Glioblastoma Segmentation using Deep Learning Techniques

University of
St Andrews

Mohak Anand Misra

Faculty Advisor: Dr David Harris-Birtill

University of St Andrews School of Computer Science

St Andrews, Scotland

14/03/2024

Contents

List of Figures **v**

List of Tables **x**

1 Introduction **2**

1.1 Motivation 3

1.2 Objectives 5

1.2.1 Primary Objectives 5

1.2.2 Secondary Objectives 6

1.3 Ethical Considerations 6

1.4 Data 6

1.5 Background 15

1.5.1 Image Segmentation 15

1.5.2 Deep Learning Techniques 16

1.5.3 Convolutional Neural Networks (CNNs) 16

2 Literature Review **19**

2.1 Clinical Background 19

2.2 Datasets 21

2.3 Segmentation Methods and Algorithms 23

2.4 State-of-the-Art Methods 25

2.4.1 U-Net . . . 25
2.4.2 V-Net . . . 27
2.4.3 Multi-path architecture . . . 28
2.4.4 Encoder–Decoder Architecture . . . 31
2.4.5 Network Cascade . . . 33
2.4.6 Transformers . . . 37
2.4.7 Summary . . . 40
2.5 Radiogenomic Classification . . . 42
2.5.1 First Position in Task 2 of the BraTS Challenge . . . 42
2.5.2 Second Position in Task 2 of the BraTS Challenge . . . 43
2.5.3 Fourth Position in Task 2 of the BraTS Challenge . . . 45
2.5.4 Summary . . . 46
2.6 Techniques for Brain Tumor Segmentation . . . 46
2.6.1 Pre-Processing techniques . . . 46
2.6.2 Post-Processing Techniques . . . 48
2.6.3 Class Imbalance . . . 48
2.7 Research Questions . . . 49

3 Implementation Methodology **51**

3.1 Overview . . . 51
3.2 Resources . . . 51
3.2.1 Lab GPU . . . 52
3.3 2D U-Net Implementation for the UCSF-PDGM dataset . . . 54
3.3.1 Model Architecture . . . 54
3.3.2 Data Generator and Preprocessing . . . 55
3.3.3 Loss . . . 57
3.3.4 Optimisation . . . 57
3.4 3D U-Net Implementation for the UCSF-PDGM dataset . . . 57

3.4.1 Model Architecture . . . 59
3.4.2 Loading the Data . . . 60
3.4.3 Loss . . . 61
3.4.4 Optimisation . . . 62
3.5 Transformer Implementation . . . 63
3.5.1 Model Architecture . . . 63
3.5.2 Loading the Data . . . 66
3.5.3 Optimiser . . . 66
3.6 Class Imbalance Experiment Implementation . . . 67
3.6.1 Experiment Basis . . . 67
3.6.2 Control Variables . . . 67
3.6.3 Independent Variable . . . 72
3.6.4 Dependent Variable . . . 76

4 Results **77**

4.1 2D U-Net Results . . . 77
4.2 3D U-Net Results . . . 81
4.3 Methylation status results . . . 85
4.4 Class Imbalance Results . . . 87

5 Evaluation **95**

5.1 Critical Evaluation of all the models . . . 95
5.2 Methylation Status Evaluation . . . 98
5.3 Class Imbalance Experiment Evaluation . . . 100
5.3.1 Flaws in the Experimental Procedure . . . 100
5.3.2 Critical Appraisal . . . 103
5.3.3 Key learnings from the experiments regardless of the results not working out . . . 106

5.4 Summary of Final Results 107

6 Conclusion **109**

6.1 Objectives 111

A Ethics Approval **113**

B Usage **115**

Bibliography **118**

List of Figures

1.1 An illustration of the UCSF-PDGM data set MRI modalities [1]. The structural modalities left to right are: T1, T1c, T2, FLAIR, together with the ground truth (Segmentation) segmentation. Tumour sub-regions are displayed per colour, with green: edema, yellow: enhancing tumour, and red: necrotic and non-enhancing tumour core. 7

1.2 The delineation of tumor classes in brain imaging, as explored in the study by Li et al. [2], reveals a structured relationship among the different tumor types. The study identifies four distinct tumor target classes, each independent yet interrelated in a hierarchical manner. Specifically, the tumor core is defined as encompassing the enhancing tumor, the non-enhancing tumor, and areas of necrosis. Beyond the tumor core, there is the complete tumor classification, which includes both the tumor core itself and the surrounding edema. This hierarchical structure is crucial for understanding the complexity and extent of brain tumors, as detailed in [2]. 8

1.3 UCSF-PDGM MRI FLAIR modality image examined using the Python nilearn library to assist with exploring the data set images [3] 9

1.4 Dice Coefficient as expressed by True Positive, False Positive, and False Negative as adapted from [4, 5] 10

1.5 Sensitivity as expressed by True Positive, False Positive, and False Negative as adapted from [4, 5] 12

1.6 Specificity as expressed by True Positive, False Positive, and False Negative as adapted from [4, 5] 12
1.7 Accuracy as expressed by True Positive, False Positive, and False Negative as adapted from [4, 5] 13
1.8 IoU as expressed by True Positive, False Positive, and False Negative as adapted from [4, 5] 14
1.9 Image illustrating the differences between Image Classification, Object Detection and Image Segmentation as taken from [6] 15
1.10 Schematic diagram of the receptive field in CNNs as taken from [7]. 17
1.11 The various layers of a CNN taken from [8] 18

2.1 An illustration of the U-Net network architecture as reported by Ronneberger et al. [9] 26
2.2 A schematic visualisation of V-Nets as reported by [10] 27
2.3 A high-level comparison between single-path and two-path Convolutional Neural Networks (CNNs) involves examining their structural differences and respective capabilities in data processing as taken from [11] 30
2.4 Schematic visualisation of the network architecture as adapted by Myronenko. The input for this process is a four-channel 3D MRI crop. It starts with an initial 3x3x3 3D convolution using 32 filters. Each green block in the network represents a ResNet-like block equipped with GroupNorm normalization. The output from the segmentation decoder, which matches the spatial dimensions of the input, comprises three channels. These channels are processed through a sigmoid function to generate segmentation maps for the three tumor subregions: Whole Tumor (WT), Tumor Core (TC), and Enhancing Tumor (ET). Additionally, there's a Variational Autoencoder (VAE) branch that reconstructs the input image. This branch is utilised only during training as a means to regularize the shared encoder. 32

2.5 The architecture of sequential convolutional networks for brain tumor delineation, adapted from the framework initially described in [12]. WNet is designated for whole tumor segmentation, TNet focuses on the tumor core, and ENet targets the enhancing tumor core. 34

2.6 Overview of the two-stage cascaded network as taken from [13] 36

2.7 The provided images, taken from [14] depict the segmentation outcomes for various categories. In the top row, the focus is on the model's segmentation results within the enhanced tumor region. Meanwhile, the bottom row illustrates how the model performs in segmenting the edema area. 40

2.8 Minh Panh's Prediction Model, as depicted in [15], employs a pre-trained EfficientNet B0 for feature extraction from MRI images. The model adapts to the four-channel input by applying a 2D convolution that transforms it into a three-channel feature map suitable for EfficientNet. The original classification head is substituted with a 256-node fully connected layer. Feature embeddings from 10 selected frames are subsequently fed into two LSTM layers with a hidden size of 32, culminating in a prediction layer with a single node to output the score. 43

3.1 2D U-Net Implementation data Pipeline for the UCSF-PDGM dataset 53

3.2 Data split in each individual DataGenerator after regularisation and preprocessing for the 2D U-Net implementation as isnpired by [16] 56

3.3 3D U-Net Pipeline for the UCSF-PDGM dataset 58

3.4 Transformer Pipeline for Methylation Status 64

3.5 Pipeline used for the experiments testing the impact of loss functions on trying to solve class imbalance 68

3.6 The 3D-UNet is composed of a contractive and an expanding path, that aims at building a bottleneck in its centermost part through a combination of convolution and pooling operations. After this bottleneck, the image is reconstructed through a combination of convolutions and upsampling. Taken from [17] 70

4.1 Training Process Visualisation for the 2D UNet 79

4.2 Ground Truth vs Prediction Results for the Implemented 2D Model for the EDEMA class. 80

4.3 Loss functions when Training the model for 50 epochs showing that the model is learning well on the training data. 82

4.4 Multi-class Dice and Jacquard Coefficient scores from Validation on the UCSF-PDFM-v2 dataset 83

4.5 Segmentation performance of the 3D U-Net on the UCSF-PDGM-v2 dataset. 84

4.6 Methylation Status Prediction vs. Count on the Testing Data as Implemented on the DGX 100. The histogram displays a narrow range of predicted values for MGMT promoter methylation (MGMT_pred), predominantly clustered around 0.146 with a few outliers extending towards 0 and 2.5e-1. The concentration of predictions around a single value suggests potential issues with the model, such as overfitting to a certain characteristic of the training data or a lack of generalization to diverse cases within the testing dataset." 86

4.7 Hybrid Loss 1 Training Graph showing the discrepancy between training and validation performance. 87

4.8 Hybrid Loss 1 Training output showing that the model is overfitting and and capturing more of the tumor area than what is present. 88

4.9 Hybrid Loss 1 Testing segmentation demonstrating the poor performance on the Testing Image 88

4.10 Hybrid Loss 2 Training graph pointing to potential over-fitting and a very erratic learning process 89
4.11 The segmentation by Hybrid Loss 2 on the training image suggests the model is unable to delienate the tumor regions and not learn any relevant patterns. 90
4.12 The segmentation performance by Hybrid Loss 2 on a Test Image reinforces the belief that the model suffers from over-fitting which has led to prove it's ineffectiveness. 91
4.13 Custom Dice Loss Training Graph. The variability and erratic behaviour of the validation loss, Dice Coefficient, and the IoU score reinforce the idea of overfitting. 93
4.14 A very accurate segmentation performed by the Custom Dice Loss on a Training Image 94
4.15 The Custom Dice Loss model was unable to accurately segment a test Image, suggesting overfitting. 94

A.1 Ethics Approval 114

List of Tables

1.1 Image segmentation techniques use different algorithms adapted from [18] . . 16

2.1 Summary of BRATS datasets over the years as taken from [19]. 22

2.2 Comparison of different models for brain tumor segmentation. In column Input, 'P' means the patch and 'I' means the image. In column Dice, 'WT' means whole tumor, 'TC' means tumor core and 'ET' means enhancing tumor. Column Dataset indicates the associated dataset with the reported segmentation performance. In column Type, 'CV' means the cross-validation on BraTS training set, 'V' means the BraTS validation set. Inspiration taken from [11] 41

2.3 Models, augmentation methods and the respective scores of the top performers in detecting the MGMT promoter methylation status challenge based on [20] 47

3.1 2D U-Net architecture, where Conv stands for 2D convolution, UpSampling - 2D upsampling, Concatenate - concatenation along the channel axis, and Dropout - dropout layer. Table inspiration taken from [21] 54

3.2 Combined 3D U-Net architecture, as taken from [22], adjusted for an input size of $240 \times 240 \times 152$, detailing the layer operations and output sizes for both the encoder and decoder paths. 59

3.3 Architecture of the Vision Transformer model as adapted from [23]. The model begins with a patch to embedding layer transforming input features to a higher dimension. This is followed by a dropout layer for regularization. The core of the model consists of transformer blocks, each containing a pre-normalization layer, a multi-head attention mechanism that captures dependencies, and a feedforward neural network. 'PreNorm' stands for Layer Normalization applied before each sub-layer, 'Attention' refers to the self-attention mechanism, and 'FeedForward' is a simple feedforward neural network. 'Identity' indicates no operation is performed on the class token. The final output is produced by the MLP head, which includes normalization, linear transformation, a GELU activation function, and dropout for regularisation. 63

3.4 3D U-Net architecture, taken from [17], detailing the convolutional layers, dropout layers, max pooling, and concatenation operations throughout the model. This architecture is specifically designed for volumetric data processing, featuring a sequence of Conv3D and Conv3DTranspose layers for down-sampling and up-sampling respectively. 69

4.1 The performance results of the 2D U-Net model up to 35 epochs on the UCSF-PDGM-v2 data per 5 epochs. 78

4.2 Overall metrics and results for 2D UNet after evaluation 80

4.3 Average Dice and Jaccard Coefficients for the 3D U-Net Model Implementation 81

4.4 Validation AUC results for the Transformer Implementation for the methylation status. 85

5.1 Comparison of different models for brain tumor segmentation with 2D models. 97

5.2 Comparison of different models for brain tumor segmentation with implemented 3D models and their loss functions. 98

5.3 Comparison of the Implemented Transformer and the AUC score for it show that there is a big gap between the literature and the transformer. This is due to the reduced training size and reduced dimensions. 100
5.4 Full performance Metrics Comparison for the three Loss Function experiments 103
5.5 Average Dice Coefficients for Loss Functions on test data 107
5.6 Comparison of all the implemented models on the UCSF-PDGM dataset. Metrics abbreviations: IoU - Intersection over Union, Prec. - Precision, Sens. - Sensitivity, Spec. - Specificity, Nec. - Necrotic, Edem. - Edema, Enh. - Enhancing, WT - Whole Tumor, TC - Tumor Core, ET - Enhancing Tumor. 108

6.1 Evaluation of Original Objectives and where they can be found in the paper. 'P' is for primary objectives and 'S' is a secondary objective. 112

Abstract

Automated brain tumor segmentation is a critical tool for supporting radiologists in the accurate diagnosis and treatment planning of patients with brain tumors. Manual segmentation is a challenging and time-consuming task that requires expert knowledge and is complicated by the variability in tumor size, location, and infiltration into adjacent healthy tissue. While fully automated segmentation solutions have yet to be adopted in clinical practice, significant progress has been made in leveraging deep neural networks to improve the precision and efficiency of brain tumor segmentation. This dissertation presents five models, including 2D and 3D U-Net architectures, developed for the University of San Francisco Preoperative Diffuse Glioma (UCSF-PDGM) dataset and a 3D U-Net trained with custom loss functions. The performance of these models was evaluated using the Dice score metric per tumor region, as well as additional metrics such as IoU Score, Sensitivity, Specificity, and Precision. The best-performing model, a 3D U-Net trained without custom loss functions, achieved Dice Scores of 0.859, 0.857, and 1.00 for whole tumor, tumor core, and enhancing tumor, respectively. Additionally, a transformer model was implemented to detect the presence of the O^6-methylguanine DNA methyltransferase (MGMT) protein, which plays a crucial role in treating gliomas. This research demonstrates the potential of deep learning techniques to enhance the accuracy and efficiency of brain tumor segmentation, ultimately supporting improved patient care.

Declaration

I declare that the material submitted for assessment is my own work except where credit is explicitly given to others by citation or acknowledgment. This work was performed during the current academic year except where otherwise stated.

The main text of this project report is 18,470 words long, including project specification and plan.

In submitting this project report to the University of St Andrews, I give permission for it to be made available for use in accordance with the regulations of the University Library. I also give permission for the title and abstract to be published and for copies of the report to be made and supplied at cost to any bona fide library or research worker, and to be made available on the World Wide Web. I retain the copyright in this work.

Acknowledgements

I would like to thank my dissertation supervisor Dr. David Harris-Birtill for advising me during the duration of the academic year and mentoring me. I would also like to thank my parents Anand Misra and Varsha Misra for constantly supporting me throughout my university journey. I would also like to thank Krish Amit Shah and Levi Finn Pastor for being a constant rock in my university life.

Chapter 1

Introduction

This project aims to implement deep learning to perform brain tumor segmentation of Magnetic Resonance Imaging (MRI). This project will focus on the analysis of a publicly available, open-source The University of California San Francisco Preoperative Diffuse Glioma MRI (UCSF-PDGM) dataset [1]. Ethical approval has been received for this project as described in Section 1.3. The School of Computer Science GPU lab client machines were used to develop both the model implementations. The main results of this project are a literature review, given in Section 2, and the consequent development of five neural network algorithms for the segmentation of brain tumor MRI images and a 3D transformer to test the presence of the methylation status. The first 2D Neural Network was developed based on an adaptation of the work done by [16]. The 3D UNet was developed to verify the reproducibility of some of the most competitive results, drawing inspiration from the work by [21], who achieved one of the highest outcomes. No existing models for this dataset were found, which further motivated the development. The performance of each of the algorithms was measured using the Dice score metric. Furthermore, the Dice scores were measured per each individual tumor region (the multi-class case). Additional metrics like Mean IoU, Precision, Sensitivity, and Specificity were also used to evaluate the performance of the model and explain the erratic behavior in training. This can be seen from Table 5.4 in Sections

5.3.2 and table 5.6 in Section 5.4. The performance of the transformer is measured and compared using the Validation AUC metric. The results of the Transformer are suboptimal due to the reduced image quality that has been used in the model and downsizing the dataset to remove unwanted classes. The best performance in terms of segmentation came from the implemented Autoencoder results. The final results of the U-Net implementations are provided in Table 5.6 and illustrated in Figures 4.1, 4.2, 4.3, 4.4, 4.7, and 4.8 in Section 4. Sample images of brain tumor segmentation per one patient are illustrated in Figures 4.2, 4.5, 4.8, 4.9, 4.11, 4.12, 4.14, and 4.15 in Section 4. In the clinical context, using convolutional neural networks to accomplish brain tumor segmentation may assist radiologists by performing automatic brain tumor segmentation, thereby saving time [24] and assisting in circumventing any possible brain tumor segmentation variability [25] that may occur in manual human-based brain tumor segmentation [25].

1.1 Motivation

The global impact of brain tumors is substantial, as they are a significant contributor to worldwide mortality rates. In the United States, an estimated 24,810 adults will be diagnosed with primary cancerous tumors of the brain and spinal cord in 2023, underscoring the severity of this health issue [26]. Similarly, in India, data reveal that annually, between 40,000 and 50,000 new cases of brain tumors are diagnosed, with a concerning 20% affecting children [27]. In Australia, brain tumors have been identified as the leading cause of cancer-related deaths among individuals under 40, as reported by the Cure Brain Cancer Foundation[28].

The survival rates for brain tumor patients are notably low and have not shown significant improvement over the past 30 years, particularly in Australia. This contrasts sharply with the advancements in survival rates for other cancer types. For instance, the 5-year relative survival rate for a cancerous brain or CNS tumor is almost 36%, and the 10-year survival rate is over 30%, with a notable variation depending on age [26]. The World Health

Organization (WHO) categorizes brain tumors by their cellular origin and aggressiveness, from least (Grade I or II, low-grade (LG) tumors) to most aggressive (Grade III or IV, high-grade (HG) tumors)[29]. HG tumors are particularly threatening, often leading to a life expectancy of no more than two years, while LG tumors may allow for many years of life expectancy.

The statistics and complexity of brain tumor diagnosis underline the crucial need for accurate detection and treatment methods. Advances in Computer-Assisted Detection (CAD) systems offer promise in improving the accuracy and timeliness of brain tumor diagnoses, potentially enhancing patient survival rates and outcomes. Integrating such systems into clinical practice represents a significant development in neuro-oncology, providing hope for more effective disease management.

In order to reduce the recurrence and mortality rates, patients undergo post-operative antineoplastic drug treatement [30]. However, statistics indicate that glioma patients have a life expectancy of only 12-15 months, with a 5-year survival rate of less than 5% [31].

O^6-methylguanine DNA methyltransferase (MGMT) is a protein found in humans and some prokaryotic organisms that plays a crucial role in the repair of DNA. This protein is particularly important in the context of cancer treatment, specifically for patients with glioma (a type of brain tumor). MGMT acts as a DNA repair protein, and its primary function is to remove alkyl groups (chemical groups containing carbon and hydrogen atoms) from the DNA molecule. When alkylating agents are used in cancer treatment, they introduce alkyl groups into the DNA of cancer cells, leading to DNA damage and eventually cell death. However, if MGMT is present and active, it can remove these alkyl groups, effectively repairing the DNA and allowing the cancer cells to survive and become resistant to the chemotherapy drugs [32]. Studies have shown that glioma patients whose tumors express high levels of MGMT tend to respond less favorably to chemotherapy compared to those with low or no MGMT expression. This is because the higher the MGMT activity, the more efficiently the cancer cells can repair the DNA damage caused by the alkylating

agents, leading to increased resistance to these antineoplastic (anti-cancer) drugs [33].

1.2 Objectives

The aim of this project was the application of deep learning techniques to the Multimodal Brain Tumour Image Segmentation Benchmark (BraTS) benchmark data set, in order to survey and improve the classification and segmentation of brain tumours within magnetic resonance imaging (MRI) brain images.

The primary objectives of this individual project include an initial pipeline implementation and secondary objectives entail the subsequent deep learning algorithm implementations as shown in detail in Figure 3.2 in Section 3.

The implementation of the primary objectives which entail the initial pipeline was completed and consisted of a BraTS brain tumour magnetic resonance imaging (MRI) image data segmentation implementation and entailed the loading in of the data, data pre-processing together with the implementation and evaluation of a brain tumour segmentation algorithm.

The following project objectives were identified:

1.2.1 Primary Objectives

- Complete a literature review which surveys the state of the art on the topic of Brain Tumour Segmentation (in MRI) using deep learning and formulate research questions.

- Implement a pipeline that will read in and pre-process the data from the open source data set (UCSF-PDGM-v2).

- Create a segmentation of brain tumour images with an implementation of a segmentation algorithm.

1.2.2 Secondary Objectives

- Create a segmentation of brain tumour images with a re-implementation of an existing algorithm as identified in the literature review.
- Implement more advanced algorithms or optimise existing deep learning algorithms to possibly find a novel solution for Brain Tumour Segmentation.
- Compare the performance and results of algorithms and the deep learning technique implementation from both my own model analysis and the results achieved by others in the literature review.

1.3 Ethical Considerations

Ethical approval for this project has been received with the following corresponding Approval Code CS17317. The full ethics approval document has been appended to the appendix and is found in Appendix A.1 at the end of this document. The University of California San Francisco Preoperative Diffuse Glioma MRI (UCSF-PDGM) data set [1] , is an open source publicly available data set which has been provided on the Cancer Imaging Archive website. This data is available from: `https://wiki.cancerimagingarchive.net/pages/viewpage.action?pageId=119705830#119705830274abcd390744380adfaae47d0aa528e`.

1.4 Data

The UCSF-PDGM dataset incorporates the brain tumor segmentation data from the BraTS 2021 Challenge, specifically focusing on MRI images of glioma brain tumors. Gliomas, which are further elaborated upon in Section 2.1, represent the primary subject of this dataset. Within the UCSF-PDGM dataset, each patient's data comprises eight structural modalities from magnetic resonance imaging (MRI). Specifically, the modalities utilized are T1, T1c, T2, and FLAIR. Accompanying these images are segmentation files, which provide detailed

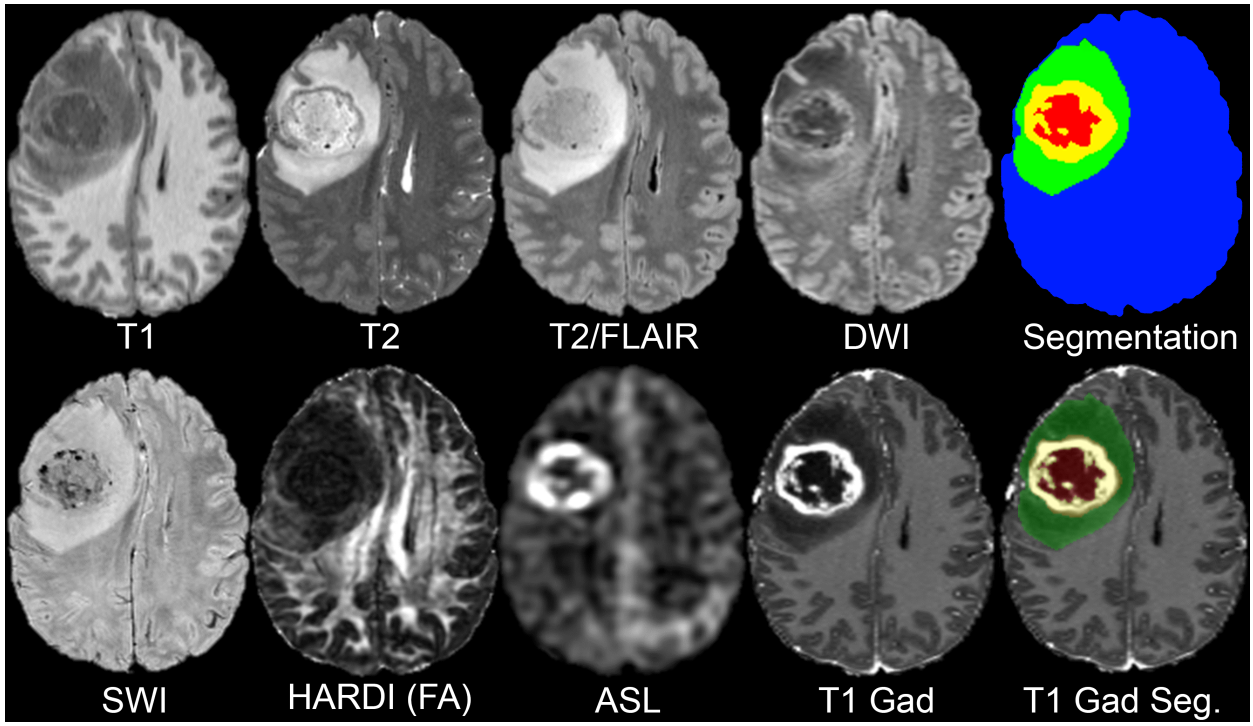

Figure 1.1: An illustration of the UCSF-PDGM data set MRI modalities [1]. The structural modalities left to right are: T1, T1c, T2, FLAIR, together with the ground truth (Segmentation) segmentation. Tumour sub-regions are displayed per colour, with green: edema, yellow: enhancing tumour, and red: necrotic and non-enhancing tumour core.

pixel-level annotations of the full extent of the tumors. These annotated segmentation images are exemplified in Figure 1.

The segmentation process for the study's data, aligned with the 2021 BraTS challenge, involved a comprehensive multi compartment approach. Initially, the images underwent an automated segmentation phase. As described in [34] this phase utilized an ensemble model comprising algorithms that had previously been successful in BraTS challenges. Following this automated process, trained radiologists conducted manual corrections to refine the segmentation accuracy. These corrections were then verified and approved by two expert reviewers.

The segmentation specifically delineated three major tumor compartments. These included the enhancing tumor, the non-enhancing or necrotic tumor, and the surrounding FLAIR abnormality, which is sometimes referred to as edema. This detailed segmentation

approach was instrumental in ensuring a high level of accuracy and reliability in tumor identification and classification. Labels were assigned per target class as 1, 2, 4 respectively [34]. While each of these four target classes are independent, there is an encompassing relationship between them as shown in Figure 2.

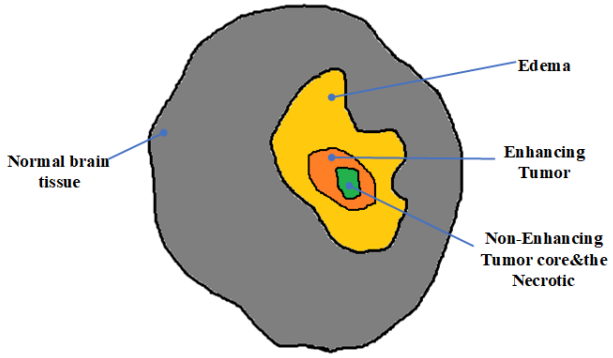

Figure 1.2: The delineation of tumor classes in brain imaging, as explored in the study by Li et al. [2], reveals a structured relationship among the different tumor types. The study identifies four distinct tumor target classes, each independent yet interrelated in a hierarchical manner. Specifically, the tumor core is defined as encompassing the enhancing tumor, the non-enhancing tumor, and areas of necrosis. Beyond the tumor core, there is the complete tumor classification, which includes both the tumor core itself and the surrounding edema. This hierarchical structure is crucial for understanding the complexity and extent of brain tumors, as detailed in [2].

All of the BraTS multimodal brain scan files are provided in the format of NIfTI (.nii.gz) [35]. NIfTI (Neuroimaging Informatics Technology Initiative) files store MRI data with the correct brain image orientation which allows the correct identification of both sides of the brain [35]. Figure 3 displays an in-depth visualisation of a FLAIR image the nilearn package [3] in Python.

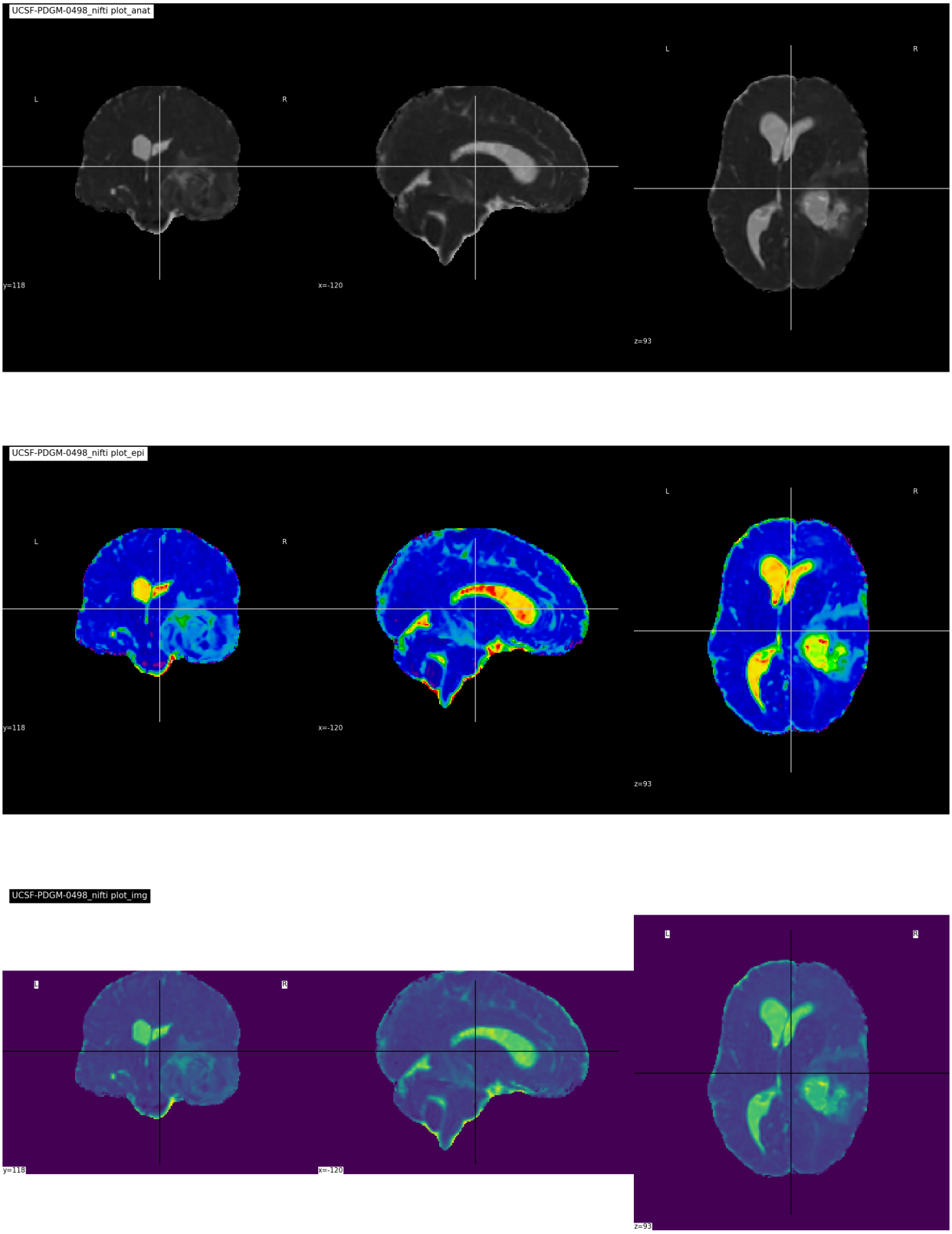

Figure 1.3: UCSF-PDGM MRI FLAIR modality image examined using the Python nilearn library to assist with exploring the data set images [3]

In glioma diagnosis, standard MRI modalities play a crucial role, each serving a specific purpose. T1-weighted MRI (T1) is primarily used to identify healthy brain [36] tissue, offering a baseline against which abnormalities can be detected. Following this, T2-weighted MRI (T2) images are employed to clearly outline edema in tumor regions [36], which typically appear as bright signals in these images. Additionally, Fluid Attenuated Inversion Recovery (FLAIR) MRI images are instrumental in differentiating between tumor-related edema and Cerebrospinal Fluid (CSF) [36]. This differentiation is achieved by suppressing the signals from water molecules, creating a distinct contrast on the image.

The introduction of the BRATS benchmark dataset has been a significant advancement in this field. This dataset is specifically designed for automated brain tumor segmentation and facilitates an objective evaluation of various glioma segmentation methods. By providing a standard set of data, BRATS allows for the consistent comparison and improvement of these segmentation techniques.

The evaluation of brain tumor segmentation in the BraTS benchmark uses metrics such as the Dice Score, Sensitivity, and Specificity to assess the accuracy of segmentation for key tumor regions: whole tumor (encompassing all tumor subregions), core tumor (all tumor regions excluding the edema), and enhancing tumor (regions indicative of active disease). These metrics are vital for quantifying the performance of segmentation algorithms.

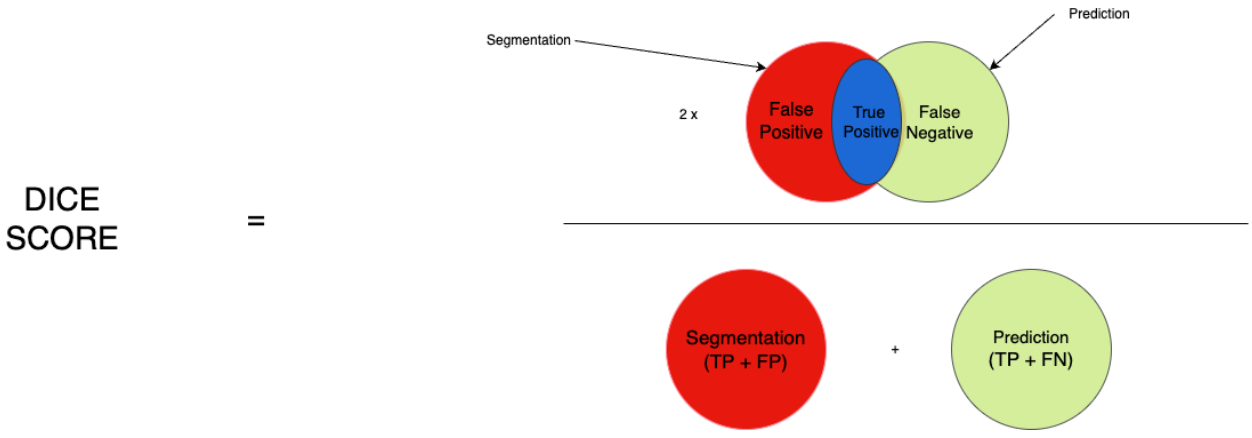

Figure 1.4: Dice Coefficient as expressed by True Positive, False Positive, and False Negative as adapted from [4, 5]

The Dice Score, a metric for spatial overlap accuracy, is defined for each tumor region. It is computed as:

$$Dice = \frac{2 \times |\mathrm{GT} \cap \mathrm{PM}|}{|\mathrm{GT}| + |\mathrm{PM}|} \times 100 \tag{1.1}$$

In this formula, "GT" stands for the ground truth voxel count within the tumor area, while "PM" denotes the predicted tumor area voxel count as determined by the segmentation algorithm. The symbol $\cap$ represents the intersection between the predicted and ground truth sets, and the vertical bars denote the cardinality of the sets, i.e., the count of voxels within.

The Dice coefficient ranges from 0 to 1, where 1 indicates perfect overlap between prediction and ground truth. Specificity measures the proportion of actual negatives correctly identified (true negative rate), and Sensitivity is the true positive rate, reflecting the proportion of actual positives correctly identified. These metrics provide a comprehensive view of a model's performance in distinguishing tumor tissue from non-tumor regions. [37].

Another evaluation metric that is used is sensitivity. Sensitivity is applied to measure the overlap percentage between the regions predicted by the model and the regions segmented by the human to the manual segmentation regions [38]. it is denoted by the equation:

$$\mathrm{Sensitivity}(GT, PM) = \frac{|PT \cap GM|}{|GM|} \tag{1.2}$$

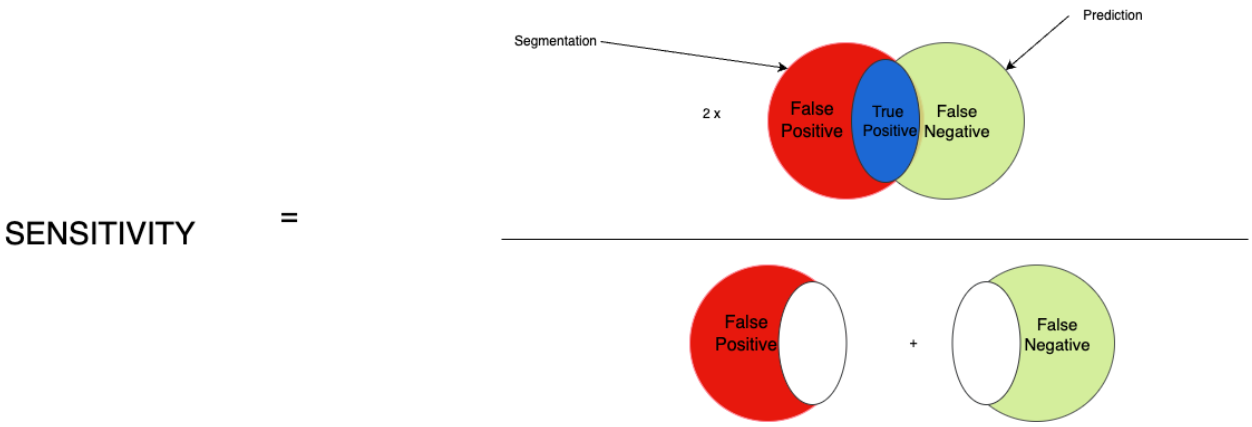

Figure 1.5: Sensitivity as expressed by True Positive, False Positive, and False Negative as adapted from [4, 5]

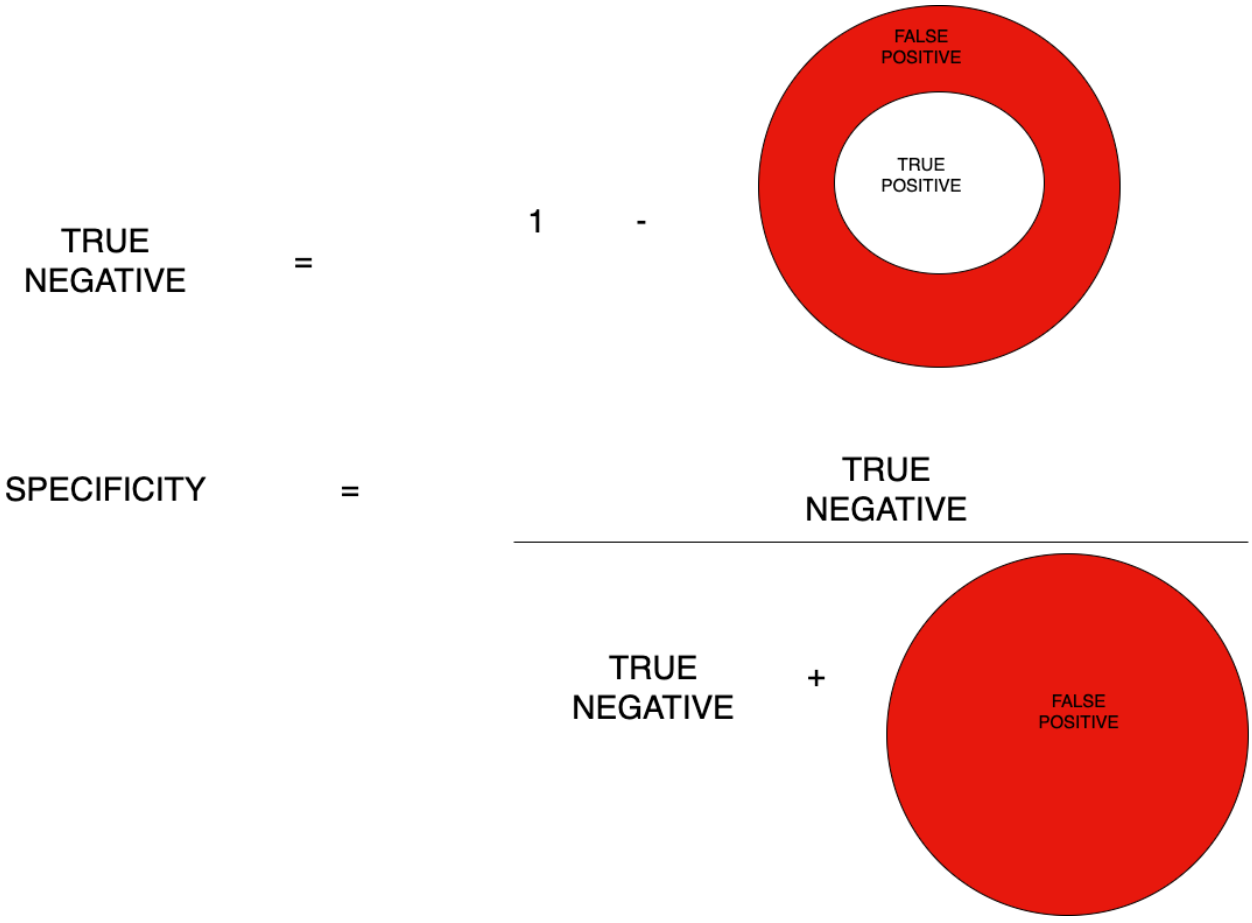

Figure 1.6: Specificity as expressed by True Positive, False Positive, and False Negative as adapted from [4, 5]

Specificity is used to correctly identify the background class in an image. This is impor-

tant because, in medical images, the background (non-region of interest) typically occupies a large fraction of the image compared to the region of interest (ROI), such as a tumor or lesion. this is due to the imbalance. between the background and ROI pixels, specificity values close to 1 (or 100%) are standard and expected for well-performing models. A high specificity value means that the model is accurately identifying the background pixels as background, which is a desirable characteristic [5].

$$\text{Specificity} = \frac{\text{Area of GT non-tumor correctly identified as non-tumor (TN)}}{\text{Total area of GT non-tumor (TN + FP)}} \tag{1.3}$$

Accuracy is defined as the number of correct predictions, consisting of correct positive and negative predictions, compared to the total number of predictions [5].

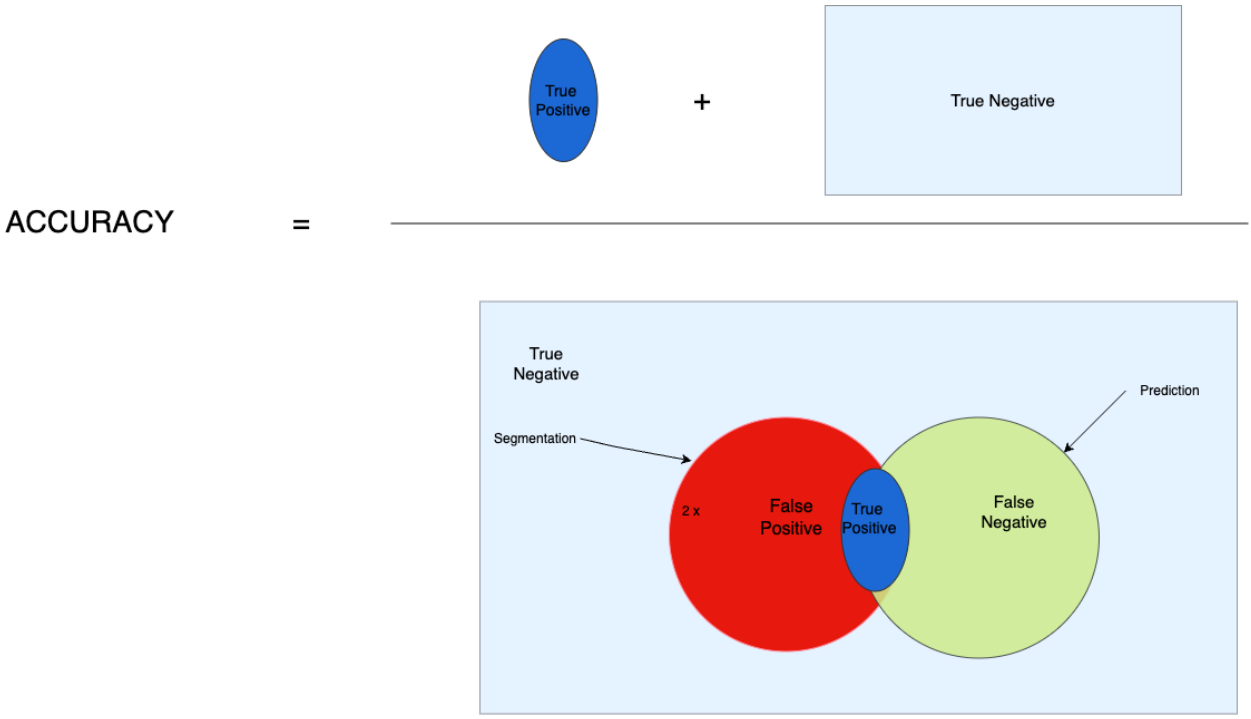

Figure 1.7: Accuracy as expressed by True Positive, False Positive, and False Negative as adapted from [4, 5]

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FN + FP} \tag{1.4}$$

The Intersection-over-Union (IoU), also known as Jaccard index or Jaccard similarity coefficient penalizes under- and over-segmentation more than the Dice Coefficient [5].

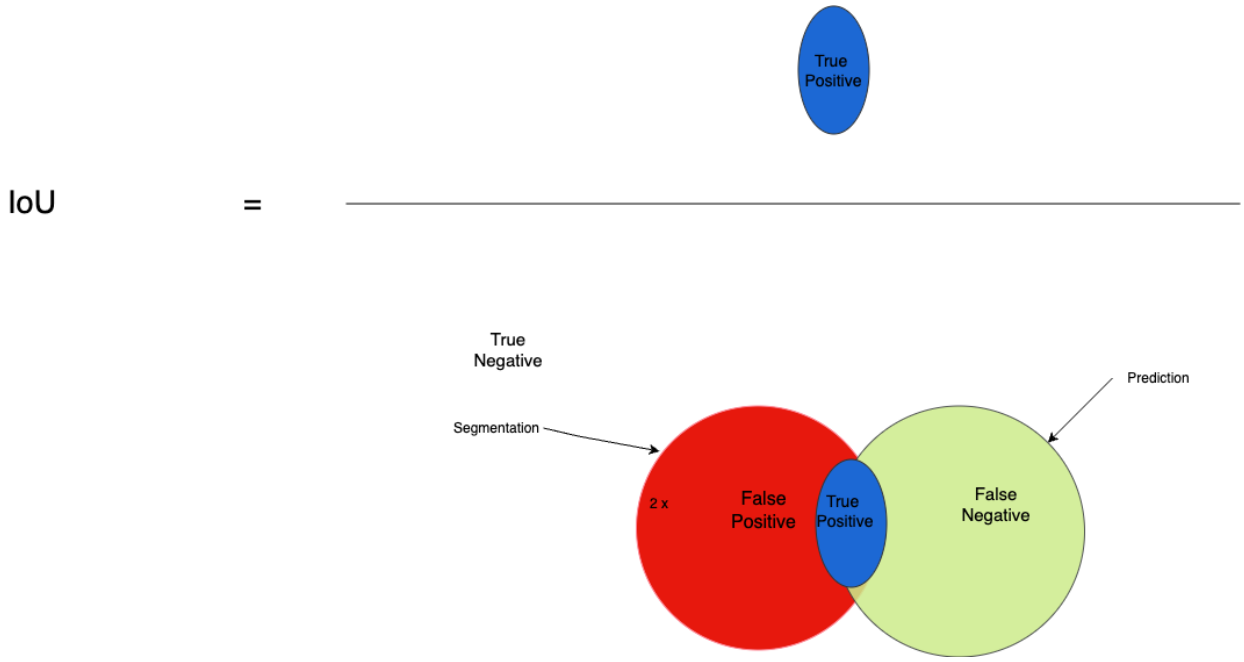

Figure 1.8: IoU as expressed by True Positive, False Positive, and False Negative as adapted from [4, 5]

$$\text{IoU} = \frac{TP}{TP + FP + FN} \tag{1.5}$$

The ROC (Receiver Operating Characteristic) curve is a plot that visualizes the diagnostic ability of a classifier by showing its true positive rate against the false positive rate at different discrimination thresholds [5]. The area under the ROC curve (AUC) is a single-value metric that summarizes the performance of a classifier [5]. A higher AUC value indicates better performance, with an AUC of 0.5 representing a random classifier [5]. The AUC score is widely used for comparing multiple classifiers and evaluating diagnostic tests and clinical trials, especially in the medical field.

$$AUC = 1 - \frac{1}{2}\left(\frac{FP}{FP + TN} + \frac{FN}{FN + TP}\right) \tag{1.6}$$

1.5 Background

1.5.1 Image Segmentation

In the realm of computer vision and machine learning, there are three foundational tasks pivotal for for understanding and interpreting images: classification, object detection, and image segmentation [18]. Figure 1.9 illustrates the differences between the following.

Figure 1.9: Image illustrating the differences between Image Classification, Object Detection and Image Segmentation as taken from [6]

Classification is the process where a machine learning model discerns which types of objects are present in a video or image. It is used for determining whether an object is in an image but it does not tell where the object is located within the image [18].

Object Detection combines classification with localisation. Models trained for object detection not only recognise objects within an image but also identify their specific locations using bounding boxes [18].

Image Segmentation is a crucial process in computer vision that breaks down an image into distinct segments to enable a more profound and accessible analysis. By analysing images at the pixel level, it achieves a depth of detail that goes beyond mere classification, making it especially valuable for precision-dependent fields like medical imaging. Here,

Algorithm	Description
Edge Detection Segmentation	Makes use of discontinuous local features of an image to detect edges and hence define a boundary of the object.
Mask R-CNN	Gives three outputs for each object in the image: its class, bounding box coordinates, and object mask
Segmentation based on Clustering	Divides the pixels of the image into homogeneous clusters.
Region-Based Segmentation	Separates the objects into different regions based on threshold value(s).

Table 1.1: Image segmentation techniques use different algorithms adapted from [18]

segmentation plays a vital role, as it allows for the exact delineation of anatomical structures and pathologies, such as tumors, by outlining their perimeters with high accuracy [18].

Table 1.1 shows the choice of algorithm and how it depends on the specific requirements of the application and the nature of the images being processed.

1.5.2 Deep Learning Techniques

Deep Learning, particularly Deep Neural Networks, have become a cornerstone in the field of brain tumor segmentation. Convolutional Neural Networks (CNNs), a specialized form of DNN, are particularly well-suited for analyzing image data due to their ability to process and learn from the spatial hierarchical patterns in images [39].

1.5.3 Convolutional Neural Networks (CNNs)

Convolutional Neural Networks (CNNs) are a foundational element in the field of computer vision, adept at performing complex tasks such as image classification, object detection, and voice recognition, as well as natural language processing. A key feature of CNNs is their ability to process images by analyzing sections of an input image, specifically the pixels within their receptive fields. [40, 41] This is achieved through convolutional layers that employ zero padding to maintain uniform dimensions and utilize filters or kernels to detect patterns within an image.

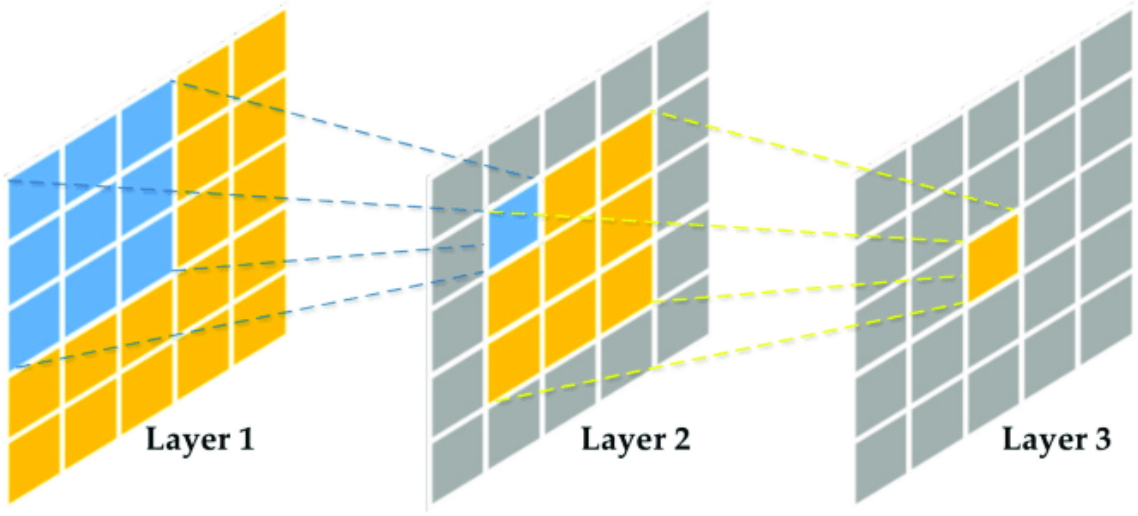

Figure 1.10: Schematic diagram of the receptive field in CNNs as taken from [7].

CNNs are inspired by the human brain's visual cortex, where each neuron responds to stimuli in a specific region, known as the receptive field. The layers in a CNN mimic this by overlapping to cover the entire visual area of an image. In the workflow of CNNs, an input image is first processed to identify objects based on color planes, then distinguished in various color spaces. [40, 41] One of CNN's strengths lies in its ability to reduce image dimensions while retaining key features, facilitating better predictions and processing efficiency.

The convolution layer, or kernel, of a CNN operates on a set formula that involves image dimensions and channels. This layer conducts convolutional operations across the image, parsing objects using a defined stride until the entire image is covered. In the case of images with multiple channels, such as RGB, the kernel adapts to match the depth of the input image [40, 41].

CNNs are designed to extract both high-level and low-level features from images, including edges, colors, and gradient orientations. The architecture of CNNs often includes layers like valid padding and same padding, which manipulate the dimensionality of the image based on the desired output. Following the convolutional layer, CNNs employ a pooling layer, which aims to reduce spatial size and computational complexity while extracting dominant features [40, 41] .

Finally, the Fully Connected Layer (FC Layer) [40] in CNNs is responsible for learning non-linear combinations of high-level features outputted by the convolutional layer. This layer flattens the image output and prepares it for further classification tasks. All these layers can be seen in Figure 1.11.

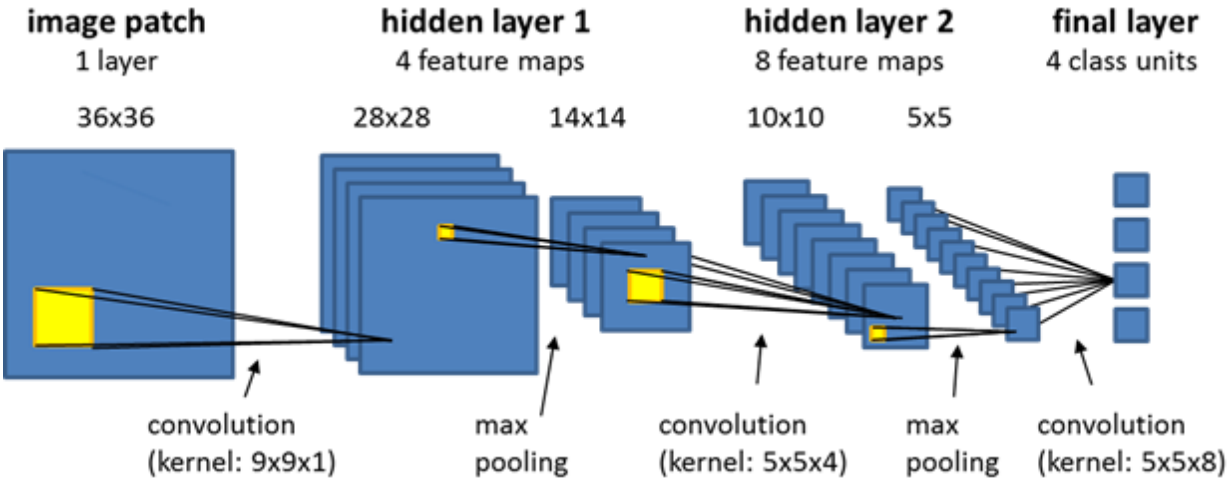

Figure 1.11: The various layers of a CNN taken from [8]

Notable examples of CNN architectures include AlexNet [42], GoogLeNet [43] , ZFNet [44], LeNet [45], and ResNet [46, 40]. Each of these architectures contributes uniquely to the field of computer vision, offering powerful tools for deep learning algorithms and AI applications.

Chapter 2

Literature Review

2.1 Clinical Background

A brain tumor represents an aggregation of cells within the brain that proliferate abnormally and uncontrollably [47]. Brain tumors can broadly be divided into two main categories. The first category includes non-cancerous, or benign, brain tumors [47]. These are typically low-grade tumors, either grade 1 or 2, and they tend to grow at a slower pace [47]. Importantly, benign brain tumors are less likely to recur after treatment, making their prognosis generally more favorable. The second category encompasses cancerous, or malignant, brain tumors. These are high-grade tumors, either grade 3 or 4, and can be more aggressive in nature [47]. Malignant brain tumors may originate within the brain, known as primary tumors, or they can metastasize to the brain from other parts of the body, termed secondary tumors [47]. The malignant nature of these tumors means they are more likely to grow back even after comprehensive treatment [47].

Glials cells are vital for supporting and insulating nerve cells in the brain and are also the origin of various types of brain tumors [48, 49]. These tumors are categorised based on the type of glial cell affected. Astrocytomas, including glioblastoma multiforme (GBM), arise from astrocytes; oligodendrogliomas originate in oligodendrocytes; and ependymomas

develop from ependymal cells [48, 49]. Glioblastoma multiforme represents the most malignant form of astrocytomas and is highly aggressive. The World Health Organization (WHO) grades gliomas I to IV based on growth rate and malignancy. Lower-grade gliomas (grade I) are slower growing and more amenable to surgical removal, while higher-grade gliomas (grades II to IV) are invasive and aggressive, with GBM being the most severe [48, 49]. Life expectancy varies significantly: patients with lower-grade tumors can live several years, while those with high-grade gliomas like GBM typically have a prognosis of about 14 to 15 months post diagnosis [48, 49]. Intensive neuroimaging is used in both groups to monitor the effectiveness of treatment and the progression of the disease.

Brain tumor segmentation methods are broadly categorized based on the level of human involvement required. These categories include manual segmentation, where a human expert outlines the tumor area; semiautomatic segmentation, which involves some degree of manual intervention combined with automated processes; and fully automatic segmentation, where the entire process is handled by algorithms without any manual input [50]. Each approach has its own set of advantages and limitations, balancing the precision of human expertise against the efficiency and consistency of automated techniques [24]. The precision required in brain tumor segmentation means that current industry practices primarily rely on manual or semi-automated techniques, where even a minimal error in image analysis is not acceptable [51]. This process, typically performed by radiologists, is not only laborious and time-intensive but also depends heavily on the expertise of the individual. Manual segmentation involves the identification and drawing of tumor regions on each image slice [24, 51]. This task is further complicated as it usually involves analyzing two-dimensional slices, limiting the use of three-dimensional data, which could provide a more comprehensive view of the tumor [24, 51]. Consequently, the results of manual segmentation can vary significantly between different radiologists.

The implementation of fully automated and highly precise brain tumor scan analyses could significantly aid medical professionals in diagnosing and treating patients. Brain

tumor segmentation, a challenging task, involves differentiating tumor tissues like active tumors, edema, and necrosis from normal brain tissues such as gray matter, white matter, and cerebrospinal fluid [24, 51]. The complexity arises from the non-uniform, varied three-dimensional structure of tumors [24, 51]. Current automatic segmentation models strive to accurately characterize tumor size, shape, location, and appearance while accommodating practical variations [24, 51]. However, as of now, no fully automatic segmentation method has been widely adopted in clinical settings due to these complexities [24].

2.2 Datasets

The BraTS (Brain Tumor Segmentation) challenge, initiated in 2012 alongside the MICCAI conference, has evolved significantly over the years. Initially, it provided participants with 50 multimodal preoperative magnetic resonance imaging (mpMRI) cases of glioma, meticulously annotated by clinical experts to segment glioma sub-regions: enhancing tumor (ET), necrotic core (NCR), and edema (ED) [19].

From 2014 to 2016, a significant portion of data was sourced from The Cancer Imaging Archive (TCIA), particularly the TCGA-GBM and TCGA-LGG collections, featuring both pre- and post-operative scans with ground truth segmentations derived from the consensus of high-performing algorithms from the 2012 and 2013 challenges [19].

In 2017, a thorough review of all TCIA datasets by board-certified neuroradiologists led to the classification of each scan into pre- or post-operative categories, with only pre-operative scans being incorporated into the BraTS challenge from then on [19].

The challenge's scope expanded from 2017 to 2020, including the prediction of overall survival for patients with glioblastoma who received a gross-total resection [19]. BraTS 2021 not only maintained its commitment to the segmentation of glioma sub-regions in a significantly expanded dataset comprising 2,000 glioma cases (or 8,000 mpMRI scans) but also added the crucial clinical objective of determining the MGMT promoter methylation sta-

Year	Total Data	Training Data	Validation Data	Testing Data	Tasks	Timepoint
2012	50	35	NA	15	Segmentation	Pre-operative
2013	60	35	NA	25	Segmentation	Pre-operative
2014	238	200	NA	38	Segmentation	Longitudinal
2015	253	200	NA	53	Segmentation, Disease progression	Longitudinal
2016	391	200	NA	191	Segmentation, Disease progression	Longitudinal
2017	477	285	46	146	Segmentation, Survival prediction	Pre-operative
2018	542	285	66	191	Segmentation, Survival prediction	Pre-operative
2019	626	335	125	166	Segmentation, Survival prediction	Pre-operative
2020	660	369	125	166	Segmentation, Survival prediction	Pre-operative
2021	2040	1251	219	570	Segmentation, MGMT classification	Pre-operative

Table 2.1: Summary of BRATS datasets over the years as taken from [19].

tus, offering insights into the likely response to certain chemotherapies [19]. The differences between the table can be seen in Table 2.1.

In this research, the UCSF-PDGM dataset, which contains a small subset of images from the BraTS 2021 dataset, was utilized. These images are identified by the BraTS21ID and BraTS Segmentation Cohort, which indicate whether the image was used during Validation, Training, or Testing. Based on the 2021 BraTS challenge, the "ground truth" labels for the training dataset, which were created by experienced neuroradiologists following a standardised protocol. These labels delineate three distinct tumor sub-regions: enhancing tumor (ET), necrotic core (NC), and vasogenic/infiltrative edema (OE) [19]. The segmentation precision scores are evaluated for the validation and test datasets, focusing on three main tumor regions: whole tumor (WT), tumor core (TC), and enhancing tumor (ET). The whole tumor region encompasses all three sub-regions, including edema, necrotic and non-enhancing core, and enhancing tumor. The equations below define these regions:

$$\mathrm{WT} = \mathrm{OE} + \mathrm{NC} + \mathrm{ET} \tag{2.1}$$

$$\mathrm{TC} = \mathrm{NC} + \mathrm{ET} \tag{2.2}$$

Regarding methylation status, the BraTS datasets have only two states: 0 (unmethylated) and 1 (methylated), while the UCSF-PDGM-v3 has an additional status of -1, which represents an intermediate state. Consequently, when evaluating the methylation status via the Validation AUC score, the indeterminate values (-1) will have to be omitted.

2.3 Segmentation Methods and Algorithms

A variety of methods and techniques have been developed and applied to realize fully automatic brain tumor segmentation. These approaches encompass region-based methods like

the Watershed technique [52] and pixel classification strategies. In brain tumor segmentation, these methods employ either supervised or unsupervised classifiers to categorize pixels within a specified feature space. Supervised segmentation methods in brain tumor analysis involve two key phases: training and testing [51]. During training, models learn to map data features to labels using accurately labeled data, which requires human intervention [51]. This process allows models to then classify unlabeled data in the testing phase [51]. Supervised methods typically offer higher accuracy compared to unsupervised methods due to the precise label guidance provided during training. Specifically for brain tumor segmentation, as in the case of the BraTS dataset, these methods leverage labeled ground truth data to train the model, enabling it to apply learned features to new, unseen data during performance testing. Unsupervised segmentation techniques in image analysis, unlike supervised methods, do not require manually labeled training data [51]. These algorithms automatically group pixels with similar characteristics, such as intensity and texture, to form homogeneous regions in the image. This process inherently decides the number of necessary classes, each corresponding to a distinct region [51, 53]. However, unsupervised methods have certain drawbacks, such as the need to predefined the number of regions, the risk of tumors being split into multiple areas, and challenges in identifying textural boundaries or clear intensity variations within tumors [53]. Supervised segmentation methods in brain tumor analysis increasingly utilize deep learning techniques, particularly Convolutional Neural Networks (CNNs). These methods have surpassed traditional ones like watershed algorithms [52] in effectiveness, establishing themselves as the leading approach in automatic segmentation. An in-depth analysis of these state-of-the-art methods will be explored in further sections.

2.4 State-of-the-Art Methods

2.4.1 U-Net

U-Net, a Fully Conventional Network (FCN) initially introduced by Ronneberger et al. [9], is a deep learning method tailored for biomedical image segmentation, particularly effective for brain tumour segmentation. It's architecture consists of an encoder and a decoder path [54], is designed for precise pixel classification in medical images where training data maybe limited. This network is adept at maintaining the spatial dimensions of the original image, utilising up sampling and down sampling instead of pooling to preserve important features [55].

The U-Net architecture, an enhancement over previous neural network models like the one cited by Ciresan et al. [56], was designed to address certain limitations in the segmentation of electron microscopy images. The earlier approach, which used a sliding-window technique for per-pixel classification, was limited by its speed and the redundancy created by overlapping patches [57]. U-Net, on the other hand, introduced a U-shaped architecture that includes a contraction path (encoder) and an expanding path (decoder), both featuring symmetrical operations [9]. This design allows U-Net to process the context of an image efficiently while maintaining high-resolution segmentation capabilities.

The U-Net architecture, as depicted in Figure 2.1, showcases its unique U-shaped design, integral to its function. This structure consists of a symmetrical arrangement of feature channels in the upsampling (expansive) and downsampling (contracting) operations [54]. The encoder path of U-Net focuses on reducing spatial dimensions to extract features, while the decoder path employs precise localization to recover spatial information, ensuring high-quality segmentation [9]. This innovative design, devoid of fully connected layers, concentrates on pixel-level context, making U-Net particularly adept for detailed segmentation tasks with limited training samples, like brain tumor segmentation [54]. The dual-path design, forming the U-shape, allows for efficient and accurate segmentation, as shown in Figure

2.1.

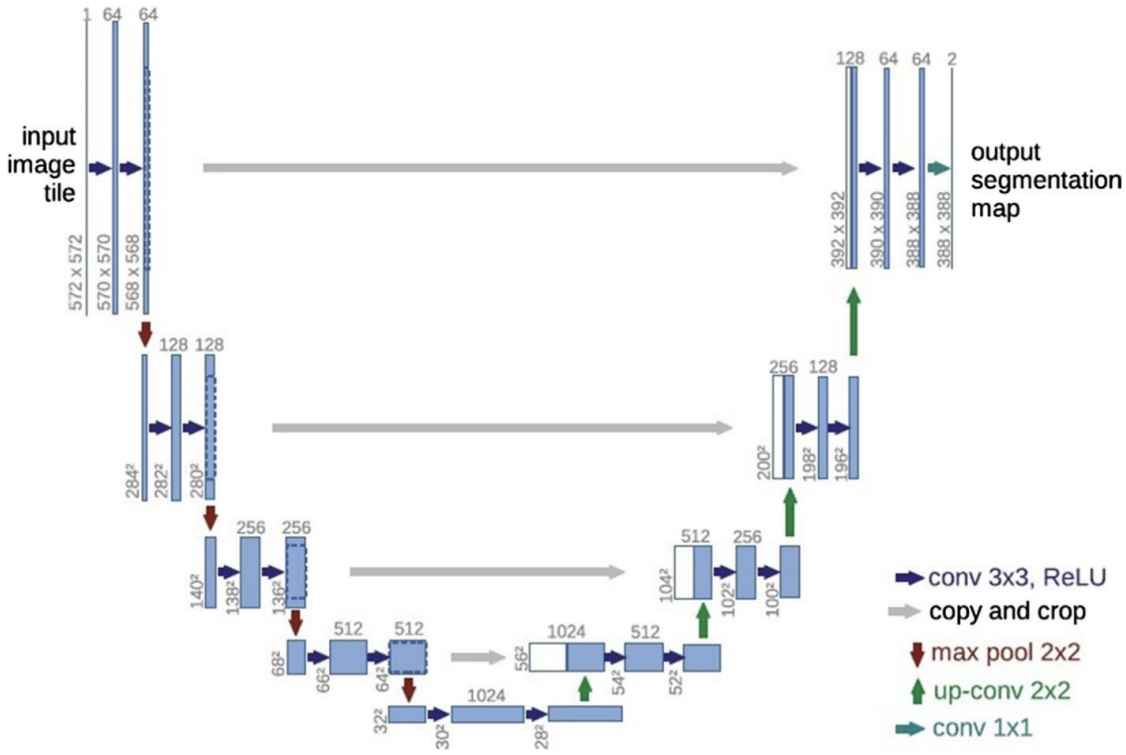

Figure 2.1: An illustration of the U-Net network architecture as reported by Ronneberger et al. [9]

The U-Net model's effectiveness is further evidenced in applications like the BraTS challenge, where adaptations by Myronenko [21] and modifications by Milletari et al. [10] demonstrated its capability to handle complex segmentation tasks in medical imaging. However, it's important to recognize and address the inherent limitations of U-Net, such as its sensitivity to data imbalance and the need for task-specific optimizations. Addressing these challenges is crucial for enhancing U-Net's effectiveness in intricate medical imaging tasks.

2.4.2 V-Net

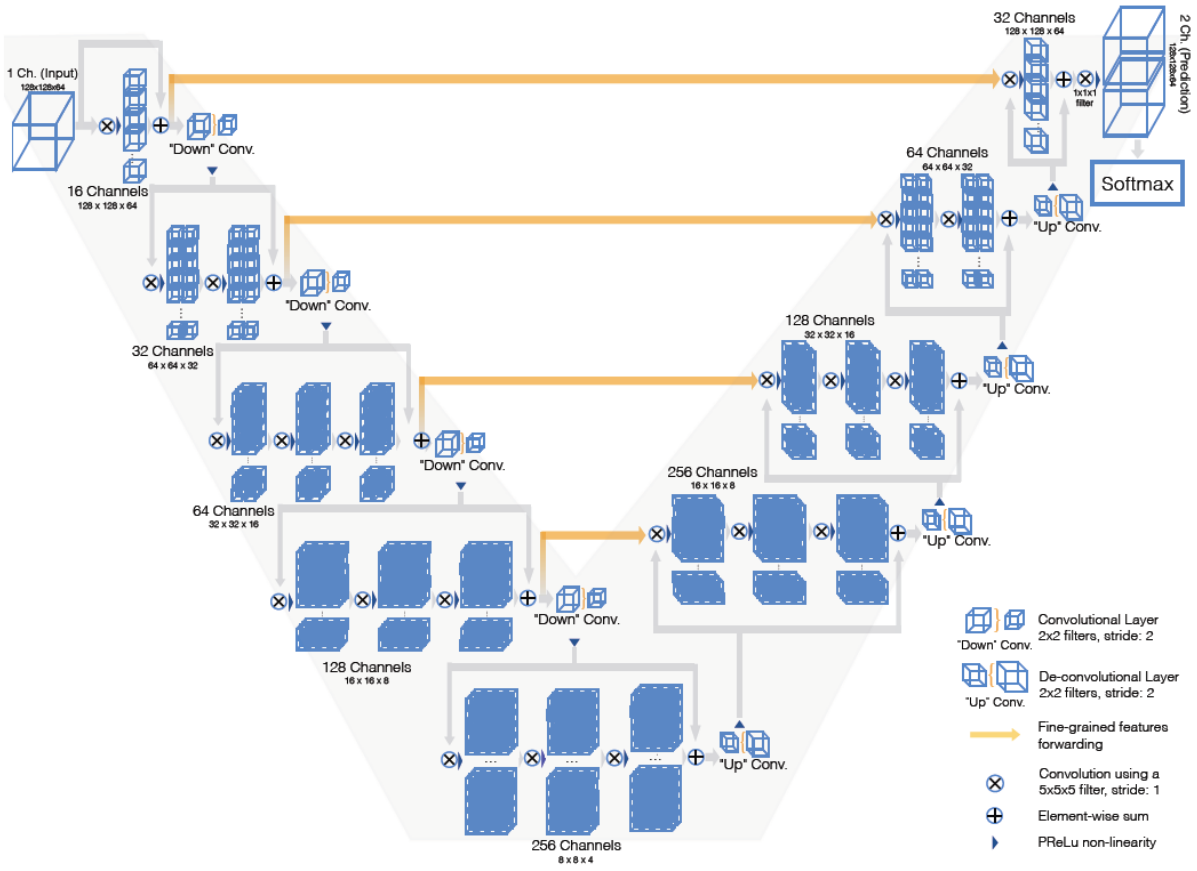

Figure 2.2: A schematic visualisation of V-Nets as reported by [10]

The V-Net architecture, an advancement of U-Net designed by Milletari et al. [10], is specifically adapted for 3D medical image segmentation. It is distinguished by its use of 3D convolutions, enabling it to process volumetric data more effectively than traditional methods that handle 3D images slice-by-slice [10]. The architecture is visible in Figure 2.2. This approach is crucial in medical imaging where the anatomy of interest often occupies a limited region within the scan, with the rest being background [10]. V-Net's dice loss layer is a key feature that addresses the common issue of foreground-background imbalance in medical images, making it suitable for binary segmentation tasks without the need for sample re-weighting [10].

In a study conducted by [58], the V-Net's application in brain tumor segmentation was

explored. This approach helped the model achieve results of 0.91 for the whole tumor (WT) and 0.83 for the tumor core (TC). The model employs a two-step process: localizing the brain tumor area and differentiating various tumor regions, focusing solely on tumor voxels [58]. Despite its advanced architecture, the model faced challenges in capturing less common tumor regions during dense-training regimes with small batch sizes [58]. This limitation indicates the need for future enhancements in V-Net's learning procedure, specifically in up-weighting smaller tumor regions to improve detection accuracy. Such improvements are crucial for ensuring that advanced models like V-Net can effectively segment all tumor regions, regardless of their size or commonality [11].

2.4.3 Multi-path architecture

In network architecture, the concept of a network path refers to the progression of data processing. Multi-path networks, contrasted with single-path ones, offer computational efficiency and enhanced feature extraction capabilities. They can simultaneously extract and process features at different scales through varied pathways. This can be seen in Figure 2.3. This multiplicity allows the combination of both global and local features: larger scale paths focus on global features, capturing overarching patterns, while smaller scale paths hone in on local features, detailing finer aspects. This approach, exemplified in studies by Urban et al. [59], Zikic et al. [60], and Pereira et al. [61], underscores the advantage of multi-path networks in complex data analysis tasks.

The work by Havaei et al. [62] pioneered multipath network solutions in brain tumor segmentation. They developed a two-pathway structure: one for local tumor information using a 7x7 convolution kernel and another for global context with a 13x13 kernel [11]. This approach allowed for the extraction of both detailed and broad features. Castillo et al. [63] furthered this concept with a three-pathway CNN, each path handling different resolution patches, enabling the network to learn features at varying spatial scales. Building on these ideas, Akil et al. [64] enhanced the network structure by incorporating overlapping patch

prediction methods, linking the target patch's center with neighboring patches for improved accuracy [11].

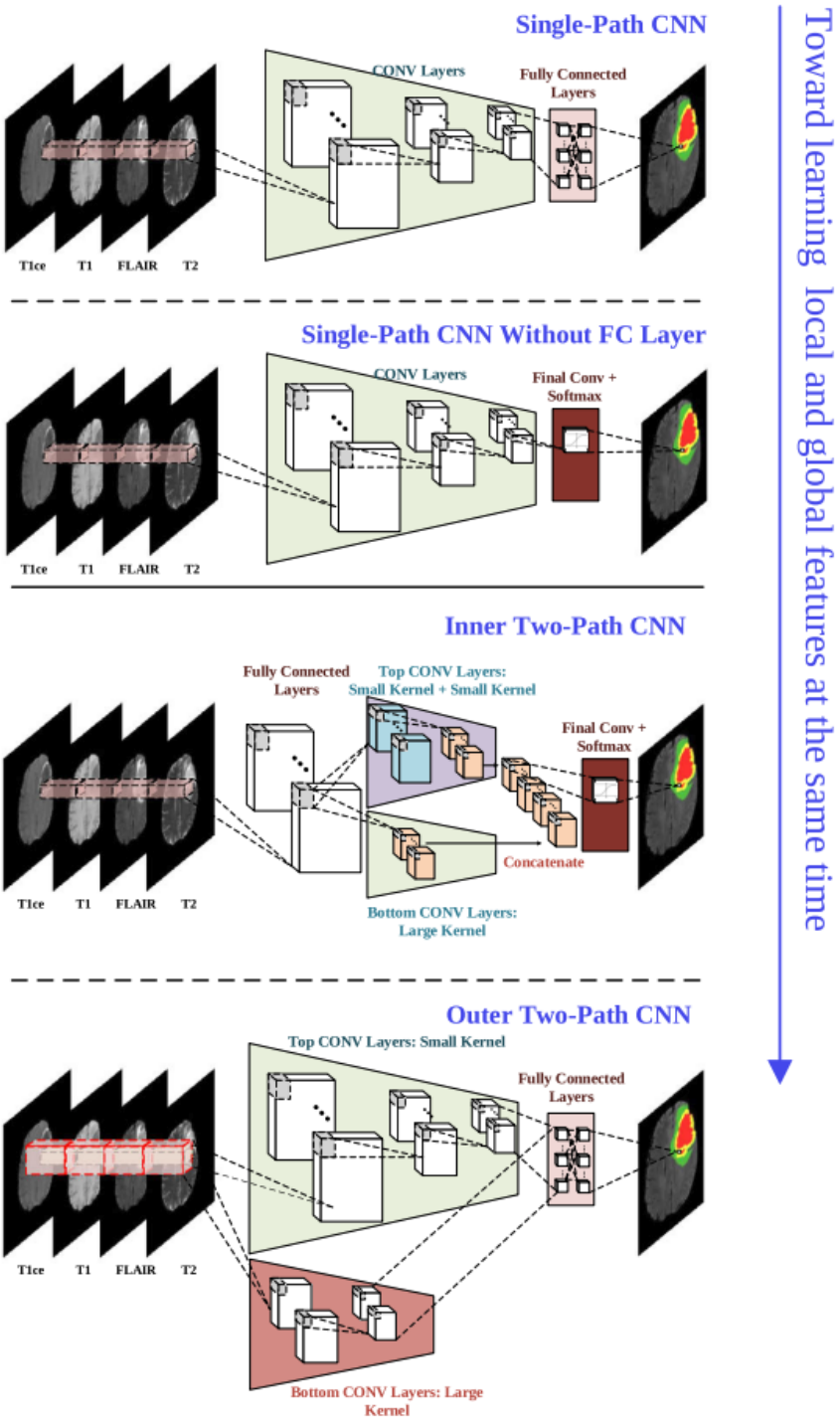

Figure 2.3: A high-level comparison between single-path and two-path Convolutional Neural Networks (CNNs) involves examining their structural differences and respective capabilities in data processing as taken from [11]

2.4.4 Encoder–Decoder Architecture

Single and multi-path networks use image patches as input, aiming to classify each patch or its central pixel. However, accurate mapping from patch level to category label poses significant challenges [11]. The segmentation performance is influenced by the input patch's size and quality. Small patches lack complete spatial information, while larger ones demand more computational resources [11]. Additionally, the feature-to-label mapping, typically performed by the final fully connected layer, can either be too simplistic or excessively complex, impacting memory usage. To address these issues, recent studies have shifted towards fully convolutional networks (FCNs) [55] and U-Net based encoder-decoder networks [9]. These approaches provide end-to-end processing from input to output segmentation map, enhancing network performance.

Jesson et al. [65, 11] introduced a multi scale loss function to standard FCNs to better model contexts in label domains. U-Net, a variant of FCN [9], excels due to its contracting and expanding paths with skip connections, enabling precise localisation and detailed feature recovery [11]. Several adaptations of U-Net have been explored for brain tumor segmentation, employing strategies like Dice loss functions, data augmentation, and zero padding to maintain output dimensions across convolutional layers. These advances demonstrate the evolving nature of neural networks in accurately segmenting brain tumors as the model had a relatively low computational cost for a new test image and out of it 95% of the time was used to compute the symmetry inputs on CPU [65]. The computation of symmetry maps parallelised on GPU provided a considerable speed-up [65, 11].

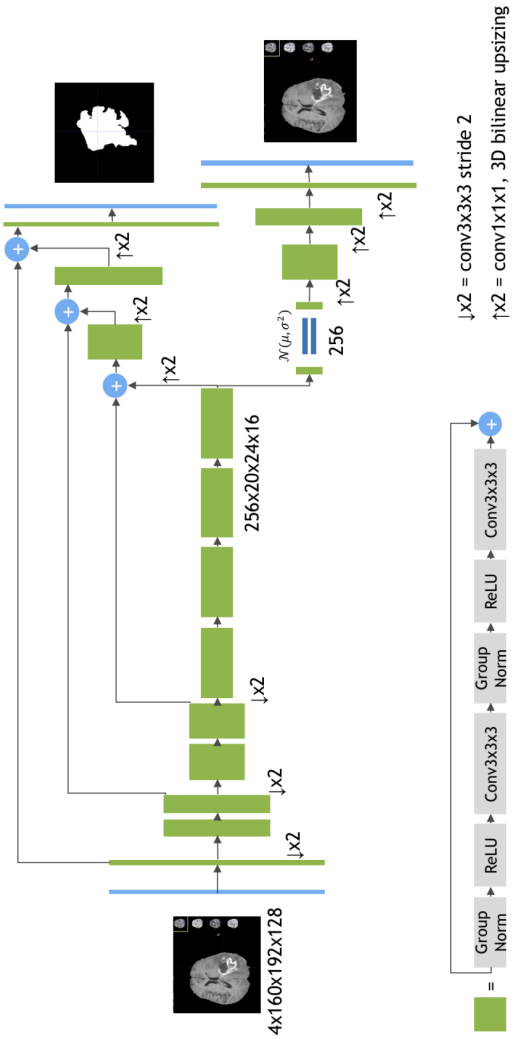

Figure 2.4: Schematic visualisation of the network architecture as adapted by Myronenko. The input for this process is a four-channel 3D MRI crop. It starts with an initial 3x3x3 3D convolution using 32 filters. Each green block in the network represents a ResNet-like block equipped with GroupNorm normalization. The output from the segmentation decoder, which matches the spatial dimensions of the input, comprises three channels. These channels are processed through a sigmoid function to generate segmentation maps for the three tumor subregions: Whole Tumor (WT), Tumor Core (TC), and Enhancing Tumor (ET). Additionally, there's a Variational Autoencoder (VAE) branch that reconstructs the input image. This branch is utilised only during training as a means to regularize the shared encoder.

The top performer in the BraTS 2018 challenge achieved remarkable results, with a Dice score of **0.82 for the enhancing tumor (ET), 0.91 for the whole tumor (WT), and 0.87 for the tumor core (TC)**, as shown in Table 2.1. This outstanding performance was achieved by employing a modified U-Net model, proposed by Myronenko (2018) [21]. The standout feature of this model was its asymmetrical structure, comprising a large encoder for in-depth image feature extraction and a smaller decoder for label reconstruction [21]. It utilised a substantial MRI volume patch size as input to the network and incorporated a variational autoencoder (VAE) for regularisation [21].

The model deployed a random crop of size 160x192x128, ensuring that most image content remained within the crop area. Additionally, it concatenated the four available 3D MRI modalities into a four-channel image as input, capturing all relevant features [21]. The architecture of the model is illustrated in Figure 2.4. Due to the large size of the input data and the limitations of using a single NVIDIA Tesla V100 32 GPU memory, the team employed a batch size of one [21]. This approach, implemented using TensorFlow, enabled the U-Net to deliver high Dice scores across different tumor regions. Notably, the additional VAE branch helped regularize the shared encoder (in the presence of limited data), not only improving performance but also consistently achieving good training accuracy regardless of random initialisation [21]. This approach is remarkable as it leveraged the power of deep learning while addressing the challenges of limited data and computational resources.

2.4.5 Network Cascade

In a network cascade, the output from an initial network, which captures broader or rougher information, is fed as input to a subsequent network [11]. This subsequent network then focuses on refining and detailing the initial output, resulting in more precise and fine-grained segmentation. This approach mirrors a coarse-to-fine strategy, leveraging the strengths of each network stage to enhance overall segmentation accuracy.

Some of the earliest examples are the by Wang et al [12]., who connected three networks

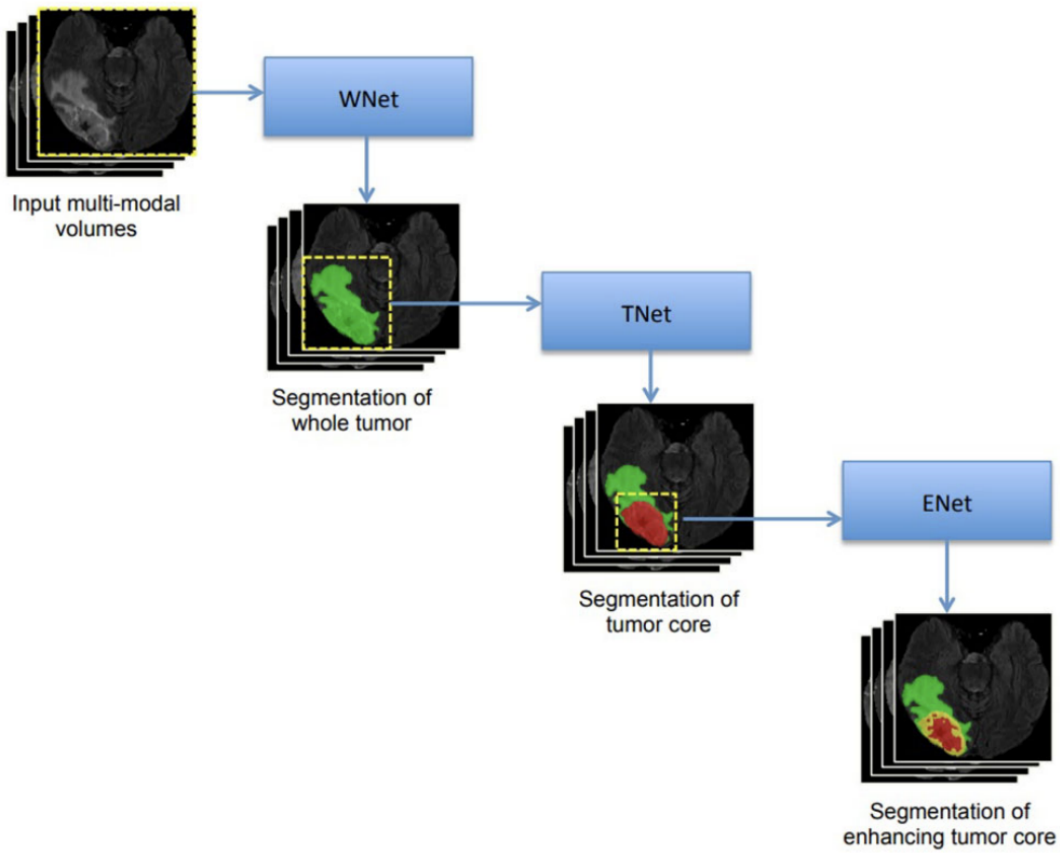

Figure 2.5: The architecture of sequential convolutional networks for brain tumor delineation, adapted from the framework initially described in [12]. WNet is designated for whole tumor segmentation, TNet focuses on the tumor core, and ENet targets the enhancing tumor core.

in series as seen in Figure 2.5. First, WNet segmented the Whole Tumor, and its segmentation output was then fed as input to TNet, which traced the Tumor Core. Finally, the segmentation result of TNet was handed over to ENet for the segmentation of the Enhancing Tumor. In this approach, the segmentation output of the upstream network served as the Region-of-Interest (RoI) for the downstream network [11, 12]. The advantage of this practice was to avoid interference caused by unbalanced data. Additionally, the introduction of anisotropic convolution and the manually cropped input effectively reduced the number of network parameters. However, there were two disadvantages: First, the segmentation effect of the downstream network heavily depended on the performance of the upstream network [11, 12]. Second, since only the upstream segmentation result was considered as input, the downstream network could not utilise other image areas as auxiliary information, which was not conducive to tasks such as tumor location detection [11, 12]. This model achieved results of **0.90, 0.84, and 0.78 for WT, TC, and ET** respectively.

First Position in the Brain Tumour Segmentation Challenge 2019

The leading CNN in the BraTS 2019 challenge was a novel two-stage cascaded U-Net which can be seen in Figure 2.6. Developed by Jiang et al. [13], this method underwent training on the BraTS 2019 dataset. Its performance, when assessed on the testing set, yielded remarkable average Dice scores: **0.88796 for the whole tumor, 0.83697 for the tumor core, and 0.83267 for the enhancing tumor** [13]. These figures highlight the model's exceptional capability in accurately segmenting different tumor regions, positioning it at the forefront of current brain tumor segmentation technologies [13].

In the innovative two-stage cascaded U-Net approach, the initial stage employs a 3D U-Net variant to generate an initial, rough segmentation [13]. This is followed by a second stage that utilises a broader 3D U-Net enhanced with dual decoders. The purpose of this enhanced stage is to refine the initial predictions, focusing on achieving a more precise and accurate segmentation map [13]. This dual-stage process effectively combines coarse and

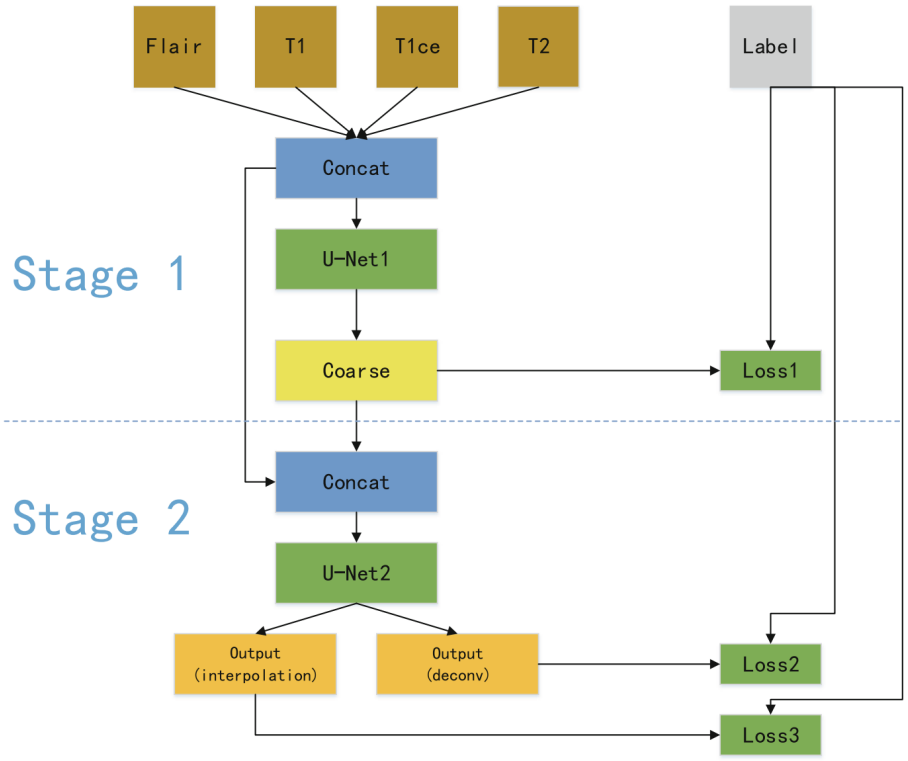

Figure 2.6: Overview of the two-stage cascaded network as taken from [13]

fine-detail segmentation for improved accuracy in brain tumor detection [13].

The two-stage cascaded U-Net methodology leverages data augmentation techniques to mitigate the risk of model over fitting [13]. This process involves varying the intensity of each channel randomly around its standard deviation, coupled with random scaling of the input intensity. Additionally, the approach includes cropping the input images and applying a random flip along the three-dimensional axes with a 50% probability [13]. These steps are crucial in enhancing the model's robustness and generalisability to diverse data sets.

A notable limitation of the two-stage cascaded U-Net is its substantial memory requirement [13]. The model necessitates cropping input images to a smaller size for feasibility, reducing them from 240 x 240 x 155 voxels to 128 x 128 x 128 voxels, to manage memory constraints. This process is essential for accommodating the model within the 12+ Gigabyte memory limit typical of a NVIDIA Titan V GPU [13]. Additionally, variability in performance was observed across different iterations of the model, indicating a degree of inconsistency in individual model outcomes.

2.4.6 Transformers

The discussion above highlights that CNN-based models, despite achieving high performance, are constrained by computational and memory resources [11]. Particularly, their effectiveness is limited in pairing different modalities for enhanced segmentation performance, as this process requires significant computational power. [11] Additionally, while methods like addition or concatenation in CNNs are efficient in terms of parameters, they might fall short in adequately representing physical features of the images [11]. On the other hand, transformers, originally developed for Natural Language Processing (NLP), have been adapted for computer vision tasks by tokenizing images, offering a different approach to handling these challenges [66].

TransUNet

In the realm of transformer-based models, Chen et al. employed the TransUNet model for image segmentation tasks, achieving Dice Scores of **0.8182, 0.8772, and 0.9191** for ET, TC, and WT, respectively [66]. The methodology of TransUNet initiates by tokenizing the input images $x \in \mathbb{R}^{H \times W \times C}$ into sequences of 2D patches [66]. Each patch, of size $P \times P$, is projected into a D-dimensional embedding space, supplemented with position embeddings to conserve spatial details, expressed as

$$z_0 = [x_{1p}E; x_{2p}E; \ldots; x_{Np}E] + E_{\text{pos}},$$

where $E \in \mathbb{R}^{(P^2 \cdot C) \times D}$ and $E_{\text{pos}} \in \mathbb{R}^{N \times D}$. This process embeds the image patches into a latent space, capturing spatial information alongside positional embeddings [66].

The transformer encoder's core consists of L layers of Multihead Self-Attention (MSA) and Multi-Layer Perceptron (MLP) blocks [66]. The output for the ℓ-th layer is defined as:

$$z'_\ell = \text{MSA}(\text{LN}(z_{\ell-1})) + z_{\ell-1},$$

$$z_\ell = \text{MLP}(\text{LN}(z'_\ell)) + z'_\ell,$$

where LN signifies layer normalization. Such an arrangement of encoder blocks within each transformer layer amplifies the representation of features.

To mitigate the detail preservation limitations inherent in transformers, TransUNet integrates a hybrid CNN-Transformer architecture [66]. The CNN part functions as a feature extractor, yielding high-resolution feature maps for subsequent tokenisation and processing by the transformer encoder. This synergy of global and local feature retention is further enhanced by a novel cascaded upsampler (CUP), which incrementally enhances the spatial resolution of the transformer's output [66]. Mimicking the U-Net's U-shaped design, this architecture enables feature consolidation at varied resolution stages via skip-connections,

thus ensuring meticulous localisation [66].

CKD-TransBTS

The TransUNet model has demonstrated impressive performance in brain tumor segmentation tasks. However, the CKD-TransBTS model takes a step further by combining the strengths of both Transformer and Convolutional Neural Network (CNN) architectures in a synergistic manner [14]. This approach aims to strike an optimal balance between the global feature extraction capabilities of Transformers and the local feature representation strengths of CNNs.

The CKD-TransBTS model is specifically designed to incorporate inductive bias and locality into its transformer model design [14]. This means that the model is designed to take advantage of the inherent structure and patterns present in the input data, such as the spatial and contextual relationships between different regions in medical images.

One key aspect of the CKD-TransBTS model is its consideration of radiological practices in MRI image interpretation. The model processes different MRI sequences, such as T1-weighted, T1Gd, T2-weighted, and T2FLAIR, in a way that mimics how radiologists interpret these correlated sequences for diagnostic purposes [14]. This approach is intended to produce more clinically accurate and relevant segmentation results.

The model employs a Dual-Branch Hybrid Encoder that processes pairs of correlated image modalities, similar to how radiologists pair modalities for diagnostic purposes [14]. This method is designed to lead to more clinically accurate and relevant segmentation.

Additionally, the CKD-TransBTS model incorporates a Convolutional Stem within its MCCA Block, which efficiently processes MRI sequences by downsampling them. However, it is worth noting that downsampling can potentially lead to some information loss [14].

In the decoder part of the model, the Feature Calibration Decoder plays a crucial role in aligning and calibrating features extracted from both the Transformer and CNN components. This step is designed to bridge any semantic gaps that may exist between the two components

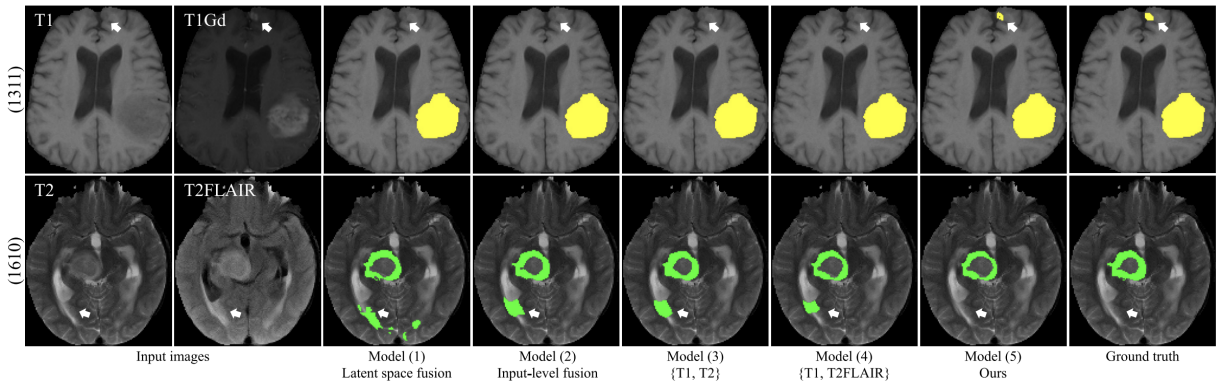

Figure 2.7: The provided images, taken from [14] depict the segmentation outcomes for various categories. In the top row, the focus is on the model's segmentation results within the enhanced tumor region. Meanwhile, the bottom row illustrates how the model performs in segmenting the edema area.

by reshaping vectors from three directions into a uniform shape, concatenating them, and then compressing the channels using a $1 \times 1 \times 1$ convolutional layer [14].

The CKD-TransBTS model is implemented using the PyTorch and MONAI frameworks and is trained on a workstation equipped with an NVIDIA 3090 GPU [14]. Despite using similar augmentation methods and resources as other models, the CKD-TransBTS model outperforms them in brain tumor segmentation tasks, achieving impressive Dice Scores of **0.8850, 0.9016**, and **0.9333** for the enhancing tumor (ET), tumor core (TC), and whole tumor (WT) regions, respectively. These results showcase the effectiveness of the CKD-TransBTS model in accurately segmenting brain tumors.

2.4.7 Summary

To summarise, the majority of cutting-edge models primarily employ variations of the U-Net architecture in their design. These models include the two-stage cascaded U-Net by Jiang et al. [13], transformer models introduced by [14] and Chen et al. [66], autoencoder U-Nets by Jesson et al. [65] and Myronenko [21], and the V-Net approach by Casamitjana et al. [58]. These models tackle various challenges: Havaei et al. [62] address segmentation speed; issues of input data variability and intra-tumoral differences are managed by Pereira et al. [61],

Model	Augmentation or Pre-processing techniques used	Input	Dimensions	Loss	Dice			Dataset	
					WT	TC	ET	Year	Type
Autoencoder [21]	Spatial Dropout, L2 norm regularisation [21]	I	3D	Dice+L2+KL	0.91	0.87	0.82	2018	V
Cascaded CNN [12]	Image cropping from 240x240x155 to 128x128x128, random scaling, random intensity shift [12]	P	3D	Dice	0.9	0.84	0.78	2017	V
Autoencoder [65]	Image cropping to 184x200x152 and standardising using mean and s.d of mask.	I	3D	Categorical CE	0.9	0.75	0.71	2017	V
V-Net [58]	Normalize the data within each input volume by subtracting the volume's mean and dividing by the volume's standard deviation [58]	I	3D	CE	0.91	0.83	0.76	2015	CV
Multipath Architecture [67]	HistogramMatching, normalisation [67]	-	2D	Dice	0.87	0.76	0.74	2014	CV
Multipath Architecture [60]	Downsampling [60]	P	2D	Log Loss	0.84	0.73	0.69	2013	V
CKD-TransBTS [14]	Random zoom, random flip in three directions, Gaussian noise, Gaussian blur and random contrast [14]	P	3D	Dice loss	0.8850	0.9016	0.9333	2021	V
TransUNet [66]	Random rotation and flipping [66]	I	3D	Dice loss	0.8182	0.8772	0.9191	2021	V

Table 2.2: Comparison of different models for brain tumor segmentation. In column Input, 'P' means the patch and 'I' means the image. In column Dice, 'WT' means whole tumor, 'TC' means tumor core and 'ET' means enhancing tumor. Column Dataset indicates the associated dataset with the reported segmentation performance. In column Type, 'CV' means the cross-validation on BraTS training set, 'V' means the BraTS validation set. Inspiration taken from [11]

and the results have bene double checked agaisnt the papers that they have been taken from.

McKinley et al. [68], and Baid et al. [69]; computational efficiency is enhanced by Kamnitsas et al. [70]; overfitting is mitigated by Jiang et al. [13] and Zhou et al. [71]; and memory constraints are addressed by Myronenko [21] and Isensee et al. [72]. However, computational demands are noted limitations in the works of Pereira et al. [61] and McKinley et al. [68], alongside memory consumption issues, and challenges with accurate predictions in smaller anatomical areas as identified by Zhao et al. [73]. In terms of Transformers the mitigation of detail preservation is handled by Chen et al. [66] and bridging of semantic gaps between the reshaped vectors is handled by [14] . A comprehensive overview of the top-performing CNN models for the BraTS challenges from 2013 to 2021 is detailed in Table 2.1.

2.5 Radiogenomic Classification

The Radiogenomic challenge has been described in Section 2.2 of this report and insight into the role the gene plays in therapy for patients with glioblastoma has been discussed in Section 1.1.

2.5.1 First Position in Task 2 of the BraTS Challenge

Firas Baba secured the top spot in the competition with a prediction accuracy of **0.62174**, utilizing a 3D CNN framework that incorporated Resnet10 and EfficientNet B0-B3. These models are renowned for their efficacy across various applications. The computational setup for the model included an NVIDIA RTX 3090 with 32GB RAM and an AMD Ryzen™ 7 5800X CPU with 8 cores, striking an optimal balance between computational efficiency and performance [74, 75].

However, the model faced challenges with the EfficientNet's batch normalization layer, which produced inconsistent outputs, thereby restricting further exploration of potentially superior models. Notably, this model did not employ ensemble methods, simplifying the computational process but at the potential cost of missing out on performance gains that

ensembling might offer. [74]

The model's validation strategy was robust, offering deep insights into its stability and performance [74]. Despite these strengths, it's important to note that the model required considerable computational resources and time, which might be a significant consideration in its practical application.

2.5.2 Second Position in Task 2 of the BraTS Challenge

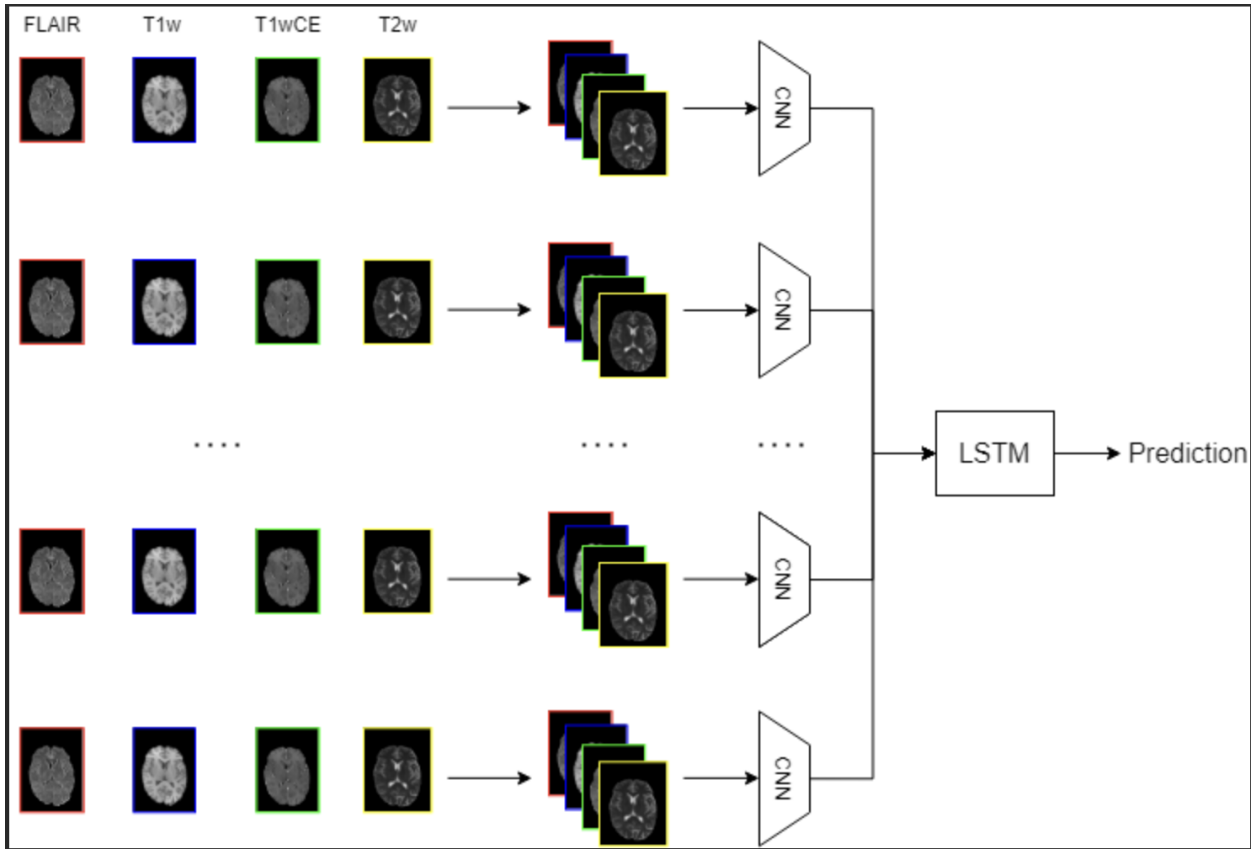

Figure 2.8: Minh Panh's Prediction Model, as depicted in [15], employs a pre-trained EfficientNet B0 for feature extraction from MRI images. The model adapts to the four-channel input by applying a 2D convolution that transforms it into a three-channel feature map suitable for EfficientNet. The original classification head is substituted with a 256-node fully connected layer. Feature embeddings from 10 selected frames are subsequently fed into two LSTM layers with a hidden size of 32, culminating in a prediction layer with a single node to output the score.

Minh Phan secured second place in the competition with a score of **0.61881** using a sophisticated CNN model complemented by an LSTM for the final prediction phase. The core of the model is an EfficientNet B0 [15], adeptly adapted to process 4-channel images by mapping them to a 3-channel feature map [15]. Additionally, its pre-trained classification head was replaced with a 256-node fully-connected layer [15].

Key to the model's data handling was Uniform Temporal Subsampling, which adeptly reduced the sequence frame count to a manageable 10 frames, ensuring a comprehensive overview of the entire sequence [15]. However, this method might overlook critical details present in the omitted frames.

The model also utilized an image concatenation strategy, merging various MRI modalities into a unified 4-channel image to enhance learning from multiple angles [15]. Nonetheless, the approach of zero-filling for missing image types could be suboptimal for handling data gaps.

EfficientNet B0's role in feature extraction capitalized on its extensive pre-training, harnessing the balance between precision and efficiency. The model's LSTM layers, tasked with processing temporal data, were ideal for discerning time-based dependencies in the data, though they introduced added complexity in training and fine-tuning compared to simpler models [15]. The choice of binary cross-entropy loss and Adam optimizer, coupled with a 15-epoch training regimen, was a strategic balance between performance and the risk of overfitting [15]. This is seen in Figure 2.8.

In preprocessing and augmentation, the model employed normalization and resizing to standardize image inputs, a critical step that risked diminishing finer details in the medical images [15]. Augmentation strategies like Horizontal Flip, ShiftScaleRotate, and RandomBrightnessContrast [15] were judiciously chosen for their simplicity and effectiveness in enhancing model robustness without excessive complexity.

Lastly, the model's validation strategy employed Stratified K-Fold Cross Validation, ensuring class balance across folds and thus bolstering model reliability. However, this method

considerably increased computational demands in terms of time and resources [15].

2.5.3 Fourth Position in Task 2 of the BraTS Challenge

David Roberts and Assam Guahati achieved fourth place in the competition, scoring **0.61562**. Their approach combined Object Detection (OD) and Classification, utilizing YOLOv5 and EfficientNet models. YOLOv5 was instrumental in achieving a mean Average Precision (mAP) of approximately 0.68, signifying proficient tumor detection [76]. However, the model's exclusive focus on axial plane images may have led to overlooking critical data from other planes [76].

The model's reliance on the T1wCE MRI series as its sole input could potentially miss nuances captured in other series, necessitating the removal of images with low pixel values [76]. In their image processing and selection strategy, they resampled and chose a specific number of slices, thereby ensuring data consistency [76]. This focus, particularly on axial images above the brain's center slice, was a strategic decision informed by brain anatomy [76].

The model's performance was also enhanced by augmentations such as CenterCrop, CLAHE, and RandomRotate90 [76], alongside standard practices of resizing and normalising images for uniformity [76]. A unique aspect of their method involved using a novel formula for Test-Time Augmentation (TTA) predictions, combining four prediction sets into a final score. For cases lacking usable T1wCE images or tumors undetected by OD, all series were compiled and processed through a 3D network.

If the TTA had 4 prediction sets y_1, y_2, y_3, y_4 then the following formula would be used for the final prediction [76]:

$$y_{\text{final}} = \frac{y_1^2 + y_2^2 + y_3^2 + y_4^2}{4} \tag{2.3}$$

In a strategic move, the team employed ImageNet pre-trained models for 3D classifi-

cation, avoiding the use of models without pre-trained weights [76]. They enhanced their 3D data with CentreCropping and noise addition, using a SAM optimizer based on Adam with a cosine_with_hard_restarts_schedule_with_warmup [76]. This approach yielded a best validation score of 0.6936. However, there was a risk of over fitting to the public leaderboard due to the complexity of the ensembling method [76].

2.5.4 Summary

From the above discussion, we can see that each model deployed a version of of an EfficientNet with different versions of pre-processing techniques. Problems around stability and performance were handled via Firas Baba [74] and their validation strategy, the handling of time-based dependencies in data was handled by [15], and the problem of handling unusable data in terms of the T1wCE modality is handled by [76]. But overall based on the results in Table 2.3 varying between 0.60-0.65, we can see that these results are not very clinically useful.

2.6 Techniques for Brain Tumor Segmentation

2.6.1 Pre-Processing techniques

Data preprocessing is an essential step in preparing raw input data, making it more suitable for neural networks. MRI images often contain artifacts resulting from the acquisition protocol and hardware. It's important to correct these artifacts before inputting the images into the network to enhance performance. A common artifact is the smooth intensity variations across the image, known as a bias field. The non-parametric nonuniform normalization (N3) [81] method has emerged as a preferred technique for correcting bias field, thanks to its simplicity and open-source availability. This technique was further refined in the N4ITK [82] version. These methods are designed for individual images. For achieving consistent intensity levels across different patients and acquisitions, the intensity normalization method

Model	Dimensions	Pre-processing and Augmentation Techniques	Score
3D CNN, EfficientNet, ResNet10 [75]	3D	Batch Normalisation [75]	0.62174
EfficientNet B0 with 2 LTSM layers [15]	2D	BatchNormalisation [15]	0.61881
EfficientNet-B3 [77]	2D	BatchNormalisation, brightness range, horizontal flip, width shift range, height shift range, rescale, zoom range [78]	0.61732
YOLOv5 and EfficientNet [76]	2D & 3D	CenterCrop, CLAHE, RandomRotate90, HorizontalFlip, VerticalFlip, RandomContrast, Cutmix (2D); CentreCropping and adding noise to the 3D data [76]	0.61562
Unet++ with Densenet121 and LTSM [79]	2D	Resizing, Chunk Averaging, Filtering [79]	0.60696
UNet [80]	2D	Resizing [80]	0.60394

Table 2.3: Models, augmentation methods and the respective scores of the top performers in detecting the MGMT promoter methylation status challenge based on [20]

proposed by Nyul and colleagues [83] can be utilized. Additionally, normalizing the image dataset to a zero mean and a standard deviation of one is another widespread preprocessing method that helps eliminate feature bias. Image cropping is also beneficial in reducing background pixels as much as possible.

2.6.2 Post-Processing Techniques

The post-processing phase is aimed at enhancing the accuracy of segmentation outcomes by reducing misclassifications or false positives. This improvement is achieved through the use of algorithms such as conditional random fields (CRF) [70, 84, 73], Markov random fields (MRF), connected component analysis [62, 84, 61], and morphological operations [61, 85] . CRF and MRF techniques are particularly effective in eliminating false positives by leveraging model predictions along with detailed image information, including pixel interactions and edge details for more precise adjustments. However, these methods are known for their high computational demands [86]. Connected component analysis focuses on identifying and isolating connected groups within the image, followed by applying thresholding to eliminate undesired segments. To further refine segmentation edges, morphological operations such as erosion and dilation are used sequentially to remove false positives effectively.

2.6.3 Class Imbalance

The class imbalance issue significantly influences the accuracy of segmentation tasks, characterized by an uneven distribution of class instances in the training data. For instance, in brain tumor segmentation, the vast majority of voxels, around 98% [62], are healthy, leading to a model preference for the dominant class. Conversely, an evenly distributed training set tends to bias the model towards tumor classes [85]. To mitigate this challenge, various strategies have been employed.

Literature has seen the adoption of loss modification techniques to counteract class imbalance. For example, Lin et al. [87] introduced a loss function that dynamically adjusts

scaling based on the model's classification confidence, thereby focusing more on incorrectly classified samples. Another approach involves the use of dice loss [88] to directly tackle the imbalance. Furthermore, some studies have applied a weighted loss function [89, 90], assigning different weights to voxels or pixels based on their prevalence in the dataset, ensuring equitable class influence on the model's loss calculation. Kuzima et al. [91] merged cross-entropy loss with Dice loss to better handle the imbalance. Additionally, techniques like hard negative mining [92, 93] have been explored, selecting voxels with the most significant errors and positive instances to refine the model's weight adjustments.

Addressing class imbalance has also been approached through two-phase training methods [62, 85, 94] , where the initial phase involves training on patches with balanced class distribution, followed by training on the actual class distribution in the subsequent phase. This method has been shown to reduce false positives effectively [85]. Moreover, hierarchical segmentation methods have been highlighted as beneficial in overcoming the challenges posed by class imbalance [84].

2.7 Research Questions

The review of the literature has identified several research questions. The following questions have been identified:

- Compare the performance of a 2D U-Net CNN and a 3D U-Net for tumor segmentation on the UCSF-PDGM v2 dataset. How do the results obtained from these architectures on this dataset compare to the performance reported in the literature on similar tumor segmentation tasks using comparable datasets?

- Does the performance of the methylation status improve with the use of a transformer and can it provide clinically useful results when it comes to tumor segmentation?

- How can custom loss functions mitigate the impact of class imbalance on the performance of segmentation models in the context of brain tumor segmentation on 3D

Models, where an unequal distribution of voxel classes exists in the UCSF-PDGM v3 dataset?

Chapter 3

Implementation Methodology

3.1 Overview

To address the research questions, the initial stage involved creating a 2D U-Net as part of the foundational pipeline, laying the groundwork for subsequent developments. Following this, a 3D U-Net was implemented, taking inspiration from Myronenko's work [21]. For the second research question, a separate Transformer model was implemented. Due to computational time constraints, a third U-Net with custom loss functions was implemented to address the issue of class imbalance, aiming to answer the corresponding research question.

3.2 Resources

Similar to the approach used in the methodology by [9], the code development was done in Python using the Tensorflow [95] library for the neural network implementations. The Tensorflow [95] version 2.2 was required due to the specific use of one-hot encoding. This required a compatible version of CUDA 12.2. A python virtual environment needs to be set up in which to run the code. The virtual environment was set up so that all the project specific libraries and inter dependent versions software are kept together in one place so as not to interfere with other programs on the computer. Other software libraries include

pandas [96], seaborn, matplotlib [97], scikit-learn [98], pytorch [99], albumentations [100], numpy [101], tqdm [102], einops [103], Efficient-Net-3D [**?**], and, segmentagtion_models_3d [104] and os.py libraries.

3.2.1 Lab GPU

For this project, accessing the Computer Science Lab's GPU resources was essential, particularly for running machine learning algorithms inspired by Ronneberger's U-Net model [9]. The GPU client assigned to me in the Lab came equipped with specific features, which were detailed in the program's run-time output. These technical properties of the GPU were critical in facilitating the computational demands of the machine learning tasks at hand. For this project, there were two GPUs that were used:

The first one originally assigned to me was the NVIDIA GeForce RTX 3060 with 12 GB of RAM space, CUDA version 12.2, and the Intel(R) Core(TM) i5-6500 CPU @ 3.20GHz (x86_64) with 4 cores per socket . Originally all the models were run on this with the exception of the Class Imbalance tests. But this GPU often resulted in Out of Memory errors occurring and required the image dimensions to be altered for the 3D U-Net Implementation and the Transformer.

To solve this, the NVIDIA DGX 100 with 8 V100-SXM2 cores with 503 GB of RAM Space was allocated to me. The GPU has 80 Intel Xeon CPU E5-2698 v4 @ 2.20GHz CPUs but operates on an older version of CUDA: 11.4. But the GPU was not prone to Out of Memory Errors as this GPU was a shared resource amongst the PhD students of the supervisor so all the models were trained on evenly across 8 cores.

In the end all the models, were implemented on the NVIDIA DGX 100.

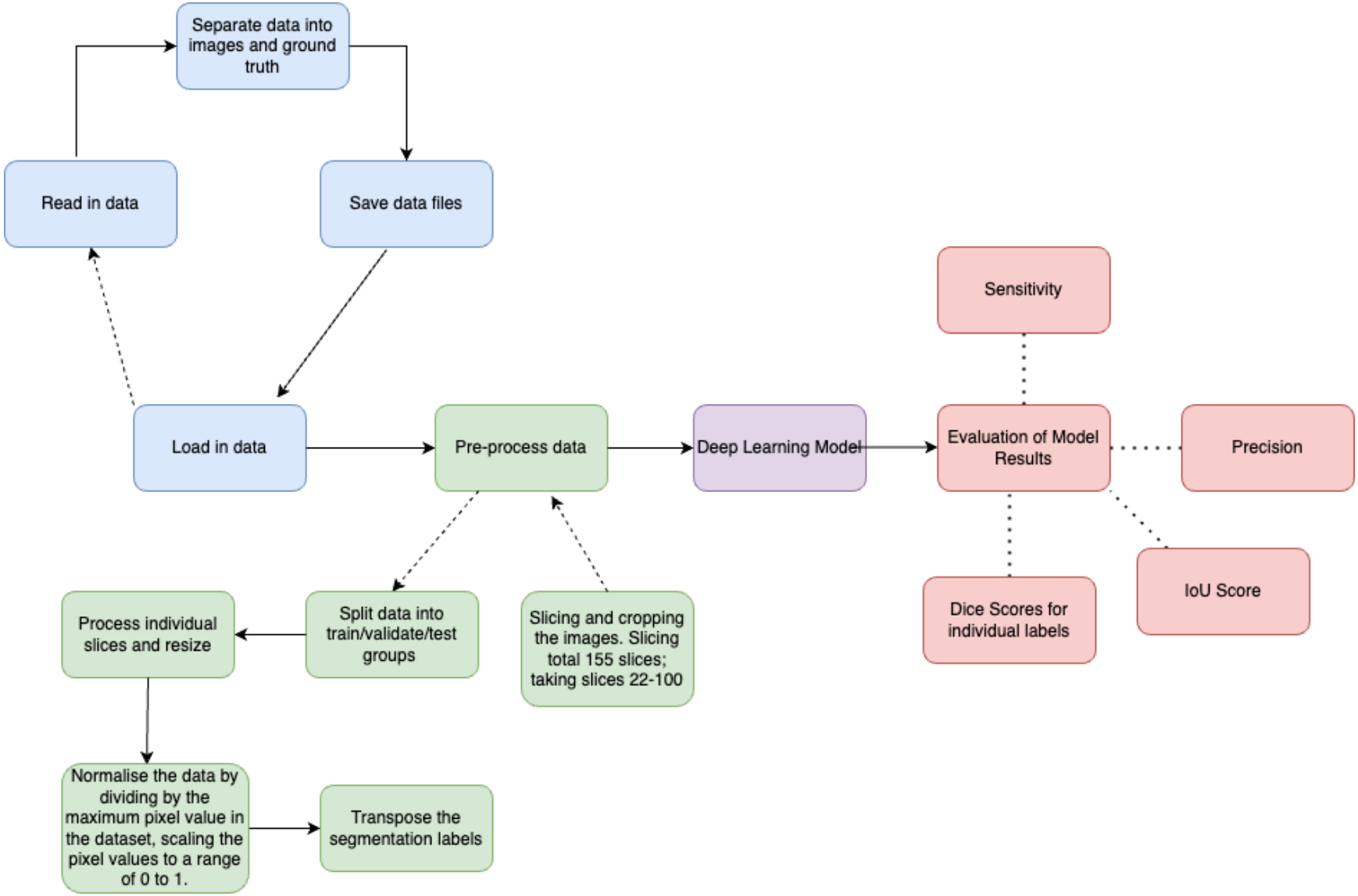

Figure 3.1: 2D U-Net Implementation data Pipeline for the UCSF-PDGM dataset

3.3 2D U-Net Implementation for the UCSF-PDGM dataset

The pipeline for the implementation inspired by [16] is shown in Figure 3.1. The following sections will give an in depth breakdown of the model.

3.3.1 Model Architecture

Name	Ops	Repeat	Output size
Input	-	1	128x128x2
Conv1	Conv (32, 3x3), ReLU, Conv (32, 3x3), ReLU	1	128x128x32
Pool1	MaxPooling2D (2x2)	1	64x64x32
Conv2	Conv (64, 3x3), ReLU, Conv (64, 3x3), ReLU	1	64x64x64
Pool2	MaxPooling2D (2x2)	1	32x32x64
Conv3	Conv (128, 3x3), ReLU, Conv (128, 3x3), ReLU	1	32x32x128
Pool3	MaxPooling2D (2x2)	1	16x16x128
Conv4	Conv (256, 3x3), ReLU, Conv (256, 3x3), ReLU	1	16x16x256
Pool4	MaxPooling2D (2x2)	1	8x8x256
Conv5	Conv (512, 3x3), ReLU, Conv (512, 3x3), ReLU	1	8x8x512
Dropout	Dropout	1	8x8x512
Up1	UpSampling2D (2x2), Conv (256, 2x2), ReLU	1	16x16x256
Merge1	Concatenate (Conv4, Up1)	1	16x16x512
Conv6	Conv (256, 3x3), ReLU, Conv (256, 3x3), ReLU	1	16x16x256
Up2	UpSampling2D (2x2), Conv (128, 2x2), ReLU	1	32x32x128
Merge2	Concatenate (Conv3, Up2)	1	32x32x256
Conv7	Conv (128, 3x3), ReLU, Conv (128, 3x3), ReLU	1	32x32x128
Up3	UpSampling2D (2x2), Conv (64, 2x2), ReLU	1	64x64x64
Merge3	Concatenate (Conv2, Up3)	1	64x64x128
Conv8	Conv (64, 3x3), ReLU, Conv (64, 3x3), ReLU	1	64x64x64
Up4	UpSampling2D (2x2), Conv (32, 2x2), ReLU	1	128x128x32
Merge4	Concatenate (Conv1, Up4)	1	128x128x64
Conv9	Conv (32, 3x3), ReLU, Conv (32, 3x3), ReLU	1	128x128x32
Output	Conv (4, 1x1), Softmax	1	128x128x4

Table 3.1: 2D U-Net architecture, where Conv stands for 2D convolution, UpSampling - 2D upsampling, Concatenate - concatenation along the channel axis, and Dropout - dropout layer. Table inspiration taken from [21]

Table 3.1 represents the U-Net architecture that consists of an encoder path for capturing context and a symmetric decoder path for enabling precise localisation [16, **?**]. The encoder

path comprises four downsampling blocks; each contains two Conv2D layers with 3x3 kernels, `ReLU` activation, and `'same'` padding [**?**, 105]. These layers are followed by a MaxPooling2D layer with a (2, 2) pool size [106]. Progressing through the blocks, the number of filters doubles from 32 up to 512 [9, 16]. The decoder path mirrors this structure with four upsampling blocks. Each block starts with an UpSampling2D layer of size (2, 2) and is followed by a Conv2D layer with 2x2 kernels [105, 107]. `ReLU` activation and `'same'` padding are used throughout [105]. Skip connections create a bridge between corresponding encoder and decoder feature maps to aid in precise localisation [9, 108]. The network culminates with a Conv2D layer that uses a 1x1 kernel and softmax activation to produce a segmentation mask across four classes [9, 16]. Convolutional kernels are initialized using the `'he_normal'` initialiser [109], and spatial dropout is applied at a rate of 0.2 after the deepest layer to encourage robustness [110].

3.3.2 Data Generator and Preprocessing

Regularisation

A custom `DataGenerator` class, inheriting from `keras.utils.Sequence`, is implemented to generate data batches on the fly [111] efficiently. This generator is crucial for the dynamic regularization process during training, employing a shuffle flag to ensure the model is exposed to a randomised sequence of data samples at the beginning of each epoch. This stochastic nature of data presentation is a form of regularization, helping the model to generalise better [112].

Preprocessing Techniques

The `DataGenerator` class also plays a vital role in data preprocessing. It handles loading FLAIR, T1c, and segmentation images for each sample. It is then resized to the target image dimensions of 128x128 pixels using the `resize` function from `skimage.transform`. Normalisation involves scaling FLAIR and T1c images by their maximum intensity value,

while segmentation masks are one-hot encoded with `tf.one_hot` and resized accordingly [16]. The dataset comprises 350 training, 100 validation, and 50 test patient cases, as seen in Figure 3.1. Each subset undergoes preprocessing steps, including image cropping and one-hot encoding, as illustrated in Figure 3.2. The one-hot encoding categorises segmentation masks into four classes: non-tumor regions are encoded as 0, and NECROTIC/CORE, EDEMA, and ENHANCING tumor regions are encoded as 1, 2, and 3, respectively. To cater to the model's input requirements and prioritise tumor-relevant sections, images are cropped from (240, 240, 155) to a focused region of (128, 128) dimensions, centering on 145 slices by excluding the outermost five slices at each end of the volume [16].

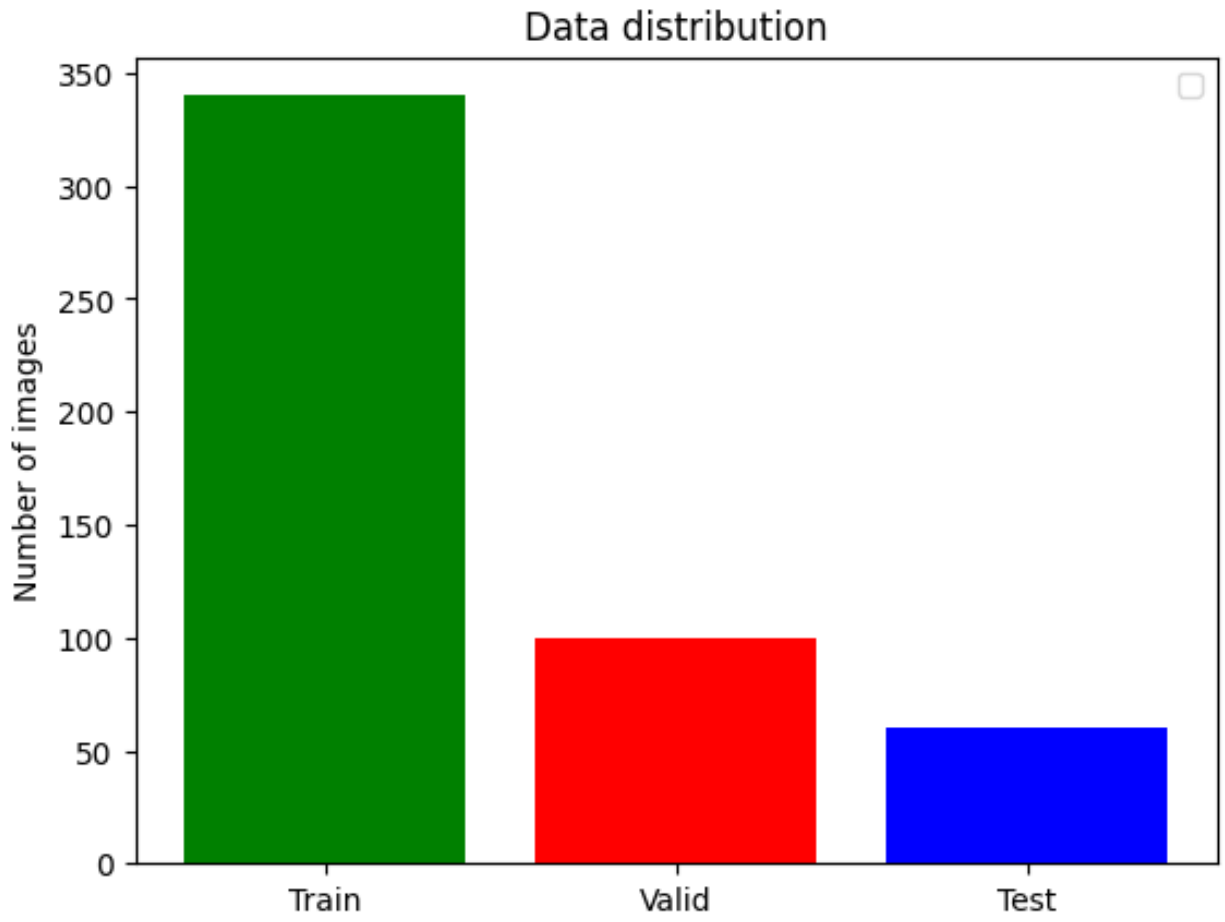

Figure 3.2: Data split in each individual DataGenerator after regularisation and preprocessing for the 2D U-Net implementation as isnpired by [16]

3.3.3 Loss

The loss function [16] deployed is Categorical Cross Entropy which can be seen in Equation 3.1.

3.3.4 Optimisation

For this experiment [16], an Adam optimizer with an initial learning rate of $\alpha_0 = 1e-4$ and utilise the `ReduceLROnPlateau` callback for dynamic learning rate adjustments [113]. This callback monitors the validation loss (`monitor='val_loss'`) and reduces the learning rate by a factor of 0.2 (`factor=0.2`) if no improvement is seen for a patience of 2 epochs (`patience=2`). The learning rate is decayed to a minimum of $1e-6$ (`min_lr=0.000001`) to prevent it from going too low. The model is trained for 36 epochs with a batch size of 1, ensuring each training image is processed precisely once per epoch.

The learning rate adjustment is governed by the following condition [113]:

$$\alpha = \max(\alpha_0 \times \texttt{factor}^{\left\lfloor \frac{\texttt{patience\_exceeded\_count}}{\texttt{patience}} \right\rfloor}, \texttt{min\_lr}) \tag{3.1}$$

where `patience_exceeded_count` is the number of consecutive epochs without improvement in validation loss after the patience threshold has been reached [113]. This helps facilitate the model's gradient descent process, optimizing the network towards higher segmentation accuracy and reliability.

3.4 3D U-Net Implementation for the UCSF-PDGM dataset

For the second part of the first research question, I implemented a 3D U-Net with an on the UCSF-PDGM-Vv2 dataset. This model was inspired by the the work of [21] and replicated the methodology of [22].

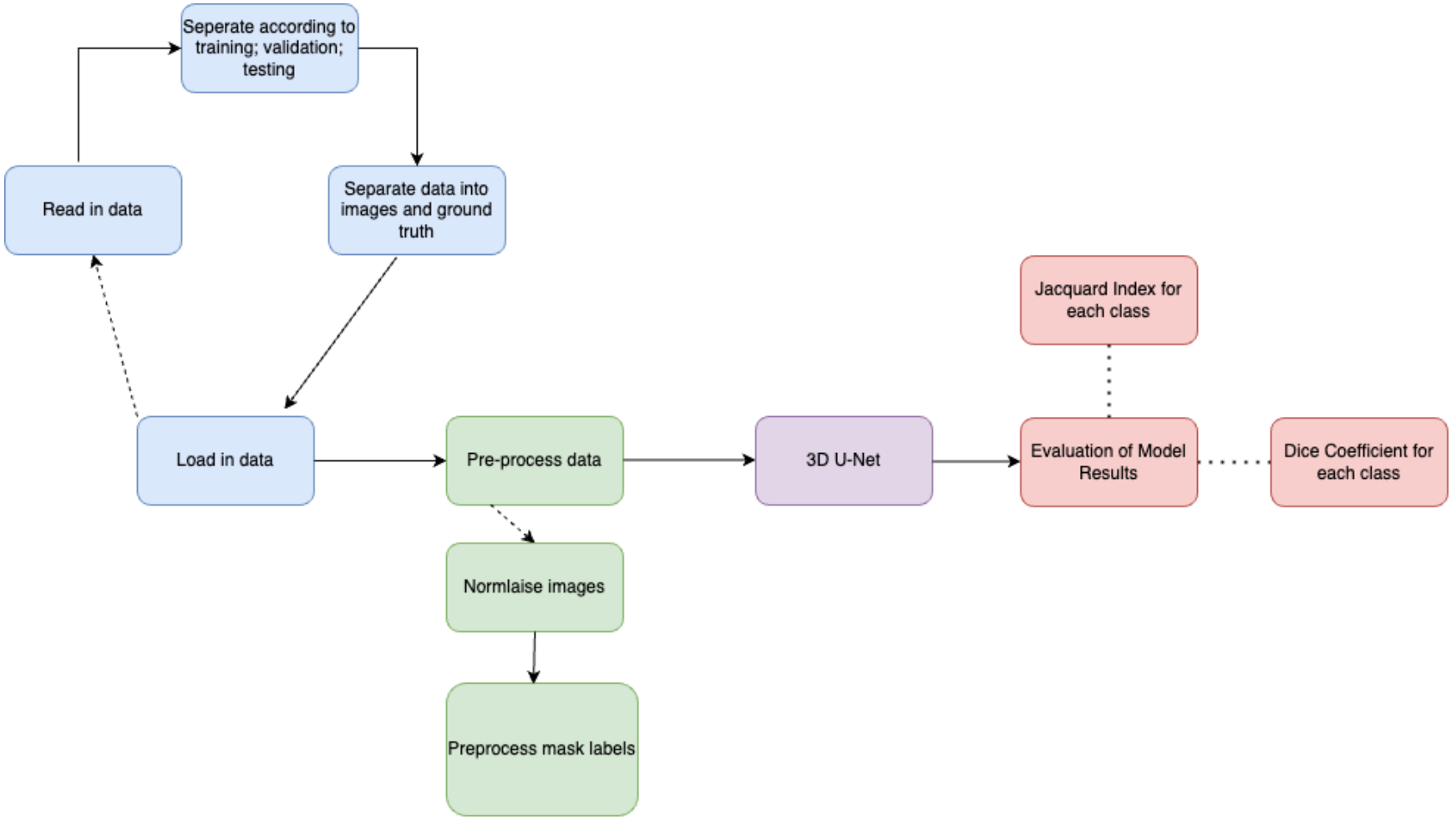

Figure 3.3: 3D U-Net Pipeline for the UCSF-PDGM dataset

3.4.1 Model Architecture

The overall model architecture can be seen in Table 3.2.

Encoder			
Name	**Ops**	**Repeat**	**Output size**
Input	-	1	4x240x240x152
Conv	Conv3d (24, 3x3x3), GroupNorm, ReLU, Conv3d (24, 3x3x3), GroupNorm, ReLU	1	24x240x240x152
Enc1	MaxPool3d (2x2x2), DoubleConv (48, 3x3x3)	1	48x120x120x76
Enc2	MaxPool3d (2x2x2), DoubleConv (96, 3x3x3)	1	96x60x60x38
Enc3	MaxPool3d (2x2x2), DoubleConv (192, 3x3x3)	1	192x30x30x19
Enc4	MaxPool3d (2x2x2), DoubleConv (192, 3x3x3)	1	192x15x15x9
Decoder			
Name	**Ops**	**Repeat**	**Output size**
Dec1	Upsample (2x2x2), DoubleConv (96, 3x3x3)	1	96x30x30x19
Dec2	Upsample (2x2x2), DoubleConv (48, 3x3x3)	1	48x60x60x38
Dec3	Upsample (2x2x2), DoubleConv (24, 3x3x3)	1	24x120x120x76
Dec4	Upsample (2x2x2), DoubleConv (24, 3x3x3)	1	24x240x240x152
Out	Conv3d (3, 1x1x1)	1	3x240x240x152

Table 3.2: Combined 3D U-Net architecture, as taken from [22], adjusted for an input size of $240 \times 240 \times 152$, detailing the layer operations and output sizes for both the encoder and decoder paths.

Encoder

The encoder is constructed from modules that downsample the input and extract features [22]. Utilising the `Down` class, the initial module starts with a `MaxPool3d` layer with a kernel size of $2 \times 2 \times 2$, which halves the spatial dimensions of the input volume. The max pooling is immediately followed by the `DoubleConv` class, comprising two sequential $3 \times 3 \times 3$ convolutions with group normalization and ReLU activation. The ReLU activation function introduces non-linearity, allowing the network to learn complex patterns. At the same time, group normalization ensures consistent training and stable gradients even with smaller batch sizes, chosen over Batch Normalization [112] to enhance model stability during training without the need for Dropout [114]. As the input progresses through each `Down` block, the number of feature channels is doubled, conforming to a design where the output of the final encoder block has its spatial dimensions reduced by a factor of 2^4 compared to the original input size, while simultaneously experiencing an increase in feature channels by a power of two. This approach maintains the integrity of the feature maps, which is crucial

for capturing fine-grained details necessary for accurate segmentation.

Decoder

Mirroring the encoder, the decoder employs the `Up` [22] class to recover spatial resolution while reducing feature channel depth progressively. Each `Up` block begins by upsampling the feature maps, potentially using trilinear interpolation with the `nn.Upsample` function if trilinear is true, or a transposed $2 \times 2 \times 2$ convolution otherwise. The upscaled feature map is then concatenated with a corresponding feature map from the encoder path, a process facilitated by the `DoubleConv` class that executes another set of $3 \times 3 \times 3$ convolutions. Conclusively, the `Out` class, a simplistic $1 \times 1 \times 1$ convolution, is applied to map the decoder's final feature representation to the desired number of segmentation classes, yielding the segmentation mask. This approach ensures that the encoder-decoder architecture of the U-Net is poised for detailed feature extraction and precise localisation [22].

3.4.2 Loading the Data

Regularisation

Several regularisation techniques have been used to mitigate the risk of overfitting and improve the model's generalization ability. Specifically, group normalization is utilized within each `DoubleConv` module as an alternative to batch normalization. Group normalization is advantageous in situations with small batch sizes, where batch normalization tends to be less effective due to insufficient batch statistics. Moreover, group normalization operates on individual examples rather than the entire batch, contributing to more stable training dynamics and better convergence [112]. Although dropout is often used as a regularization technique to prevent co-adaptation of features, it is deliberately omitted here. The exclusion of dropout helps preserve the integrity of the feature maps, ensuring that the salient features essential for accurate segmentation are retained during training [114].

Preprocessing Techniques

The code performs several preprocessing steps on the UCSF-PDGM-v2 dataset. Firstly, it loads the MRI images of different modalities (FLAIR, T1, T1c, T2) using the `load_img` function from the `nibabel` library. The images are not resized due to the strength of the GPU mentioned in Section 3.2, which preserves the entire range of pixel values [22]. The resized images are normalized using the `normalize` function, which scales the pixel values to the range $[0, 1]$. The preprocessed images are stacked along a new axis to create a 4D array of shapes $(4, 155, 240, 240)$. For the training and validation phases, the corresponding tumor segmentation masks are also loaded and preprocessed. The masks are resized, clipped to binary values, and preprocessed to separate the different tumor regions (whole tumor, tumor core, enhancing tumor) using the `preprocess_mask_labels` function [22]. The preprocessed masks are stacked along a new axis to create a 4D array of shapes $(3, 155, 240, 240)$.

Data Augmentation

Data augmentation techniques are applied to the pre-processed images and masks during the training and validation phases [22]. The specific augmentations used are determined by the `get_augmentations` function based on the phase (train or valid) [22]. This has its limitation as not all the 501 images have been assigned a `BratsSegmentationCohort`. The augmented images and masks are returned as a dictionary by the `__getitem__` method of the `BratsDataset` class. No data augmentation is applied during the testing phase. Overall, the model is trained on 216 images, validated on 71, and tested on 206 images.

3.4.3 Loss

Two loss functions were used: Binary Cross-Entropy (BCE) loss and Dice loss. The BCE loss is calculated using the `BCEWithLogitsLoss` function from PyTorch, which combines a sigmoid activation function and binary cross-entropy loss [22]. The Dice loss is a custom implementation that measures the overlap between the predicted segmentation and the ground

truth. The Dice loss is calculated as 1 minus the Dice coefficient, where the Dice coefficient is defined as:

$$\text{Dice} = \frac{2 \times \text{Intersection}(\text{Pred}, \text{Target})}{\text{Sum}(\text{Pred}) + \text{Sum}(\text{Target})} \tag{3.2}$$

where Intersection(Pred, Target) represents the element-wise multiplication of the predicted segmentation and the target segmentation, summed over all elements, and Sum(Pred) and Sum(Target) represent the sum of all elements in the predicted and target segmentations, respectively [22].

The final loss used for training the model is the sum of the BCE loss and the Dice loss, which is computed in the `BCEDiceLoss` class.

3.4.4 Optimisation

Like the 2D model, an Adam Optimiser is used with an initial learning rate of $\alpha_0 = 5e - 4$. The learning rate is adapted over time using a `ReduceLROnPlateau` scheduler. This helps in fine-tuning the learning process by taking more significant steps when the loss landscape is smooth and reducing the step size when approaching a minimum, thereby preventing overshooting and helping the model to converge more effectively [113].

The optimizer operates within the `Trainer` class framework, which facilitates the training of the neural network, `UNet3d`, for segmentation tasks. Each training epoch consists of an optimization phase where gradients are accumulated for 25 steps before an optimization step is taken. This accumulation strategy allows for a more stable and reliable gradient estimation, especially useful when working with a small batch size of 1, as it simulates a larger batch size [22, 21]. The `Trainer` class also handles the zeroing of gradients at the beginning of each optimization step, ensuring that the optimization is based on fresh gradient information from the current batch of data. The model was trained for 50 epochs.

3.5 Transformer Implementation

To answer the second research question, I implemented a transformer model which took inspiration for [23] to see if the model provide as competitive AUC scores like the ones in Table 2.3 as the UCSF-PDGM-v2 model inherits images from the BraTS 2021 dataset.

3.5.1 Model Architecture

Transformer Model Architecture			
Component	**Operation**	**Repeat**	**Configuration**
Patch to Embedding	Linear	1	in_features=512, out_features=512
Dropout	Dropout	1	p=0.1
Transformer Layer 0	Attention	1	dim=512, heads=8, mlp_dim=1024
PreNorm LayerNorm	LayerNorm	2	normalized_shape=512
Attention to_qkv	Linear	1	in_features=512, out_features=24576
Attention to_out	Linear	1	in_features=8192, out_features=512
Transformer Layer 1	FeedForward	1	dim=512, mlp_dim=2048
PreNorm LayerNorm	LayerNorm	2	normalized_shape=512
FeedForward Net	Linear	2	in_features=512, out_features=2048; 2048, out_features=512
Classification Head	Linear	1	in_features=512, out_features=1024
	GELU	1	-
	Dropout	1	p=0.1
	Linear	1	in_features=1024, out_features=1

Table 3.3: Architecture of the Vision Transformer model as adapted from [23]. The model begins with a patch to embedding layer transforming input features to a higher dimension. This is followed by a dropout layer for regularization. The core of the model consists of transformer blocks, each containing a pre-normalization layer, a multi-head attention mechanism that captures dependencies, and a feedforward neural network. 'PreNorm' stands for Layer Normalization applied before each sub-layer, 'Attention' refers to the self-attention mechanism, and 'FeedForward' is a simple feedforward neural network. 'Identity' indicates no operation is performed on the class token. The final output is produced by the MLP head, which includes normalization, linear transformation, a GELU activation function, and dropout for regularisation.

The model's architecture, drawing from the Vision Transformer (ViT) [23, 115], has been adeptly modified for the domain of 3D medical image analysis, bridging the Transformer's adeptness in handling long-range dependencies with the spatial specificity afforded by 3D convolutions [116]. At the outset, patch embeddings disaggregate the input 3D image into a sequence of flattened, fixed-size patches, which are linearly projected to a higher-dimensional space (`dim`). This segmentation enables the model to interpret the image analogously to word sequences in natural language processing tasks, thus effectively processing volumetric data

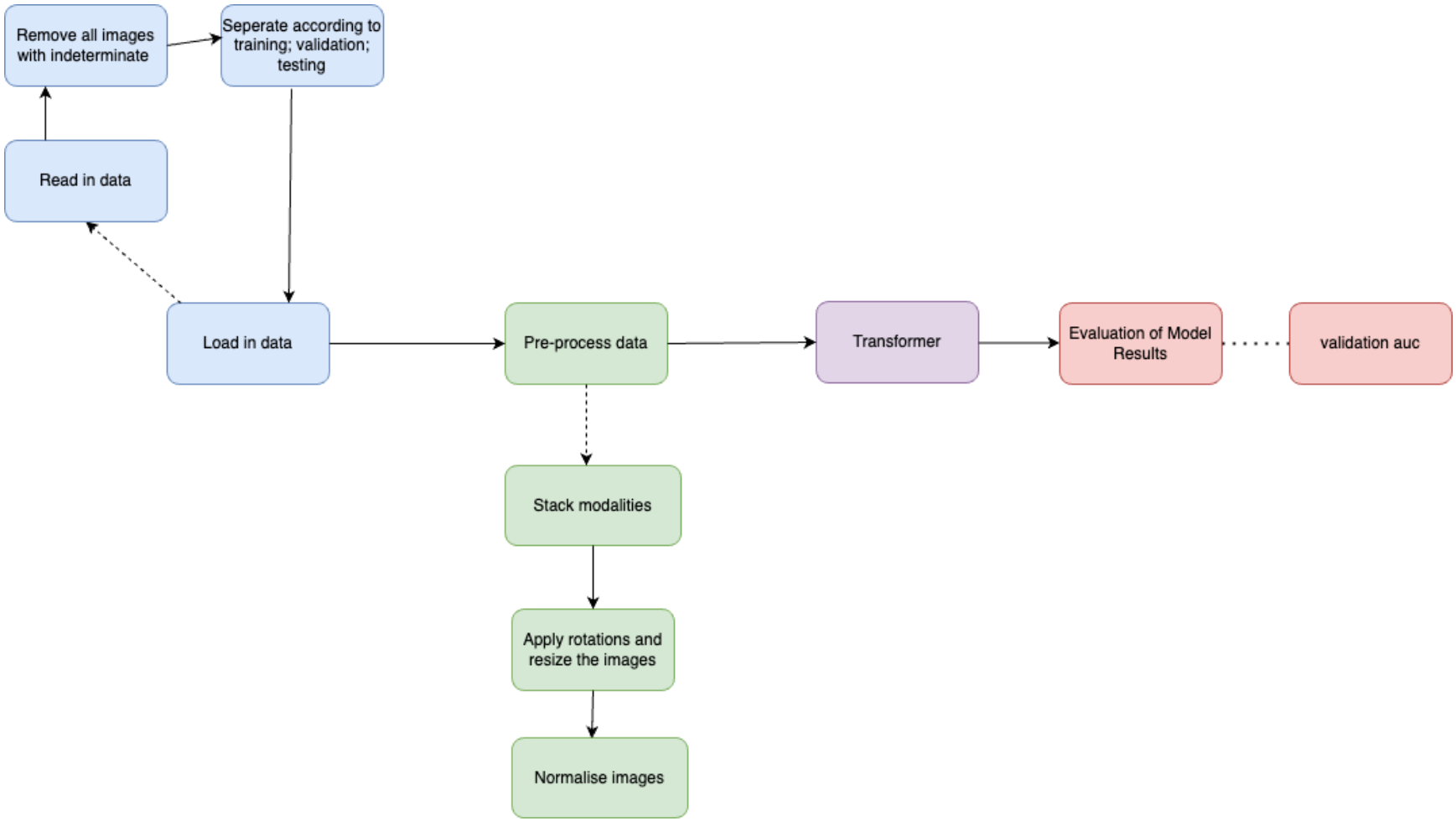

Figure 3.4: Transformer Pipeline for Methylation Status

and capturing intrinsic local and global interrelations [117].

The inclusion of positional embeddings is pivotal, as it imparts the essential spatial context to the patches, a feature inherently absent in the Transformer's architecture. The positional data imbue the model with the ability to recognize the relative positioning of patches within the 3D image, forming a foundational aspect of its learning capabilities [118]. Complementarily, the introduction of a learnable class token (`cls_token`) at the commencement of the patch sequence offers a holistic representation of the input, which, after transformation through the Transformer encoder, becomes instrumental in the classification task [115].

At the heart of the architecture lies the Transformer encoder, a synergistic composition of multi-head self-attention (`Attention`) mechanisms and position-wise feed-forward networks (`FeedForward`), which together empower the model to identify and accentuate pertinent inter-patch features and interactions essential for the classification objective. Augmenting these layers are `PreNorm` (Layer Normalization) and residual connections, which collectively enhance training stability and the propagation of information throughout the network [119, 120].

The culmination of this process occurs in the MLP head, where the processed representation is subjected to a series of operations, including Layer Normalization, linear transformation, GELU activation, and dropout, to distill and relay the decisive features for the final classification [121, 122, 114]. The forward pass of the model encapsulates the intricate processes from patch embedding to classification, rendering it a proficient system for 3D medical image analysis, which is an evolution from ViT's original 2D scope, hereby modified for 3D inputs.

3.5.2 Loading the Data

Preprocessing techniques and Regularisation

The images are loaded in NIfTI format via the `nibabel` library. Rotational data augmentation is applied conditionally, with the potential for no rotation or rotation by 90, 180, or 270 degrees. Image resizing is performed to adjust the spatial resolution to a uniform size of $128 \times 128 \times 128$ due to GPU constraint, ensuring consistency across the dataset [23]. In this experiment, only the FLAIR modality was tested.

For images with a depth different from the predefined size, padding is introduced to normalize the depth dimension, filling any additional space with zeros. The images undergo normalization, where pixel intensities are adjusted to range between 0 and 1, enhancing the model's convergence efficiency. The normalization ensures that the data fed into the model is standardized and accommodates the spatial and intensity variations.

Data Augmentation

The processed images are stacked along the depth dimension and then expanded to add a singleton dimension, making them compatible with the expected input format for the model [23].

3.5.3 Optimiser

An Adam optimizer with an initial learning rate of $\alpha_0 = 1e-4$ was used for this experiment. The model is trained for ten epochs for each modality. The batch size is set to 1, meaning that for each modality, the model is trained on 327 volumes and validated on 82 volumes. When the transformer is run on all the modalities, the model is trained on 1635 volumes and validated against 410 volumes.

3.6 Class Imbalance Experiment Implementation

3.6.1 Experiment Basis

Class imbalance has attracted much attention in recent years, as discussed in Section 2.6.3. This experiment aims to improve the segmentation accuracy and provide competitive Dice Coefficient results with the models in Table 2.2.

3.6.2 Control Variables

To ensure a fair comparison and minimize confounding factors, the following control variables were implemented:

1. Model Architecture;
2. Pre-Processing Techniques; and
3. Optimiser

Model Architecture

As illustrated in Table 3.4 and Figure 3.3, the model architecture design effectively captures and processes features at various scales for accurate segmentation. The network's encoder path consists of multiple layers of 3D convolutions, which extract hierarchical features from the input data [13]. These convolutional layers progressively increase the network's representational capacity, enabling it to learn complex patterns and structures present in the medical images [9]. Dropout layers are strategically placed after the convolutional layers to regularise the network and mitigate overfitting [114].

Pooling layers play a crucial role in the encoder by reducing the spatial dimensions of the feature maps [123]. This downsampling operation allows the network to capture higher-level features by increasing the receptive field of the subsequent convolutional layers [124]. The decoder path of the network employs transposed convolutions to upsample the feature

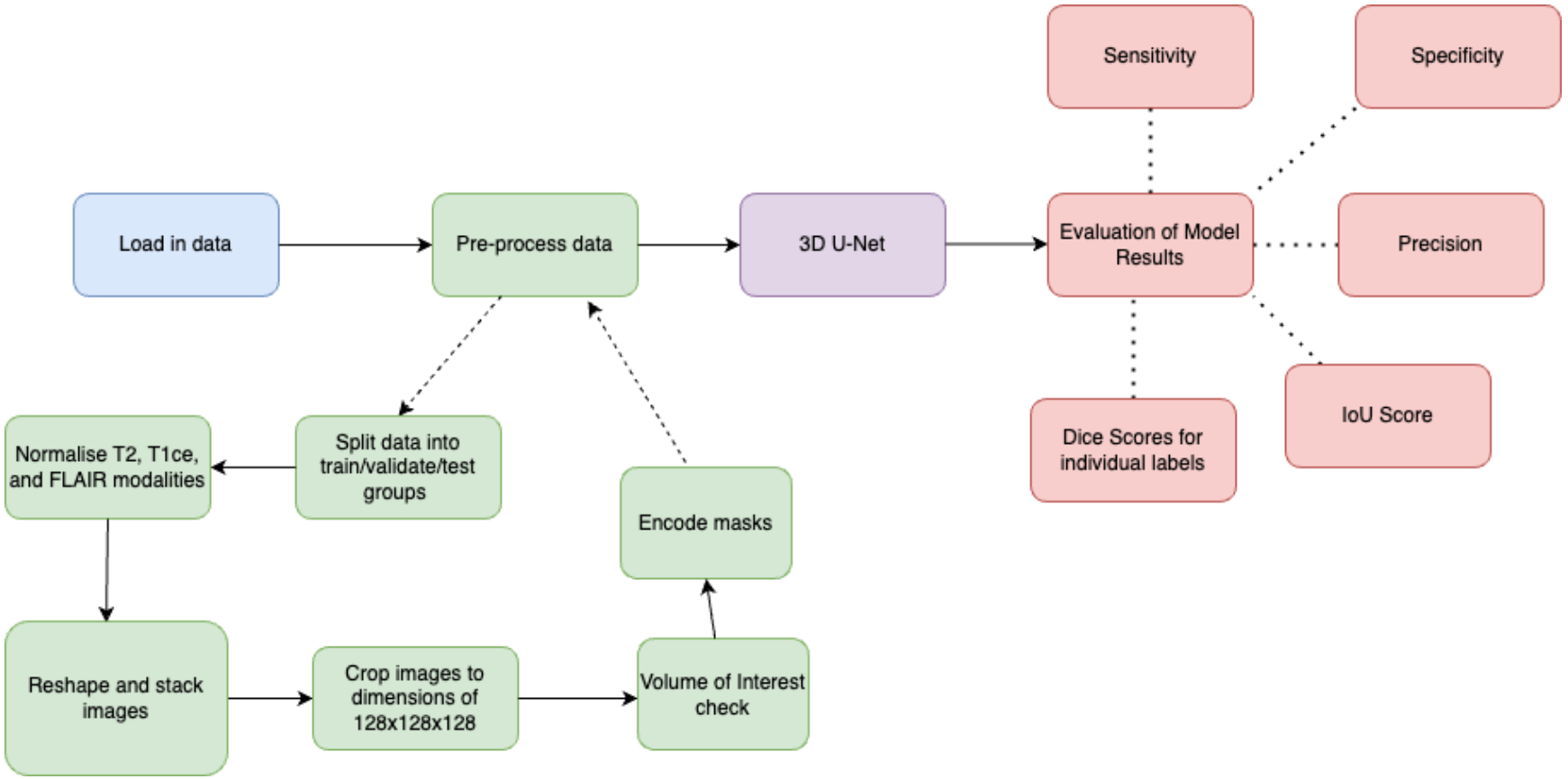

Figure 3.5: Pipeline used for the experiments testing the impact of loss functions on trying to solve class imbalance

Name	Operation	Repeat	Output Size
Input	Input Layer	1	128x128x128x3
Conv1	Conv3D (16) + Dropout	1	128x128x128x16
Conv2	Conv3D (16)	1	128x128x128x16
Pool1	MaxPooling3D	1	64x64x64x16
Conv3	Conv3D (32) + Dropout	1	64x64x64x32
Conv4	Conv3D (32)	1	64x64x64x32
Pool2	MaxPooling3D	1	32x32x32x32
Conv5	Conv3D (64) + Dropout	1	32x32x32x64
Conv6	Conv3D (64)	1	32x32x32x64
Pool3	MaxPooling3D	1	16x16x16x64
Conv7	Conv3D (128) + Dropout	1	16x16x16x128
Conv8	Conv3D (128)	1	16x16x16x128
Pool4	MaxPooling3D	1	8x8x8x128
Conv9	Conv3D (256) + Dropout	1	8x8x8x256
Conv10	Conv3D (256)	1	8x8x8x256
UpConv1	Conv3DTranspose (128)	1	16x16x16x128
Merge1	Concatenate	1	16x16x16x256
Conv11	Conv3D (128) + Dropout	1	16x16x16x128
Conv12	Conv3D (128)	1	16x16x16x128
UpConv2	Conv3DTranspose (64)	1	32x32x32x64
Merge2	Concatenate	1	32x32x32x128
Conv13	Conv3D (64) + Dropout	1	32x32x32x64
Conv14	Conv3D (64)	1	32x32x32x64
UpConv3	Conv3DTranspose (32)	1	64x64x64x32
Merge3	Concatenate	1	64x64x64x64
Conv15	Conv3D (32) + Dropout	1	64x64x64x32
Conv16	Conv3D (32)	1	64x64x64x32
UpConv4	Conv3DTranspose (16)	1	128x128x128x16
Merge4	Concatenate	1	128x128x128x32
Conv17	Conv3D (16) + Dropout	1	128x128x128x16
Conv18	Conv3D (16)	1	128x128x128x16
Output	Conv3D (4)	1	128x128x128x4

Table 3.4: 3D U-Net architecture, taken from [17], detailing the convolutional layers, dropout layers, max pooling, and concatenation operations throughout the model. This architecture is specifically designed for volumetric data processing, featuring a sequence of Conv3D and Conv3DTranspose layers for down-sampling and up-sampling respectively.

maps back to higher resolutions, keeping the same filter size [10]. This upsampling process is essential for reconstructing the segmented regions with precise spatial details [10].

A vital aspect of the model's architecture is the presence of skip connections between the encoder and decoder paths. These connections carry fine-grained information from the encoder to the corresponding resolution levels in the decoder [9, 108]. Skip connections enable the network to accurately localize and reconstruct the segmented regions by preserving spatial details that may be lost during pooling operations [125, 71]. The fusion of high-resolution features from the encoder with the upsampled features in the decoder enhances the network's ability to capture intricate details necessary for precise segmentation [10, 126].

The model also incorporates an excitation module inspired by the squeeze-and-excitation

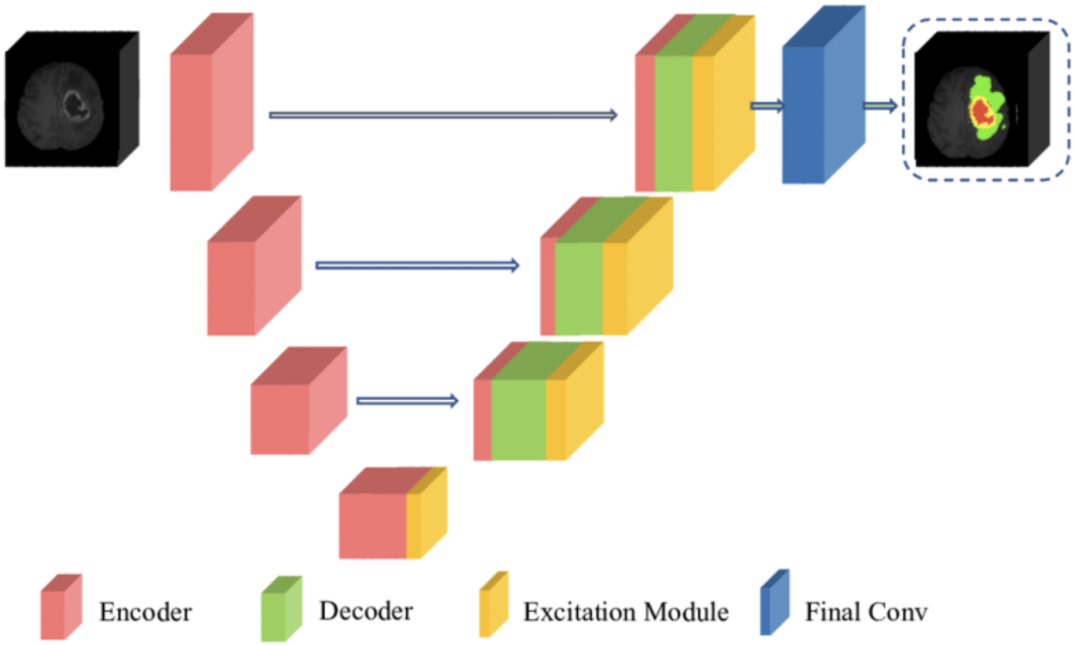

Figure 3.6: The 3D-UNet is composed of a contractive and an expanding path, that aims at building a bottleneck in its centermost part through a combination of convolution and pooling operations. After this bottleneck, the image is reconstructed through a combination of convolutions and upsampling. Taken from [17]

concept [70, 127]. This module focuses on enhancing the representation of specific modalities, such as T1, T1cE, and FLAIR, over others. By performing dynamic channel-wise feature recalibration, the excitation module allows the network to adaptively weight the importance of different modalities based on their relevance to the segmentation task [128, 129]. This mechanism improves the network's representational capacity and effectively utilizes the complementary information provided by multiple modalities[130, 70].

The final convolution layer of the network serves as the critical decision-making component. It maps the feature-rich representations from the decoder to the desired number of output classes, generating a 3D segmentation map [131, 132]. This layer typically employs a softmax activation function that is well-suited for multi-class classification problems. The softmax function converts the network output into probability distributions, assigning each input voxel to one of the predefined classes [46]. By leveraging the learned features and the spatial information preserved throughout the network, the final convolution layer determines the class membership of each voxel, ultimately producing the segmented output [124].

Pre-Processing Techniques

Using `StandardScaler`, the T2, T1ce, and FLAIR modalities are normalized per volume. The reshaping operations that flatten the image along all axes except the last one [17] display the normalization. These normalized images are stacked along a new axis to create a multi-channel image, effectively combining the information from T2, T1ce, and FLAIR modalities into one array [17]. The images are cropped from 240x240x155 to the dimensions of 128x128x128 voxels, ensuring that the volumes are divisible by 64. Cropping removes irrelevant areas from the scan and prepares the images to extract patches. The mask volumes are checked to ensure a sufficient volume of interest [17]. Volumes, where the non-zero labels (i.e., the part of the image containing the tumor or pathology) make up at least 1% of the total volume, are kept for further processing [17]. This step ensures that the model trains on images with meaningful information rather than on those with almost no relevant features.

Finally, the masks are one-hot encoded [17]. The ground truth masks will be provided in a $128 \times 128 \times 128 \times 3$ numpy array format, where the fourth dimension represents the four classes or labels corresponding to the segmentation task.

Optimiser

An Adam optimizer with an initial learning rate of $\alpha_0 = 1e{-}4$ was used and `ReduceLROnPlateau` callback was used for dynamic learning rate adjustments [113]. This callback monitors the validation loss (`monitor='val_loss'`) and reduces the learning rate by a factor of 0.2 (`factor=0.2`) if no improvement is seen for a patience of 2 epochs (`patience=2`).

The learning rate adjustment is governed by the following condition [113]:

$$\alpha = \max(\alpha_0 \times \texttt{factor}^{\left\lfloor \frac{\texttt{patience\_exceeded\_count}}{\texttt{patience}} \right\rfloor}, \texttt{min\_lr}) \tag{3.3}$$

3.6.3 Independent Variable

In this experiment, the independent variables are the loss functions used to compile the model. In this case, the loss functions are: Custom Dice Loss, Hybrid Loss 1 and Hybrid Loss 2 which are inspired by [38]

The following describes each loss function and how it is calculated:

Cross-entropy loss

We can write the cross-entropy loss for a multi-class task as [38]:

$$L_{ce\_loss} = -\frac{1}{N} \sum_{i=1}^{N} \sum_{k=0}^{K-1} g_{i,k} \log p_{i,k} \tag{3.4}$$

where $g_{i,k} \in \{0, 1\}$ and $p_{i,k} \in [0, 1]$ denote the manual annotation and the predictions of the Softmax at each pixel i for class k, N represents the number of pixels.

Dice Loss

There are many criteria for evaluating image segmentation algorithms, among which Dice Similarity Coefficient (DSC) is the most widely applies. Milletari et al. [10] propose a differentiable version of it to optimise the model. The Dice loss for each subclass can be expressed as [38]:

$$DL_k = 1 - \frac{2\sum_{i=1}^{N} p_{i,k} g_{i,k}}{\sum_{i=1}^{N} p_{i,k} + \sum_{i=1}^{N} g_{i,k}} \tag{3.5}$$

where N, $g_{i,k}$, and $p_{i,k}$ are the same as in Formula 1. So we can write the Dice loss for enhance region as:

$$DL_{\text{enh\_loss}} = 1 - \frac{2\sum_{i=1}^{N} p_{i,4} g_{i,4}}{\sum_{i=1}^{N} p_{i,4} + \sum_{i=1}^{N} g_{i,4}} \tag{3.6}$$

The Dice loss for background region is as below:

$$DL_{\text{bg\_loss}} = 1 - \frac{2\sum_{i=1}^{N} p_{i,0} g_{i,0}}{\sum_{i=1}^{N} p_{i,0} + \sum_{i=1}^{N} g_{i,0}} \tag{3.7}$$

The Dice loss for complete region can be written as:

$$DL_{\text{com\_loss}} = 1 - \frac{2\sum_{i=1}^{N} p_{i,\text{com}} g_{i,\text{com}}}{\sum_{i=1}^{N} p_{i,\text{com}} + \sum_{i=1}^{N} g_{i,\text{com}}} \tag{3.8}$$

where

$$p_{i,\text{com}} = p_{i,4} + p_{i,3} + p_{i,1} + p_{i,2},$$

$$g_{i,\text{com}} = g_{i,4} + g_{i,3} + g_{i,1} + g_{i,2},$$

denote the predicted values and manual segmentation label for complete region, respectively.

The Dice loss for tumor core region can be written as [38]:

$$DL_{\text{core\_loss}} = 1 - \frac{2\sum_{i=1}^{N} p_{i,\text{core}} g_{i,\text{core}}}{\sum_{i=1}^{N} p_{i,\text{core}} + \sum_{i=1}^{N} g_{i,\text{core}}} \tag{3.9}$$

where

$$p_{i,\text{core}} = p_{i,4} + p_{i,3} + p_{i,1}$$

$$g_{i,\text{core}} = g_{i,4} + g_{i,3} + g_{i,1}$$

denote the predicted values and manual segmentation label for tumor core region, respectively.

The combined Dice loss can be defined as [38]:

$$DL_{\text{combined\_loss}} = DL_{\text{bg\_loss}} + DL_{\text{com\_loss}} + DL_{\text{core\_loss}} + DL_{\text{enh\_loss}} \tag{3.10}$$

Categorical Focal Loss

Categorical focal loss is used [133]. It is designed to give more relative weight to minority class examples during training, helping the model focus on learning these underrepresented classes better.

$$FL(p_t) = \alpha \cdot (1 - p_t)^{\gamma} \cdot (L_{ce\_loss}, y_{\text{pred}})) \tag{3.11}$$

Recall Loss

Although model trained with unbalanced data may make high accuracy predictions, the target area may be partially detected or missing, which is very harmful in medical applications [38]. Sensitivity, therefore, can be used to address the imbalance issues, shifting emphasis to the minority.

Sensitivity is one of the highly regarded characteristics when evaluating the performance of image segmentation algorithms [38]. The recall loss for each subclass is as below:

$$RL_k = 1 - \frac{\sum_{i=1}^{N} p_{i,k} g_{i,k}}{\sum_{i=1}^{N} g_{i,k}} \tag{3.12}$$

where N, $g_{i,k}$, and $p_{i,k}$ are the same as in Formula 1. So we can write the recall loss for enhance region as [38]:

$$RL_{\text{enh\_loss}} = 1 - \frac{\sum_{i=1}^{N} p_{i,4} g_{i,4}}{\sum_{i=1}^{N} g_{i,4}} \tag{3.13}$$

The recall loss for complete region is as below [38]:

$$RL_{\text{com\_loss}} = 1 - \frac{\sum_{i=1}^{N} p_{i,\text{com}} g_{i,\text{com}}}{\sum_{i=1}^{N} g_{i,\text{com}}} \tag{3.14}$$

where

$$p_{i,\text{com}} = p_{i,4} + p_{i,3} + p_{i,1} + p_{i,2},$$

$$g_{i,\text{com}} = g_{i,4} + g_{i,3} + g_{i,1} + g_{i,2},$$

denote the predicted values and manual segmentation label for complete region, respectively.

The recall loss for tumor core region is as below [38]:

$$RL_{\text{core\_loss}} = 1 - \frac{2\sum_{i=1}^{N} p_{i,\text{core}} g_{i,\text{core}}}{\sum_{i=1}^{N} g_{i,\text{core}}} \tag{3.15}$$

where

$$p_{i,\text{core}} = p_{i,4} + p_{i,3} + p_{i,1},$$

$$g_{i,\text{core}} = g_{i,4} + g_{i,3} + g_{i,1},$$

denote the predicted values and manual segmentation label for tumor core categories, respectively.

Combined Loss

$$CL_{dc_loss} = \alpha L_{ce\_loss} + \beta RL_{com\_loss} + \gamma RL_{core\_loss} + \delta RL_{enh\_loss} \quad (13) \tag{3.16}$$

Hybrid Loss

Two hybrid loss functions were used which included recall loss, combined Dice loss and cross-entropy loss, to better balance recall and precision. The first one named HL1 can be written as [38]:

$$HL_{ce\_rl\_loss} = \alpha L_{ce\_loss} + \beta RL_{com\_loss} + \gamma RL_{core\_loss} + \delta RL_{enh\_loss} \quad (13) \tag{3.17}$$

where δ, α, β, and γ are weights of different losses.

The second one named $HL2$ can be written:

$$HL_{cb\_rl\_loss} = \alpha DL_{combined\_loss} + \beta RL_{com\_loss} + \gamma RL_{core\_loss} + \delta RL_{enh\_loss} \quad (14) \tag{3.18}$$

where δ, α, β, and γ are weights of different losses.

These algorithms were inspired by the work of [38] as they yielded some of the best results on 2D models but there was no implementation of these algorithms on 3D data.

Custom Dice Loss Function

The custom loss function is inspired by the Kaggle submission [17]. It can be denoted by:

$$\text{total\_loss} = \text{DiceLoss}(\mathbf{wt}) + \alpha \times \text{CategoricalFocalLoss}() \tag{3.19}$$

where $alpha = 1$ and $wt = [0.25, 0.25, 0.25, 0.25]$.

3.6.4 Dependent Variable

The dependent variable in this case are the IoU, Sensitivity, Specificity, and Dice Coefficients for the Edema, Enhancing, and Necrotic tumor regions based on the loss functions mentioned above.

Chapter 4

Results

4.1 2D U-Net Results

The results presented in Table 4.1 were obtained by training the model on the dataset for up to 35 epochs. The model's predictions were generated using batch implementation with a batch size of 100, and the results were evaluated after every 5 epochs to highlight the validation performance during training. The training process and the comparison between ground truth and predictions can be visualized in Figures 4.1 and 4.2, respectively. In total, the model took a total of 2 hours to finish training.

From Figure 4.1, we can observe that the model's accuracy quickly reaches a high level on the training set and remains stable, while the validation accuracy, although slightly lower, is also stable after an initial phase. This indicates good generalisation performance without significant overfitting.

epoch	val_accuracy	val_loss	val_mean_io_u	val_precision	val_sensitivity	val_specificity	Dice Coefficients		
							Edema	Enhancing	Necrotic
0	0.9858	0.0736	0.86143	0.98614	0.98569	0.99538	0.1416	0.0173	0.0198
5	0.9909	0.02749	0.39616	0.99163	0.98988	0.99719	0.5456	0.4110	0.1976
10	0.9938	0.01859	0.48583	0.99401	0.99286	0.99799	0.7180	0.5394	0.4022
15	0.994	0.01714	0.73389	0.99471	0.99365	0.99822	0.743	0.5837	0.4291
20	0.9946	0.01713	0.75532	0.99474	0.99375	0.99823	0.7478	0.5845	0.4322
25	0.9947	0.01711	0.76005	0.99476	0.99378	0.99824	0.748	0.5849	0.4326
30	0.9947	0.01706	0.76231	0.99477	0.99378	0.99824	0.748	0.5864	0.4337
34	0.9947	0.01709	0.76572	0.99478	0.99382	0.99825	0.7482	0.5860	0.4336

Table 4.1: The performance results of the 2D U-Net model up to 35 epochs on the UCSF-PDGM-v2 data per 5 epochs.

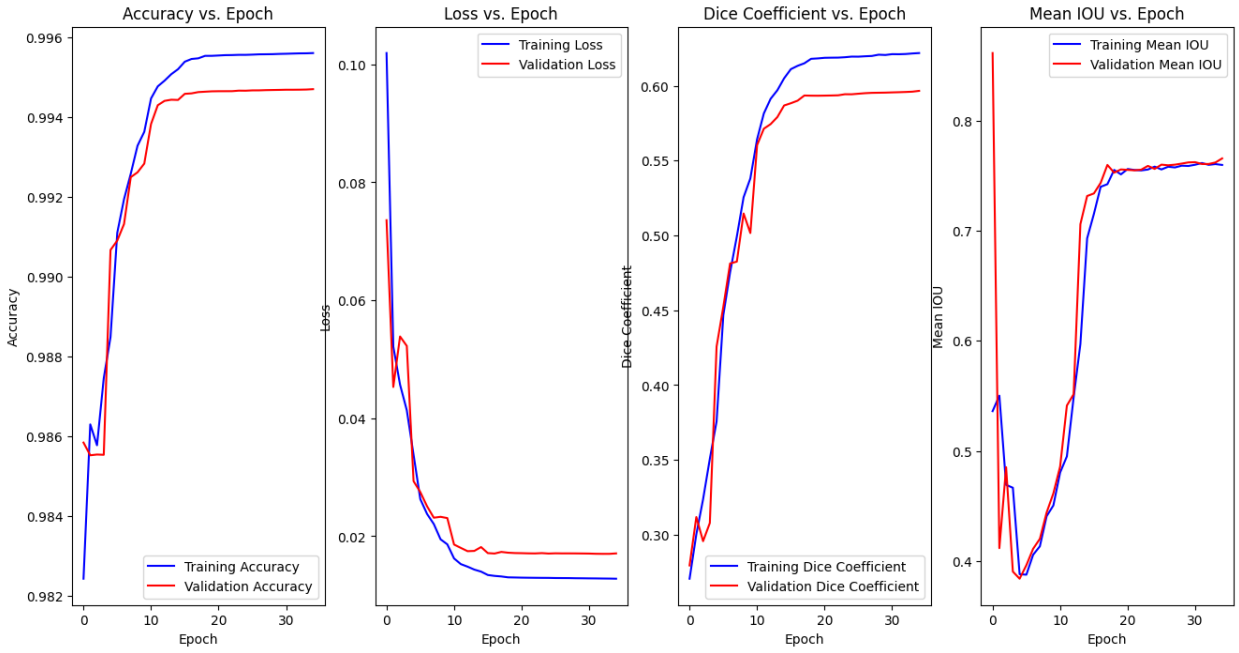

Figure 4.1: Training Process Visualisation for the 2D UNet

The loss curves for both training and validation decrease sharply and converge, which is indicative of good learning progress. Furthermore, the validation loss is slightly higher than the training loss but follows a similar trajectory, closely tracking the training loss.

Additionally, the Dice Coefficient suggest that the model is learning to segment the images well. However, there are some fluctuations in the Dice Coefficient curves, particularly in the validation curve, which suggest that the model's performance on the validation set is less stable.

This instability in the validation performance could be attributed to the variability present in the validation data or slight over fitting tendencies. However, the latter possibility seems less likely given the closeness of the training and validation accuracy curves, which indicates the model has good generalization capabilities.

When it came to the predictions on the test set, the model took 25 seconds to evaluate a batch size of 100 with a learning rate of 0.001. These results can be seen in Table 4.2 which were calculated through `model.eval()`.

Overall, the results presented in Table 4.1 and the training/validation curves in Figure 4.1

suggest that the model is learning effectively and achieving good segmentation performance, with a reasonable balance between training and validation performance, indicating successful generalisation to unseen data. This is further reinforced by the segmentation in Figure 4.2 of the EDEMA class.

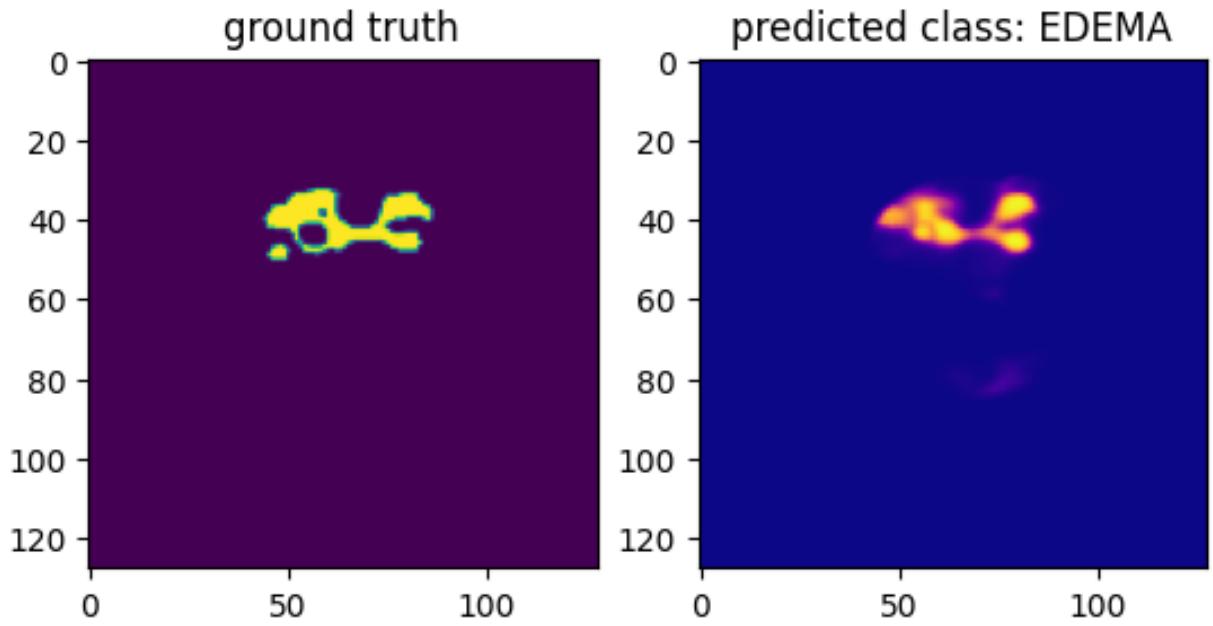

Figure 4.2: Ground Truth vs Prediction Results for the Implemented 2D Model for the EDEMA class.

No.	Metric	Result
0	Loss	0.013634
1	Accuracy	0.995221
2	Mean IoU	0.764063
3	Dice Coef	0.624600
4	Precision	0.995288
5	Sensitivity	0.994204
6	Specificity	0.998408
7	Dice Coef Necrotic	0.521996
8	Dice Coef Edema	0.750305
9	Dice Coef Enhancing	0.651414

Table 4.2: Overall metrics and results for 2D UNet after evaluation

4.2 3D U-Net Results

Metric	Value
WT dice	0.8590
WT jaccard	0.7680
TC dice	0.8570
TC jaccard	0.7640
ET dice	1.0000
ET Jaccard	1.0000

Table 4.3: Average Dice and Jaccard Coefficients for the 3D U-Net Model Implementation

The results from the training and validation processes can be seen in Table 4.3 and Figure 4.4. The model took a lengthy training time of approximately 18 hours for 50 epochs on the available data. The following results have been acquired on images of size 240x240x155.

From Figure 4.3, we can observe that the model's training accuracy is relatively stable and high, which suggests good learning on the training set. The validation accuracy also remains high and stable, indicating that the model generalizes well to unseen data. Both training and validation loss curves decrease over time, which is a desirable trend. However, the validation loss appears slightly higher than the training loss, which could indicate slight overfitting tendencies.

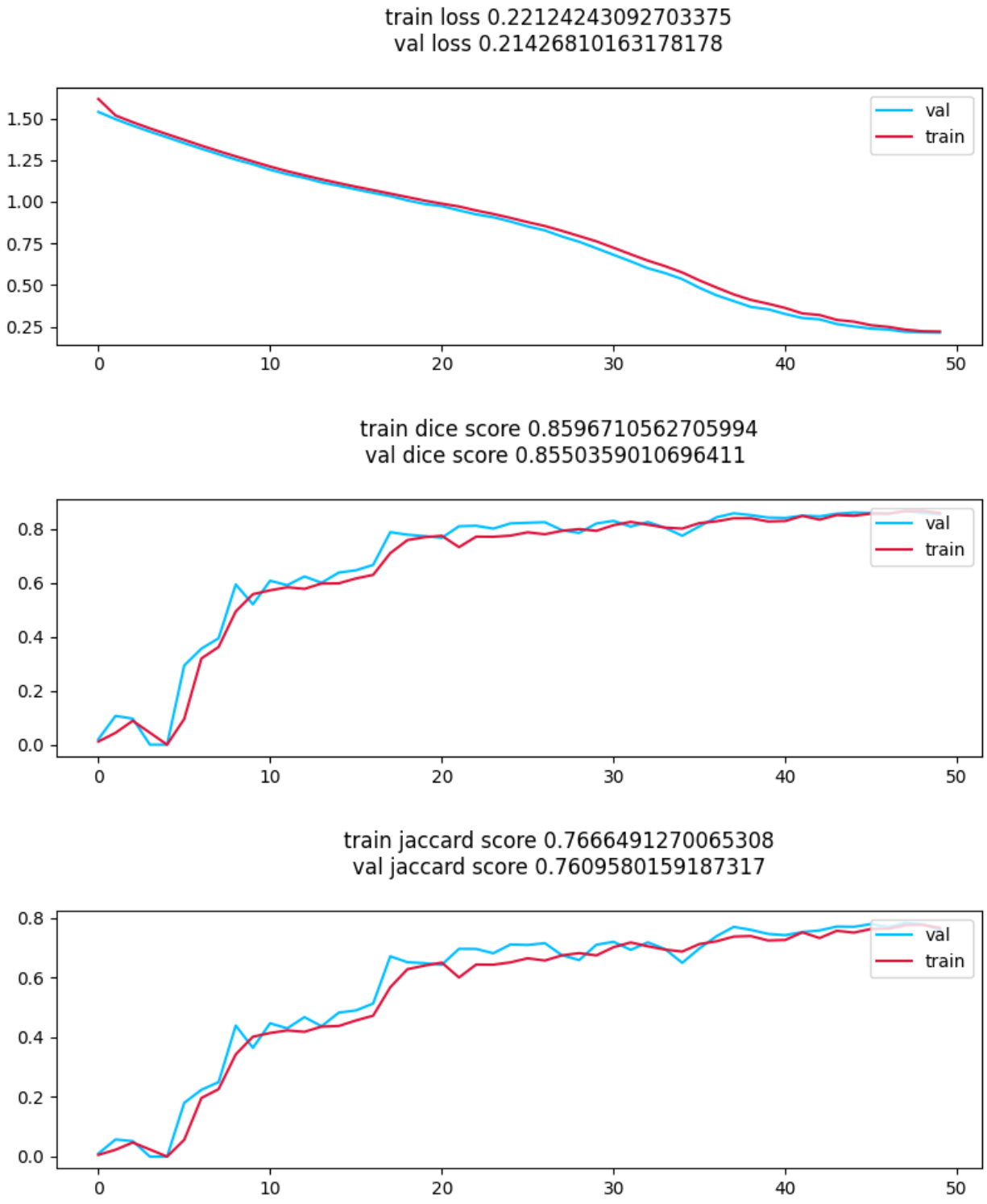

Figure 4.3: Loss functions when Training the model for 50 epochs showing that the model is learning well on the training data.

From Figure 4.4, training Dice coefficient are high, showing that the model is learning the training data well. The validation Dice coefficient exhibits some fluctuations but remains high overall, suggesting the model is good at generalising to the validation set. Similarly, the training Intersection over Union (IoU) performance stabilizes at a high value, while the validation IoU has fluctuations but remains within an acceptable range.

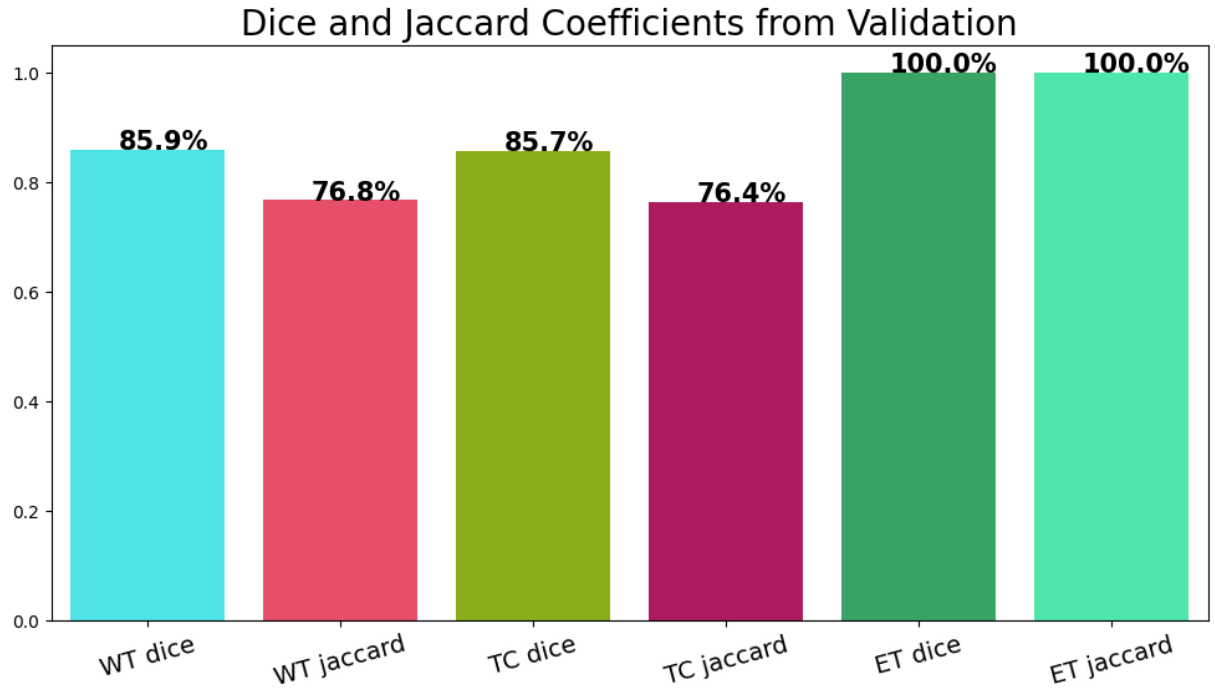

Figure 4.4: Multi-class Dice and Jacquard Coefficient scores from Validation on the UCSF-PDFM-v2 dataset

Overall, the results presented in Figures 4.4 and 4.5 suggest that the implemented model is effectively learning the training data and generalising reasonably well to the validation set, as evidenced by the high and stable performance metrics. However, there are indications of slight overfitting tendencies, such as the high Dice and IoU scores for the Expanding Tumors. Nevertheless, the model appears to strike a good balance between learning the training data and generalizing to unseen data, with room for potential improvements through further fine-tuning or regularisation techniques.

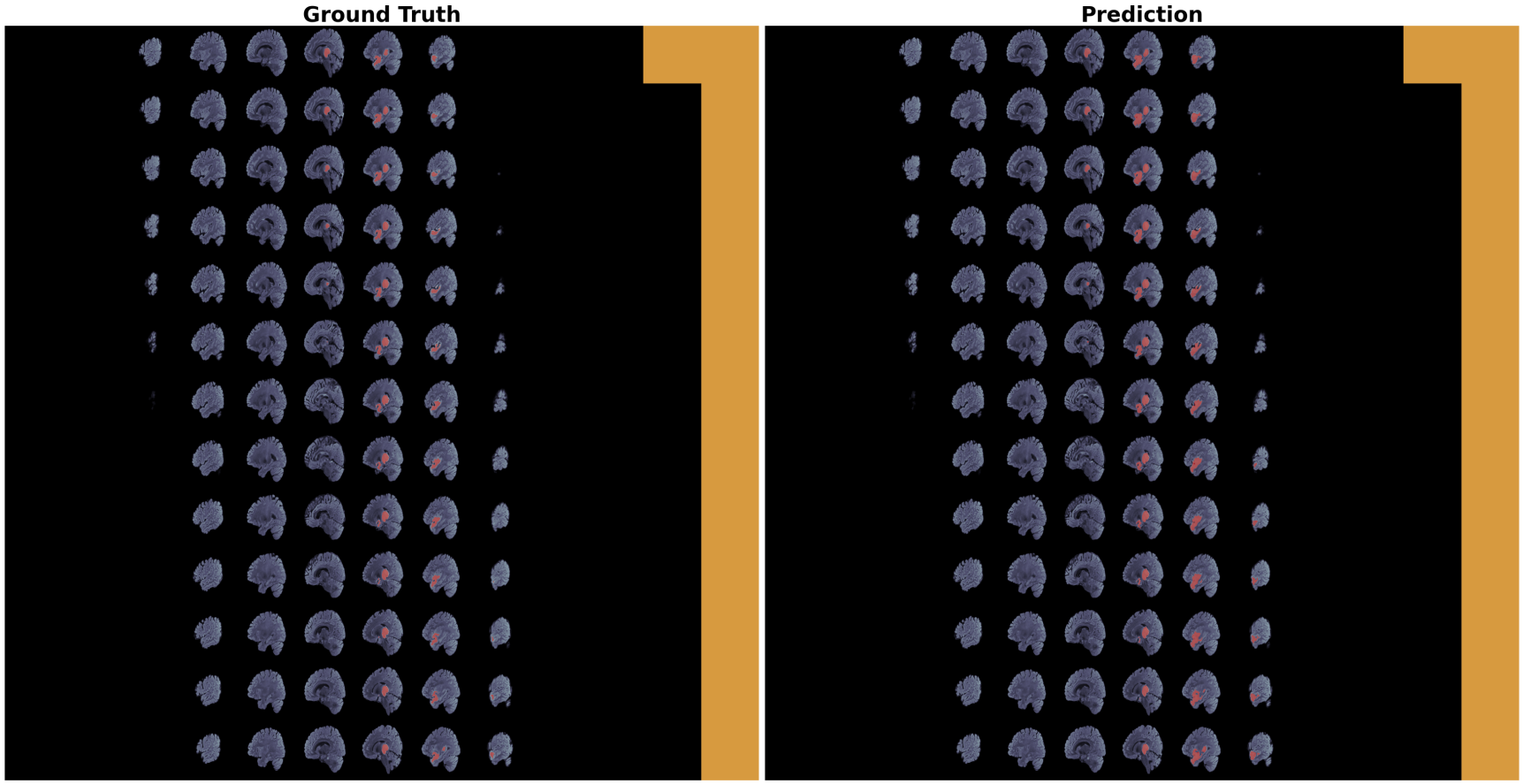

Figure 4.5: Segmentation performance of the 3D U-Net on the UCSF-PDGM-v2 dataset.

4.3 Methylation status results

From Figure 4.6, we can see that the distribution of the predicted MGMT status has a narrow concentration, with a significant number of predictions occurring between approximately 1×10^{-5} and 1.46×10^{-1}. The clustering of predictions in this range, with little to no spread beyond it, suggests a lack of variation in the predictions, which may indicate that the model is not confident in its predictions—a potential sign of underfitting. The narrowness of the prediction range might also be a result of the model not capturing the underlying complexity of the data, potentially due to the reduced image size used during training.

Modality Type	**Validation AUC Score**
FLAIR	0.5863
T1	0.5871
T1c	0.5629
T2	0.5861
all	0.5875

Table 4.4: Validation AUC results for the Transformer Implementation for the methylation status.

The score after validating the model on the test set is 0.4795. Table 4.4, shows that the validation AUC scores for various modalities and the combined 'all' category. All the AUC scores are around 0.58 which suggests moderate performance in distinguishing between the positive and negative classes. The scores are quite close to each other, indicating that no single modality outperforms the others. As discussed, in Section 1.4, if the AUC score is close to 0.5, it essentially becomes a Random Classifier which suggests the model did not do well when it came to predicting the methylation status.

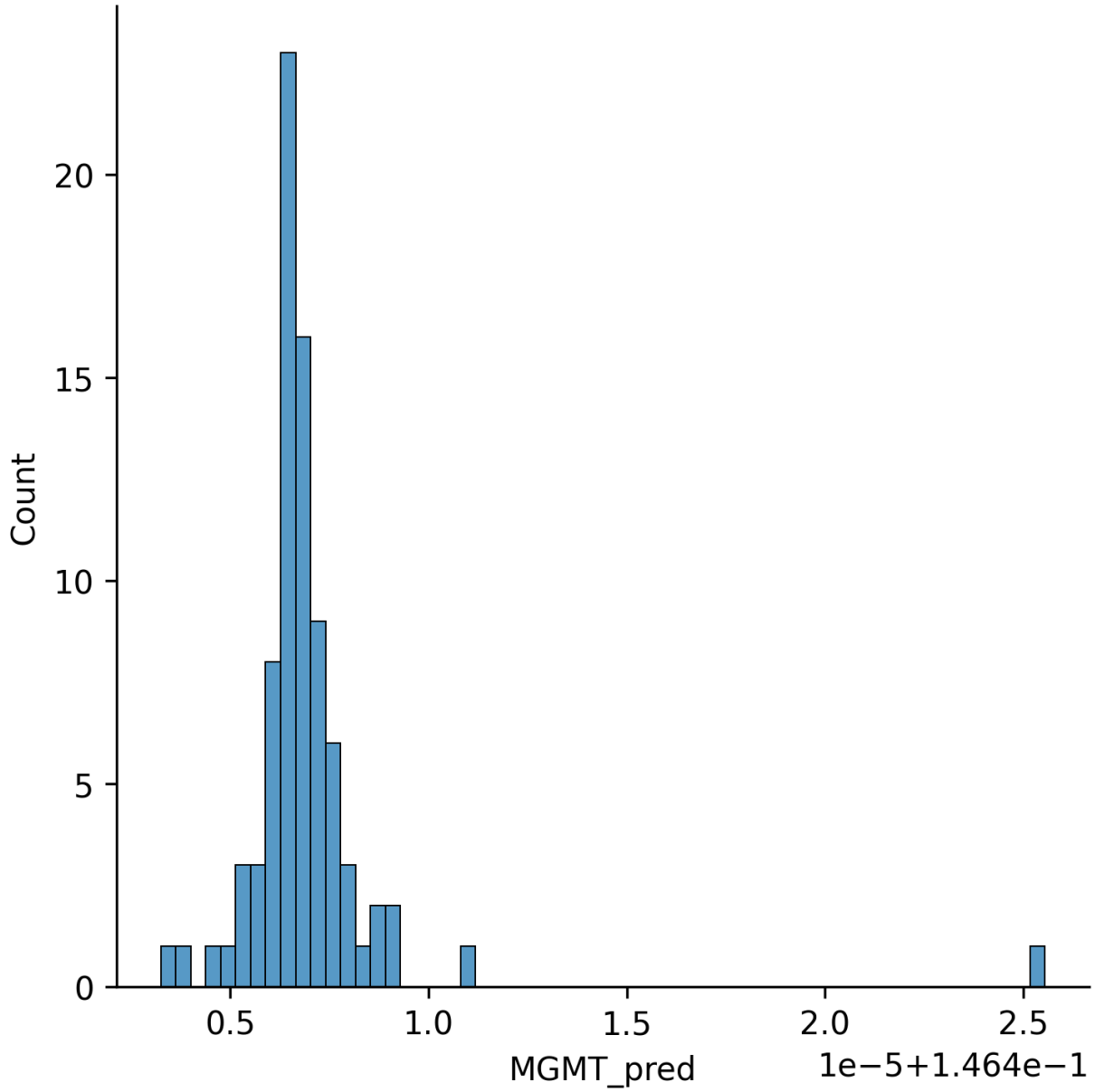

Figure 4.6: Methylation Status Prediction vs. Count on the Testing Data as Implemented on the DGX 100. The histogram displays a narrow range of predicted values for MGMT promoter methylation (MGMT_pred), predominantly clustered around 0.146 with a few outliers extending towards 0 and 2.5e-1. The concentration of predictions around a single value suggests potential issues with the model, such as overfitting to a certain characteristic of the training data or a lack of generalization to diverse cases within the testing dataset."

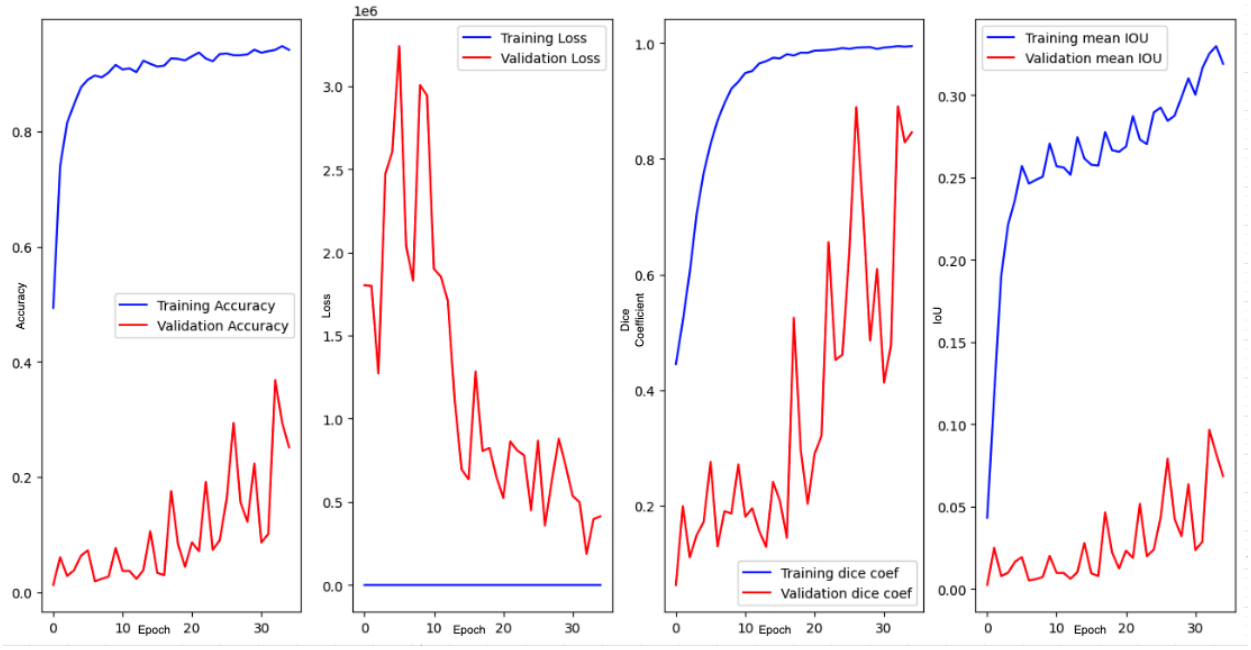

Figure 4.7: Hybrid Loss 1 Training Graph showing the discrepancy between training and validation performance.

4.4 Class Imbalance Results

Hybrid Loss 1

Analysing Figure 4.7, we can infer that while the training accuracy improves and stabilizes, the validation accuracy is very erratic and doesn't show a clear trend of improvement. This discrepancy between training and validation performance indicates that over-fitting has likely occurred during the training process.

From the training loss graph, we can see that the training loss decreases as expected, which is a good sign of learning. However, the validation loss remains very high and shows significant variance, only slowly trending downwards. This suggests that the model may not be generalizing well to unseen data.

The training mean IoU improves over time, while the validation mean IoU exhibits low values and high variance. Since the mean IoU measures the overlap between predicted and ground truth segmentation masks, its low value on the validation set suggests that the model predictions are not aligning well with the actual labels.

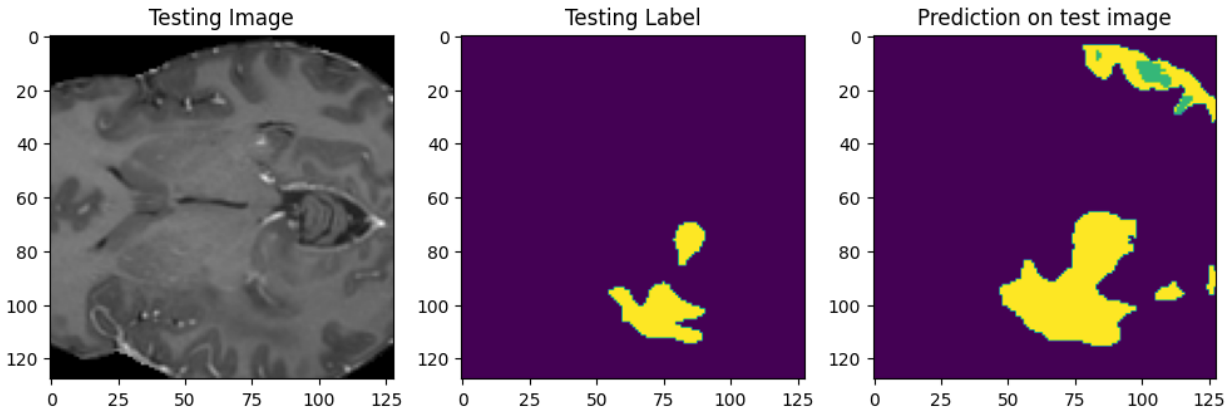

Figure 4.8: Hybrid Loss 1 Training output showing that the model is overfitting and and capturing more of the tumor area than what is present.

Similar to the mean IoU, the training dice coefficient shows an upward trend, but the validation dice coefficient remains low and erratic. This reinforces the potential overfitting issue, as the model appears to perform well on the training data but struggles to generalize to the validation set.

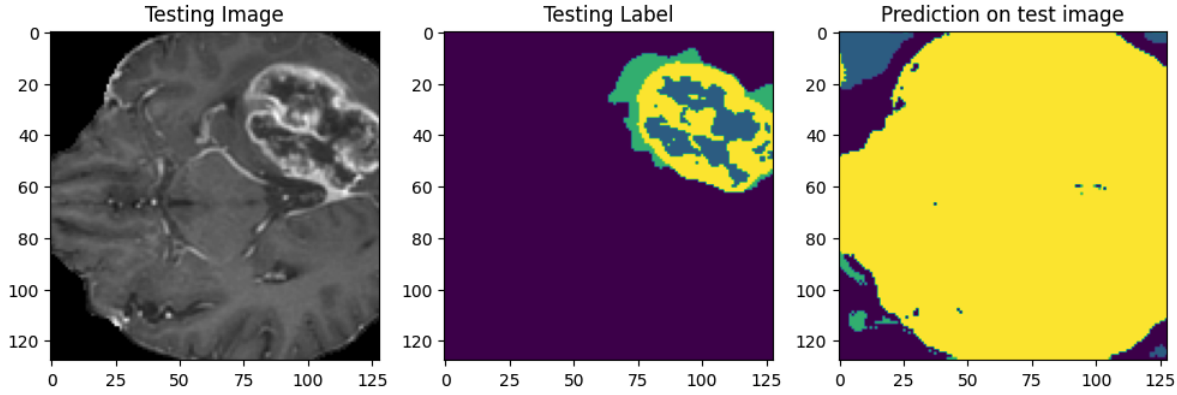

Figure 4.9: Hybrid Loss 1 Testing segmentation demonstrating the poor performance on the Testing Image

This overfitting issue is further reinforced by Figure 4.8, where the predicted segmentation captures more of the tumor area but also includes a large region not present in the ground truth label, indicating over-segmentation tendencies. Furthermore, we can see that in Figure 4.9, the model is unable to segment the tumor region and segments the entire brain instead.

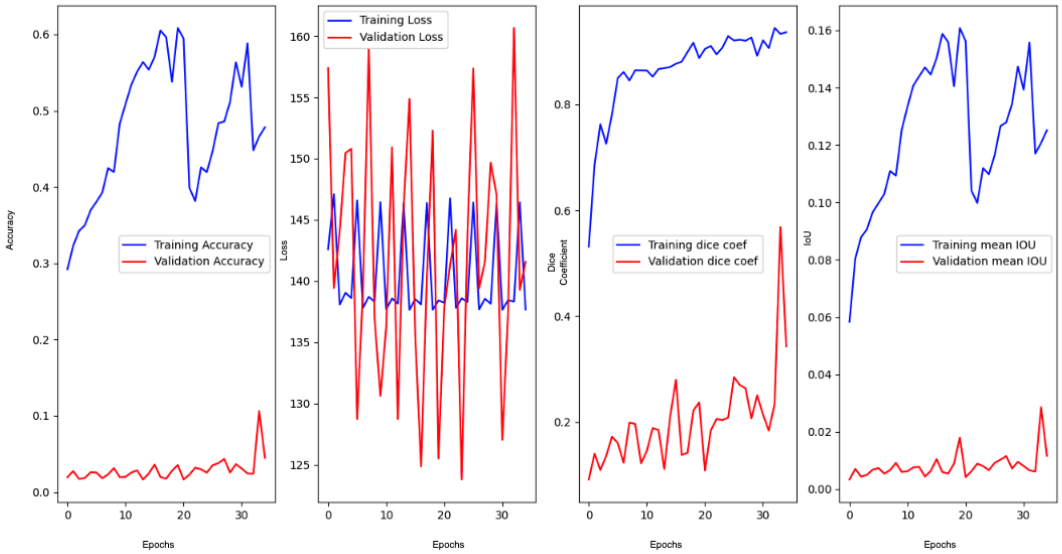

Figure 4.10: Hybrid Loss 2 Training graph pointing to potential over-fitting and a very erratic learning process

Hybrid Loss 2

From Figure 4.10, we can observe that both the training and validation accuracy are relatively low, with the training accuracy being higher than the validation accuracy. This discrepancy between training and validation performance again suggest potential over fitting, where the model performs better on the training data it has seen during training compared to the unseen validation data.

The loss graph shows very high variance for the validation loss compared to the training loss. The validation loss values are also significantly higher and more sporadic than the training loss, indicating that the model may not be generalising well to the validation data.

While the training Dice Coefficient is generally trending upwards, showing some learning progress, the validation Dice coefficient remains low and erratic. This again points to a poor match between the model's predictions and the ground truth labels in the validation set.

Similarly, the training mean IoU (Intersection over Union) is gradually increasing, which is a positive sign. However, the validation mean IoU is very low, suggesting poor segmenta-

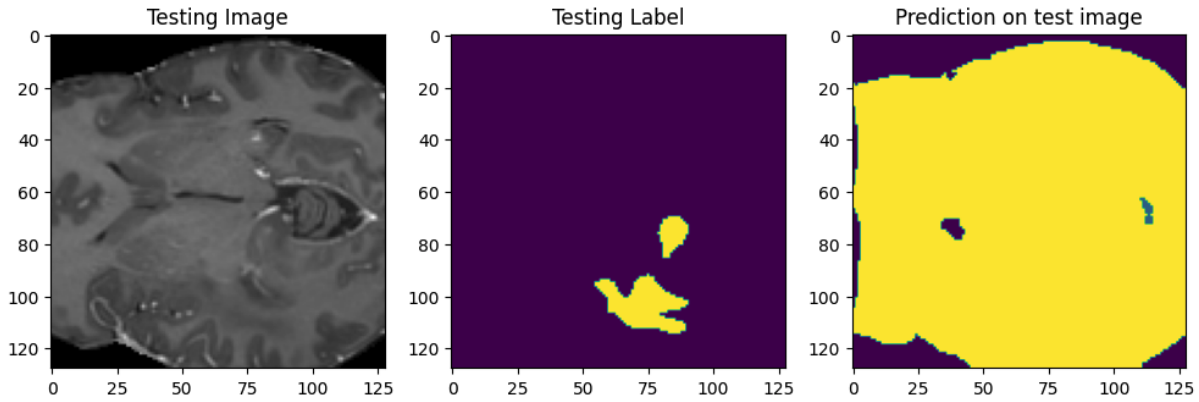

Figure 4.11: The segmentation by Hybrid Loss 2 on the training image suggests the model is unable to delienate the tumor regions and not learn any relevant patterns.

tion performance of the model on the validation set.

From Figure 4.11, we can see that the segmentation on the training image looks suboptimal. While the model captures some areas of the tumor, it fails to accurately delineate the tumor regions, suggesting that the model is struggling with effective feature extraction or has not learned the relevant patterns effectively.

This issue is further reinforced in Figure 4.12, where the segmentation on the test image is highly inaccurate. Most of the tumor region is either not detected or incorrectly marked, reinforcing the issues highlighted by the poor validation metrics observed in Figure 4.10.

In summary, the analysis of Figure 4.10 and the segmentation examples in Figures 4.11 and 4.12 strongly suggest that the model is suffering from over fitting, leading to poor generalisation performance on unseen data. The model appears to struggle with accurately segmenting tumor regions, potentially due to ineffective feature extraction or an inability to learn the relevant patterns effectively during training.

Custom Dice Loss

From Figure 4.13 we can see that the training accuracy is high and stable, while validation accuracy is highly variable and generally lower. Once again reinforcing the general problem of over fitting. The training loss shows a downward trend, which is good. However, the

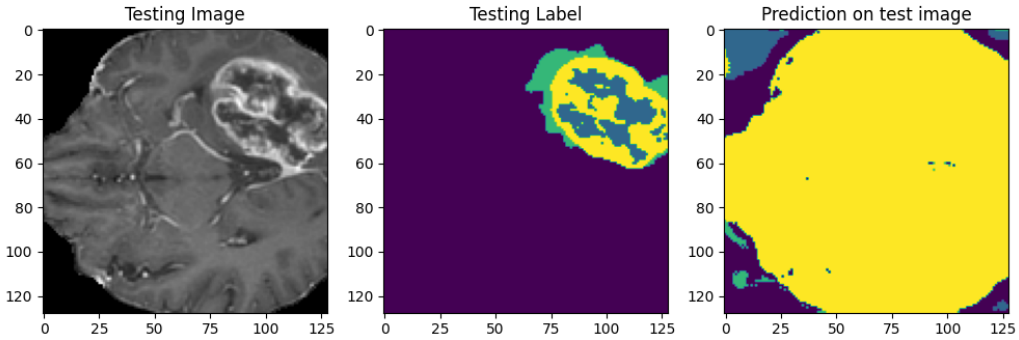

Figure 4.12: The segmentation performance by Hybrid Loss 2 on a Test Image reinforces the belief that the model suffers from over-fitting which has led to prove it's ineffectiveness.

validation loss is quite erratic and does not show a consistent decrease, suggesting the model may not be generalizing well. The Dice coefficient for the training data shows a good upwards trend, while the validation Dice coefficient is again erratic and generally lower, reinforcing the overfitting issue. The training mean IoU is trending upwards, while the validation means IoU starts low and has high variability. A higher training mean IOU indicates that the model is learning to segment the training data effectively, but the lower and variable validation mean IOU indicates the model may not be performing as well on unseen data.

From Figure 4.11, the models seems to be performing reasonably well on the training image, closely matching the test label. This is consistent with the high training accuracy and Dice coefficient. On the test image in Figure 4.15, the prediction not as precise as the training image as the model segments the entire brain rather than the tumor region. Some of the tumor regions are missed, and some false positives are present, which correlates with the validation metrics showing lower performance.

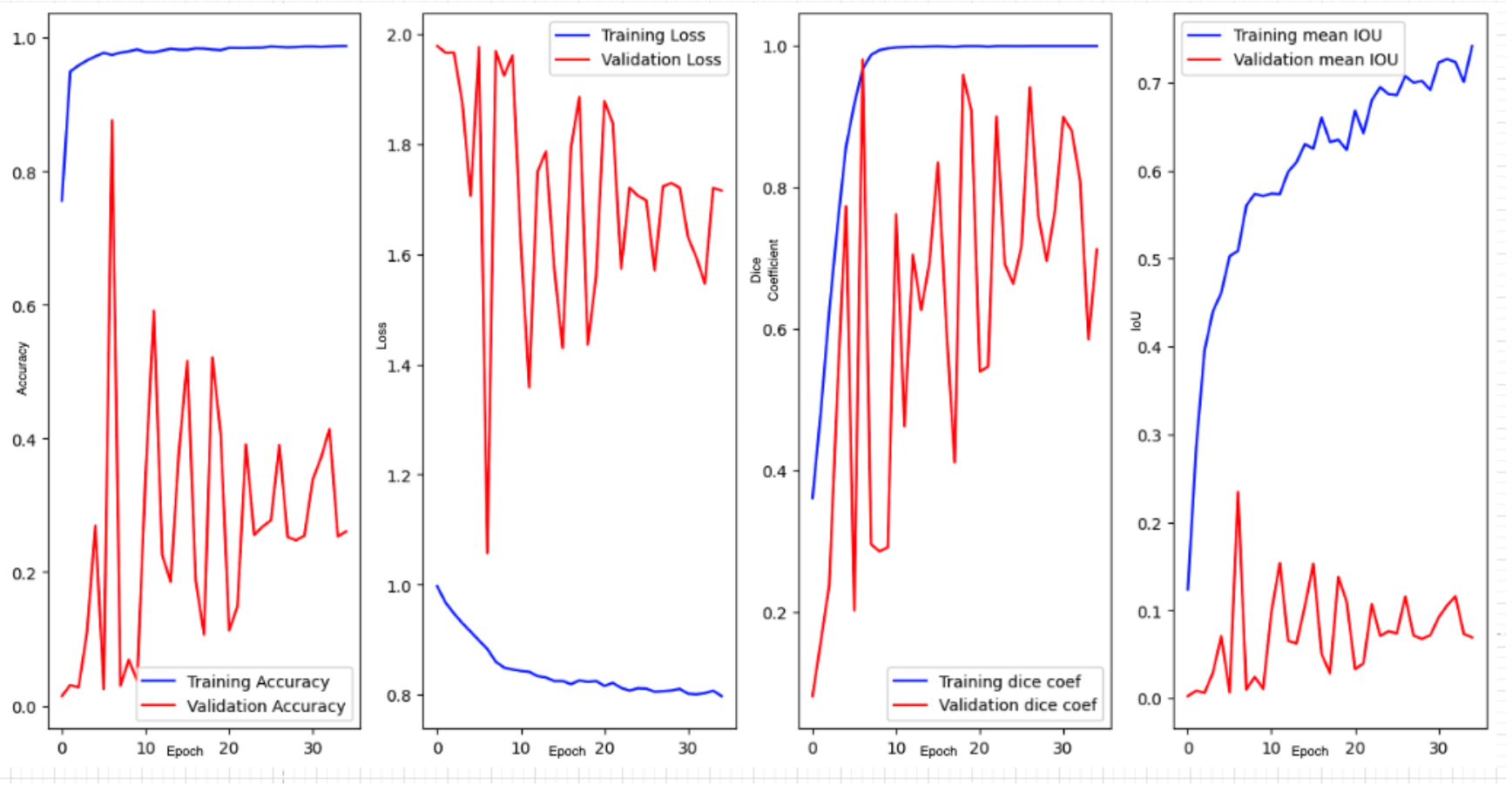

Figure 4.13: Custom Dice Loss Training Graph. The variability and erratic behaviour of the validation loss, Dice Coefficient, and the IoU score reinforce the idea of overfitting.

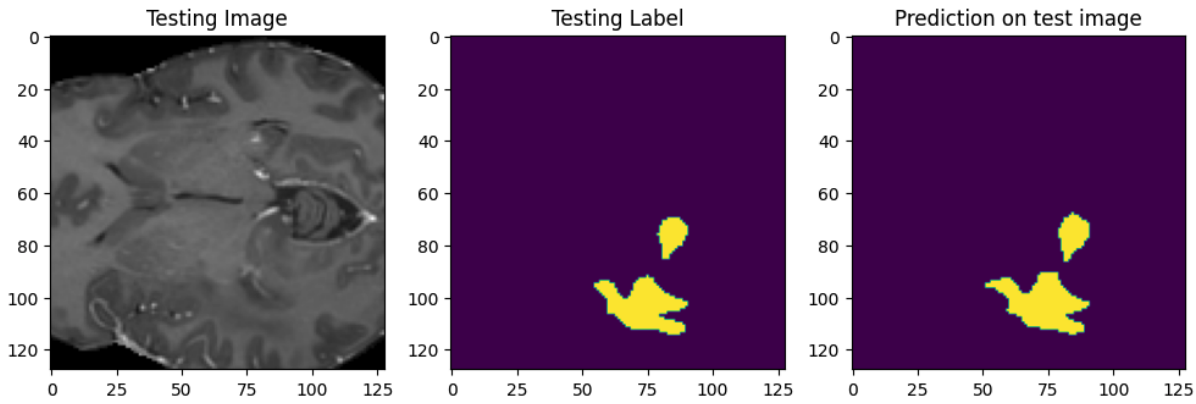

Figure 4.14: A very accurate segmentation performed by the Custom Dice Loss on a Training Image

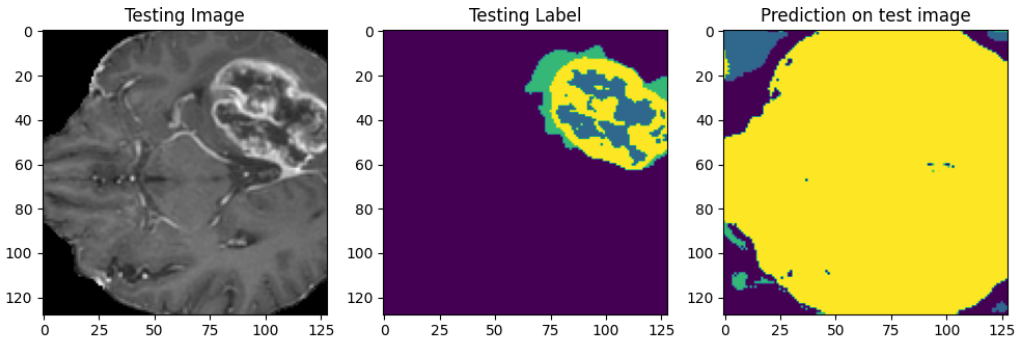

Figure 4.15: The Custom Dice Loss model was unable to accurately segment a test Image, suggesting overfitting.

Chapter 5

Evaluation

5.1 Critical Evaluation of all the models

2D U-Net

Table 5.1 compares different models for brain tumor segmentation using 2D architectures. While the implemented 2D U-Net achieves Dice scores of 0.6204, 0.5868, and 0.5220 for the whole tumor (WT), tumor core (TC), and enhancing tumor (ET) regions, respectively, it still lags behind the Multipath architectures proposed by Kleesiek et al. [67] and Zikic et al. [60].

The performance of the implemented 2D U-Net model can be attributed to several factors despite its lower scores compared to the work of Zikic et al. [60]. The model's nine-layer architecture is more complex than Zikic's, which has only five layers. This increased depth enables the 2D U-Net to capture more intricate features and patterns from the data, potentially enhancing its representational power. However, it is essential to note that the 2D U-Net differs from other models as it only captures data for Edema, Tumor Core, and Enhancing regions.

The 2D U-Net achieved Dice scores of 0.52, 0.75, and 0.65 for the Necrotic, Edema, and Enhancing classes, respectively, as shown in Table 5.1. These scores were calculated using the

formulas presented in Section 2.2 [19]. It is crucial to acknowledge that these results cannot be directly compared to the performance of models implemented on the UCSF-PDGM-v2 dataset, as currently, no available models have been applied to this specific dataset. The lack of comparable models on the same dataset limits the ability to make direct performance comparisons and draw definitive conclusions about the 2D U-Net's effectiveness compared to other approaches.

Secondly, the implemented model is trained on a larger dataset of over 350 images, while Zikic's model is trained on only 20 high-grade cases [60]. Training on a more diverse and extensive dataset can help the model generalize better and capture a broader range of tumor variations. However, it is essential to note that the quality and consistency of the dataset also play a crucial role in model performance. Zikic et al. experimented on the [60] BraTS 2016 dataset, and post-2016, this dataset was no longer used (as seen in Section 2.2 [19]). Furthermore, the implemented model undergoes optimization techniques, contributing to its performance. Despite these advantages, the implemented model's performance is still suboptimal compared to the Multipath architecture proposed by Kleesiek et al. [67]. One reason for this difference could be the additional pre-processing steps employed by Kleesiek et al. [67], such as using a 3D slicer for image normalization. In contrast, the implemented model relies on manual regularization and pre-processing techniques.

Several strategies could be explored to improve the performance of the implemented model. Adopting the two-step normalization and augmentation methods proposed by Kleesiek et al., [67] could help standardize the input data and increase the model's robustness. Additionally, incorporating post-processing techniques like edge-preserving filters and graph-cut optimization [67] could refine the segmentation results and improve the overall performance.

In conclusion, the implemented 2D U-Net model's performance can be attributed to its increased depth, larger training dataset, and optimization techniques. However, there is room for improvement by incorporating advanced pre-processing and post-processing methods and exploring alternative architectures, like multipath architectures and training strategies.

Model	Dimensions	Loss	Dice		
			WT	**TC**	**ET**
Multipath Architecture [67]	2D	Dice	0.87	0.76	0.74
Multipath Architecture [60]	2D	Log Loss	0.84	0.73	0.69
Implemented 2D U-Net	2D	Categorical CE	**0.6204**	**0.5868**	**0.5220**

Table 5.1: Comparison of different models for brain tumor segmentation with 2D models.

3D U-Net

From Table 5.2, we can see that the 3D Implementation for the UCSF-PDGM-v2 dataset suffers from the same problem as the 2D model, which is that there are not that many comparable models implemented on the dataset. But in comparison, to the other 3D U-Net models, the implemented model has performed equally well and with similar competitive results to the other 3D models in Table 2.2. This can be attributed to similar pre-processing techniques like group normalisation and dropout layers like the ones present in [13, 21]. The use of the loss function Binary Cross Entropy and Dice Loss [22] helped the model perform better than models that are just based on Dice Loss as they give more relative weight to minority classes during training and can help the model focus on learning these underrepresented classes better [133]. This is visible through Figure 4.3 which reinforces the stability of the model's training accuracy on the training set and loss curves decreasing over time.

But from Table 5.2, we can see that the model is overfit when it comes to the Enhancing Tumor (ET) class. This maybe attributed to the model being trained on only 216 (43% images) and validated against only 71 images (14% of all images). This has occured because the models were split on the basis of the BraTS 21 Segmentation Cohort column in the UCSF-PDGM-metadata-v2.csv that came with the dataset [1]. Training and validating the data against a higher number of images could improve the accuracy. This is seen through the CKD-TransBTS model [14] which is trained on 834 images, tested validated against 208, and tested against 209 images. Training on more images could help push the implemented 3D U-Net closer to the Transformer implementations as the model is currently performing better

in terms of WT against the TransUNet [66] (0.86 vs 0.82). Potentially using SpatialDropout layers on the model could help the model perform as good as the Autoencoder implemented by [21] as currently the model is just 0.01 off in terms of Dice Coefficient and 0.05 in terms of WT.

Overall, this was a very successful implementation that yielded near competitive results with the literature review.

Model	**Dimensions**	**Loss**	**Dice**		
			WT	**TC**	**ET**
Autoencoder [21]	3D	Dice+L2+KL	0.91	0.87	0.82
2 Stage Cascaded UNet [13]	3D	Dice	0.91	0.86	0.8
Autoencoder [65]	3D	Categorical CE	0.9	0.75	0.71
V-Net [58]	3D	CE	0.91	0.83	0.76
CKD-TransBTS [14]	3D	Dice loss	0.89	0.90	0.93
TransUNet [66]	3D	Dice loss	0.82	0.88	0.92
Implemented 3D U-Net	3D	Binary Cross Entropy+ Dice Loss	**0.86**	**0.86**	**1.00**

Table 5.2: Comparison of different models for brain tumor segmentation with implemented 3D models and their loss functions.

5.2 Methylation Status Evaluation

The implemented Transformer's (using FLAIR modality) score of **0.4795** on the test set is significantly lower than the validation AUC scores mentioned in the first table. This substantial discrepancy suggests that the model appears to struggle with the reduced image dimensions, resulting in its inability to capture essential features [134]. The model's subpar performance on the test set raises serious concerns about its generalization capability and practical utility.

Figure 4.6 reveals a tight distribution of the predictions, which, coupled with the low test AUC score, strongly indicates that the model needs to learn more variability from the training data [135]. The consistency of the AUC scores across different modalities in the validation set and their notable drop in the test set further reinforces the notion that the model's performance is severely hindered by the reduced image dimensions.

To address these shortcomings, a potential improvement would be to use full image sizes,rather than 128x128x128 volumes, enabling the model to extract more discriminative features. Additionally, when loading data in the `Dataset` class, incorporating all available modalities, rather than just FLAIR images, could provide a more comprehensive view of the model's performance [136]. Furthermore, using a dataset with binary values of 0 (unmethylated) and 1 (methylated), instead of the three values present in the UCSF-PDGM dataset, may simplify the learning process and improve the model's ability to distinguish between the two classes.

It is crucial to acknowledge that the downsizing of the training data, compared to the 1010 images used by all the models in Table 5.3, has likely contributed to the lower score [15]. This highlights the importance of utilizing a sufficiently large and representative dataset to ensure the model's robustness and generalization capability.

The implemented model closely resembles the one developed by Phan et al. [15]. To further enhance the results, instead of stacking all the volumes together using patch embeddings, concatenating the images into a unified 4-channel image may prove beneficial. Moreover, employing a more robust validation strategy could bolster the model's reliability and help mitigate overfitting.

Furthermore, deploying advanced pre-processing techniques, such as CentreCropping and noise adjustment on the 3D data, can potentially elevate the scores and improve the model's robustness [137]. However, it is crucial to recognize that, as highlighted in the literature review, a transformer trained solely on methylation status may not be sufficient to significantly improve the model's performance .

The results of the implemented tarsnformer (0.4795) align with the observations made by Minh Panh [15], suggesting that the current performance is not clinically useful. If the Validation AUC score is close to 0.5061, the model essentially behaves as a random classifier, merely guessing the methylation status without any meaningful understanding of the underlying patterns [5].

Model	Score
3D CNN, EfficientNet, ResNet10 [75]	0.62174
EfficientNet B0 with 2 LTSM layers [15]	0.61881
EfficientNet-B3 [77]	0.61732
YOLOv5 and EfficientNet [76]	0.61562
Unet++ with Densenet121 and LTSM [79]	0.60696
UNet [80]	0.60394
Implemented Transformer (FLAIR modality)	**0.4795**

Table 5.3: Comparison of the Implemented Transformer and the AUC score for it show that there is a big gap between the literature and the transformer. This is due to the reduced training size and reduced dimensions.

5.3 Class Imbalance Experiment Evaluation

5.3.1 Flaws in the Experimental Procedure

The strategy used when training the model involved training on the train dataset, validating against the train dataset, and finally testing on the test set. This approach led to the misleading results shown in Table 5.4, where the implemented model for Hybrid Loss 1 appears to surpass all other models in terms of Dice coefficients. Applying the formulas from Section 2.2 to calculate the Dice coefficients for each region yields seemingly impressive results: Whole Tumor (WT) is 0.905, Tumor Core (TC) is 0.910, and Enhancing Tumor (ET) is 0.9054. These values exceed those of any 3D model listed in Table 5.2, ostensibly indicating superior performance. However, a deeper analysis into the segmentation performances in into further evaluation reveals that these results are highly misleading and should be cautiously approached. This is through Figures 4.8 and 4.9 which show the performance of the model in segmenting the test and training images.

The experiments faced significant hurdles due to the misapplication of equations described in Section 3.6.3, specifically the loss functions of Hybrid Loss 1, Hybrid Loss 2, and Custom Dice Loss, which are designed for 2D U-Net models, to 3D U-Net architectures.

This incorrect implementation of loss functions resulted in subpar outcomes and compromised the overall effectiveness of the experiments. Furthermore, the individual loss functions that make up the loss functions are implemented for entire tumor regions like (Whole Tumor, Tumor Core, and Enhancing Tumor) rather than labels that make up these tumor regions like Necrotic, Edema, and Enhancing. Looking at Table 5.2, a good Dice score would categorize anything above 0.75 for the labeled regions as satisfactory. While models like Hybrid Loss 1 achieve results upwards of 0.89 for each labeled region, the results are flawed because of the poor segmentation performance, which can be seen in the corresponding figures (not provided in the given text). Even the Custom Dice Loss model, which incorporated the Dice Loss and Categorical Focal Loss methods from the segmentation_models_3D package [104], failed to yield satisfactory results. The implementations highlight the critical importance of selecting loss functions tailored to the specific architecture.

For the first hybrid loss function (HL1), the parameters ($\alpha = 20$, $\beta = 0.5$, $\gamma = 1$, and $\delta = 0.5$ were chosen based on their optimal performance in the work of Huang et al. [38]. The recall loss component in HL1 aimed to enhance the network's Sensitivity, reflected in the higher sensitivity scores. However, this improvement came at the notable expense of low Intersection over Union (IoU) scores, indicating a trade-off between Sensitivity and overall segmentation accuracy. Intriguingly, the Dice scores for all classes increased, with high Dice coefficients for specific tumor regions suggesting a close alignment between the model predictions and ground truth in those areas. The Dice coefficients of 0.3827, 0.3740, and 0.3611 from Table 5.4 for HL2 are lower than the results in Table 5.2, indicating that HL2 performs worse than the models in the literature that have scores upwards of 0.70 for all coefficients.

The IoU scores in Table 5.6 tell a starkly different story, suggesting a very poor overlap between the predicted segments and the ground truth segmentation. This discrepancy underscores the crucial need to consider multiple evaluation metrics when assessing the actual quality of segmentation outputs [138, 5]. Moreover, the low Precision of **0.2616** for HL1

implies that a significant proportion of the model's optimistic predictions are false positives, indicating a high misclassification rate. Similarly, the Sensitivity of 0.2616 suggests that the model fails to identify many positive cases when classifying the different classes.

The assignment of parameters $\alpha = 1$, $\beta = 0.1$, $\gamma = 1.3$, and $\delta = 0.5$ for the second hybrid loss function (HL2) yielded the worst results among the experiments. HL2 exhibited the lowest precision and sensitivity values, although the dice scores for the different classes were fairly close to each other. The precision and sensitivity scores suggest that the model struggled to accurately identify and delineate the tumor regions. Changing the parameter values could potentially improve the learning performance of this loss function[139]. In the Critical Appraisal section of the report, the optimisation of the parameters to better improve the scores will be discussed.

The mean IoU of **0.088** for the Custom Dice Loss model is low but higher than the mean IoU for Hybrid Loss 2 (**0.0146**), indicating a slightly better overlap between predicted segments and ground truth. However, the Precision of **0.2835** remains low, suggesting that many optimistic predictions are false positives. The sensitivity score is the same, indicating that many positives are not detected. The specificity of **0.7611** is relatively high, suggesting better performance at identifying true negatives. The Dice Coefficients (0.843, 0.8324, 0.8065 for Necrotic, Edema, and Enhancing, respectively) are relatively high, especially compared to Hybrid Loss 2 (0.3827, 0.3740, 0.3611 for Necrotic, Edema, and Enhancing respectively), which suggests the model is performing well for these regions.

The custom Dice loss function balances Sensitivity and specificity, resulting in a decent performance on training and test data. While it does not reach the performance levels of Hybrid Loss 1, it represents an improvement over Hybrid Loss 2.

HL2 demonstrated the worst learning out of the three experiments, as evidenced by the extremely low Mean IoU of **0.0146**. The IoU score suggests almost no overlap between the predicted segments and the ground truth segmentation masks. The Precision of **0.0552** is very low, indicating that very few of the optimistic predictions made by the model are

true positives. Similarly, the low sensitivity value means the model fails to identify most positive cases correctly. While the specificity of **0.6854** is moderate, indicating an acceptable performance at identifying true negatives, this metric is less relevant in segmentation tasks, which primarily focus on positive detection and delineation [140].

5.3.2 Critical Appraisal

	Mean IoU	Precision	Sensitivity	Specificity	Dice Coefficients for Tumor Regions		
					Necrotic	Edema	Enhancing
Hybrid Loss 1	0.0787	0.2616	0.2616	0.7539	**0.9139**	**0.8957**	**0.9054**
Hybrid Loss 2	0.0146	0.0552	0.0552	0.6854	0.3827	0.3740	0.3611
Custom Dice Loss	0.088	0.2835	0.2835	0.7611	0.843	0.8324	0.8065

Table 5.4: Full performance Metrics Comparison for the three Loss Function experiments

The implemented 3D model's performance, while showing promise in certain aspects, needs to catch up to the state-of-the-art models discussed in the literature review. The Dice Coefficients achieved by the model, particularly with the Hybrid Loss 1 function, are relatively high and state of the art for the tumor regions (**0.9139, 0.8957, and 0.9054 for Necrotic, Edema, and Enhancing regions, respectively**). However, these scores alone do not provide a comprehensive picture of the model's performance.

When examining other metrics, such as Mean IoU, Precision, and Sensitivity, the model's limitations become more apparent. The low Mean IoU scores, ranging from **0.0146 to 0.088**, indicate a significant discrepancy between the predicted segments and the ground truth segmentation [5]. This suggests that the model struggles to accurately delineate the tumor regions despite the high Dice Coefficients. The Precision and Sensitivity scores further reinforce this observation, with values as low as **0.0552 and 0.2835**, respectively, implying a high rate of false positives and a limited ability to identify positive cases accurately [5].

In contrast, the models discussed in the literature review, such as the Autoencoder by Myronenko [21] and the CKD-TransBTS by [14], demonstrate superior performance across

multiple metrics. Myronenko's Autoencoder achieves Dice Coefficients of **0.91, 0.87, and 0.82** for the Whole Tumor (WT), Tumor Core (TC), and Enhancing Tumor (ET) regions, respectively, while the CKD-TransBTS model attains similarly high scores of **0.8850, 0.9016, and 0.9333** for the same regions. The design of these models makes them fine-tuned to address the challenges specific to brain tumor segmentation by incorporating techniques such as spatial dropout, regularization, and advanced augmentation methods. For example, Myrenko [21] has a loss function that combines Dice Loss, Mean Squared Error, and KL Divergence to optimize learning and generalizing. In Myrenko's [21] model, KL Divergence regularizes the encoder by comparing the distribution of the encoded data to a prior distribution, typically a Gaussian [141]. This term encourages the encoder to produce latent vectors that follow a known distribution, which can improve the robustness and generalization of the model, and L2 Loss penalizes the differences between the reconstructed outputs of the autoencoder branch against the input image [142].

The performance gap can be bridged between the implemented 3D model and the state-of-the-art-models through the following ways:

1. **Architecture Enhancements**: The use of 3D convolutions to extract hierarchical features, dropout layers for regularisation, and the preservation of spatial details has its limitations, which could be addressed by incorporating features of the CKD-TransBTS model [14]. The implemented 3D U-Net model is very reliant on a single encoder-decoder path. While this architecture has proven effective in capturing and processing features at various scales, it may not be optimal for capturing long-range dependencies and global context [115]. Using a hybrid encoder to combine convolutional layers with self-attention mechanisms allows the model to capture both local and global dependencies effectively [14]. Adding the hybrid encoder could improve the implemented model's ability to capture relationships between different regions of the input data and enhance segmentation accuracy. Another feature that can improve the segmentation accuracy is the feature calibration decoder, as it helps to bridge the

semantic gaps between the reshaped vectors by reshaping them into a uniform shape, concatenating them, and compressing the channels using a 1x1x1 convolutional layer. By integrating a hybrid encoder, convolutional stem, feature calibration decoder, and advanced augmentation techniques [14], the implemented model could improve its ability to capture long-range dependencies, handle multi-modal data, align features from different paths, and enhance its robustness and generalization.

2. **Optimising Computational Efficiency**: The computational efficiency of the implemented 3D model can be significantly improved by adopting a cascaded architecture similar to the two-stage cascaded U-Net proposed by Jiang et al. [13]. In their approach, the first stage, U-Net, generates an initial, coarse segmentation map, which is then refined by the second stage, U-Net. This cascaded design allows for more efficient utilization of computational resources by focusing on progressively refining the segmentation rather than processing the entire input simultaneously. Moreover, the two-stage cascaded U-Net employs a patch-based processing approach to manage memory constraints. By processing smaller subsets of the input data, the model can effectively handle the high memory requirements associated with 3D volumetric data. Implementing a similar patch-based processing strategy in the current 3D model could significantly improve its computational efficiency and make training on available hardware resources more feasible. The effectiveness of the two-stage cascaded U-Net is evident from its impressive performance on the BraTS 2019 dataset. As reported by Jiang et al. [13], their model achieved Dice scores of **0.91 for the whole tumor (WT), 0.86 for the tumor core (TC), and 0.8 for the enhancing tumor (ET) regions**. These results demonstrate the model's ability to accurately segment brain tumors while maintaining computational efficiency through its cascaded architecture and patch-based processing. Adapting the key features of the two-stage cascaded U-Net, such as the cascaded refinement strategy and patch-based processing, to the implemented 3D model holds great promise for enhancing its computational efficiency

and segmentation performance. The model can achieve similar or even better results by leveraging these techniques while effectively managing memory constraints and computational resources.

3. **Hyperparameter Tuning**: To find the best parameters for the hybrid loss function, a systematic approach such as grid search or random search can be employed [143]. Grid search involves defining a range of values for each parameter and exhaustively evaluating the model's performance for every possible combination of parameter values. This approach can be computationally expensive, especially when dealing with many parameters like the Hybrid Loss and Custom Dice Loss models [144]. Alternatively, random search can be used to explore the parameter space more efficiently. In random search, parameter values are sampled randomly from a defined distribution, and the model's performance is evaluated for each sampled set of parameters. This method is more effective than grid search in high-dimensional spaces and can often find better parameter configurations in less time [143].

5.3.3 Key learnings from the experiments regardless of the results not working out

The findings from the experiments (Figures 4.7, 4.8, 4.9, 4.10, 4.11, 4.12, 4.13, 4.14, 4.15, and Table 5.4) highlight the critical necessity of comprehensive evaluation for segmentation models, going beyond single metrics like Dice coefficients. The discrepancies between the Dice scores and other metrics, such as IoU, Precision, and Sensitivity, raise concerns about the reliability and generalizability of the model's predictions. It is imperative to meticulously examine the quality of the segmentation outputs using multiple evaluation criteria to gain a holistic understanding of the model's performance and identify potential limitations or biases in the predictions [5] when performing similar tasks.

Model Name	Dice Coefficient
Hybrid Loss 1	0.860
Hybrid Loss 2	0.389
Custom Dice Loss	0.841

Table 5.5: Average Dice Coefficients for Loss Functions on test data

5.4 Summary of Final Results

The following tables summarise the key findings from all the implemented experiments. Table 5.6 summarises the calculated metric scores for each model along with the respective Loss functions.

						Dice Coefficients					
Model	**Loss Fun.**	**IoU**	**Prec.**	**Sens.**	**Spec.**	**Nec.**	**Edem.**	**Enh.**	**WT**	**TC**	**ET**
2D U-Net	CE	0.764	0.995	0.994	0.998	0.522	0.751	0.651	-	-	-
3D U-Net	CE+Dice	0.768	-	-	-	-	-	-	0.859	0.857	1.00
HL1 U-Net	HL1	0.079	0.262	0.262	0.754	0.914	0.896	0.905	-	-	-
HL2 U-Net	HL2	0.015	0.055	0.055	0.685	0.383	0.374	0.361	-	-	-
Custom Dice	Dice+FL	0.088	0.284	0.284	0.761	0.843	0.832	0.807	-	-	-

Table 5.6: Comparison of all the implemented models on the UCSF-PDGM dataset. Metrics abbreviations: IoU - Intersection over Union, Prec. - Precision, Sens. - Sensitivity, Spec. - Specificity, Nec. - Necrotic, Edem. - Edema, Enh. - Enhancing, WT - Whole Tumor, TC - Tumor Core, ET - Enhancing Tumor.

Chapter 6

Conclusion

In summary, this thesis investigated brain tumor segmentation using MRI scans by applying deep learning techniques. The research began with an extensive literature review that established a robust foundation for understanding the current state-of-the-art and recent advancements in segmentation methods, highlighting their crucial role in the medical diagnostic process as seen through Tables 2.2 and 2.3 [24, 51]. The core of this study focused on the development and evaluation of three distinct segmentation models: a custom-designed 2D U-Net [9, 16], a 3D U-Net inspired by the seminal work of Myronenko [21], and the innovative application of advanced loss functions to effectively tackle the challenges posed by class imbalance in the dataset [38].

The research progressively delved into the processing of the UCSF-PDGM dataset, emphasizing the potential and challenges associated with applying deep learning techniques in the intricate domain of medical image analysis. Although the 2D U-Net achieved notable performance, it demonstrated limitations when directly compared to the sophisticated architectures reported in the literature [60, 67]. The adapted 3D U-Net model addressed the intricacies of volumetric data, representing a significant advancement towards future methodological improvements as seen through Figure 4.5 and the high Dice Coefficient of 0.86 for the WT region[72].

Exploring advanced loss functions yielded mixed but informative results, underscoring the importance of rigorous evaluation and the potential pitfalls associated with misaligned training strategies. Furthermore, the attempt to predict methylation status using a transformer model revealed the constraints imposed by limited data and reduced image dimensions, suggesting avenues for enhancing model performance through expanded datasets and more comprehensive training approaches [15].

A comparative analysis of the developed models against state-of-the-art segmentation techniques provided a comprehensive overview of the current landscape and the necessary advancements to bridge the existing gaps. The thesis discussed architectural optimizations, computational efficiency improvements, hyperparameter tuning, and multi-faceted evaluation strategies, outlining a roadmap for elevating brain tumor segmentation models to clinical applicability [62, 61].

This research highlights the delicate balance between methodological rigor and innovative exploration, establishing a foundation for future studies that have the potential to drive groundbreaking advancements in medical image analysis and a wide range of healthcare applications through deep learning.

From a medical perspective, this work underscores the transformative potential of deep learning in automating brain tumor segmentation from MRI scans. The ability of these sophisticated algorithms to process and analyze large-scale datasets like UCSF-PDGM indicates promising clinical implications. Such models could significantly aid radiologists by improving diagnostic accuracy, reducing variability, and saving valuable time [145].

Addressing the substantial impact of brain tumors, which are associated with high morbidity and mortality rates, demands innovations that can enhance diagnostic precision and treatment efficacy [26]. The utilization of CNNs and transformer models for glioma segmentation and methylation status prediction suggests a shift towards personalized medicine, potentially improving patient outcomes, particularly for those affected by high-grade tumors with historically poor survival rates [48].

Ultimately, this project aims to promote the integration of AI-assisted diagnostics within the field of neuro-oncology, potentially ushering in a paradigm shift in patient care [**?**]. By streamlining and optimizing the segmentation process, these technologies could facilitate earlier interventions, more precise treatment planning, and improved survival rates for patients diagnosed with brain tumors [28, 31].

6.1 Objectives

The following table summarises the list of objectives from Section 1.2 that have been accomplished and where to find them in the paper.

Type	Objective	Work Completed
P	Complete a literature review which surveys the state of the art on the topic of Brain Tumour Segmentation (in MRI) using deep learning and formulate research questions.	Literature review in Section 2 covers the comain background, network architectures, the difference in the dataset over the years, preprocessing and post-processing techniques and methods for class imblance.
P	Implement a pipeline that will read in and pre-process the data from the open source dataset (UCSF-PDGM-v2).	Inspired by [16], Figure 3.1 shows the implemented 2D U-Net pipeline and Section 3.3 discusses the implementation.
P	Create a segmentation of brain tumour images with an implementation of a segmentation algorithm.	In Section 4.1, the results of the 2D U-Net are discussed. This is seen through Figure 4.2, and Table 4.2.
S	Create a segmentation of brain tumour images with a reimplementation of an existing algorithm as identified in the literature review.	In Sections 3.4, Myrenko's [21] model was reimplemented with inspiration taken from [22]. The segmentation pipeline is outlined in Figure 3.3 and the segmentation results have been analysed in Figure 4.2. The performance is compaired with the Literature Review in Section 5.1.
S	Implement more advanced algorithms or optimise existing deep learning algorithms to possibly find a novel solution for Brain Tumour Segmentation.	The implementation of algorithmic Loss functions ,inspired by [38] have been implemented in Section 3.6.3 through the Pipeline in Figure 3.5. These algorithms were tested and trialled with augmentation techniques used by [17].
S	Compare the performance and results of algorithms and the deep learning technique implementation from both my own model analysis and the results achieved by others in the literature review.	The performance of the algorithmic Loss functions has been reported and evaluated in Sections 5.4 and through Tables 5.4. The performance of the Transformer implementation has been evaluated via Section 5.2.

Table 6.1: Evaluation of Original Objectives and where they can be found in the paper. 'P' is for primary objectives and 'S' is a secondary objective.

Appendix A

Ethics Approval

University of St Andrews | FOUNDED 1413

School of Computer Science Ethics Committee

09 November 2023

Dear Mohak,

Thank you for submitting your ethical application which was considered by the School Ethics Committee.

The School of Computer Science Ethics Committee, acting on behalf of the University Teaching and Research Ethics Committee (UTREC), has approved this application:

Approval Code:	CS17317	**Approved on:**	09.11.23	**Approval Expiry:**	09.11.28
Project Title:	Deep Learning for Cancer Detection (and Segmentation) in Medical Imaging				
Researcher(s):	Mohak Misra				
Supervisor(s):	David Harris-Birtill				

The following supporting documents are also acknowledged and approved:

1. Application Form

Approval is awarded for 5 years, see the approval expiry data above.

If your project has not commenced within 2 years of approval, you must submit a new and updated ethical application to your School Ethics Committee.

If you are unable to complete your research by the approval expiry date you must request an extension to the approval period. You can write to your School Ethics Committee who may grant a discretionary extension of up to 6 months. For longer extensions, or for any other changes, you must submit an ethical amendment application.

You must report any serious adverse events, or significant changes not covered by this approval, related to this study immediately to the School Ethics Committee.

Approval is given on the following conditions:

- that you conduct your research in line with:
  - the details provided in your ethical application
  - the University's Principles of Good Research Conduct
  - the conditions of any funding associated with your work
- that you obtain all applicable additional documents (see the 'additional documents' webpage for guidance) before research commences.

You should retain this approval letter with your study paperwork.

Yours sincerely,

Wendy Boyter

SEC Administrator

School of Computer Science Ethics Committee
Dr Olexandr Konovalov/Convenor, Jack Cole Building, North Haugh, St Andrews, Fife, KY16 9SX
Telephone: 01334 463273 Email: ethics-cs@st-andrews.ac.uk

Figure A.1: Ethics Approval

Appendix B

Usage

The code for this thesis is open and publicly accessible on Github: https://github.com/mohakmisra8/CS4099FinalSubmission/tree/master

On University of St Andrews, School of Computer Science GPUs—the code should be run inside a Docker container built with the latest TensorFlow image from https://catalog.ngc.nvidia.com/orgs/nvidia/containers/tensorflow/tags. Instructions for setting up Docker can be found on the school's systems wiki: https://systems.wiki.cs.st-andrews.ac.uk/index.php/Docker. The DockerFile file included in the source code on GitHub can be used to build the Docker container to install all the necessary Python libraries.

The data has not been included in the source code since it is around 700 GB. The UCSF-PDGM dataset can be downloaded using The Cancer Imaging Archive data retriever tool: https://www.ibm.com/aspera/connect/ on https://wiki.cancerimagingarchive.net/pages/viewpage.action?pageId=119705830#1197058307e324bc732ef415a86e5ac4b072a65f2.

The data has not been included in the source code since it is around 700 GB. The UCSF-PDGM dataset can be downloaded using The Cancer Imaging Archive data retriever tool: https://www.ibm.com/aspera/connect/ on https://wiki.cancerimagingarchive.ne

t/pages/viewpage.action?pageId=119705830#1197058307e324bc732ef415a86e5ac4b072a65f2.

Use the command:

```
docker run \ -v /path/to/model/source:/workspace \
-v /path/to/dataset:/data \ -w /workspace \
--gpus all \ --shm-size=400g \ -it \ -p 8888:8888 \
--rm \ <docker-image-name>
```

To run the Implemented 2D U-Net, follow these steps in the command line:

1. `cd 2D_Implementation`
2. Run all the cells in the `2D_Notebook.ipynb`

To run the Implemented 3D U-Net, follow the following steps in the command line:

1. `cd 3D_Implementation`
2. Run the following command `python Train_Run.py`

To run the Transformer, follow the following steps in the command line:

1. `cd Transformer`
2. `python Transformer.py`

To run the U-Net with the Custom Dice Loss, follow the following steps in the command line:

1. `cd ClassImbalanceExperiments`
2. `cd ClassImbalanceExperiments/CustomDiceLoss/`
3. `python customDiceLoss.py`

To run the U-Net with the Hybrid Loss 1, follow the following steps in the command line:

1. `cd ClassImbalanceExperiments`
2. `cd ClassImbalanceExperiments/HybridLoss1/`
3. `python hybrid_loss_1.py`

To run the U-Net with the Hybrid Loss 2, follow the following steps in the command line:

1. `cd ClassImbalanceExperiments`
2. `cd ClassImbalanceExperiments/HybridLoss2/`
3. `python hybrid_loss_2.py`

Bibliography

[1] T. Nolan, "The university of california san francisco preoperative diffuse glioma mri (ucsf-pdgm)." https://wiki.cancerimagingarchive.net/pages/viewpage.action?pageId=119705830, 2023. Accessed: December 29, 2023.

[2] X. Li, G. Luo, and K. Wang, "Multi-step cascaded networks for brain tumor segmentation." https://www.arxiv-vanity.com/papers/1908.05887/, 2019.

[3] nilearn developers, "nilearn: Machine learning for neuroimaging in python." https://github.com/nilearn/nilearn, 2023. Accessed: December 29, 2023.

[4] NewbieTTN, "Precision, recall, sensitivity, specificity." https://newbiettn.github.io/2016/08/30/precision-recall-sensitivity-specificity/, 2016. Accessed: 2024-03-17.

[5] D. Müller, I. Soto-Rey, and F. Kramer, "Towards a guideline for evaluation metrics in medical image segmentation," *BMC Research Notes*, vol. 15, no. 1, p. 210, 2022.

[6] R. Kanjee, "Object segmentation vs. object detection - which one should you use?." https://www.linkedin.com/pulse/object-segmentation-vs-detection-which-one-should-you-ritesh-kanjee/, 2022. Accessed: 30 December 2023.

[7] "Schematic diagram of the receptive field in cnns from msf-net: Multi-scale feature learning network for classification of surface defects of multifarious sizes." Scientific Figure on ResearchGate, 2023. Available from: https://www.researchgate.net/f

igure/Schematic-diagram-of-the-receptive-field-in-CNNs_fig1_353545214 [accessed 30 Dec, 2023].

[8] J. Heyman, "Different pooling layers for cnn." https://jacobheyman702.medium.com/different-pooling-layers-for-cnn-4652a5103d62, 2021. Accessed: 23/1/2024.

[9] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," *CoRR*, vol. abs/1505.04597, 2015.

[10] F. Milletari, N. Navab, and S. A. Ahmadi, "V-net: Fully convolutional neural networks for volumetric medical image segmentation," in *2016 4th International Conference on 3D Vision (3DV)*, pp. 565–571, IEEE, 2016.

[11] Z. Liu, L. Tong, L. Chen, Z. Jiang, F. Zhou, Q. Zhang, X. Zhang, Y. Jin, and H. Zhou, "Deep learning based brain tumor segmentation: a survey," *Complex & Intelligent Systems*, vol. 9, pp. 1001–1026, 2023.

[12] G. Wang, W. Li, S. Ourselin, and T. Vercauteren, "Automatic brain tumor segmentation using cascaded anisotropic convolutional neural networks," *CoRR*, vol. abs/1709.00382, 2017.

[13] Z. Jiang, C. Ding, M. Liu, and D. Tao, "Two-stage cascaded u-net: 1st place solution to brats challenge 2019 segmentation task," in *International MICCAI Brainlesion Workshop*, (Cham), pp. 231–241, Springer, Oct 2019.

[14] J. Lin, J. Lin, C. Lu, H. Chen, H. Lin, B. Zhao, Z. Shi, B. Qiu, X. Pan, Z. Xu, B. Huang, C. Liang, G. Han, Z. Liu, and C. Han, "Ckd-transbts: Clinical knowledge-driven hybrid transformer with modality-correlated cross-attention for brain tumor segmentation," *IEEE Transactions on Medical Imaging*, vol. 42, no. 8, pp. 2451–2461, 2023.

[15] M. Phan, "Discussion on rsna-miccai brain tumor radiogenomic classification." https://www.kaggle.com/competitions/rsna-miccai-brain-tumor-radiogenomic-classification/discussion/280033, 2021. Accessed: 23/01/2024.

[16] Rastislav, "3d mri brain tumor segmentation u-net." https://www.kaggle.com/code/rastislav/3d-mri-brain-tumor-segmentation-u-net/notebook, Year. Accessed: 8/2/2024.

[17] L. Halder, "3d mri brain tumor segmentation u-net." Kaggle Notebook, 2021. Accessed: 2024-03-14.

[18] "Classification, object detection and image segmentation." https://developer.qualcomm.com/software/qualcomm-neural-processing-sdk/learning-resources/image-segmentation-deeplab-neural-processing-sdk/classification-object-detection-segmentation, n.d. Accessed: 30 December 2023.

[19] S. Bakas, M. Reyes, A. Jakab, S. Bauer, M. Rempfler, A. Crimi, R. Shinohara, C. Berger, S. Ha, M. Rozycki, M. Prastawa, E. Alberts, J. Lipkova, J. Freymann, J. Kirby, M. Bilello, H. Fathallah-Shaykh, R. Wiest, J. Kirschke, and V. Vilaplana, "Identifying the best machine learning algorithms for brain tumor segmentation, progression assessment, and overall survival prediction in the brats challenge," 04 2019.

[20] Radiological Society of North America (RSNA), "Brain tumor ai challenge 2021." https://www.rsna.org/rsnai/ai-image-challenge/brain-tumor-ai-challenge-2021, 2021. Accessed: 23/1/2024.

[21] A. Myronenko, "3d mri brain tumor segmentation using autoencoder regularization," in *Brainlesion glioma, multiple sclerosis, stroke and traumatic brain injuries* (A. Crimi, S. Bakas, H. Kuijf, F. Keyvan, M. Reyes, and T. van Walsum, eds.), vol. 11384 of *BrainLes lecture notes in computer science*, Cham: Springer, 2018.

[22] Polomarco, "Brats20 3dunet + 3dautoencoder." https://www.kaggle.com/code/polomarco/brats20-3dunet-3dautoencoder#Data, 2020. Accessed: 8/2/2024.

[23] FUNKYBOY, "Vit vision transformer 3d with one mri type." https://www.kaggle.com/code/super13579/vit-vision-transformer-3d-with-one-mri-type#Functions-to-load-images, 2021. Accessed: 2024-03-13.

[24] N. Gordillo, E. Montseny, and P. Sobrevilla, "State of the art survey on mri brain tumor segmentation," *Magnetic Resonance Imaging*, vol. 31, no. 8, pp. 1426–1438, 2013.

[25] D. R. White, A. S. Houston, W. F. Sampson, and G. P. Wilkins, "Intraand inter-operator variations in region-of-interest drawing and their effect on the measurement of glomerular filtration rates," *Clinical nuclear medicine*, vol. 24, no. 3, pp. 177–181, 1999.

[26] "Brain tumor: Statistics." https://www.cancer.net/cancer-types/brain-tumor/statistics, 2023. Accessed: December 29, 2023.

[27] "Cancer today." http://gco.iarc.fr/today/home. Accessed: December 28, 2023.

[28] "Facts and stats." https://www.curebraincancer.org.au/page/8/facts-stats, 2018. Accessed on: December 28, 2023.

[29] D. N. Louis *et al.*, "The 2016 world health organization classification of tumors of the central nervous system: A summary," *Acta Neuropathologica*, vol. 131, no. 6, pp. 803–820, 2016.

[30] B. Kaina, M. Christmann, S. Naumann, and W. Roos, "Mgmt: Key node in the battle against genotoxicity, carcinogenicity and apoptosis induced by alkylating agents," *DNA Repair (Amst)*, vol. 6, pp. 1079–1099, 2007.

[31] D. Bocangel, S. Sengupta, S. Mitra, and K. Bhakat, "p53-mediated down-regulation of the human dna repair gene o6-methylguanine-dna methyltransferase (mgmt) via interaction with sp1 transcription factor," *Anticancer Res*, vol. 29, pp. 3741–3750, 2009.

[32] J. Molnár, H. Engi, J. Hohmann, P. Molnár, J. Deli, O. Wesolowska, K. Michalak, and Q. Wang, "Reversal of multidrug resistance by natural substances from plants," *Curr Top Med Chem*, vol. 10, pp. 1757–1768, 2010.

[33] B. Tugcu, L. Postalci, O. Gunaldi, O. Tanriverdi, and H. Akdemir, "Efficacy of clinical prognostic factors on survival in patients with glioblastoma," *Turk Neurosurg*, vol. 20, pp. 117–125, 2010.

[34] E. Calabrese, J. Villanueva-Meyer, J. Rudie, A. Rauschecker, U. Baid, S. Bakas, S. Cha, J. Mongan, and C. Hess, "The university of california san francisco preoperative diffuse glioma mri (ucsf-pdgm)," 2022.

[35] Safe Software, "Nifti (neuroimaging informatics technology initiative) data format." `https://docs.safe.com/fme/html/FME-Form-Documentation/FME-ReadersWriters/nifti/nifti.htm`. Accessed: December 29, 2023.

[36] G. Vishnuvarthanan, M. P. Rajasekaran, N. A. Vishnuvarthanan, T. A. Prasath, and M. Kannan, "Tumor detection in t1, t2, flair and mpr brain images using a combination of optimization and fuzzy clustering improved by seed-based region growing algorithm," 2017. First published: 21 March 2017.

[37] T. F. Monaghan, S. N. Rahman, C. W. Agudelo, A. J. Wein, J. M. Lazar, K. Everaert, and R. R. Dmochowski, "Foundational statistical principles in medical research: Sensitivity, specificity, positive predictive value, and negative predictive value," *Medicina*, vol. 57, p. 503, May 2021. Published online 2021 May 16.

[38] D. Huang, M. Wang, L. Zhang, *et al.*, "Learning rich features with hybrid loss for brain tumor segmentation," *BMC Medical Informatics and Decision Making*, vol. 21, no. Suppl 2, p. 63, 2021.

[39] R. Ranjbarzadeh, A. B. Kasgari, S. J. Ghoushchi, S. Anari, M. Naseri, and M. Bendechache, "Brain tumor segmentation based on deep learning and an attention mechanism using mri multi-modalities brain images," *Scientific Reports*, vol. 11.

[40] R. Awati, "What are convolutional neural networks?: Definition from techtarget." `https://www.techtarget.com/searchenterpriseai/definition/convolutional-neural-network`, 2023. Accessed: 30 December 2023.

[41] A. Géron, *Hands-on Machine Learning with Scikit-Learn, Keras & TensorFlow: Concepts, Tools, and Techniques to Build Intelligent Systems*. O'Reilly Media, Inc., 2 ed., 2019.

[42] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems* (F. Pereira, C. Burges, L. Bottou, and K. Weinberger, eds.), vol. 25, Curran Associates, Inc., 2012.

[43] C. Szegedy, W. Liu, Y. Jia, P. Sermanet, S. Reed, D. Anguelov, D. Erhan, V. Vanhoucke, and A. Rabinovich, "Going deeper with convolutions," *arXiv preprint arXiv:1409.4842*, 2014.

[44] M. D. Zeiler and R. Fergus, "Visualizing and understanding convolutional networks," 2013.

[45] Y. Lecun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[46] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," 2015.

[47] "Brain tumours." `https://www.nhs.uk/conditions/brain-tumours/`, 2023. Accessed: 30 December 2023.

[48] H. Zong, R. G. Verhaak, and P. Canoll, "The cellular origin for malignant glioma and prospects for clinical advancements," *Expert Review of Molecular Diagnostics*, vol. 12, no. 4, pp. 383–394, 2012.

[49] B. H. Menze, A. Jakab, S. Bauer, J. Kalpathy-Cramer, K. Farahani, J. Kirby, *et al.*, "The multimodal brain tumor image segmentation benchmark (brats)," *IEEE Transactions on Medical Imaging*, vol. 34, no. 10, pp. 1993–2024, 2014.

[50] E. S. Biratu, F. Schwenker, Y. M. Ayano, T. G. Debelee, L. Rundo, C. Militello, V. Conti, F. Zaccagna, and C. Han, "A survey of brain tumor segmentation and classification algorithms," *Journal of Imaging*, vol. 7, no. 9, p. 179, 2021.

[51] A. Wadhwa, A. Bhardwaj, and V. S. Verma, "A review on brain tumor segmentation of mri images," *Magnetic Resonance Imaging*, 2019. Jaypee Institute of Information Technology, Noida, India and Ajay Kumar Garg Engineering College, Ghaziabad, India.

[52] K. Nurzynska, "Deep learning as a tool for automatic segmentation of corneal endothelium images," *Symmetry*, vol. 10, no. 3, p. 60, 2018.

[53] M. Schmidt, "Automatic brain tumor segmentation," tech. rep., Technical report TR05-13, 2005. Subjects: Automatic segmentation, Magnetic Resonance Imaging, Brain tumors.

[54] H. Lamba, "Understanding semantic segmentation with unet: A salt identification case study." Towards Data Science, Feb 2019.

[55] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3431–3440, 2015.

[56] D. Ciresan, A. Giusti, L. M. Gambardella, and J. Schmidhuber, "Deep neural networks segment neuronal membranes in electron microscopy images," in *Advances in neural information processing systems*, pp. 2843–2851, 2012.

[57] H. Jing, "Biomedical image segmentation - u-net." `https://jinglescode.github.io/2019/11/07/biomedicalimage-segmentation-u-net/`, July 2020.

[58] A. Casamitjana, M. Catà, I. Sánchez, M. Combalia, and V. Vilaplana, "Cascaded v-net using roi masks for brain tumor segmentation," in *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries*, vol. 10670 of *Lecture Notes in Computer Science*, Springer International Publishing, 2018.

[59] G. Urban, M. Bendszus, F. Hamprecht, and J. Kleesiek, "Multimodal brain tumor segmentation using deep convolutional neural networks," in *MICCAI BraTS (Brain Tumor Segmentation) Challenge. Proceedings, Winning Contribution*, pp. 31–35, 2014.

[60] D. Zikic, Y. Ioannou, M. Brown, and A. Criminisi, "Segmentation of brain tumor tissues with convolutional neural networks," in *Proceedings MICCAI-BRATS*, vol. 36, pp. 36–39, 2014.

[61] S. Pereira, A. Pinto, V. Alves, and C. A. Silva, "Brain tumor segmentation using convolutional neural networks in mri images," *IEEE Transactions on Computational Imaging*, vol. 35, no. 5, pp. 1240–1251, 2016.

[62] M. Havaei, A. Davy, D. Warde-Farley, A. Biard, A. Courville, Y. Bengio, C. Pal, P. Jodoin, and H. Larochelle, "Brain tumor segmentation with deep neural networks," *Medical Image Analysis*, vol. 35, pp. 18–31, 2017.

[63] L. Castillo, L. Daza, L. Rivera, and P. Arbeláez, "Volumetric multimodality neural network for brain tumor segmentation," in *13th International Conference on Medical Information Processing and Analysis*, vol. 10572, p. 105720E, International Society for Optics and Photonics, 2017.

[64] M. Akil, R. Saouli, R. Kachouri, *et al.*, "Fully automatic brain tumor segmentation with deep learning-based selective attention using overlapping patches and multi-class weighted cross-entropy," *Medical Image Analysis*, vol. 63, p. 101692, 2020.

[65] A. Jesson and T. Arbel, "Brain tumor segmentation using a 3d fcn with multi-scale loss," in *International MICCAI Brainlesion Workshop*, pp. 392–402, Springer, 2017.

[66] J. Chen *et al.*, "Transunet: Transformers make strong encoders for medical image segmentation," *arXiv preprint arXiv:2102.04306*, 2021.

[67] J. Kleesiek, A. Biller, G. Urban, U. Kothe, M. Bendszus, and F. Hamprecht, "Ilastik for multi-modal brain tumor segmentation," *Proceedings MICCAI BraTS (brain tumor segmentation challenge)*, pp. 12–17, 2014.

[68] R. McKinley, M. Rebsamen, R. Meier, and R. Wiest, "Triplanar ensemble of 3d-to-2d cnns with label-uncertainty for brain tumor segmentation," in *International MICCAI Brainlesion Workshop*, (Cham), pp. 379–387, Springer, October 2019.

[69] U. Baid, S. Talbar, S. Rane, S. Gupta, M. H. Thakur, A. Moiyadi, and A. Mahajan, "Deep learning radiomics algorithm for gliomas (drag) model: A novel approach using 3d unet based deep convolutional neural network for predicting survival in gliomas," in *International MICCAI Brainlesion Workshop*, (Cham), pp. 369–379, Springer, September 2018.

[70] K. Kamnitsas, C. Ledig, V. Newcombe, J. Simpson, A. Kane, D. Menon, D. Rueckert, and B. Glocker, "Efficient multi-scale 3d cnn with fully connected crf for accurate brain lesion segmentation," *Med. Image Anal.*, vol. 36, pp. 61–78, 2017.

[71] C. Zhou, S. Chen, C. Ding, and D. Tao, "Learning contextual and attentive information for brain tumor segmentation," in *International MICCAI Brainlesion Workshop*, (Cham), pp. 497–507, Springer, September 2018.

[72] F. Isensee, P. Kickingereder, W. Wick, M. Bendszus, and K. H. Maier-Hein, "No new-net," in *International MICCAI Brainlesion Workshop*, (Cham), pp. 234–244, Springer, September 2018.

[73] X. Zhao, Y. Wu, G. Song, Z. Li, Y. Zhang, and Y. Fan, "A deep learning model integrating fcnns and crfs for brain tumor segmentation," *Med. Image Anal.*, vol. 43, pp. 98–111, 2018.

[74] F. Baba, "Discussion on rsna-miccai brain tumor radiogenomic classification." `https://www.kaggle.com/c/rsna-miccai-brain-tumor-radiogenomic-classification/discussion/281347`, 2021. Accessed: 23/01/2024.

[75] F. Baba, "Brain tumor radiogen." `https://github.com/FirasBaba/brain-tumor-radiogen`, 2021. Accessed: 23/01/2024.

[76] A. N. or Username (if available), "Discussion on rsna-miccai brain tumor radiogenomic classification." `https://www.kaggle.com/c/rsna-miccai-brain-tumor-radiogenomic-classification/discussion/280029`, 2021. Accessed: 24/01/2024.

[77] C. Soares, "Discussion on rsna-miccai brain tumor radiogenomic classification." `https://www.kaggle.com/c/rsna-miccai-brain-tumor-radiogenomic-classification/discussion/287713`, 2021. Accessed: 24/01/2024.

[78] C. Soares, "Tf efficientnet transfer learning strat split." `https://www.kaggle.com/code/cedricsoares/tf-efficientnet-transfer-learning-strat-split/notebook`, 2021. Accessed: 24/01/2024.

[79] N. N. T. T. D. T. K. V. D. Tung Vu Son, Truong Bui Nhat, "Discussion on rsna-miccai brain tumor radiogenomic classification." https://www.kaggle.com/c/rsna-miccai-brain-tumor-radiogenomic-classification/discussion/280402, 2021. Accessed: 24/01/2024.

[80] I. Lashkov, "Discussion on rsna-miccai brain tumor radiogenomic classification." https://www.kaggle.com/competitions/rsna-miccai-brain-tumor-radiogenomic-classification/discussion/281374, 2021. Accessed: 25/01/2024.

[81] J. Sled, A. Zijdenbos, and A. Evans, "A nonparametric method for automatic correction of intensity nonuniformity in mri data," *IEEE Transactions on Medical Imaging*, vol. 17, no. 1, pp. 87–97, 1998.

[82] N. J. Tustison, B. B. Avants, P. A. Cook, Y. Zheng, A. Egan, P. A. Yushkevich, and J. C. Gee, "N4itk: Improved n3 bias correction," *IEEE Transactions on Medical Imaging*, vol. 29, no. 6, pp. 1310–1320, 2010.

[83] L. Nyul, J. Udupa, and X. Zhang, "New variants of a method of mri scale standardization," *IEEE Transactions on Medical Imaging*, vol. 19, no. 2, pp. 143–150, 2000.

[84] G. Wang, W. Li, S. Ourselin, and T. Vercauteren, "Automatic brain tumor segmentation based on cascaded convolutional neural networks with uncertainty estimation," *Front. Comput. Neurosci.*, vol. 13, p. 56, 2019.

[85] S. Hussain, S. M. Anwar, and M. Majid, "Brain tumor segmentation using cascaded deep convolutional neural network," in *2017 39th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 1998–2001, 2017.

[86] N. Gordillo, E. Montseny, and P. Sobrevilla, "State of the art survey on mri brain tumor segmentation," *Magn. Reson. Imaging*, vol. 31, pp. 1426–1438, 2013.

[87] T. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," 2018.

[88] L. Sun, S. Zhang, H. Chen, and L. Luo, "Brain tumor segmentation and survival prediction using multimodal mri scans with deep learning," *Front. Neurosci.*, vol. 13, p. 810, 2019.

[89] P. Mlynarski, H. Delingette, A. Criminisi, and N. Ayache, "Deep learning with mixed supervision for brain tumor segmentation," *J. Med. Imaging*, vol. 6, p. 034002, 2019.

[90] S. Iqbal, M. Ghani Khan, T. Saba, Z. Mehmood, N. Javaid, A. Rehman, and R. Abbasi, "Deep learning model integrating features and novel classifiers fusion for brain tumor segmentation," *Microsc. Res. Tech.*, vol. 82, pp. 1302–1315, 2019.

[91] A. Kuzina, E. Egorov, and E. Burnaev, "Bayesian generative models for knowledge transfer in mri semantic segmentation problems," *Front. Neurosci.*, vol. 13, p. 844, 2019.

[92] P. Kao, T. Ngo, A. Zhang, J. Chen, and B. Manjunath, "Brain tumor segmentation and tractographic feature extraction from structural mr images for overall survival prediction," in *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries* (A. Crimi, S. Bakas, H. Kuijf, F. Keyvan, M. Reyes, and T. van Walsum, eds.), vol. 11384, (Berlin/Heidelberg, Germany), pp. 128–141, Springer International Publishing, 2019.

[93] P. Kao, F. Shailja, J. Jiang, A. Zhang, A. Khan, J. Chen, and B. Manjunath, "Improving patch-based convolutional neural networks for mri brain tumor segmentation by leveraging location information," *Front. Neurosci.*, vol. 13, 2020.

[94] M. I. Razzak, M. Imran, and G. Xu, "Efficient brain tumor segmentation with multiscale two-pathway-group conventional neural networks," *IEEE Journal of Biomedical and Health Informatics*, vol. 23, no. 5, pp. 1911–1919, 2019.

[95] "TensorFlow: An end-to-end open source machine learning platform." https://tensorflow.org/.

[96] "pandas: Powerful data structures for data analysis, time series, and statistics." https://pandas.pydata.org/.

[97] J. D. Hunter *et al.*, "Matplotlib: A plotting library for the python programming language and its numerical mathematics extension numpy." https://matplotlib.org/.

[98] "scikit-learn: Machine learning in python." https://scikit-learn.org/stable/.

[99] "PyTorch: An open source machine learning framework." https://pytorch.org/.

[100] "Albumentations: Fast and flexible image augmentations." https://github.com/albumentations-team/albumentations.

[101] "NumPy: The fundamental package for scientific computing with python." https://numpy.org/.

[102] "tqdm: A fast, extensible progress bar for python and cli." https://github.com/tqdm/tqdm.

[103] A. Rogozhnikov, "Einops: Clear and reliable tensor manipulations with einstein-like notation," in *International Conference on Learning Representations*, 2022.

[104] R. Solovyev, A. A. Kalinin, and T. Gabruseva, "3d convolutional neural networks for stalled brain capillary detection," *Computers in Biology and Medicine*, vol. 141, p. 105089, 2022.

[105] V. Dumoulin and F. Visin, "A guide to convolution arithmetic for deep learning," *arXiv preprint arXiv:1603.07285*, 2016.

[106] D. Scherer, A. M"uller, and S. Behnke, "Evaluation of pooling operations in convolutional architectures for object recognition," in *International conference on artificial neural networks*, pp. 92–101, Springer, 2010.

[107] A. Odena, V. Dumoulin, and C. Olah, "Deconvolution and checkerboard artifacts," *Distill*, vol. 1, no. 10, p. e3, 2016.

[108] M. Drozdzal, E. Vorontsov, G. Chartrand, S. Kadoury, and C. Pal, "The importance of skip connections in biomedical image segmentation," in *Deep learning and data labeling for medical applications*, pp. 179–187, Springer, 2016.

[109] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," in *Proceedings of the IEEE international conference on computer vision*, pp. 1026–1034, 2015.

[110] J. Tompson, R. Goroshin, A. Jain, Y. LeCun, and C. Bregler, "Efficient object localization using convolutional networks," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 648–656, 2015.

[111] Shervine, "Keras: How to generate data on the fly." `https://stanford.edu/~shervine/blog/keras-how-to-generate-data-on-the-fly`, Year. Accessed: 8/2/2024.

[112] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *Proceedings of the 32nd International Conference on Machine Learning*, 2015.

[113] "Reducelronplateau." `https://keras.io/api/callbacks/reduce_lr_on_plateau/`, 2024. Accessed: 2024-03-19.

[114] N. Srivastava *et al.*, "Dropout: a simple way to prevent neural networks from overfitting," *The Journal of Machine Learning Research*, vol. 15, no. 1, pp. 1929–1958, 2014.

[115] A. Dosovitskiy *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," in *International Conference on Learning Representations*, 2021.

[116] J. Devlin *et al.*, "Bert: Pre-training of deep bidirectional transformers for language understanding," in *Proceedings of the 2019 Conference of the North American Chapter of the Association for Computational Linguistics: Human Language Technologies, Volume 1*, 2019.

[117] J. Gehring *et al.*, "Convolutional sequence to sequence learning," in *Proceedings of the 34th International Conference on Machine Learning*, vol. 70, 2017.

[118] A. Vaswani *et al.*, "Attention is all you need," in *Advances in Neural Information Processing Systems*, vol. 30, 2017.

[119] J. L. Ba *et al.*, "Layer normalization," *arXiv preprint arXiv:1607.06450*, 2016.

[120] K. He *et al.*, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pp. 770–778, 2016.

[121] A. Rosenfeld *et al.*, "Constructive language generation for effective meeting summarization," in *Proceedings of the 2019 Conference on Empirical Methods in Natural Language Processing and the 9th International Joint Conference on Natural Language Processing*, 2019.

[122] D. Hendrycks and K. Gimpel, "Gaussian error linear units (gelus)," *arXiv preprint arXiv:1606.08415*, 2016.

[123] H. Noh, S. Hong, and B. Han, "Learning deconvolution network for semantic segmentation," in *2015 IEEE International Conference on Computer Vision (ICCV)*, pp. 1520–1528, 2015.

[124] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and

fully connected crfs," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 40, no. 4, pp. 834–848, 2018.

[125] S. Jégou, M. Drozdzal, D. Vazquez, A. Romero, and Y. Bengio, "The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation," in *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 1175–1183, 2017.

[126] Özgün Çiçek, A. Abdulkadir, S. S. Lienkamp, T. Brox, and O. Ronneberger, "3d u-net: Learning dense volumetric segmentation from sparse annotation," 2016.

[127] J. Hu, L. Shen, and G. Sun, "Squeeze-and-excitation networks," in *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 7132–7141, 2018.

[128] A. G. Roy, N. Navab, and C. Wachinger, "Concurrent spatial and channel squeeze excitation in fully convolutional networks," 2018.

[129] A.-M. Rickmann, A. G. Roy, I. Sarasua, N. Navab, and C. Wachinger, "'project excite' modules for segmentation of volumetric medical scans," 2019.

[130] H. Chen, Q. Dou, L. Yu, J. Qin, and P.-A. Heng, "Voxresnet: Deep voxelwise residual networks for brain segmentation from 3d mr images," *NeuroImage*, vol. 170, pp. 446–455, 2018. Segmenting the Brain.

[131] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," 2015.

[132] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017.

[133] "TensorFlow Documentation on CategoricalFocalCrossentropy." https://www.tensorflow.org/api_docs/python/tf/keras/losses/CategoricalFocalCrossentropy, 2023. Accessed: 2023-03-12.

[134] I. Goodfellow, Y. Bengio, and A. Courville, *Deep Learning*. MIT Press, 2016. http://www.deeplearningbook.org.

[135] Y. Bengio, A. Courville, and P. Vincent, "Representation learning: A review and new perspectives," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 35, no. 8, pp. 1798–1828, 2013.

[136] R. L. Figueroa, Q. Zeng-Treitler, S. Kandula, and L. H. Ngo, "Predicting sample size required for classification performance," *BMC Medical Informatics and Decision Making*, vol. 12, no. 1, p. 8, 2012.

[137] F. Isensee, J. Petersen, S. Kohl, P. Jaeger, and K. Maier-Hein, "nnu-net: Breaking the spell on successful medical image segmentation," 04 2019.

[138] A. A. Taha and A. Hanbury, "Metrics for evaluating 3d medical image segmentation: analysis, selection, and tool," *BMC Med Imaging*, vol. 15, p. 29, 2015. Research Support, Non-U.S. Gov't.

[139] T.-Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 2, pp. 318–327, 2020.

[140] V. Yeghiazaryan and I. Voiculescu, "An overview of current evaluation methods used in medical image segmentation," Tech. Rep. RR-15-08, Department of Computer Science, Oxford, UK, 2015.

[141] A. Dhinakaran, "Understanding kl divergence: A guide to the math, intuition, and practical use of kl divergence — including how it is best used in drift monitoring." ht

tps://towardsdatascience.com/understanding-kl-divergence-f3ddc8dff254, Feb 2023. Published in Towards Data Science.

[142] A. Shekhar, "What are l1 and l2 loss functions?." https://amitshekhar.me/blog/l1-and-l2-loss-functions, Aug 2019. Accessed: 2024-03-21.

[143] J. Bergstra and Y. Bengio, "Random search for hyper-parameter optimization," *Journal of Machine Learning Research*, vol. 13, no. Feb, pp. 281–305, 2012.

[144] P. Liashchynskyi and P. Liashchynskyi, "Grid search, random search, genetic algorithm: A big comparison for nas," 2019.

[145] E. J. Topol, "High-performance medicine: the convergence of human and artificial intelligence," *Nature Medicine*, vol. 25, no. 1, pp. 44–56, 2019.