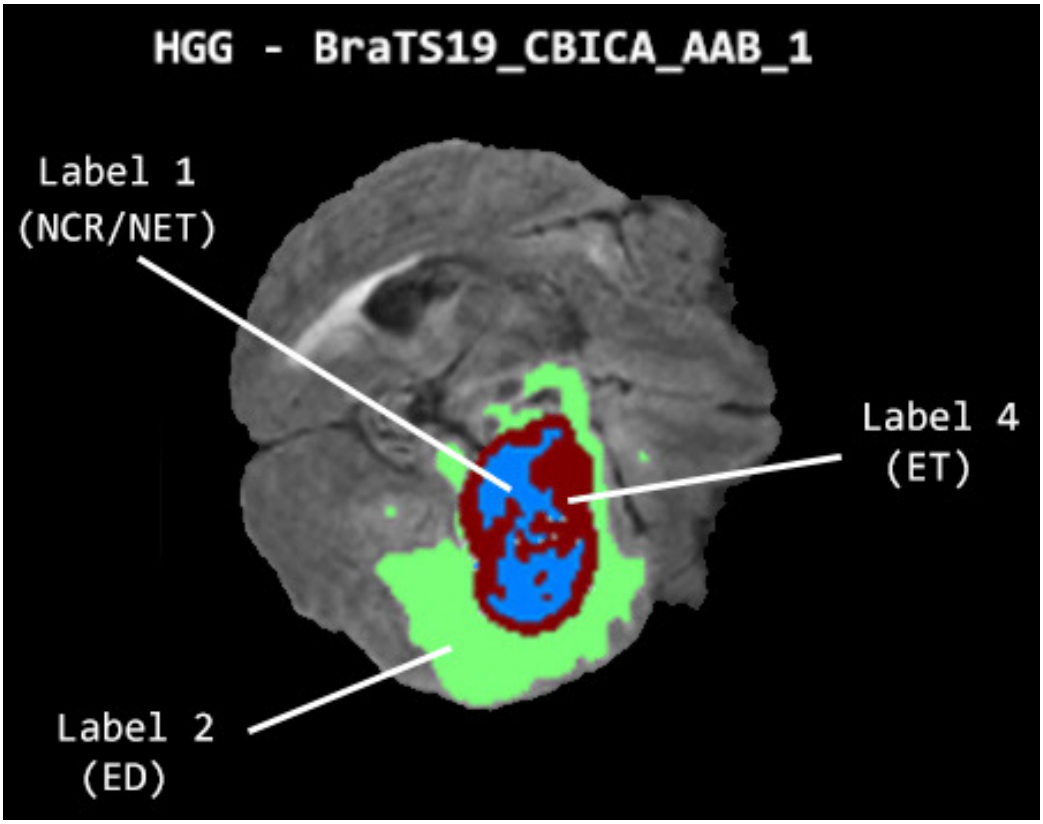


Figure 2: Label1: Necrotic and non-enhancing tumor core (NCR/NET). Label 2: Peritumoral edema (ED). Label 4: GD-enhancing tumor (ET).

U-Net aimed to create a high-performing network that could work with limited training data, addressing the challenge of scarce annotated images in the medical field.

The process of image segmentation Deep Learning models have solved the limitations discussed above. Several deep-learning models are used for image segmentation, such as U-Net, Fully Convolutional Networks (FCN), and Mask R-CNN. However, all of these models roughly follow the following procedure for image segmentation.[6]

Similar to FCN [9] and SegNet [1], U-Net [13] uses a network entirely of convolutional layers to perform the task of semantic segmentation. The network architecture is symmetric, having an Encoder that extracts spatial features from the image, and a Decoder that constructs the segmentation

map from the encoded features. The Encoder follows the typical formation of a convolutional network. It involves a sequence of two 3 ×3 convolution operations, which is followed by a max pooling operation with a pooling size of 2 × 2 and stride of 2. This sequence is repeated four times, and after each downsampling 3 the number of filters in the convolutional layers are doubled. Finally, a progression of two 3 × 3 convolution operations connects the Encoder to the Decoder. On the contrary, the Decoder first up-samples the feature map using a 2×2 transposed convolution operation [17], reducing the feature channels by half. Then again a sequence of two 3 × 3 convolution operations is performed. Similar to the Encoder, this succession of up-sampling and two convolution operations is repeated four times, halving the number of filters in each stage. Finally, a 1×1 convolution operation is performed to generate the final segmentation map. All convolutional layers in this architecture except for the final one use the ReLU (Rectified Linear Unit) activation function [8]; the final convolutional layer uses a Sigmoid activation function. Perhaps, the most ingenious aspect of the U-Net architecture is the introduction of skip connections. In all the four levels, the output of the convolutional layer, prior to the pooling operation of the Encoder is transfered to the Decoder. These feature maps are then concatenated with the output of the upsampling operation, and the concatenated feature map is propagated to the successive layers. These skip connections allow the network to retrieve the spatial information lost by pooling operations [2]. The network architecture is illustrated in 3

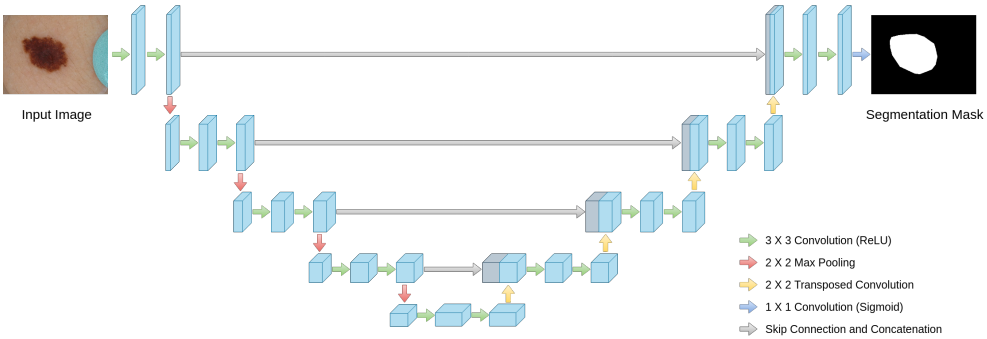


Figure 3: The U-Net Architecture. The model comprises an encoder and a decoder pathway, with skip connections between the corresponding layers.

2.4.1 Architecture of U-Net

The model features a distinctive U-shaped structure, comprising two main parts: the contracting path (encoder) and the expanding path (decoder). The encoding path captures context, and the decoding path enables precise localization.

Contracting Path (Encoder)

- **Convolutional Layers:** Convolution Layers are the primary components of the contracting path. In the originally proposed model, each block consists of two consecutive 3×3 convolutional layers followed by a Rectified Linear Unit (ReLU) activation function. By stacking multiple convolutional layers, U-Net learns increasingly complex features.

- **Activation Functions**: After each convolution operation, a ReLU activation function is applied. The role of ReLU here is crucial as it introduces non-linearities into the system, which allows for learning more complex patterns in data that are not possible with just linear transformations.

- **Max Pooling**: Following the convolutional layers, a 2×2 max pooling operation with stride 2 is used. This step reduces the spatial dimensions by half. However, it captures abstract information (that makes the model invariant to small shifts and distortions).

- **Feature Doubling:** After each max pooling step, the subsequent convolutional layer doubles the number of filters used. For example, if a layer starts with 64 feature channels, it will have 128 channels after the next pooling and convolution operations. By doubling the number of feature channels, the network can maintain or even increase its capacity to represent information despite the reduction in spatial resolution. This is crucial because the risk of losing important details increases as the image size reduces

Expansive Path (Decoder)

Aims to recover spatial information and generate the segmentation map using up-convolution (or transposed convolution). Each block includes: Upsampling of the feature map to increase image size. A 2×2 convolution to halve the number of feature channels. Two 3×3 convolutions followed by ReLU activation. U-Net also uses skip connections.

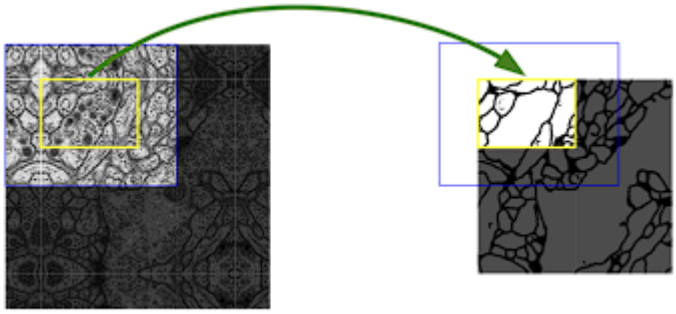


Figure 4: . U-Net: "Overlap-tile strategy for seamless segmentation of arbitrary large images (here segmentation of neuronal structures in EM stacks). Prediction of the segmentation in the yellow area, requires image data within the blue area as input. Missing input data is extrapolated by mirroring" [18]

What are skip connections?

Skip connections significantly contribute to U-Net's effectiveness. By merging feature maps from the contracting path directly with the expanding path, U-Net combines low-level detail information with high-level contextual information across the network

- **Recover Spatial Hierarchies**: These connections allow U-Net to concatenate high-resolution features from the contracting path with upsampled outputs from the expanding path. This helps recover spatial hierarchies lost during pooling operations in the contracting phase.[13]

The authors tested U-net on several datasets, notably the ISBI cell tracking challenge 2015 where they bested the previous SoTA on segmentation of brain tumor cells captured by phase-contrast microscopy (9% improvement in IOU score), and cervical cancer cells captured by differential interference contrast microscopy (31% improvement in IOU score).

1. **U-Net with Residual Connections:** Drozdzal et al. [2] explored the use of short and long skip connections in a Unet-like model (modifying [5] by adding an expanding path and corresponding connections from the contracting path). They noted that the copy and concatenation of features in U-Net's contracting path with features in the expanding path are akin to long skip connections and so choose to sum rather than concatenate the features in their models. The combination with

short and long skip connections led to better convergence and training stability relative to variants of the network that either utilized only one type of connection, or neither.

2. **3D U-Net :** [19] directly extended U-net in a different dimension, proposing a variant that utilized 3D convolutions in place of 2D convolutions for full volumetric segmentation. Aside from reducing the number of output features in every layer of the the contracting path by half, save for those directly preceding downsampling operations, the 3D U-net was identical to the original U-net (see 5). Given sparsely annotated training data (volumes with only a few slices annotated), the authors used the 3D U-net to produce dense volumetric segmentation of Xenopus kidney embryos captured by confocal microscopy in two tasks. The first was a 'semi-automated" segmentation task where dense (complete) volume segmentation was produced for a sparsely annotated training sample, achieving a 7% higher IOU score relative to a 2D U-Net. The second was a fully-automated segmentation task where a dense volume segmentation was produced for an unlabeled volume on which the network had not been trained, achieving an 18% higher IOU compared to a 2D U-Net.

3. **V-Net:** Milletari et al. [7] combined the above ideas in "V-Net", a 3D U-net with residual blocks applied to the task of 3D prostate MRI segmentation (see 6). The integration of greater spatial context and residual learning led to remarkable performance benefits, being on par with the then SoTA model on the "PROMISE 2012" challenge dataset at a reduced training convergence time common to residual networks. Unlike U-net [13] and 3D Unet [19] the authors eschew batch normalization and follow the increasingly common trend of eliminating pooling layers, performing downsampling via convolutions kernels of size 2x2x2 and a stride of two. They also performed segmentation on the entire image patch as opposed to previous works which only segmented the central section of the image patch. Another major contribution of the authors was the proposal of a soft dice loss which they used in their loss function in an attempt to directly optimize the network for segmentation accuracy. This version led to 13% greater performance than one trained using multinomial logistic loss with sample weighting. The resulting segmentation maps were not only more accurate, they were

also smoother and more visually pleasing. LLdice is a soft DICE loss applied to the decoder output ppred to match the segmentation mask ptrue:

$$LLdice = \frac{2 \times \sum_i^N PtruePpred}{\sum_i^N Ptrue^2 + \sum_i^N Ppred^2}$$

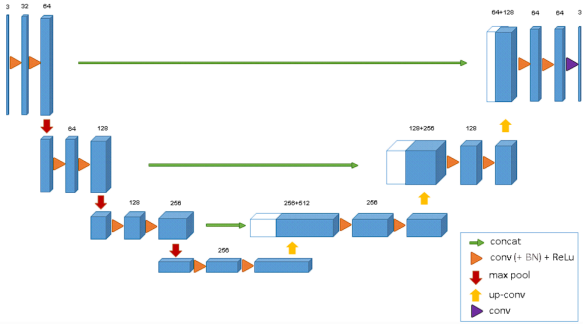


Figure 5: 3D U-Net: "The 3D u-net architecture. Blue boxes represent feature maps. The number of channels is denoted above each feature map." [19]

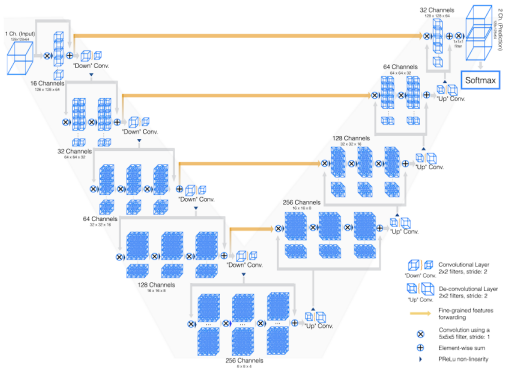


Figure 6: V-Net: "Schematic representation of our network architecture." "...processes 3D data by performing volumetric convolutions." [7]

3 Dataset and Methods

All BraTS mpMRI scans are available as NIfTI files (.nii.gz) and describe a) native (T1) and b) post-contrast T1-weighted (T1Gd), c) T2-weighted (T2), and d) T2 Fluid Attenuated Inversion Recovery (T2-FLAIR) volumes, and were acquired with different clinical protocols and various scanners from multiple data contributing institutions. We intend to release the associated de-identifed DICOM (.dcm) files after the conclusion of the challenge.

All the imaging datasets have been annotated manually, by one to four raters, following the same annotation protocol, and their annotations were approved by experienced neuro-radiologists. Annotations comprise the GD-enhancing tumor (ET — label 4)[12], the peritumoral edematous/invaded tissue (ED — label 2)[2], and the necrotic tumor core (NCR — label 1), as described both in the BraTS 2012-2013 TMI paper and in the latest BraTS summarizing paper. The ground truth data were created after their pre-processing, i.e., co-registered to the same anatomical template, interpolated to the same resolution (1 mm3) and skull-stripped.

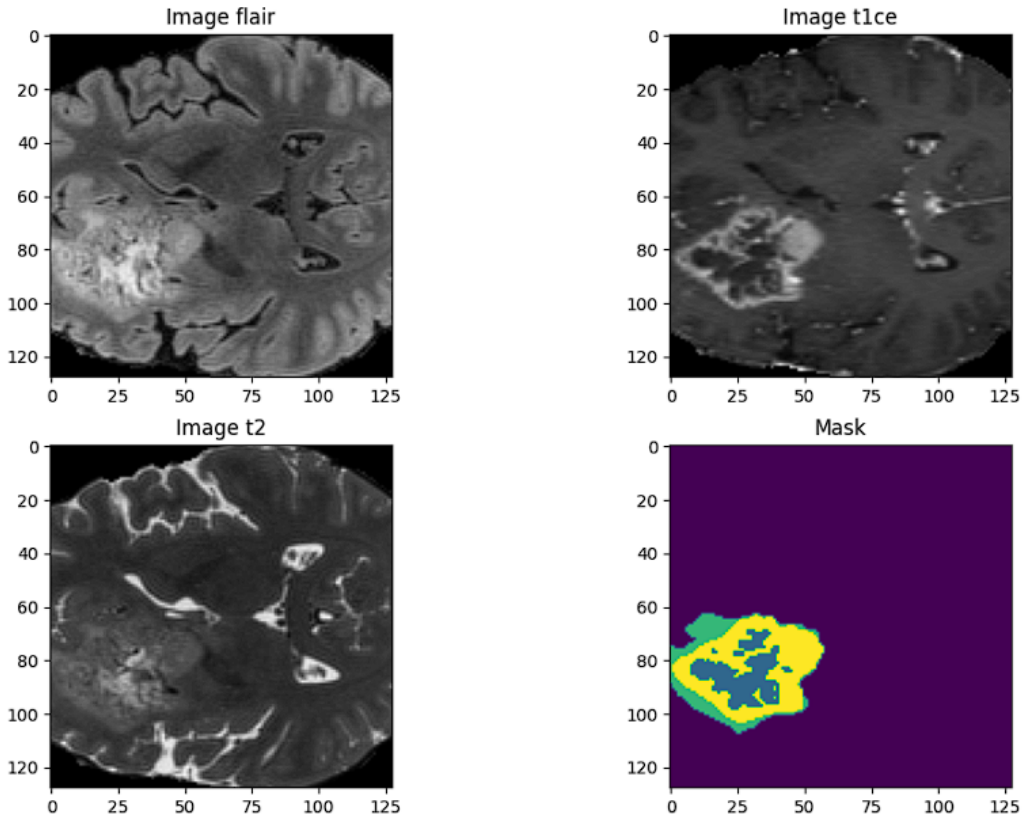


Figure 7: Where t1ce flair and t2 are displayed horizontally and their masked image

When working with the data we had, there were certain things we were sure of from our first attempts, and we kept these principles throughout the project. These basic principles included the elements of the data that needed to be organized. First of all, we realized that we needed to reduce the size and computational complexity of the data, and we found that there were a lot of redundant fields in our data. We implemented the following solutions for these problems:

1-Reducing the Size of the Data: The values in our data maxed out to 1890. We used the Min-Max Scalar to control this and make calculations easier. The Min-Max Scalar allows the data to be scaled within a certain range. In this way, by compressing our data between 0 and 1, we both accelerated the calculation processes and enabled the algorithms to work

efficiently.

$$Xscaled = \frac{X - Xmin}{Xmax - Xmin}$$

2-Cleaning Unnecessary Fields: We observed that we had a lot of unnecessary fields in our data. These fields were causing us to spend unnecessary processing power in the analysis and modeling processes. We cropped our 240x240x155 (Figure **??**) data to 128x128x128 (Figure 9)and got a closer image, making our data more compact and analyzable.

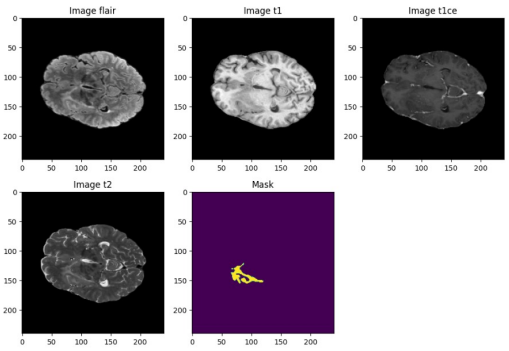


Figure 8: Data and mask as a 240x240x155

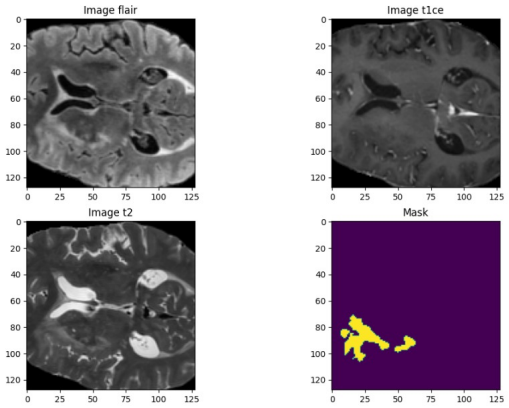


Figure 9: Cropped Data

We then did some research on how to use the data we had more efficiently and found the following information (Figure 10:)

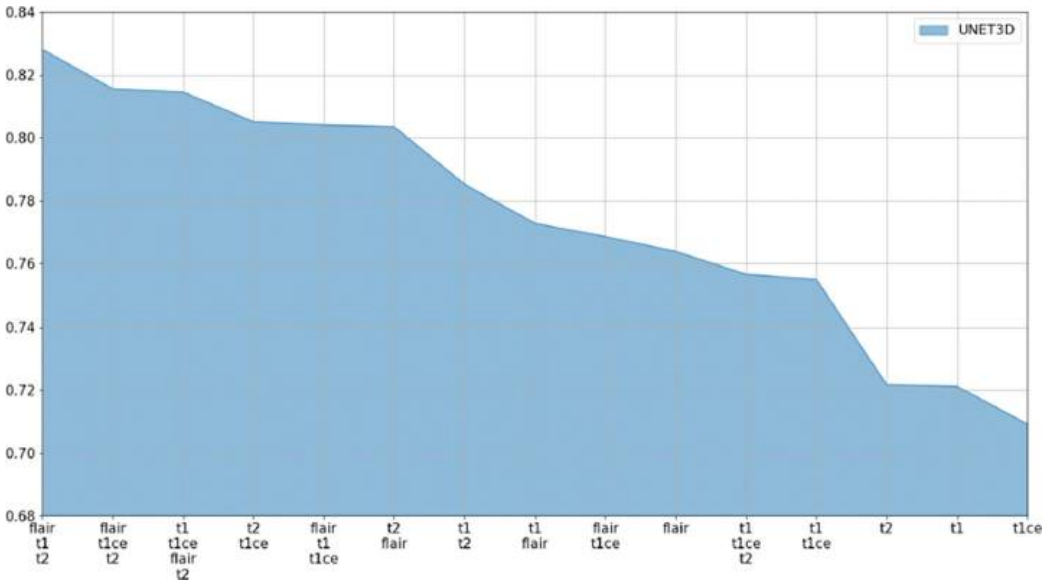


Figure 10: Accuracy with respect to combinations of sequences for UNET3D. In x axis the different combination of sequences. In y axis the accuracy

As a result of a research on Brats 2018 data, we see how the accuracy compares when using different combinations and singles. And from there, we developed a strategy to evolve into two different models, one of our models would be based on flair, which had the highest success rate as a single, and the other was to try a different combination of t1ce t2 and flair without t1.

1. **Segmentation with Flair Inputs with VNet Model:** We used the VNet model to perform segmentation with single inputs. However, there were a few key challenges we faced with this model:

    - **Heavy Dispersion:** The model's segmentations showed a rather heavy distribution. In particular, this caused the model's outputs to be inhomogeneous and overestimated in some regions.
    - **Operation on a Single Channel:** The VNet model operated on a single channel only. This limited the model's performance and hindered its ability to work with multi-channel data.

    We developed several strategies to overcome these problems:

    - **Different Normalization Techniques:** To improve the segmentation results, we considered applying different normalization

techniques. These techniques can improve the performance of the model by making the distribution of the data more balanced.

- **Multi-Channel Input Strategies:** We explored multi-channel input strategies to enable the model to work more effectively with multi-channel data. This can enable the model to process more data types and learn richer features.

2. **Segmentation with T1ce,T2,Flair Inputs with 3D U-Net 3D U-Net Model:** On the other hand, we tried segmentation with triple inputs using the 3D U-Net model, which also faced some significant challenges:

   - **Memory Problem:** The 3D U-Net model required a large amount of memory. Therefore, when working with the model, we ran into out-of-memory issues and had to use very small chunk sizes, which reduced efficiency and increased computation times during the training of the model.
   - **Single Channel Problem:** The problem we experienced with the VNet model of processing on only one channel also appeared in the 3D U-Net model. This prevented us from utilizing the full potential of multi-channel data sets and limited the overall performance of the model.

     As a solution to these problems, we considered the following:

   - **Memory Management:** To alleviate the memory issue, we planned to implement more optimized memory management techniques and also aimed to reduce memory requirements by making changes to the model architecture.
   - **Multichannel Inputs:** To overcome the single-channel problem, we considered developing methods to enable the model to work with multi-channel inputs. This could allow the model to acquire more information and perform more accurate segmentation.

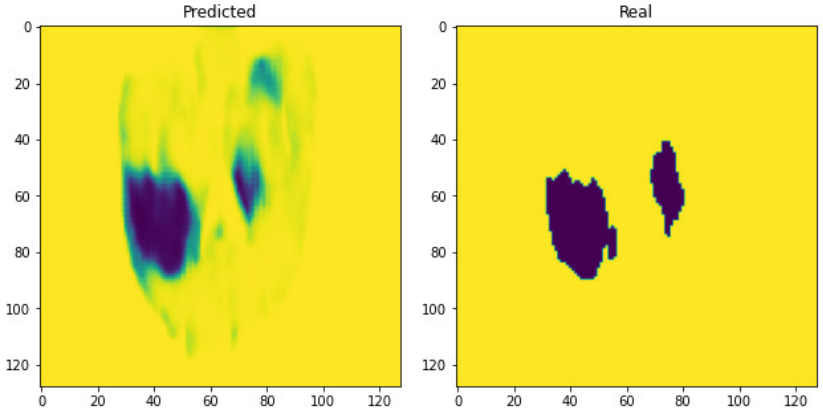


Figure 11: First result with Vnet and with just Flair with no class

But despite all this, we managed to get results, and although the results did not satisfy us, we obtained results that could be considered successful, but our accuracy results were freezing more positive results than necessary because our accuracy results were based on a single mask.

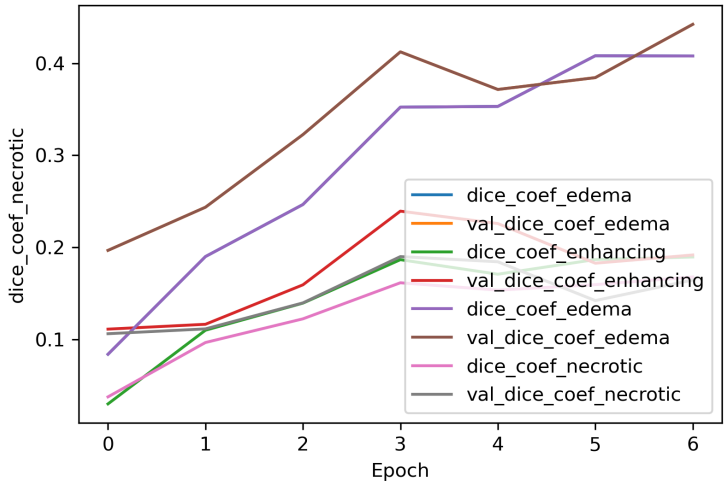


Figure 12: Bad results on dice score

We saw that our first model accidentally shallow the classes but, overall results was good enough on single class as shown on Figure 11. We redesigned

our model again to classify tumor classes.

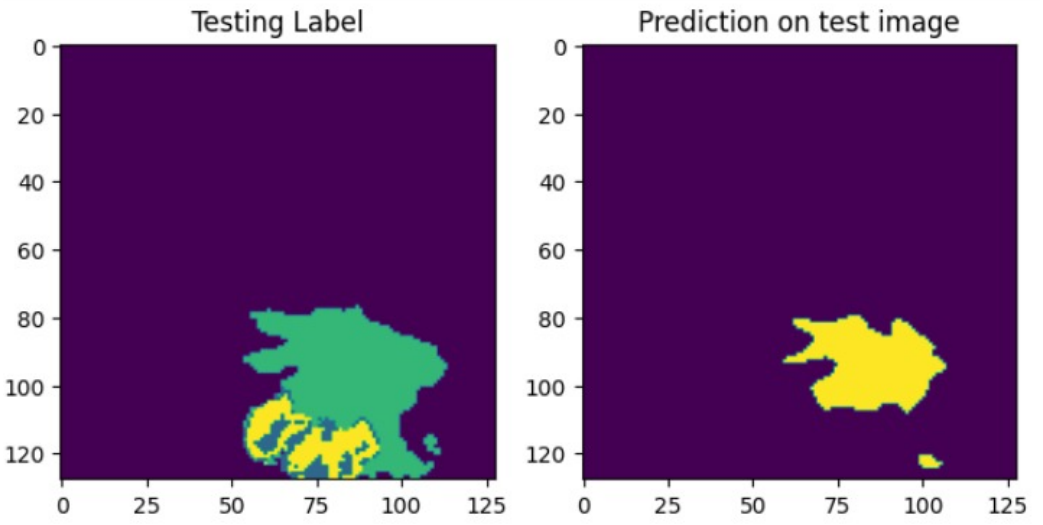


Figure 13: First result with Unet and T1ce, T2 and Flair combination

Since we have observed that the unified model will be more efficient from now on, we have decided to continue and develop this 3D Unet model with we decided to use most efficient scans on single input, flair scans according to Figure 10.

We tried new things with our model, added excavated things and removed some things.

3.0.1 Dice Loss

In a CNN-based setting, the weights $\theta \in \Theta$ are often updated using gradient descent. For this purpose, the loss function ℓ computes a real valued cost $\ell(Y, \tilde{Y})$ based on the comparison between the ground truth Y and its prediction $\tilde{Y}$ in each iteration. Y and $\tilde{Y}$ contain the values $y_{b,c,i}$ and $\tilde{y}_{b,c,i}$, respectively, pointing to the value for a semantic class $c \in \mathcal{C} = [\mathrm{C}]$ at an index $i \in \mathcal{I} = [\mathrm{I}]$ (e.g. a voxel) of a batch element $b \in \mathcal{B} = [\mathrm{B}]$ The exact update of each θ depends on $d\ell(Y, \tilde{Y})/d\theta$, which can be computed via the generalized chain rule. With $\omega = (b, c, i) \in \Omega = \mathcal{B} \times \mathcal{C} \times \mathcal{I}$, we can write:[14]

$$\frac{d\ell(Y,\tilde{Y})}{d\theta} = \sum_{b\in\mathcal{B}} \sum_{c\in\mathcal{C}} \sum_{i\in\mathcal{I}} \frac{\partial\ell(Y,\tilde{Y})}{\partial\tilde{y}_{b,c,i}} \frac{\partial\tilde{y}_{b,c,i}}{\partial\theta} = \sum_{\omega\in\Omega} \frac{\partial\ell(Y,\tilde{Y})}{\partial\tilde{y}_{\omega}} \frac{\partial\tilde{y}_{\omega}}{\partial\theta}. \tag{1}$$

The Dice similarity coefficient (DSC) over a subset $\phi \subset \Omega$ is defined as:

$$\mathrm{DSC}(Y_\phi, \tilde{Y}_\phi) = \frac{2|Y_\phi \cap \tilde{Y}_\phi|}{|Y_\phi| + |\tilde{Y}_\phi|}. \tag{2}$$

This formulation of $\mathrm{DSC}(Y_\phi, \tilde{Y}_\phi)$ requires Y and $\tilde{Y}$ to contain values in $\{0, 1\}$. In order to be differentiable and handle values in relaxations such as the *soft* DSC (sDSC) are used [3, 11]. Furthermore, in order to allow both Y and $\tilde{Y}$ to be empty, a smoothing term ϵ is added to the nominator and denominator such that $\mathrm{DSC}(Y_\phi, \tilde{Y}_\phi) = 1$ in case both Y and $\tilde{Y}$ are empty. This results in the more general formulation of the Dice loss (DL) computed over a number of subsets $\Phi = \{\phi\}$:

$$\mathrm{DL}(Y, \tilde{Y}) = 1 - \frac{1}{|\Phi|} \sum_{\phi \in \Phi} \mathrm{sDSC}(Y_\phi, \tilde{Y}_\phi) = 1 - \frac{1}{|\Phi|} \sum_{\phi \in \Phi} \frac{2 \sum_{\varphi \in \phi} y_\varphi \tilde{y}_\varphi + \epsilon}{\sum_{\varphi \in \phi} (y_\varphi + \tilde{y}_\varphi) + \epsilon}. \tag{3}$$

Note that typically all ϕ are equal in size and define a partition over the domain Ω, such that $\bigcup_{\phi \in \Phi} \phi = \Omega$ and $\bigcap_{\phi \in \Phi} \phi = 0$. In $d\mathrm{DL}(Y, \tilde{Y})/d\theta$ from Eq. 1, the derivative $\partial \mathrm{DL}(Y, \tilde{Y})/\partial \tilde{y}_\omega$ acts as a scaling factor. In order to understand the underlying optimization mechanisms we can thus analyze $\partial \mathrm{DL}(Y, \tilde{Y})/\partial \tilde{y}_\omega$. Given that all ϕ are disjoint, this can be written as:

$$\frac{\partial \mathrm{DL}(Y, \tilde{Y})}{\partial \tilde{y}_\omega} = -\frac{1}{|\Phi|} \left(\frac{2 y_\omega}{\sum_{\varphi \in \phi^\omega} (y_\varphi + \tilde{y}_\varphi) + \epsilon} - \frac{2 \sum_{\varphi \in \phi^\omega} y_\varphi \tilde{y}_\varphi + \epsilon}{\left(\sum_{\varphi \in \phi^\omega} (y_\varphi + \tilde{y}_\varphi) + \epsilon \right)^2} \right), \tag{4}$$

with ϕ^ω the subset that contains ω. As such, it becomes clear that the specific action of DL depends on the exact configuration of the partition Φ of Ω and the choice of ϵ. Next, we describe the most common choices of Φ and ϵ in practice. Then, we investigate their effects in the context of missing or empty labels. Finally, we present a simple heuristic to tune both.

3.0.2 Tversky loss layer

The output layer in the network consists of c planes, one per class ($c = 2$ in lesion detection). We applied softmax along each voxel to form the loss. Let P and G be the set of predicted and ground truth binary labels, respectively.

The Dice similarity coefficient D between two binary volumes is defined as:

$$D(P, G) = \frac{2|PG|}{|P| + |G|} \tag{5}$$

If this is used in a loss layer in training , it weighs FPs and FNs (precision and recall) equally. In order to weigh FNs more than FPs in training our network for highly imbalanced data, where detecting small lesions is crucial, we propose a loss layer based on the Tversky index . The Tiversky index is defined as:

$$S(P, G; \alpha, \beta) = \frac{|PG|}{|PG| + \alpha|P \setminus G| + \beta|G \setminus P|} \tag{6}$$

where α and β control the magnitude of penalties for FPs and FNs, respectively.

To define the Tversky loss function we use the following formulation:

$$T(\alpha, \beta) = \frac{\sum_{i=1}^{N} p_{0i} g_{0i}}{\sum_{i=1}^{N} p_{0i} g_{0i} + \alpha \sum_{i=1}^{N} p_{0i} g_{1i} + \beta \sum_{i=1}^{N} p_{1i} g_{0i}} \tag{7}$$

where in the output of the softmax layer, the p_{0i} is the probability of voxel i be a lesion and p_{1i} is the probability of voxel i be a non-lesion. Also, g_{0i} is 1 for a lesion voxel and 0 for a non-lesion voxel and vice verse for the g_{1i}. The gradient of the loss in Equation 7 with respect to p_{0i} and p_{1i} can be calculated as:

$$\frac{\partial T}{\partial p_{0i}} = 2\frac{g_{0j}(\sum_{i=1}^{N} p_{0i} g_{0i} + \alpha \sum_{i=1}^{N} p_{0i} g_{1i} + \beta \sum_{i=1}^{N} p_{1i} g_{0i}) - (g_{0j} + \alpha g_{1j}) \sum_{i=1}^{N} p_{0i} g_{0i}}{(\sum_{i=1}^{N} p_{0i} g_{0i} + \alpha \sum_{i=1}^{N} p_{0i} g_{1i} + \beta \sum_{i=1}^{N} p_{1i} g_{0i})^2} \tag{8}$$

$$\frac{\partial T}{\partial p_{1i}} = -\frac{\beta g_{1j} \sum_{i=1}^{N} p_{0i} g_{0i}}{(\sum_{i=1}^{N} p_{0i} g_{0i} + \alpha \sum_{i=1}^{N} p_{0i} g_{1i} + \beta \sum_{i=1}^{N} p_{1i} g_{0i})^2} \tag{9}$$

Using this formulation we do not need to balance the weights for training. Also by adjusting the hyperparameters α and β we can control the trade-off between false positives and false negatives. It is noteworthy that in the case of $\alpha = \beta = 0.5$ the Tversky index simplifies to be the same as the Dice coefficient, which is also equal to the F_1 score. With $\alpha = \beta = 1$, Equation 2 produces Tanimoto coefficient, and setting $\alpha + \beta = 1$ produces the set of F_β scores. Larger βs weigh recall higher than precision (by placing more emphasis on false negatives). We hypothesize that using higher βs in our

generalized loss function in training will lead to higher generalization and improved performance for imbalanced data; and effectively helps us shift the emphasis to lower FNs and boost recall.[15]

4 Training Technics

4.1 Training Procedures

Our network architecture is trained with randomly sampled patches of size 128x128x128 voxels and batch size 4. We refer to an epoch as an iteration over 213 batches and train for a total of 30 epochs. Training is done using the adam optimizer [16] with an initial learning rate $lr_{init} = 1 * 10^{-3}$, and a there is a reducing learning rate algorithm that reducing the learning rate on plateaus. When there is no improvement on validation loss learning rate is decreasing simultaneously at least $lr_{limit} = 1 * 10^{-6}$.

4.2 Loss Function

5 Result

5.1 Evaluation

We saw that Dice is better on our dataset

Table 4: Results of evaluation of trained on different losses

Loss	accuracy	precision	recall	DC	DC ED	DC ET	DC NCR
Dice+Tversky	93,76%	94,26%	93,61%	0.5180	0.5158	0.3185	0.2649
Cross-Entropy	93,77%	95,70%	9259%	0.4893	0.4532	0.2804	0.2447

5.2 Predictions on Test Set

5.2.1 Dice + Tversky

Here you can see the test results we obtained using dice and tversky loss. The texts in the images show the abnormalities seen in the scans. For example, the text "NECROTIC/CORE" indicates that cell death is occurring in that area of the brain. The text "EDEMA" indicates swelling of the brain tissue.

The text "ENHANCING" indicates that there is abnormally high blood flow in that area of the brain.

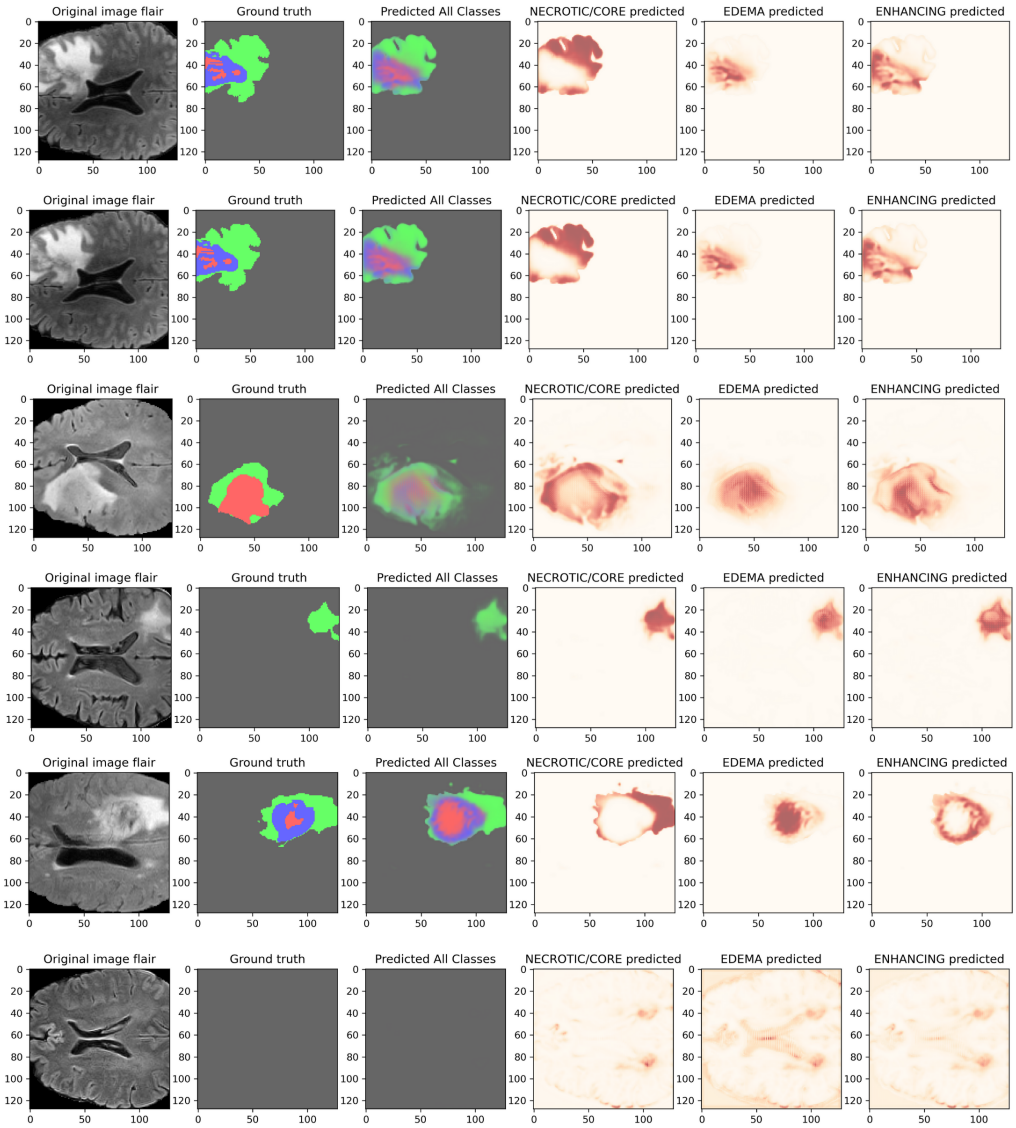
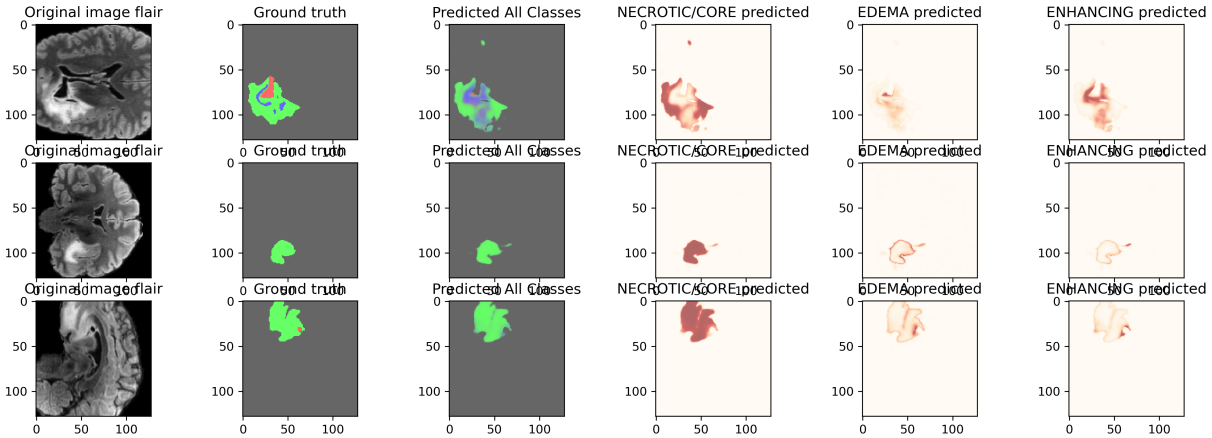


Figure 14: Result with Tversky + Dice

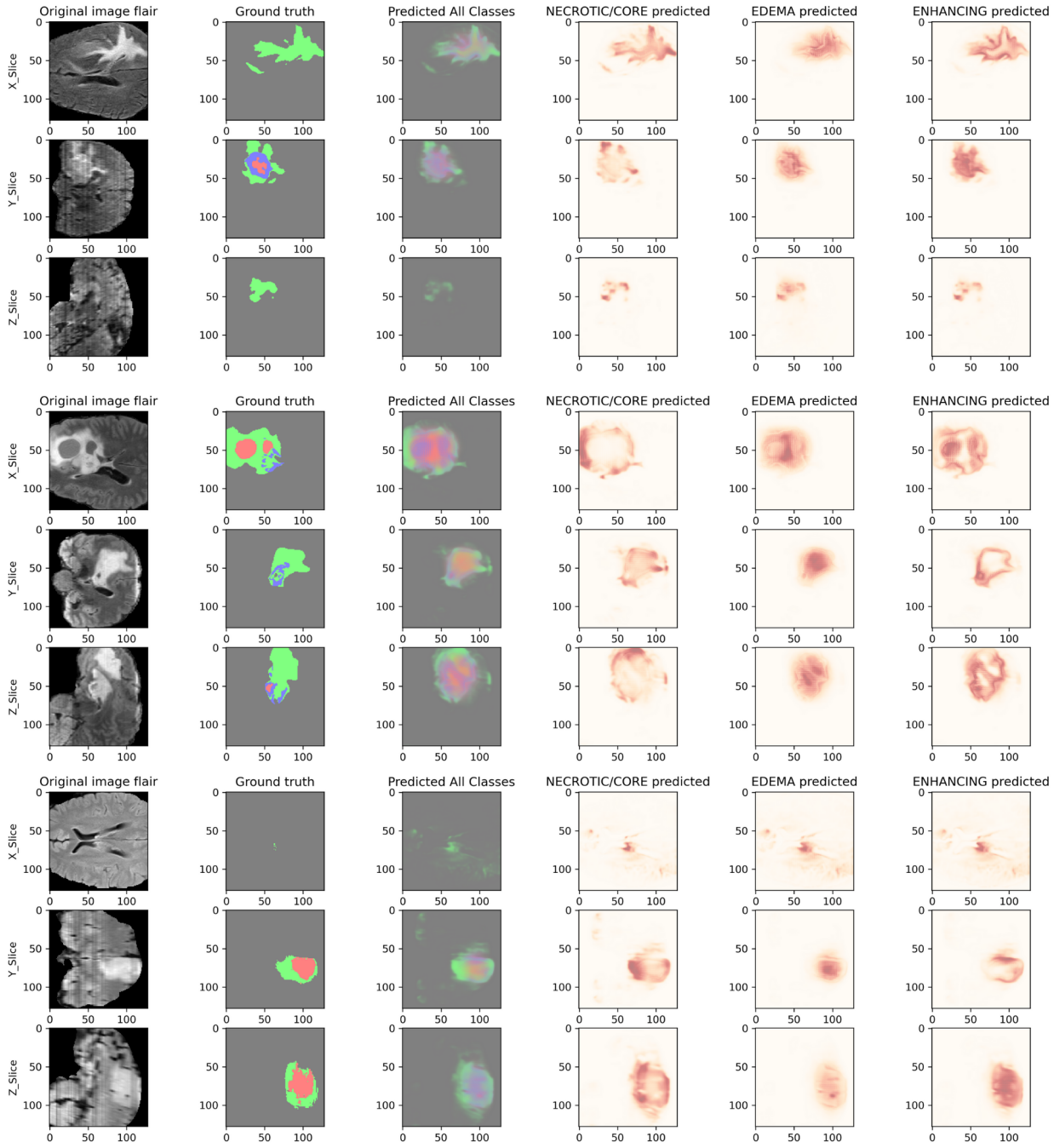


The results here actually satisfied us, but while the drawings were being created, it seemed that the top predictions overrode the bottom predictions in the collective predictions. However, our success rate was quite high, and it satisfied us, especially even though we made predictions based on a single mri. In the first prediction chart we shared (Figure 10), we achieved a higher success rate than with flair alone, and we trained them as 850 train 250 val 150 test.

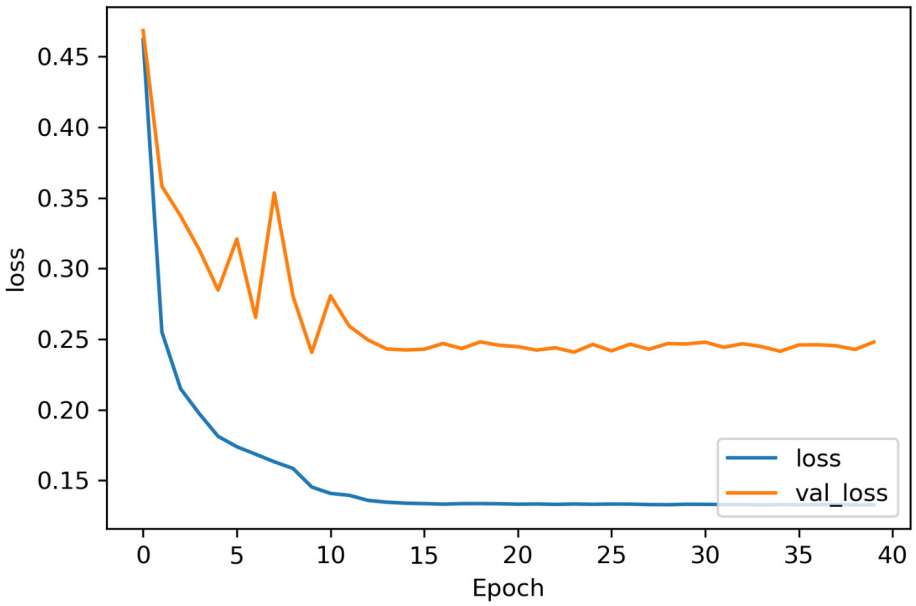
5.2.2 Cross Entropy Loss

In this section, we will examine the results obtained by our model trained on the Brats dataset using cross entropy loss and Unet. While the cross entropy loss function is used to measure how close the model's predictions are to the real values in the dataset, Unet is known as an effective neural network architecture for medical image segmentation.

And the result of Cross Entropy Loss:


Original image flair
Ground truth
Predicted All Classes
NECROTIC/CORE predicted
EDEMA predicted
ENHANCING predicted
X_Slice
Y_Slice
Z_Slice
0
50
100

And this is the state of our loss functions during the training period.



6 Conclusion

This section provided an overview of the methodology used to train a deep learning model for brain tumor segmentation on the BraTS dataset. We discussed the BraTS challenge, a significant event in the field that provides a platform for evaluating automatic tumor segmentation techniques. MRI, the imaging modality used in the challenge, was explained along with the different sequences employed to capture various tumor characteristics.

We then focused on the U-Net architecture, a popular deep learning model specifically designed for medical image segmentation. We explored the U-Net architecture, including its contracting and expanding paths, skip connections, and their functionalities. We also highlighted variations of U-Net, such as those incorporating residual connections and 3D convolutions for volumetric segmentation tasks.

By understanding the BraTS dataset, MRI as the underlying imaging modality, and the U-Net architecture with its variations, we have established

a foundation general evaluation into the details of the model's training process, loss function selection, and the achieved segmentation results given in the previous section of the report.

References

[1] Vijay Badrinarayanan, Alex Kendall, and Roberto Cipolla. Segnet: A deep convolutional encoder-decoder architecture for image segmentation. *CoRR*, abs/1511.00561, 2015.

[2] Michal Drozdzal, Eugene Vorontsov, Gabriel Chartrand, Samuel Kadoury, and Chris Pal. The importance of skip connections in biomedical image segmentation, 2016.

[3] Michal Drozdzal, Eugene Vorontsov, Chartrand Gabriel, Kadoury Samuel, and Pal Chris. The Importance of Skip Connections in Biomedical Image Segmentation. In *DLMIA 2016, LABELS 2016, LNCS*, volume 10008, pages 179–187, 2016.

[4] Gloria Guzmán Pérez-Carrillo. The rsna-asnr-miccai brats 2021 benchmark on brain tumor segmentation and radiogenomic classification. 09 2021.

[5] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition, 2015.

[6] Nabil Ibtehaz and M Sohel Rahman. Multiresunet: Rethinking the u-net architecture for multimodal biomedical image segmentation. *Neural networks*, 121:74–87, 2020.

[7] Reza Kalantar, Gigin Lin, Jessica Winfield, Christina Messiou, Susan Lalondrelle, Matthew Blackledge, and Dow-Mu Koh. Automatic segmentation of pelvic cancers using deep learning: State-of-the-art approaches and challenges. 08 2021.

[8] Yann LeCun, Y. Bengio, and Geoffrey Hinton. Deep learning. *Nature*, 521:436–44, 05 2015.

[9] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. *CoRR*, abs/1411.4038, 2014.

[10] Neil Micallef, Dylan Seychell, and Claude Bajada. Exploring the u-net++ model for automatic brain tumor segmentation. *IEEE Access*, PP:1–1, 09 2021.

[11] Fausto Milletari, Nassir Navab, and Seyed Ahmad Ahmadi. V-Net: Fully convolutional neural networks for volumetric medical image segmentation. *Proceedings - 2016 4th International Conference on 3D Vision, 3DV 2016*, pages 565–571, 2016.

[12] Torsten Rohlfing, Natalie M Zahr, Edith V Sullivan, and Adolf Pfefferbaum. The sri24 multichannel atlas of normal adult human brain structure. *Human brain mapping*, 31(5):798–819, 2010.

[13] Olaf Ronneberger, Philipp Fischer, and Thomas Brox. U-net: Convolutional networks for biomedical image segmentation, 2015.

[14] Sofie Tilborghs, Jeroen Bertels, David Robben, Dirk Vandermeulen, and Frederik Maes. The dice loss in the context of missing or empty labels: introducing ϕ and . In *International Conference on Medical Image Computing and Computer-Assisted Intervention*, pages 527–537. Springer, 2022.

[15] Sofie Tilborghs, Jeroen Bertels, David Robben, Dirk Vandermeulen, and Frederik Maes. *The Dice Loss in the Context of Missing or Empty Labels: Introducing*

$$\Phi$$

and

$$\epsilon$$

, page 527–537. Springer Nature Switzerland, 2022.

[16] Dmitry Ulyanov, Andrea Vedaldi, and Victor Lempitsky. Instance normalization: The missing ingredient for fast stylization, 2017.

[17] Matthew Zeiler, Dilip Krishnan, Graham Taylor, and Robert Fergus. Deconvolutional networks. pages 2528–2535, 06 2010.

[18] Teofilo Zosa. Catalyzing clinical diagnostic pipelines through volumetric medical image segmentation using deep neural networks: Past, present, future. 03 2021.

[19] Özgün Çiçek, Ahmed Abdulkadir, Soeren Lienkamp, Thomas Brox, and Olaf Ronneberger. 3d u-net: Learning dense volumetric segmentation from sparse annotation. 06 2016.