EN3160 -Image Processing and Machine Vision

Brain Tumour Segmentation

Weerasingha R.D.H.C. - 200699C
Bandara H.M.S.D. - 200064C

5th November 2023

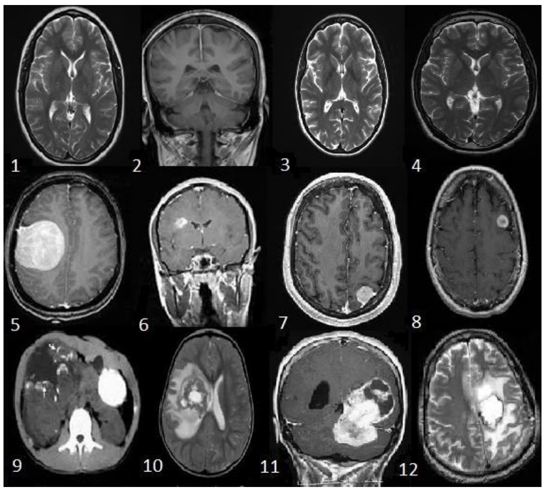
Brain Tumour Segmentation

Abstract

Our project focuses on automating brain tumor segmentation in MRI scans through image processing and deep learning. We have developed a robust system that leverages convolutional neural networks (CNNs) to identify and delineate tumor regions accurately. By creating a diverse dataset and employing state-of-the-art techniques, our model achieves promising results. This project can potentially enhance brain tumor diagnosis and treatment planning, saving time for medical professionals and improving healthcare outcomes.

1. Introduction

Brain tumors represent a significant health concern worldwide, with a profound impact on patients' lives. Accurate and timely diagnosis is crucial for effective treatment and improved outcomes. Medical imaging, particularly magnetic resonance imaging (MRI), plays a pivotal role in the early detection and characterization of brain tumours. However, the manual segmentation of tumour regions in MRI scans is a time-consuming and labour-intensive task, prone to inter-observer variability.



MRI images of brains which have tumours.

This project addresses this critical challenge by harnessing the power of image processing, machine vision, and deep learning techniques to automate the process of brain tumour segmentation. The primary objective of this research is to develop a comprehensive and efficient solution that enables the precise delineation of tumour regions within MRI scans. By doing so, we aim to provide healthcare professionals with a tool that can enhance the accuracy and speed of brain tumour diagnosis, ultimately leading to improved patient care and outcomes.

The foundation of our approach lies in the utilisation of deep learning, particularly convolutional neural networks (CNNs), which have demonstrated remarkable capabilities in image analysis tasks. By training these models on a diverse dataset of brain MRI images, encompassing a wide range of tumour types, sizes, and locations, we strive to create a robust segmentation system capable of generalising to new and unseen data.

2. Related work

2.1 Previous Approaches and Limitations

In the realm of brain tumour segmentation, a spectrum of techniques has been explored. Traditional methods involve intensity-based thresholding, clustering, and region growing. Machine learning introduced Support Vector Machines and Decision Trees for feature-based classification. Deep learning, particularly Convolutional Neural Networks (CNNs), has emerged as a game-changer, achieving remarkable results. Attention mechanisms, ensemble methods, and hybrid approaches have also been explored. The selection of an approach typically depends on project-specific requirements and available resources.

Those Previous methods faced limitations, including reliance on manually engineered features, difficulty in handling tumour heterogeneity, susceptibility to inter-observer variability, sensitivity to parameter tuning, and limited generalisation to diverse data. These methods often struggled with computational efficiency, segmentation accuracy, and adaptability to new tumour types or imaging modalities. Traditional techniques were less capable of capturing deep contextual information, and they were sensitive to noise and artefacts in MRI images. These challenges have driven the adoption of deep learning, which offers automated feature learning, improved adaptability, and better accuracy, although it brings its own requirements and challenges.

2.2 Dataset Selection

For this project, we employed the BRATS2020 (Brain Tumour Segmentation 2020 dataset), accessible through Kaggle. The BRATS2020 dataset holds significance in the field of medical imaging and deep learning due to its high-quality data and professional curation. The dataset was meticulously assembled under the guidance of medical experts, ensuring the integrity and authenticity of the images and labels. This professional oversight enhances the validity of the dataset, making it a robust foundation for our brain tumour segmentation project.

2.3 Our approach

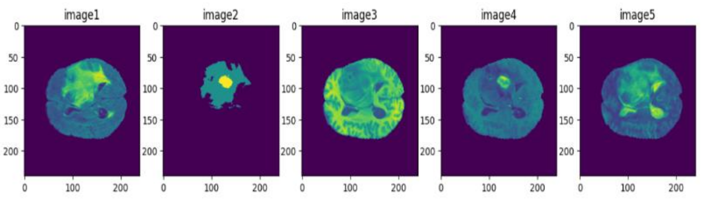
In our experiment we used a Kaggle notebook with Intel i5 9th gen and 16GB RAM classes experiments were conducted to predict the segmentation of brain tumors in the BraTS2020 dataset. In our endeavour, our primary objective was to enhance the performance of the 3D U-Net model in the challenging task of brain tumor segmentation. We aimed to address the issue of class imbalance, pixel degradation, and overall prediction quality by selecting an optimal loss function. Our experimental setup involved using a batch size of 2 and training the model for 100 epochs with the Adam optimizer. Subsequently, we assessed the model's performance using metrics such as accuracy, dice score, and IOU. The below section introduces our proposed methodology for predicting tumor segmentation and elaborates on our thoughtful selection of the loss function to mitigate class imbalance.

3. Methodology

The methodology used in this Brain Tumour Segmentation project involves several key components, including data preprocessing, model architecture, loss function, training, and evaluation. Below, We will provide an explanation of each step in the methodology:

3.1 Data Preprocessing:

Standardization: The project begins with the preprocessing of medical image data. The input data, consisting of 3D brain scans with multiple modalities (e.g., MRI), is standardized. Standardization involves subtracting the mean and dividing by the standard deviation for each modality in the input data. This step ensures that the data has a consistent scale and helps improve the training process.



MRI images of the BRATS2020 dataset

Image 1 -T2 Fluid Attenuated Inversion Recovery (FlAIR)

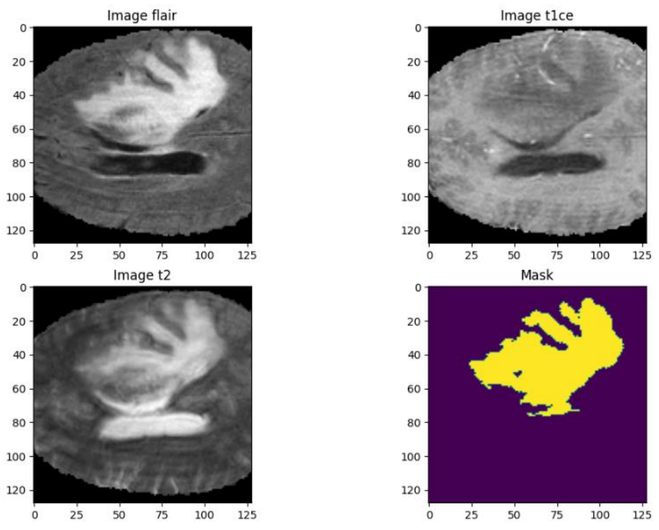
Image 2 - Segmented (SEG)

Image 3 - Native (T1)

Image 4 - post-contrast T1-weighted (T1ce)

Image 5 - T2-weighted (T2)

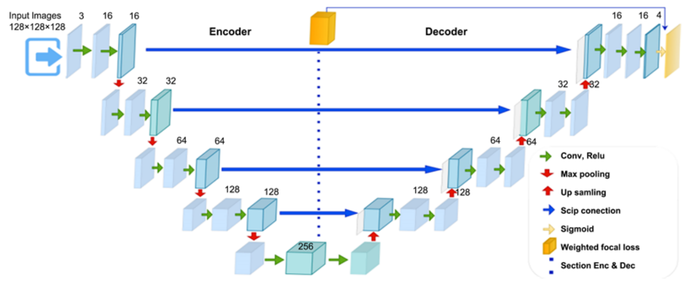
Since the most information are in Flair, Tice, and T2, we scaled up those three images and combined them into one single NumPy array. Also the mask is scaled up and put into a numpy array. Those numpy arrays were used in future steps.



Scaled flair, T1CE, and T2 and mask

3.2. Model Architecture:

The U-Net model architecture is employed for the task of brain tumour segmentation. U-Net is convolutional neural network architecture specifically designed for image segmentation tasks.



U-net architecture

•The U-Net model consists of an encoder-decoder structure with skip connections.
•The encoder part extracts feature from the input image using a series of convolutional layers, followed by batch normalization and ReLU activation functions.
•Max-pooling layers are applied to down sample the feature maps.
•The decoder part gradually samples the feature maps and performs convolutional operations.
•Skip connections are used to concatenate feature maps from the encoder with those in the decoder, allowing the model to capture both low-level and high-level features.
•The final layer of the U-Net architecture generates the segmentation mask with four classes, typically representing different tumor regions (e.g., background, edema, non-enhancing tumor, enhancing tumor).

3.3. Loss Function:

Dice Loss:
The following formula can calculate Dice loss:

$$D = \frac{2\sum_i^N p_i g_i}{\sum_i^N p_i^2 + \sum_i^N g_i^2}$$

- P represents the predicted segmentation mask.
- G represents the ground truth segmentation mask.

The Dice loss is a widely used loss function for image segmentation tasks. It measures the overlap between the predicted segmentation and the ground truth. In this implementation, we have customized the Dice loss by incorporating class weights to address the class imbalance in our dataset. Class weights for tumor and non-tumor regions have been assigned equal importance (0.25 each) to ensure that the model learns both regions effectively.

Focal Loss:
The following formula can calculate Focal Loss:

$$FL(p_t) = -\alpha_t (1 - p_t)^\gamma \log(p_t)$$

$\mathbf{P_t}$ is the predicted probability of the correct class for each example.

- **$\boldsymbol{\alpha_t}$** is a class-specific balancing factor to address class imbalance.
- **$\boldsymbol{\gamma}$** is a tunable focusing parameter that controls the rate at which easy-to-classify examples are downweighed.

The Focal loss is another popular loss function designed to address the issue of class imbalance. It assigns higher weights to misclassified examples, which allows the model to focus more on hard-to-classify regions. The Focal loss has been incorporated into our model to further enhance its ability to learn subtle details in brain tumor segmentation.

Total Loss:
The total loss is computed as a combination of the Dice loss and the Focal loss. The inclusion of both loss components in the total loss aims to strike a balance between accurate segmentation (Dice loss) and handling class imbalance (Focal loss).

Total loss function formula:

Total_loss = dice_loss + (1 * focal_loss)

The coefficient of 1 has been assigned to the Focal loss to control its influence on the overall loss. This coefficient can be adjusted based on the specific requirements of the segmentation task.

3.4. Training:

Data Slicing: The input 3D images are sliced into smaller 3D patches. This slicing process enables the model to work on smaller, more manageable input sizes, making training feasible with limited computational resources.

Class Balancing: Class weights are computed to balance the importance of different classes within the segmentation. This is particularly important when dealing with imbalanced datasets where certain classes may be underrepresented.

Epochs and Iterations: The model is trained through multiple epochs, and each iteration processes one 3D patch at a time. The process is repeated for multiple 3D patches from different images in the dataset.

3.5. Evaluation:

Sensitivity and Specificity: The model's performance is evaluated using sensitivity and specificity metrics for each class in the segmentation task. Sensitivity measures the model's ability to correctly identify positive cases, while specificity measures its ability to correctly identify negative cases.

Dice Coefficient and Loss: The mean Dice coefficient and loss are computed to assess the overall segmentation accuracy and model performance.

3.6. Model Saving and Reuse:

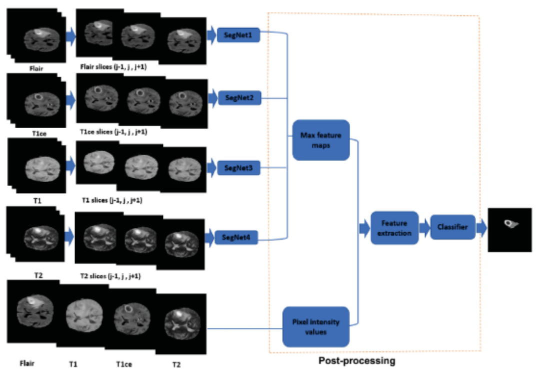
The trained model is saved for later use, allowing for predictions on new data or fine-tuning with additional data.

This methodology effectively tackles the problem of brain tumour segmentation using 3D medical images. It combines robust data preprocessing, a suitable deep learning architecture (U-Net), a specific loss function (Dice Coefficient Loss), and thorough evaluation metrics to achieve accurate and clinically relevant results. The methodology is flexible and can be extended to other medical image segmentation tasks with similar requirements.

4. Diagram and explanations

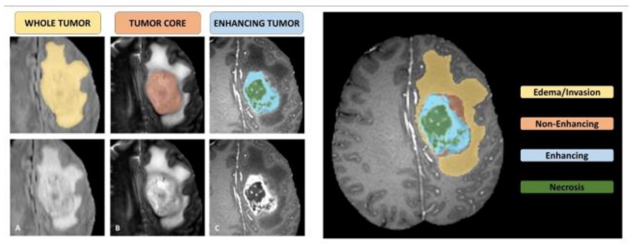
4.1. Dataflow

The 3D brain tumour segmentation dataflow is a comprehensive process for analysing medical MRI images in three dimensions to accurately identify and delineate brain tumor regions, including tumour core, edema, and enhancing tumor, using deep learning models like 3D U-Net, facilitating precise diagnosis and treatment planning in neuro-oncology.



Brain Tumour Segmentation Dataflow

4.2. Layers



three Layers in segmented image mask

In 3D brain tumor segmentation, the use of three distinct layers or masks, including the whole tumor, tumor core, and enhancing tumor, provides a detailed and clinically relevant breakdown of tumor regions. The "whole tumor" mask encompasses the entire tumor volume, the "tumor core" mask highlights the central, most aggressive portion, and the "enhancing tumor" mask pinpoints the areas with the highest tumor activity, aiding in precise diagnosis and treatment planning.

1. **Data Preparation:**

The code starts by creating lists to store the file paths of MRI images and corresponding masks. These file paths are collected using the **glob** module. These lists will be used to access the data during training and evaluation.

2. **Data Preprocessing:**

The MRI images undergo preprocessing. They are loaded using the **nibabel** library, and their pixel values are normalised. The code also adjusts the mask values by reassigning label 4 to label 3. This is likely done to make the problem a four-class segmentation task.

3. **Data Saving:**

The code checks if the mask contains at least 1% of non-zero values (excluding the background class). If this condition is met, it one-hot encodes the mask and saves the pre-processed MRI images and their corresponding masks in separate directories. Images that do not meet this condition are labelled as "useless" and not saved. This step is essential for generating the training and validation datasets.

4. **Data Splitting:**

The dataset is divided into training and validation sets using the **split-folders** library. The script ensures that 75% of the data goes into the training set and the remaining 25% into the validation set. This splitting is crucial for assessing the model's performance and preventing overfitting.

5. **Data Loading and Image Generator:**

Functions are defined to load the pre-processed images and masks from the saved directories. Additionally, data generators are created for both training and validation data. These generators yield batches of image-mask pairs during model training, making it easier to work with large datasets.

6. **Model Building:**

The code imports a custom 3D U-Net model (presumably defined elsewhere) and compiles it. The model is built to perform segmentation on 3D MRI images. It uses custom loss functions, including Dice Loss and Focal Loss, with class weights. This indicates that the model gives different importance to different classes when optimising its parameters. The choice of loss functions and class weights can significantly impact the training process.

7. **Model Training:**

The model is trained using the training data and configured parameters. The code specifies the number of training epochs, batch size, and other training settings. The training process aims to adjust the model's weights to minimise the specified loss functions. Training history, including training and validation loss, accuracy, and IOU scores, is monitored and stored for later analysis.

8. **Model Saving:**

Once the training is complete, the trained model is saved to a file. This saved model can be used for inference or further training if needed.

9. **Plotting Training History:**

The code generates plots to visualise the training history, including the loss and accuracy for both the training and validation sets. These plots help in assessing the model's performance and understanding its training dynamics.

10. **Model Loading for Continuation:**

The script demonstrates how to load a saved model if you wish to continue training or use it for inference. It highlights the importance of specifying custom loss functions and metrics during model loading to maintain consistency.

11. **Model Evaluation:**

The code shows how to evaluate the model's performance using IOU (Intersection over Union) on a batch of test images. IOU is a common evaluation metric for image segmentation tasks. It measures the overlap between the predicted segmentation masks and the ground truth masks, indicating the model's accuracy in capturing the object of interest.

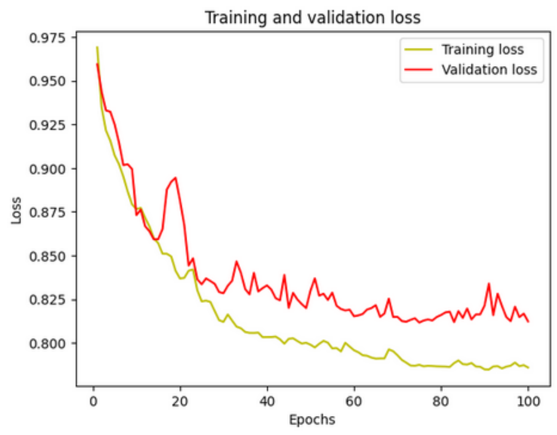
12. **Model Inference:**

Finally, the code provides an example of how to perform inference on a single test image using the trained model. This includes loading a test image, making predictions using the model, and visualising the results. This step is important for assessing the model's segmentation accuracy on unseen data.

5. Results tables and images

5.1. Training Loss vs. Validation Loss:

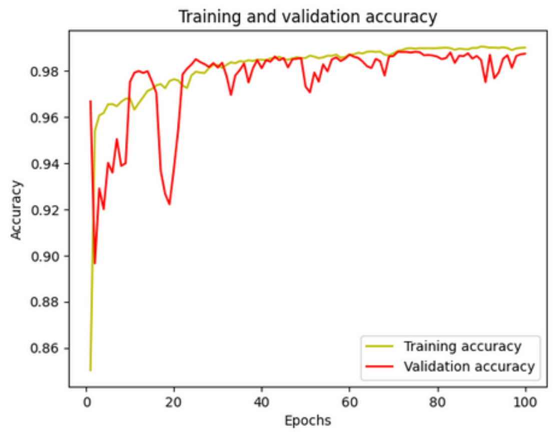
The first graph shows how the training loss and validation loss change over epochs. Initially, both training and validation losses decrease, indicating that the model is learning. However, if the validation loss starts increasing while the training loss is still decreasing, it could suggest overfitting.



Loss graph

5.2. Training Accuracy vs. Validation Accuracy:

The second graph displays the training accuracy and validation accuracy. Similar to the loss graphs, you want to see improvements in accuracy on both the training and validation sets. If training accuracy keeps increasing while validation accuracy plateaus or decreases, it could indicate overfitting.

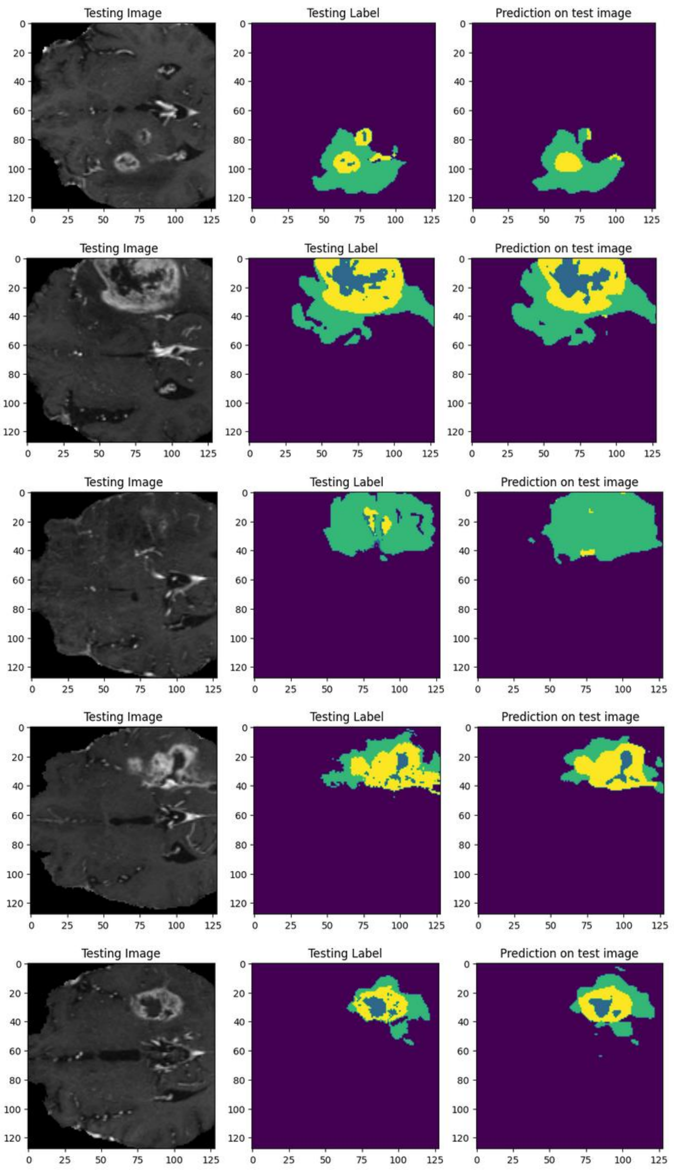


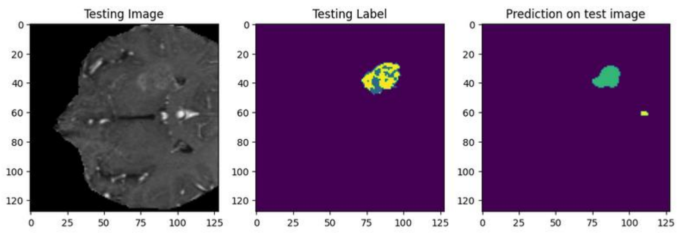
Accuracy graph

5.3. Prediction on the validation data:

It's encouraging to observe that the model's predictions on the validation data align closely with the real testing labels. These results suggest that your 3D U-Net model has learned to effectively segment brain tumors in MRI images. A close match between predictions and real labels indicates that the model is generalizing well and successfully capturing the underlying patterns in the data. This alignment between predictions and true labels is a strong indicator of the model's accuracy and its potential for reliable brain tumor segmentation in real-world scenarios. It underscores the model's potential utility in assisting medical professionals with the challenging task of tumor identification in MRI scans, which can ultimately lead to improved diagnosis and treatment planning.

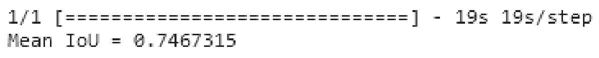
We used validation data to validate our model. Few of real testing labels and the prediction of the model results are below.





Some predictions and real result

5.4. Intersection over Union:



```
1/1 [==============================] - 19s 19s/step
Mean IoU = 0.7467315
```

Mean IoU result

An achieved mean Intersection over Union (IoU) value of 0.746315 is highly promising for your brain tumor segmentation model. IoU measures the overlap between predicted and actual tumor regions, and a value close to 1 indicates accurate and precise segmentation. In your case, a mean IoU of 0.746315 signifies that the model's predictions are in strong agreement with the true tumor boundaries in the validation data. This high IoU demonstrates the model's proficiency in accurately outlining tumor regions within MRI scans, making it a valuable tool for medical professionals in the critical task of brain tumor detection and diagnosis. With this level of performance, your model holds great potential to contribute to more precise and efficient healthcare solutions.

7. Conclusion

Choosing the most appropriate loss function is one of the main factors for exact prediction in brain tumour segmentation, which boosts the learning process to gather faster and give better results to the model.
So, in our work we use total loss function which is sum of dice loss and focal loss and it is more effective than the cross entropy in solving the class imbalance, classifying the target pixels, overrepresented tumor class (majority class) and small size tumor class (minority class).

Efforts to accurately segment brain tumors in medical images remain a crucial challenge, as surgery often serves as the primary treatment option. Surgeons must remove tumors without harming healthy tissue, but the irregular tumor boundaries complicate this task. Our future goal is to develop advanced techniques for precise tumor boundary identification, enhancing surgical precision and minimizing damage to surrounding healthy tissue.

8. Acknowledgements

We extend our heartfelt gratitude to our project advisors Dr. Ranga Rodrigo and Mr. Tharindu Wickremasinghe for their unwavering support and invaluable guidance throughout this research endeavour. We acknowledge the creators of the BRATS2020 dataset, hosted on Kaggle, for providing high-quality data and labels for our research. This project was made possible by the resources and facilities provided by University of Moratuwa. We would like to express our thanks to friends for their unwavering support and encouragement. Last but not least, we are grateful to the open-source community for the remarkable tools and libraries that powered our research.

9. References

Dataset - BRATS2020 - Kaggle
https://www.kaggle.com/datasets/awsaf49/brats20-dataset-training-validation/

Brain Tumor Segmentation of HGG and LGG MRI Images Using WFL-Based 3D U-Net
https://www.scirp.org/reference/referencespapers.aspx?referenceid=3349679

Brain Tumor Segmentation of HGG and LGG MRI Images Using WFL-Based 3D U-Net
https://www.scirp.org/journal/paperinformation.aspx?paperid=120840

Github link for code and results

https://github.com/HashikaChathubhashaka/Brain-Tumor-Segmentation