GBM6700E – Efficient segmentation of 3D MRI brain tumor using various U-Net architectures

Clémence Duplat (2407959)

December 12, 2024

1 Introduction

In the past years, the use of deep learning techniques in the medical domain has significantly increased. Indeed, it allow us to have a fast and precise way to analyse medical images. For example, this is widely used in the area of brain tumor detection (classification) and segmentation.
However, multiple deep learning architectures exist and each has his own strengths and weakness. Therefore, it could be useful to compare some of them precisely in terms of accuracy and computational efficiency.
If we know that we have a tumor on the scan, we can identify where it is located by performing segmentation. The project aims to find an efficient method for 3D Brain MRI segmentation by comparing 3 architectures: 3D U-Net, 2.5D U-Net and 2D U-Net. We will use the BraTS (Brain Tumor Segmentation) dataset, which consists of multi-modal 3D MRI scans.

2 Data Preparation

2.1 Dataset Explanation

In this project we use the Brats dataset that contains multimodal MRI scans 1.

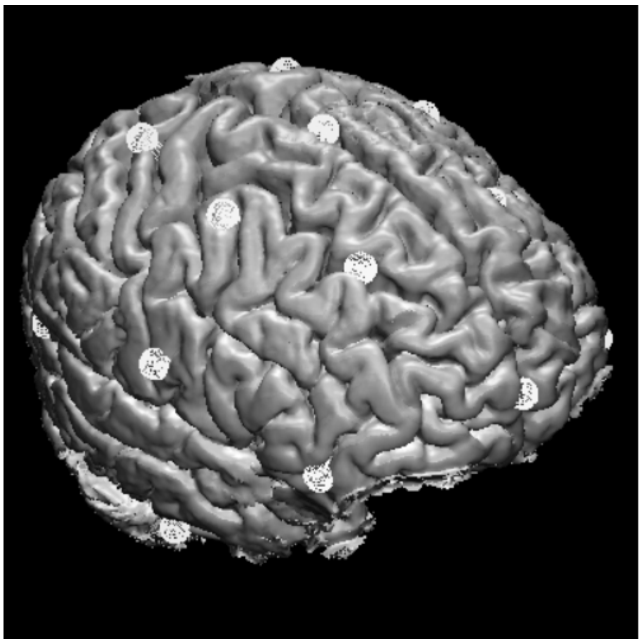

Figure 1: 3D MRI image

Those images depend on the echo time (TE) and the repetition time (TR) chosen during the MRI scan, allowing us to reveal different properties of the tissue:

1. T1 : highlight fat-rich tissues, useful to indetify boundaries
2. T1ce : T1 with contrast enhancement which improves the visibility of abnormalities
3. T2 : highlight fluid content
4. FLAIR : Fluid attenuated inversion recovery, suppresses fluid signals

The images are NIfTI files (Neuroimaging Informatics Technology Initiative) and are composed of 3 dimensions. One slice is one dimension, and the images are a serie of two-dimensional images. The 3 dimensions refer to 3 different planes that each provides us a different perspective:

1. Axial (Transverse) Plane
2. Coronal (Frontal) Plane
3. Sagittal (Lateral) Plane

The images come with expert-annotated segmentation masks. it will delineate the tumor into various sub-regions, such as the necrotic and non-enhancing tumor core, the peritumoral edema, and the enhancing tumor. The annotations are the following and can be seen in Figure 2:

- Label 0: Not Tumor (NT) volume
- Label 1: Necrotic and non-enhancing tumor core (NCR/NET)
- Label 2: Peritumoral edema (ED)
- Label 3: Missing (No pixels in all the volumes contain label 3)
- Label 4: GD-enhancing tumor (ET)

As there are no pixels with the label 3, we will be replacing label 3 with label 4 so that there is continuity between the labels.

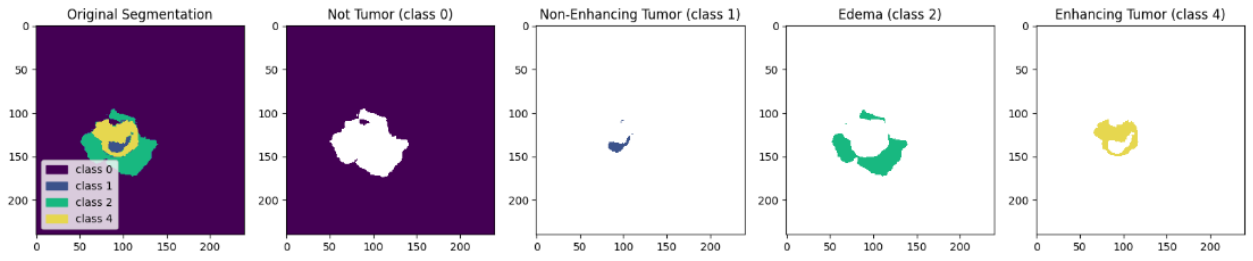

Figure 2: 2D segmentation

We could also create a binary mask of shape with different gray intensities. We can plot in Figure 3 the segmentation of a 2D plane across the different modalities in order to help visualizing the tumor.

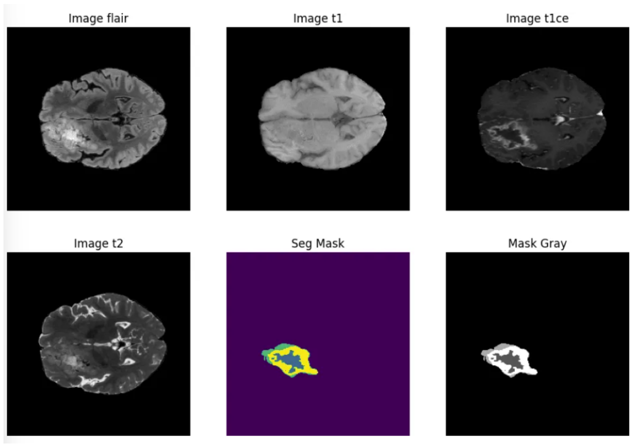

Figure 3: GLI African Dataset Modalities Visualization

2.2 Dataset Preprocessing

We need to prepare our MRI scans in order to train our different U-Net models later. This will ensure that we have consistent input formats, an improvement of the model performance and a shorter training time.
This can be done by methods like resampling (to ensure spatial consistency), intensity normalization and cropping.
Additionally we can use a pretrained model in order to be able to run the code on our computet. Here we will use MONAI Model Zoo (3D) or Kaggle(2D).

2.3 Splitting the dataset

We are going to split our dataset in 3 parts in order to train and evaluate our model effectively.

- Training: 80 % of the dataset to learn the mapping between input and output images
- Validation : 10% of the dataset to prevent overfitting (model learns the noise and will not act well on unseen data) and find the best set of hyperparameters. This happens druing the training.
- Testing : 10% of the dataset for evaluating how our model perform on unseen data. This happens after the training, and is done using the best set of hyperparameters or the best model.

3 Evaluating the model

3.1 Loss function

We need an appropriate loss function when training our CNN such that the model improve.
We can use the Cross-Entropy loss or the dice loss function. The cross-entropy loss measures the difference between the predicted probability distribution and the true labels. Additionally we can use the Dice Loss function that is a measure of the overlap between the true and the predicted segmentation.

We will also use the Adam Optimizer, which allow us for an adaptive learning rates and a faster convergence. We choose an initial learing rate of $1e-3$.
An epoch is one complete pass through the entire training dataset during training. This is an hyperparameter that will need to be tuned during the validation step. The steps per epoch are the number of batches processed in each epoch, and this is also an hyperparameter.

3.2 Metrics

The metrics are used during the training, validation and testing. During the validation they are used to decide when to stop training or adjust hyperparameters. During the tetsing they are used as a final evaluation.

3.2.1 Dice Coefficient (DC)

It is a measure of the overlap between the true and the predicted segmentation, and its formula is shown in Figure 4. A value of 1 indicate a perfect overlap (very good segmentation performance) while a value of 0 indicates no overlap at all (poor segmentation performance).

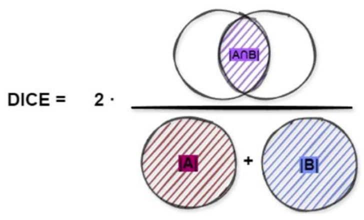

Figure 4: Dice Coefficient

3.2.2 Intersection Over Union (IoU)

It is another way to measure the overlap between the true and the predicted segmentation, but it is stricter then the Dice Coefficient. It is shown in Figure 5.

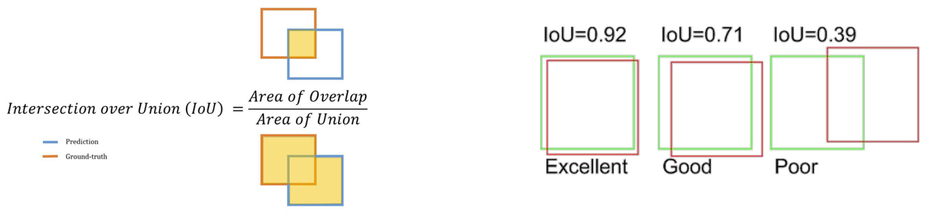
Intersection over Union (IoU) = Area of Overlap / Area of Union
Prediction
Ground-truth
IoU=0.92
IoU=0.71
IoU=0.39
Excellent
Good
Poor

Figure 5: IoU

3.2.3 Mean Accuracy (MA)

It measures how accurately the model classifies pixels into the correct categories. A high accuracy means that the model performs well overall. However, it will not focus on our region of interest. When the region of interest is small compared to the background, looking only at the MA is thus always enough.

3.2.4 Other

To go further we could even evaluate:

- Sensitivity : Recall or True Positive Rate
- Precision : Positive Predictive Value.
- Specificity: True Negative Rate

Additionally, it is interesting to measure the training, validation and application time.

The memory usage, how much RAM is needed to store and process data, is preferred to be low.

4 Architectures

4.1 2D U-Net architecture

We have two main parts that are forming a ”U”: a contracting path and an expansive path.

The **contracting path** acts as a feature detector. At each step we have a convolutional layer, that will highlight the important features in the image, and a max-pooling layer that will reduce the size of the image while keeping the most relevant information. The deeper we go in the path, the more abstract the extracted features become.
Between the two path, we have the **bottleneck** that will make a compact representation of all the extracted features.
The **expansive path** is a mirror of the contracting path: it will take the compact and extracted features to reconstruct the image, but this time highlighting the region of interest (here the tumor). It will use upsampling layers (Transposed Convolutions) to increase the size of the image step by step.
We also notice that we have **skip connections**, that will directly transfer the details from the contracting path to the expansive path. This will ensure that the network combines the low level features in the starting with the high level features in the expansive path. After this concatenation we perform a double convolution to refine the combined features.

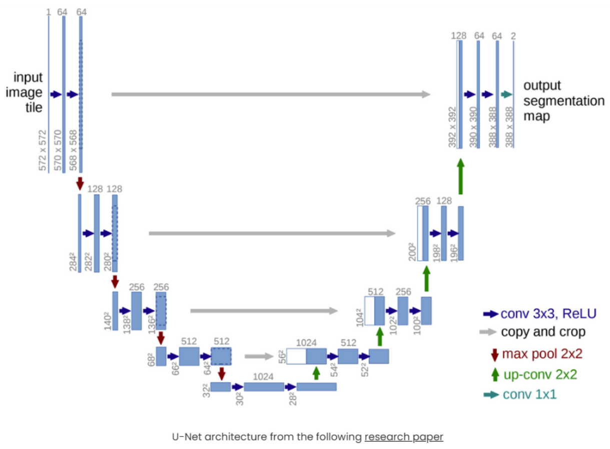

Figure 6: 2D U-Net architecture

4.2 3D U-Net architecture

Here, the difference with the 2D U-Net architecture is that we use 3D convolutions and 3D pooling layers.

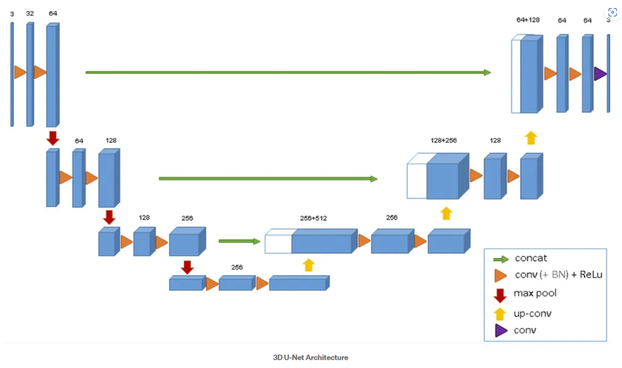

Figure 7: 3D U-Net architecture

4.3 2.5D U-Net architecture

This architecture will combine the strengths of the 2D U-Net architecture and 3D U-Net architecture. The model proposed in the paper 1 works as follow:

1. **Projections with Maximum Intensity Projections (MIP):**
   It will project the 3D MRI data onto multiple 2D planes using MIPs. For the amount of projection directions we choose equidistant angles as in the paper. This way we have a simpler data to process, but it will still retain the 3D context.

2. **Segmentation:** using the 2D U-Net architecture on each plane
3. **Reconstruction:** into a 3D MRI data. This will be done using a trainable reconstruction operator. This operator will ensure that the information from each projection is aligned correctly in 3D space.

It will thus process 2D slices from different orientations in order to capture volumetric information from 3D data and reconstruct the 2D slices into a 3D volume.

5 Results

5.1 2D U-Net

We can track the model progress over the number of epochs in Figure 8.
We see on the accuracy plot that the training and validation accuracy increases both closely to each other over the epochs. This indicates a good generalization without overfitting. We see on the loss plot that both curves are decreasing close to each other.
On the Dice Coefficient plot we see an increase of this metric both for the training and the validation, before reading a steady-sate around 0.65.
The Mean IoU plot shows some fluctuations during the early epoch for the validation curve, but it stabilizes as the model converges.

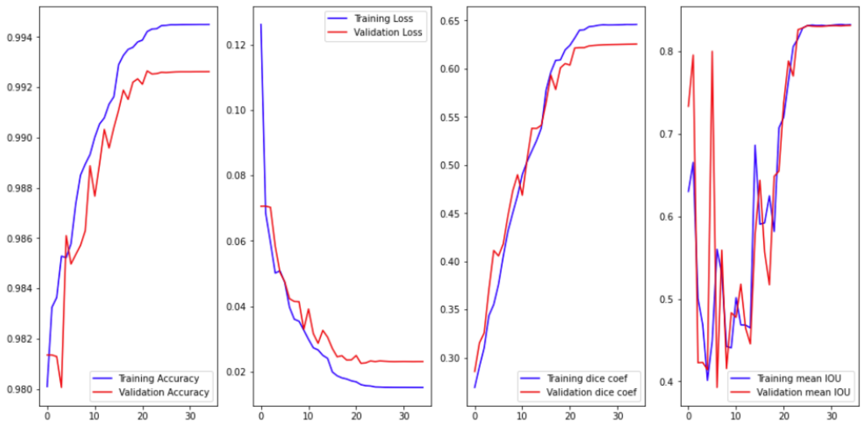

Figure 8: 2D U-Net: Learning curves

Across all metrics, we see that the model converges well after 19 epochs. Taking a bigger epoch may lead to overfitting (model will not perform well on unseen data). Therefore we will use this, the best saved model, to predict segmentation on our test dataset. We can see an example of this prediction in Figure 9.

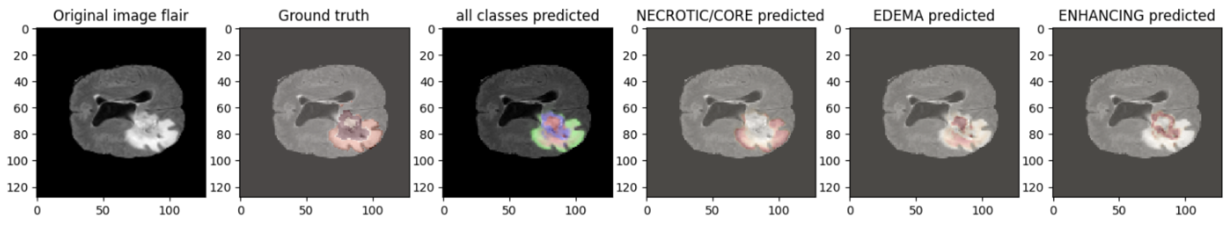

Figure 9: 2D U-Net: Prediction example

The Model Evaluation on the test set for one single axial scan is

```
Loss : 0.0267
Accuracy : 0.9931
MeanIOU : 0.8426
Dice coefficient : 0.648
```

Additionally we have the Dice Coefficients for different regions shown in Table 1

Table 1: 2D U-Net: Dice Coefficients for Tumor Segmentation Regions in axial scan

Region	**Dice Coefficient**
Enhancing Tumor (ET)	0.7395
Whole Tumor (WT)	0.648
Tumor Core (TC)	0.5916

We see that the training and application time are very short, we indeed talk in ms for each step.

However, those are the results for a single scan (axial). In order to compare those results to the 3D U-Net, we need to combine the 2D predictions slice-by slice into a single 3D volume. We thus need to take into account the 3 different views (axial, coronal and sagittal). We will see that the axial images perform well, while the coronal and sagittal are performing less well (dropping to 0.6). We have the same drop of performance for the DC and IoU. The 2D U-Net is therefore not always an optimal model for our 3D BratsDataset. However the 2D U-Net acts better on the BratsDataset compared to the results proposed in the paper 1, but using a different dataset.

5.2 3D U-Net

In Figure 10 we can make the same observations in Figure 8, but this time the curves will reach a different steady-state. We can see that after more or less 30 epochs, the performance of the model does not improve anymore. To avoid the overfitting (that model learns the noise in the data too), we will thus choose to keep 30 epochs for our best model.

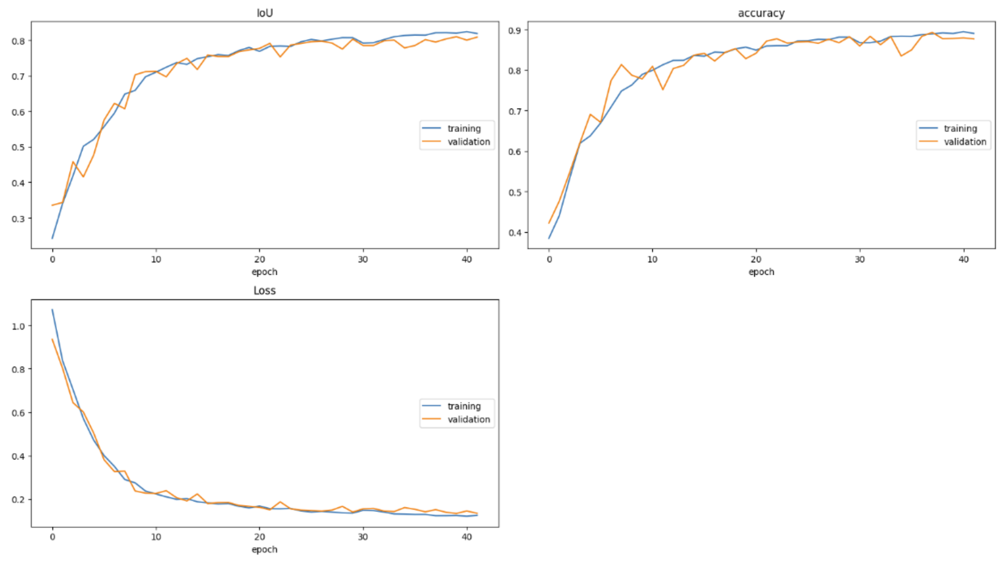

Figure 10: 3D U-Net: Learning curves

Using our best saved model we can see how our model performs on unseen data by using it on the test set. The visualization can normally be seen in a 3D animation but I was unable to put it in my overleaf document. I can thus show one slice in Figure 11. We see that, compared to the 2D U-Net, the image has been cropped beforehand in order to delete the background information and thus gain some computational efficiency.

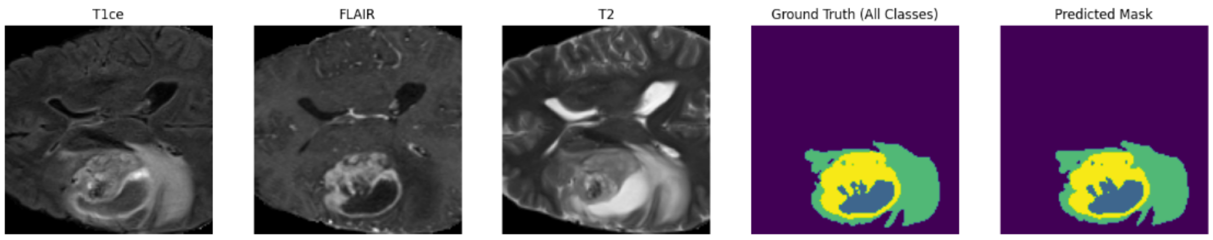

Figure 11: 3D U-Net: Prediction

The Model Evaluation (for a 3D image) on the test set is

```
Loss : 0.1191
Accuracy : 0.8939
MeanIOU : 0.8252
Dice coefficient : 0.86171
```

We can even look at the Dice Coefficient for the different tumor parts in Table 2.

Table 2: 3D U-Net: Dice Coefficients for Tumor Segmentation Regions

Region	**Dice Score**
Enhancing Tumor (ET)	0.65446
Whole Tumor (WT)	0.86171
Tumor Core (TC)	0.75757

We see that the training time and application time of this model is much higher than for the 2D U-Net model (we are talking in seconds instead of ms for each step). Therefore, it was really important to use a pre-trained model to reduce the timing and the memory usage.

5.3 2.5D U-Net

Unfortunately, I wasn't able to implement the 2.5D U-Net architecture proposed in the paper. However I found another paper 7 that presented a different 2.5D U-Net Architecture and directly applied it on the Brats Dataset. The architecture will still process the volumetric data more efficiently by using different 2D slice rather that the 3D volumes, but this time it will not use the MIPs. The model works as follow (and is shown in Figure 12) :

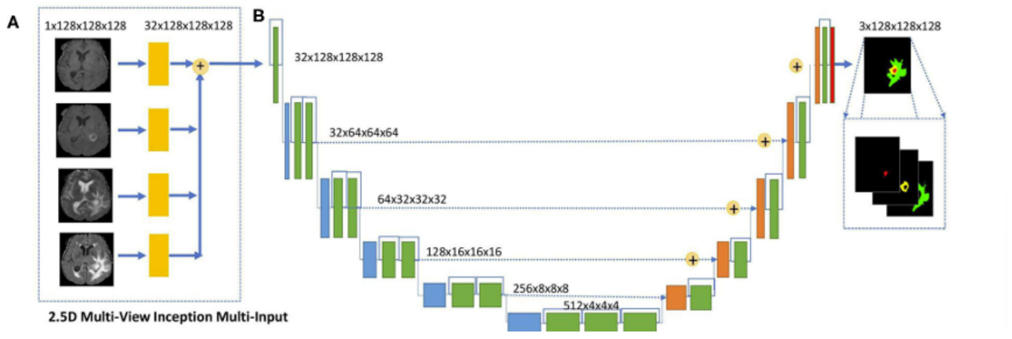

Figure 12: 2.5D Multi-View Inception Multi-input

- Multi-Modal input: each input is represented by the 4 IRM modalities (T1, T1c, T2 and FLAIR) and each modality is processed independently at the beginning. This will extract low-level features.
- 2D Convolution layers: to extract local features from the slices in different anatomical planes (axial, coronal and sagittal).

- 2.5D Multi-View inception block (Figure 13): combine the information from different planes in order to capture the contextual information in 3D (without processing the full 3D image). The kernels are anistropic, meaning they treat one dimension differently from the others.
- Features from all modalities and anatomical planes are fused to create a comprehensive representation of the region of interest.
- 2D U-Net: we pass those fused features in the 2D U-Net architecture.

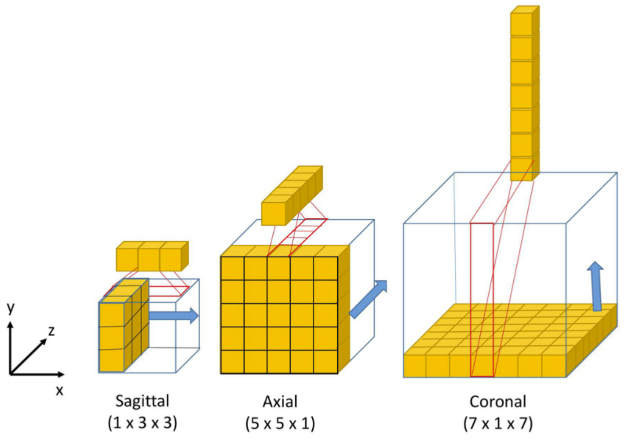

Figure 13: 2.5D Multi-View inception block

In the paper, the model is trained on the BraTS DataSet. The performance during the validation and traing are given in Figure 14.

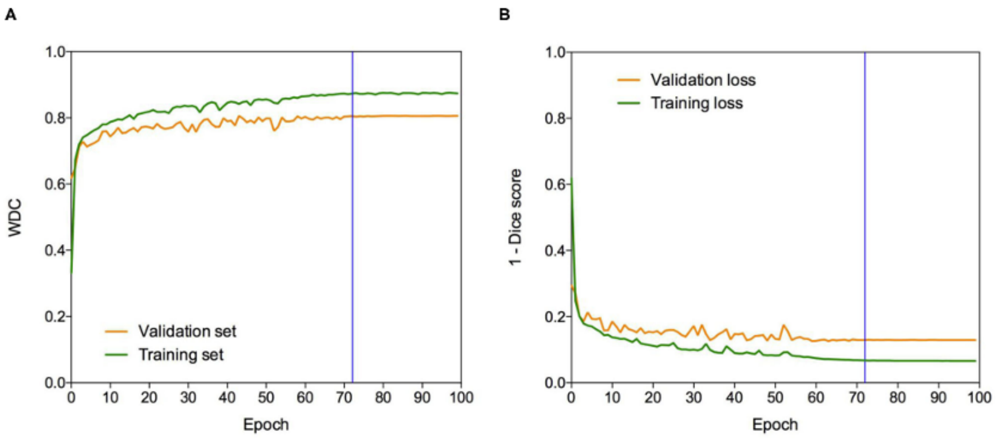

Figure 14: 2.5D: Learning curves. Example of (A) WDC (Weighted-DC) evaluation metric and (B) (1-Dice score) loss function evolution during training on the BraTS 2019 dataset. The blue vertical line shows the moment at which the model reached the best performance in the validation dataset. The best models were obtained systematically before the 100 epochs.

We can even look at the Dice Coefficient for the different tumor parts in Table 3.

Table 3: 2.5D U-Net: Dice Coefficients for Tumor Segmentation Regions

Region	**Dice Score**
Enhancing Tumor (ET)	0.775
Whole Tumor (WT)	0.865
Tumor Core (TC)	0.789

Additionally, this model ensure that the memory consumption is low compared to the fully 3D U-Net.

6 Conclusion

By comparing the different tables in terms of DC, IoU, Accuracy Scores and the different timings we can conclude the following:

The 3D U-Net will perform volumetric segmentation using 3D convolutions but can be computationally efficient and time consuming. The 2D U-Net is more efficient but processes individual MRI slices without volumetric context. The 2.5D U-Net will be a compromise between both: processing projections of 3D data with 2D convolutions. It is thus faster and retain some 3D information. We even see that the 2.5D U-Net model performs better than the 3D U-Net model. This could be explained by the fact that the 2.5D U-Net focus on the most relevant spatial planes. This is summarized in the following Table 4.

Architecture	**Advantages**	**Disadvantages**
2D U-Net	Fast, low memory usage	Lack volumetric context
2.5D U-Net	Efficient, captures some 3D context	Requires a good preprocessing
3D U-Net	Full volumetric context	High memory and training time

Table 4: Comparison of U-Net Architectures

7 References

Documents:

1. https://arxiv.org/pdf/1902.00347
2. https://arxiv.org/pdf/1505.04597
3. https://blog.paperspace.com/unet-architecture-image-segmentation/#:~:text=U-Net%20Architecture%20For%20Image%20Segmentation.%20Image%20segmentation%20makes%20it%20easier
4. https://apprendre-le-deep-learning.com/u-net-une-architecture-pour-la-segmentation-d-images/#:~:text=D%C3%A9couvrez%20U-Net,%20l'architecture%20de%20r%C3%A9seau%20de%20neurones%20r%C3%A9volutionnaire%20pour%20la
5. https://www.frontiersin.org/journals/neurology/articles/10.3389/fneur.2021.609646/full?utm_source=chatgpt.com
6. https://learnopencv.com/3d-u-net-brats/
7. https://doi.org/10.3389/fneur.2021.609646

Code:

1. 2D U-Net: https://www.kaggle.com/code/krn0209/unet2d https://www.kaggle.com/code/zeeshanlatif/brain-tumor-segmentation-using-u-net
2. 3D U-Net: https://github.com/hodlen/brats2020-keras/tree/master https://github.com/spmallick/learnopencv/tree/master/Training_3D_U-Net_Brain_Tumor_Seg
3. 2.5D U-Net: https://github.com/mehta-lab/microDL/tree/main