

UNIVERSITY OF BAHRAIN

College of Science

Department of Mathematics



Mathematical and Computational Convolutional Neural Network Methods for MRI Segmentations

A Thesis Submitted in Partial Fulfilment of the Requirements for the
Master Degree in Science in Mathematics

Submitted By

Huda Jaffer Alsaeed

20071197

Supervised By

Dr. Abdulla Eid

(Associate Professor)

Dr. Anwar A.S Mohamed

(Assistant Professor)

University of Bahrain

July 2025



Dissertation Deposit Form

Form Code: UDNGS025

Author Name:	Huda Jaffer Alsaeed
Thesis Title:	Mathematical and Computational Convolutional Neural Network Methods for MRI Segmentations
Year of Publication:	2025
Department:	Department of Mathematics
College:	College of Science
E-mail:	20071197@uob.stu.edu.bh
Contact No.	36699014

Author's Declaration: I agree to the following Conditions:

The above thesis will be available (in the University of Bahrain Library and externally) and reproduced as necessary at the discretion of the University of Bahrain. It may also be digitized by the University of Bahrain and made available via the university's repository or via other parties on the Internet.

This condition shall apply to all copies including electronic copies.

The above thesis has been provided on the understanding that it is copyright protected material and that no quotation from this thesis may be published without paper acknowledgement.

Sign. of the Author:	huda.	Date:	22-July-2025
----------------------	-------	-------	--------------



Thesis Discussion Committee Decision

Form Code: UDNGS013

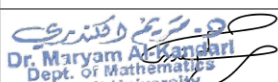
Student Name:	Huda Jaafar AlSaeed
Student ID. No.	20071197
Department:	Department of Mathematics
College:	College of Science
Program:	M.Sc. in Mathematics
Student GPA:	3.45
Thesis Title:	Mathematical and Computational Convolutional Neural Network Methods for MRI Segmentations

The Discussion Committee met on Sunday 16 June 2025 in _____ upon discussing the student whose details are stated above in public. The discussion committee recommends the following:

The student is awarded the grade _____ in the thesis course MATHS 599.

- ☐ Without making any changes to the thesis.
- ☒ After making the modifications attached with this decision.

Discussion Committee Members Approval:

1.	Supervisor Name:	Dr. Abdulla Eid Dr. Anwar A.S. Mohamed	Sign.:	
2.	Internal Examiner:	Dr. Nasser Metwally	Sign.:	
3.	External Examiner:	Dr. Maryam Mohammad Alkandari	Sign.:	 Dr. Maryam Alkandari Dept. of Mathematics Kuwait University

ABSTRACT

Segmenting brain tumors using magnetic resonance imaging (MRI) is essential for effective clinical diagnosis and treatment planning. Conventional segmentation techniques, such as thresholding, region-growing algorithms, and active contour models, face considerable challenges. These methods struggle with brain tumor segmentation because they require manual input and struggle with variations in intensity of MRI. They also fail to capture tumor's irregular shapes and indistinct edges due to limited morphological feature extraction. Additionally, traditional methods often handle 2D slices separately, overlooking vital 3D contextual information crucial for precise volumetric analysis of brain tumors. Deep learning has transformed medical image segmentation by overcoming various constraints. Convolutional Neural Networks (CNNs), especially fully convolutional architectures such as U-Net and V-Net, have shown impressive achievements in automating the segmentation of brain tumors. These deep learning methods are adept at learning hierarchical features straight from data, managing 3D volumetric information, and ensuring spatial consistency across slices—all crucial skills for tackling the complexities of brain tumor segmentation. This study provides an in-depth comparison of two deep learning architectures for brain tumor segmentation, 3D U-Net and V-Net. Using the BraTS 2020 dataset for evaluation, our 3D U-Net implementation achieved a segmentation accuracy of 98.26 percent with an IoU of 0.6792. In contrast, V-Net outperformed with a 98.65 percent accuracy and an IoU of 0.7330. The findings emphasize V-Net's particular strengths in addressing tumor boundary uncertainties through its residual connections and advanced feature learning abilities. Compared to traditional methods, both deep learning models offer notable improvements in processing speed, consistency, and capability to capture complex tumor characteristics without manual intervention. The findings highlight the significant impact of deep learning in neuro-oncology imaging, providing more dependable and efficient tools for clinical decision-making.

Keywords: Brain tumor segmentation, MRI, traditional methods, deep learning, 3D U-Net, V-Net, convolutional neural networks

Contents

APPROVAL	A
DECLARATION	B
ABSTRACT	C
TABLE OF CONTENTS	D
LIST OF FIGURES	G
LIST OF TABLES	I
ACKNOWLEDGMENTS	J
DEDICATION	K
1 Introduction	1
1.1 Overview	1
1.2 Research Problem	2
1.3 Research Questions	3
1.4 Research Objectives	3
1.5 Contribution to the Field	4
1.6 Thesis Outline	4
2 Mathematical Preliminaries	6
2.1 Introduction	6
2.2 Convolution Operation	7
2.2.1 Continuous Convolution	9
2.2.2 Discrete Convolution	10
2.3 The Characteristics and Functionalities of Different Convo- lution Types	10
2.3.1 Two Dimensional Convolutions	10

2.3.2	One Dimensional Convolution	11
2.3.3	Three Dimensions Convolutions for Volumetric Data	12
2.3.4	Activation Functions	13
2.3.4.1	Rectified Linear Unit (ReLU)	13
2.3.4.2	SoftMax Function	13
2.4	Fourier Transform for Images	14
2.5	Magnetic Resonance Imaging (MRI) Processing	16
2.6	Types of Brain MR Images	19
2.7	Sparsity of MR Images	20
2.7.1	Compressed Sensing (CS) in MRI	21
2.7.2	Data Compression	21
2.7.3	Disturbance Suppression and Image Quality	22
2.7.4	Sparse Dictionary Learning	22
3	Literature Review	23
3.1	Definition of Image Segmentation	23
3.2	Traditional Approaches for Medical Image Segmentation . .	27
3.2.1	Thresholding Methods	27
3.2.2	Region-Growing Method	31
3.2.3	Active Contour Method	35
3.3	Deep Learning for Medical Image Segmentation	37
3.3.1	Introduction	37
3.3.2	Convolutional Neural Networks (CNNs)	38
3.3.2.1	Residual Neural Networks	43
3.3.2.2	Batch Normalization	43
3.3.2.3	The Encoder-Decoder Structure	44
3.3.2.4	The Theoretical Framework of Simple 3D U-Net Model	49
3.3.2.5	The Theoretical Framework of V-Net Model	53
4	Numerical Implementations	56

4.1	Chapter Overview	56
4.2	Dataset Used	57
4.3	Dataset Pre-processing	59
4.4	The Composition of Used Models	61
4.4.1	3D U-Net Model	61
4.4.2	V-Net Model	65
4.5	Evaluation Metrics	68
4.5.1	Dice Coefficient	68
4.5.2	Total Loss	70
4.5.2.1	Dice Loss	70
4.5.2.2	Focal Loss	70
4.5.2.3	Combined Dice and Focal Loss Behavior .	71
4.5.3	Intersection over Union (IoU)	72
4.5.4	Accuracy	73
4.5.5	The Training Process	74
5	Result Analysis and Comparison	75
5.1	Result Analysis	75
5.2	Observation	80
5.3	Comparison with Other Models	83
5.4	Discussion	87
6	Conclusion and Future works	89
6.1	Conclusion	89
6.2	Future Works and Recommendations	90
	References	92
	Abstract (in Arabic)	

List of Figures

2.1	Example of 2D convolution operation: input matrix I convolved with kernel K produces output matrix C	8
2.2	Effect of convolution filters on a grayscale image. (a) original image, (b) result after applying vertical edge detection filter, (c) result after applying horizontal edge detection filter.	9
3.1	The types of input MRI of brain (FLAIR, T1CE, T2) and the output segmentation image (Mask)	25
3.2	Segmentation using Otsu thresholding	31
3.3	An example of SRG with threshold	33
3.4	Generated seed points and annotations using the proposed approach	34
3.5	An illustration of a FC neural network	42
3.6	The convolutional autoencoder's structure	46
4.1	Examples from BraTS 2020 dataset	58
4.2	Samples of the flair, t1ce, t2 and mask	60
4.3	3D U-Net Architecture by layers	63
4.4	3D U-Net Architecture by block diagram	64
4.5	V-Net Architecture by block diagram	67
4.6	Visual representation of the Dice coefficient showing predicted, ground truth, and overlapping (green) areas.	69
5.1	The Training vs Validation Accuracy Graph of 3D U-Net model	78
5.2	The Training vs Validation Loss Graph of 3D U-Net model	78

5.3	The Training vs Validation Accuracy Graph of V-Net model	79
5.4	The Training vs Validation Loss Graph of V-Net model . . .	79
5.5	Examples of prediction accuracies for some images using 3D U-Net model	81
5.6	Examples of prediction accuracies for some images using V-Net model	82
5.7	The U-Net architecture	84
5.8	The V-Net architecture	85

List of Tables

3.1	Class Values and Meanings	25
5.1	Results for 3D U-Net and V-Net Models	76
5.2	Comparison the results of models	84

ACKNOWLEDGMENTS

With heartfelt gratitude flowing from the depths of my heart and with words that may never fully capture my appreciation, I extend my sincerest thanks and profound appreciation to everyone who supported me on this journey and to all who laid the foundation for the realization of this work. I express my deepest gratitude to Dr. Anwar for his insightful guidance and steadfast support which served as a beacon throughout this journey. My sincere thanks also go to Dr. Abdullah for his valuable feedback and unwavering encouragement, both of which greatly contributed to the completion of this thesis. To my beloved family: Your love and patience have embraced me at every stage and your presence has been a constant source of strength and warmth. To my dear parents-the heartbeat of every prayer, the pillar of my soul and the endless source of giving- I owe you my deepest gratitude. All I have achieved is through the grace of Allah and then by your unwavering support. For all of this, I am eternally grateful. This achievement is not mine alone, but a reflection of your faith, your help and heartfelt prayers.

DEDICATION

To the one whose presence brought a unique meaning to my journey. To my son, Abdullah, whose shining eyes were my driving force to persevere and give my best. Your journey taught me the strength of patience and the value of steady progress, reminding me that every step forward, no matter how small, leads to meaningful achievement. To my son Hussain and my daughter Hawraa, who endured my absence, silence and preoccupation, and who despite their young age exemplified patience and unwavering support. To my life partner, my dear husband, my backbone , my first supporter and steadfast pillar. Your constant presence, encouragement, and belief in me were instrumental to my continued progress throughout this endeavor. To my dear parents, your endless prayers,encouragement and unconditional support have been a guiding light during moments of difficulty and doubt. May Allah bless you for everything you have given me. To my family, your love and support have surrounded me throughout this journey. This accomplishment is yours as much as it is mine. This work is the result of shared effort and is sincerely dedicated to those who, after Allah played a vital role in its completion.

Chapter 1

Introduction

1.1 Overview

Brain tumors become more common with age, contributing to the increased incidence of these diseases. In medical imaging of brain tumor patients, segmentation is vital to identify affected brain regions. It is critical for diagnosis, therapy planning, disease progression, and tumor delineation (Du, Cao, Liang, Chen, & Zhan, 2020).

Traditional manual segmentation relies on expert-defined features, but inconsistencies arise from image intensity variations and other factors. Thus, these methods lack precision and robustness, necessitating repeated manual adjustment. However, manual tumor segmentation is often labor intensive and time consuming, particularly due to the diverse tissue regions in tumors and the complexity of three-dimensional multimodal brain magnetic resonance imaging scans. Manual segmentation is error-prone due to dependence on image quality and operator subjectivity. Inaccuracies and redundant segmentation are common, making it necessary to develop automated segmentation techniques that offer greater accuracy for clinical use.

Semi-automated methods-traditional approaches- require feature engineering and machine learning for segmentation. In contrast, deep learning (e.g., CNNs) learns representations directly from images.

Key challenges in automated brain tumor segmentation include: Tumor

detection relies on pixel intensity comparisons, but gray-scale offsets reduce tissue contrast, obscuring boundaries. However, the gray-scale offset fields tend to reduce the contrast between the tissues and hence the tumor boundary is less conspicuous. Tumors vary in size, shape, location, and tissue involvement, complicating algorithm design. Tumor mass effects distort brain anatomy, introducing irregular background data that challenge segmentation (Hussain et al., 2022).

Automated segmentation systems can significantly enhance the efficiency and precision of tumor identification and treatment planning, minimizing human error and improving patient outcomes (Buchholz & Jug, 2022). To address these challenges, fully automated segmentation methods have been developed. These methods eliminate the need for manual intervention in tumor detection and typically use deep learning (DL). These methods improve segmentation effectiveness, reliability, and reproducibility. In addition, fully automated methods have reported very high accuracy and precision, their training demands large amounts of labeled data and they are computationally intensive (Bhatt et al., 2021). This study evaluates two CNN-based automated methods for tumor segmentation, addressing challenges in localization, denoising, and accuracy-particularly for tumors with diffuse boundaries.

1.2 Research Problem

The research problem addressed in this study centers on applying deep learning models based on the accuracy and effectiveness of brain tumor segmentation in MRI. In the recent times, CNNs have been found to give high accuracy in computer vision activities such as medical image segmentation compared to the conventional methodologies as they learn the features through the data itself. Nevertheless, the use of manually designed features in conjunction with CNNs for brain tumor segmentation is, according to

literature (Jalab & Hasan, 2019), not thoroughly investigated. The key challenges include how to effectively with CNN-based models and how this approach can improve accuracy, particularly in addressing the complexity of tumor shapes, sizes, and the distortion of surrounding tissues. This study aims to bridge this gap by developing a novel method that incorporates CNN based architectures for enhanced segmentation of brain tumors in MRI.

1.3 Research Questions

1. How do deep learning-based segmentation principles (e.g., feature extraction, spatial consistency) apply to MRI brain tumor segmentation?
2. How do the segmentation performance (e.g., Dice score, IoU) of 3D U-Net and V-Net differ statistically when evaluated on the BraTS2020 dataset?
3. What are the correlations between the size, shape, and localization of brain tumors achieved by U-Net and V-Net models?
4. Which model (U-Net or V-Net) better generalizes to rare tumor subtypes in clinical settings?

1.4 Research Objectives

Thesis work intends to accomplish the following:

1. Implement and evaluate two 3D deep learning models(U-Net and V-Net) for brain tumor segmentation using the BraTS2020 dataset.
2. Quantitatively compare model performance using segmentation metrics (Accuracy, IoU, Dice and Focal loss).

3. Qualitatively analyze differences in tumor boundary delineation and localization between U-Net and V-Net.
4. Identify clinical implications of each model's performance for diagnosis and treatment planning.

1.5 Contribution to the Field

This study contributes to the field of neuro-oncology imaging by:

1. Demonstrating the superiority of 3D deep learning models(U-Net and V-Net) over traditional 2D methods in capturing tumor spatial complexity.
2. Providing empirical evidence of V-Net's residual connections improving boundary segmentation accuracy(IoU: 0.63 vs. U-Net's 0.68).

1.6 Thesis Outline

This thesis is structured as follows:

- Chapter 2 (Literature Review): Provides a comprehensive review of both traditional and deep learning-based segmentation methods, establishing a solid theoretical foundation.
- Chapter 3 (Methodology): Clearly explains the proposed approach, including the use of 3D U-Net and V-Net architectures, along with the evaluation metrics employed.
- Chapter 4 (Implementation): Details the experimental setup, covering data preprocessing techniques and optimization strategies for 3D MRI segmentation.

- Chapter 5 (Results and Analysis): Presents the findings with a comparative analysis of the two models, supported by empirical evidence.
- Chapter 6 (Conclusion and Future Work): Summarizes key contributions and suggests potential directions for future research.

Chapter 2

Mathematical Preliminaries

2.1 Introduction

This chapter outlines the core mathematical concepts and operations that form the basis of the architectures used in this study, particularly focusing on 3D U-Net and V-Net. The chapter starts by covering the fundamentals of convolutional operations, demonstrating how they extract spatial features from MRI scans. It then delves into the differences between discrete and continuous convolution, emphasizing their significance in medical image processing. Following this, the chapter explores Fourier transforms and their role in analyzing frequency-domain representations of MRI data, essential for noise reduction and image enhancement. A significant emphasis is placed on 3D convolutions, extending traditional 2D operations to volumetric data, which enables the model to capture depth-related tumor features. The chapter also examines loss functions, such as Dice loss and Focal loss, detailing their mathematical formulations and explaining how they

enhance segmentation accuracy, particularly in handling imbalanced tumor sub-regions. Lastly, the chapter introduces sparsity and compressed sensing in MRI to improve computational efficiency while maintaining diagnostic quality. These concepts are foundational for the deep learning models developed in this thesis, ensuring they are both theoretically robust and practically effective for brain tumor segmentation.

2.2 Convolution Operation

Convolution is one of the simplest mathematical operations used in the processing of grid-type data, including time series, movies, and photographs. It plays a key role in signal processing, typically used for analyzing and transforming signals. As noted by (Kiranyaz et al., 2021), convolution operations are fundamental in deep learning and are frequently used in neural networks to extract features from raw input data. A typical convolution involves sliding a small kernel over the input data, where each kernel element is multiplied by the corresponding input element, and the results are summed to form a single value in the output feature map.

An example of this process is illustrated in Figure 2.1. Here, a 3×3 input matrix I is convolved with a 2×2 kernel K . As shown, when the kernel K with elements j, k, l, m overlaps with the top-left corner of the input matrix I with elements a, b, d, e , the resulting value in the output matrix C is computed as the sum of element-wise products: $a.j + b.k + d.l + c.m$. This operation is systematically repeated as the kernel slides across the entire input matrix, generating a new feature map C where each value highlights local patterns such as edges, textures, and shapes.

According to (Aloysius & Geetha, 2017), this ability to extract meaningful patterns makes convolution especially powerful for visual tasks in deep learning. It enables the network to learn spatial hierarchies of features in the data, layer by layer.

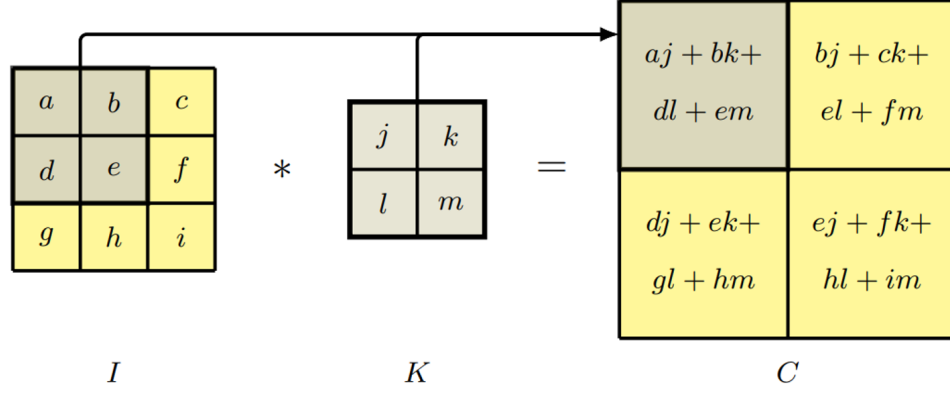


Figure 2.1: Example of 2D convolution operation: input matrix I convolved with kernel K produces output matrix C .

To further illustrate the effect of convolution operations on real-world data, Figure 2.2 shows the results of applying convolution-based edge detection filters to a grayscale image of a bicycle.

The original image Figure 2.2-a is processed using vertical and horizontal edge detection filters. These filters are convolution kernels designed to highlight gradients in specific directions:

Vertical edge detection Figure 2.2-b emphasizes vertically running edges by detecting changes in intensity in the horizontal direction.

Horizontal edge detection Figure 2.2-c emphasizes horizontal structures by detecting changes in intensity in the vertical direction.

These convolutional transformations enhance structural features such as the spokes of the bicycle and the bricks in the background, which are useful for downstream tasks such as object detection or image classification.

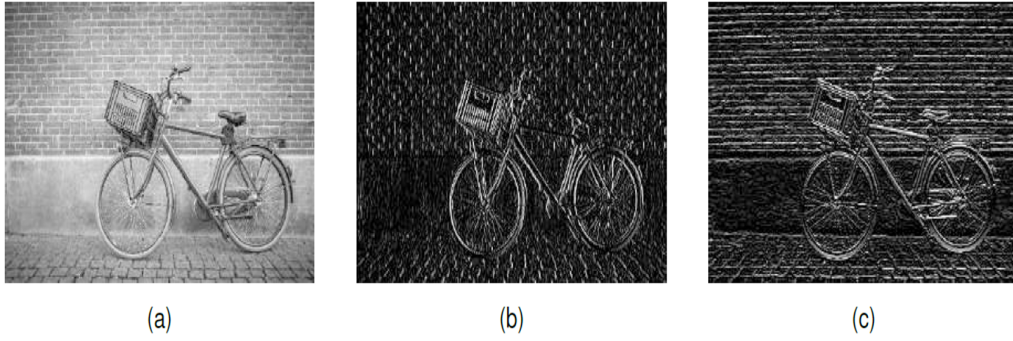


Figure 2.2: Effect of convolution filters on a grayscale image. (a) original image, (b) result after applying vertical edge detection filter, (c) result after applying horizontal edge detection filter.

2.2.1 Continuous Convolution

In the setting of analog signals, convolution is distinct as (Pinaya, Vieira, Garcia-Dias, & Mechelli, 2020):

$$(f * g)(t) = \int_{-\infty}^{\infty} f(\tau)g(t - \tau)d\tau \quad (2.1)$$

where

- τ : The integration variable that represents a time or spatial shift.
- t : The variable that represents the output time or spatial position of the convolved function.
- $f(\tau)$: The input function.
- $g(t)$: The kernel or filter function.

Having discussed continuous convolution, we now turn to its discrete counterpart.

2.2.2 Discrete Convolution

Convolution is the way by which two sequences are multiplied to form another sequence for discrete signals. It is defined as follows:

$$(f * g)[n] = \sum_{m=-\infty}^{\infty} f[m]g[n-m] \quad (2.2)$$

where

- n : The discrete index that represents the output position of the convolved sequence.
- m : The summation index represents a discrete shift.
- $f[n]$: The input sequence.
- $g[n]$: The kernel or filter sequence.

2.3 The Characteristics and Functionalities of Different Convolution Types

2.3.1 Two Dimensional Convolutions

Especially in Computer Vision (CV), 2D convolutions are components of convolutional neural networks (CNNs). It utilizes an input of two dimensions to obtain the features, such as edges and textures from the two-dimensional input of images in this case. For basic recognition processes such as object recognition, and picture categorization, these characteristics are mandatory (Bhatt et al., 2021).

Equation (2.3) represents the mathematical expression for a 2D convolution:

$$F(x,y) = \sum_{i=-m}^m \sum_{j=-m}^m K(i,j) \cdot I(x+i,y+j) \quad (2.3)$$

where:

- m : The spatial extent of the kernel.
- $F(x,y)$: Output feature maps.
- $K(i,j)$: Kernel values.
- $I(x,y)$: Input image values.

2D convolutions occur in natural language processing(NLP), CV, and signal processing to enhance the viability of facial recognition systems. Such applications include access control, identification verification, and classification tasks, which are possible due to 2D convolutions (Brar, Aggarwal, Nanda, Saxena, & Gautam, 2024).

By using the 2D convolutional layer, sequential data such as time series, audio signals, and textual information can be processed. Because of their superior temporal profile capturing ability, they are widely used in tasks such as speech recognition and text classification. These models are helpful in tasks such as speaker identification and emotion detection, as well as in revealing underlying patterns in sequential data(Kiranyaz et al., 2021).

2.3.2 One Dimensional Convolution

A 1D convolutional layer is a tool used in deep learning to analyze sequential data—things like sound waves, stock prices over time, or even sentences broken down into words. Unlike image processing (which uses 2D convolutions for height and width), a 1D convolution works along a single direction, like time or sequence order (Kiranyaz et al., 2021).

Imagine sliding a small window (called a filter or kernel) step-by-step over your data. At each stop, the filter multiplies its values with the corresponding part of the sequence and adds everything up to produce a single

number. By doing this across the whole sequence, it picks up on local patterns—like a sudden spike in a sensor reading, a repeated rhythm in audio, or a common phrase in text (Y. Kim, n.d.).

Equation (2.4) represents the mathematical expression for a one-dimensional convolution:

$$F(n) = \sum_{i=-m}^m K(i) \cdot I(n+i) \quad (2.4)$$

where:

- $F(n)$: Output sequence.
- $K(i)$: Kernel values.
- $I(n)$: Input sequence values.

2.3.3 Three Dimensions Convolutions for Volumetric Data

Motion and medical images are examples of 3D data for which 3D convolutions are developed. They are very useful in tasks such as tumor segmentation since they are able to evaluate spatial as well as temporal features concurrently. These models are created through the analysis of 3D images, helping to study global organ architecture and correlations between spaces, which increases diagnostic accuracy.

Equation (2.5) represents the mathematical expression for a 3D convolution:

$$F(x,y,z) = \sum_{i=-m}^m \sum_{j=-m}^m \sum_{l=-m}^m K(i,j,l) \cdot I(x+i,y+j,z+l). \quad (2.5)$$

where:

- $F(x,y,z)$: Output s.

- $K(i, j, l)$: Kernel values.
- $I(x, y, z)$: Input volume values.

In tumor segmentation, doctors benefit from using 3D convolutions to determine the size and location of tumors, assisting in selecting specific therapies(Xu, Dehghani, Corrigan, Caulfield, & Moloney, 2016).

2.3.4 Activation Functions

Activation functions introduce non-linearity into neural networks, enabling them to learn complex patterns. Two fundamental activation functions used in this study are defined below:

2.3.4.1 Rectified Linear Unit (ReLU)

The ReLU activation function is defined as:

$$\text{ReLU}(x) = \begin{cases} x & \text{if } x > 0 \\ 0 & \text{if } x \leq 0 \end{cases}. \quad (2.6)$$

This piecewise linear function provides sparse activations and avoids the vanishing gradient problem while maintaining computational efficiency.

2.3.4.2 SoftMax Function

For multi-class classification, the SoftMax function normalizes outputs to class probabilities:

$$\text{SoftMax}(\mathbf{z})_i = \frac{e^{z_i}}{\sum_{j=1}^C e^{z_j}}, \quad \text{for } i = 1, \dots, C. \quad (2.7)$$

where:

- $\mathbf{z} = (z_1, \dots, z_C) \in \mathbb{R}^C$ is the input vector of raw scores (logits).

- C is the number of classes.
- The denominator ensures that $\sum_{i=1}^C \text{SoftMax}(\mathbf{z})_i = 1$.

The SoftMax function provides a probability distribution over K possible classes.

2.4 Fourier Transform for Images

The Fourier Transform is a common image processing approach that can change a picture from the spatial domain into the frequency domain. Mainly, it shows that an image can be represented as a sum of sine and cosine functions of different frequencies. The original image resides in the spatial domain, while the Fourier Transform describes the image in the frequency space.

The Fourier image has a set frequency from the spatial domain image for each of its points. A large number of fields, such as image processing, image filtering, image reconstruction, and image compression, rely on the Fourier Transform. Casting a new light on the Discrete Fourier Transform (DFT) for digital images, this paper (Buchholz & Jug, 2022) is dedicated to this topic.

For a square image of size $N \times N$, the two-dimensional DFT is given by:

$$F(u, v) = \sum_{x=0}^{N-1} \sum_{y=0}^{N-1} f(x, y) e^{-j2\pi \left(\frac{ux}{N} + \frac{vy}{N} \right)}. \quad (2.8)$$

Since a rectangular periodic sequence is exactly equal to a finite-length sequence, the DFT can indeed perform the operation of convolution for finite arrays, which is its main strength. The Discrete Fourier Transform is also called the Fourier Transform of finite-duration sequences. Note that, in contrast to the continuous Fourier Transform, the DFT operates on and processes sequences of finite lengths and discrete variables.

The DFT samples the frequencies of an image but does not sum all frequencies. Instead, it represents the spatial domain image as a sum of a finite number of frequencies. The number of frequencies corresponds to the number of pixels in the spatial domain image.

It is extremely important to note that the Fourier Transform usually yields complex values because the DFT uses a complex exponential function. Therefore, an image's DFT consists of:

- **Phase (argument):** Depicts the position and balance of the frequency.
- **Amplitude (modulus, Euclidean distance):** Depicts the energy density of the frequency.

Normally, high-frequency components are located at the edges of an image, while low-frequency components are in the middle of the image.

As it allows data to be transformed from the frequency domain back to the spatial domain, the Inverse Fourier Transform is one of the fundamental concepts of signal processing. This procedure helps in understanding how a signal behaves in space or time. For instance, in MRI, the Inverse Fourier Transform plays a crucial role in relating frequency and spatial domains.

The Fourier Transform extends the Fourier series, which describes periodic functions as a sum of sinusoidal components. To generalize this for non-periodic functions, the Fourier Transform expresses them as a sum of sine and cosine functions over a continuous frequency spectrum. Similarly, the Inverse Fourier Transform can be used to reconstruct a function from the frequency domain back to the time (or spatial) domain.

The general formula for the Inverse Fourier Transform is:

$$f(x, y) = \sum_{u=0}^{N-1} \sum_{v=0}^{N-1} F(u, v) e^{j2\pi(\frac{ux}{N} + \frac{vy}{N})}. \quad (2.9)$$

Magnetic resonance imaging (MRI) relies heavily on Fourier Transforms due to their valuable properties. Among these properties is the abil-

ity to simplify data by analyzing complex computational signals. In MRI, phase alteration is crucial, as movements in the spatial domain correspond to phase variations in the frequency domain. Additionally, convolution plays a significant role in image filtering and enhancement, according to the convolution theorem, which states that convolution in one domain is equivalent to multiplication in the other.

Besides image reconstruction, the Fourier Transform is critical for other MRI processing tasks such as:

- Edge detection
- Noise removal
- Image enhancement

Low-frequency components are related to the overall image structure, while high-frequency components are often associated with noise. Adjusting the presence of specific frequencies in an image can enhance its clarity, making bright and detailed images more prominent.

2.5 Magnetic Resonance Imaging (MRI) Processing

Magnetic Resonance Imaging (MRI) is a painless action of scanning the human body to produce cross-sectional, three-dimensional images of the internal structures of the human body using electromagnetic waves. It is used commonly for disease detection, confirmation, and surveillance of disease status in the body. MRI depends on sophisticated mechanisms that respond to shifts in the vector position of protons within the water content in biological tissues (Talo, Baloglu, Yildirim, & Acharya, 2019).

MRI machines employ strong magnetic field that aligns the body protons along the magnetic field direction. Whenever an RF pulse kinda knocks the

protons off their pedestal and causes the net magnetization to begin spinning perpendicular to the magnetic field. When the RF pulse is switched off, the protons return to the longitudinal alignment to the main magnetic field releasing energy. This, the released energy is picked by MRI sensors, and it is used to build the image. Dependence of the time required for protons to re-establish alignment with the magnetic field in addition to the emitted power relies on the tissue type and the molecular surrounding, thus assisting in the determination of one tissue from another.

In MRI, actual data is collected in k-space or Fourier data while conducting scanning session. This data encodes the frequency information regarding the object that is imaged and is produced with the help of magnetic field gradient. Every point in k-space contains the whole image information, but with different frequency precision. The data thus pass through processing before being transformed to the spatial domain to give the final image.

The initial image enhancement process of MRI is the reformation of the image from k-space data. This is usually done by performing a two-dimensional Inverse Fourier Transform, which re-creates data into a Spatial Domain data. This process embeds the frequency information contained in each microform and generates a primary image of specific spatial localization. To achieve a high quality of reconstruction the Nyquist theorem should be obeyed in the collection of k-space data to avoid aliased patterns (artifact arising from undersampled k-space data).

A typical MRI can be subject to artifacts due to the inhomogeneity in the magnetic field, the variations in the receiver coil sensitivity and the like which may cause variations in the intensity distribution of the image. Such things as the bias field are corrected to endeavour to reduce these variations and make the image more uniform. Furthermore, normalization is conducted for the reasons of obtaining intensity invariability for better comparability, which is rather important for understanding of images of the various patients or scans of one patient at different times.

Another form of artifacts is noise; it originates from systematic limitations and thermal impact in MRI data. Such noise can make irrelevant certain details of the image. In order to improve the image quality different restoration techniques, and pre-processing methods are used on the observed images. These are for instance methods like Gaussian filtering, Wavelet shrinkage and the Non-Local Means (NLM). They influence the removal of votes from random noise while retaining essential information, including edges and substructures. These techniques consist of, for instance: Gaussian Filter: This technique makes the image clearer while maintaining overall details by uniformly reducing noise throughout the image. Wavelet Shrinkage: This technique detects and eliminates noise from a picture without compromising small features like edges and subtle shapes. By comparing several aspects of a picture to identify commonalities, Non-Local Means (NLM) can eliminate random noise while maintaining crucial details like edges and substructures. In order to accurately diagnose and identify health issues using MRI scans, these approaches are utilized to enhance the final image's quality so that it is clearer and contains finer details.

An important form of MRI pre-processing is image segmentation that involves partitioning the image into areas of interest for instance, tissues or organs. This step is important in clinical applications like identification of tumor, tissue differentiation and measurement of structures. There are still four most widely used methods namely discrete thresholding, continuous histogram based techniques, region growing and statistical approaches and new and advancing algorithms known as machine learning algorithms.

There is another important concept used in MRI known as image registration whereby image from different modality or from different time point are realigned to match in terms of position. This is especially helpful in cases where multiple scans are made or where MRI is done with other procedures like CT or PET scans. Image-registration process can be accomplished based upon a number of transform types, for example, rigid, affine,

or non-affine transform, based on the requirements of the task.

Those artifacts are motion blur, signal loss due to susceptibility artifacts and Gibbs ringing that affect MRI. These distortions may be due to patient motion, the susceptibility effects from local magnetic fields, or too little sampling. Some motion artifacts include respiratory motion, cardiac motion, and other motions such as head motion; various methods are used to correct these motions such as retrospective motion correction, phase correction, and filtering out of artifacts.

Finally, the MRI yield high contrast and a wide range of brightness levels of the processed images, with options to integrate segmented regions or view functional activity. The post-processing tools can also create 3D representations which provide integrated overviews of the spatial relationships of the structures under investigation.

2.6 Types of Brain MR Images

T1-Weighted MRI T1-Weighted MRI is an imaging technique that generates high-resolution pictures of brain anatomy, highlighting distinct contrasts between gray matter, white matter, and cerebrospinal fluid (CSF). Tumors typically appear darker in these images compared to normal tissue (Balafar, Ramli, Saripan, & Mashohor, 2010).

T2-Weighted MRI This highlights water content, causing edema (fluid accumulation) and tumors to appear hyperintense (bright). It is ideal for detecting swelling associated with tumors (K. C. Kim, Jang, & Seo, 2023).

FLAIR (Fluid-Attenuated Inversion Recovery) Reduces the CSF signal to highlight abnormalities like tumors or lesions around the ventricles. Tumors stand out brightly against a darkened CSF backdrop (Amin, Sharif, Haldorai, Yasmin, & Nayak, 2022).

T1ce(T1 Contrast-Enhanced) A T1 scan conducted following the injection of gadolinium-based contrast highlights leaks in the blood-brain bar-

rier, making active tumors appear more luminous (Ronneberger, Fischer, & Brox, 2015).

2.7 Sparsity of MR Images

MRI can therefore be characterized as a sparse recovery problem from both a mathematical and computational perspective. In sparse data structures, the majority of elements in a matrix are zero or small, meaning that most of the information is concentrated in a limited number of significant values. This property can be utilized to optimize MRI procedures, including image compression, image reconstruction, and storage.

While original MRI data is weak in certain domains (e.g., wavelet or gradient domain), the k-space data set itself is generally dense. However, the volume and density of k-space data obtained can be incredibly high, yet much of the information is granular. Some key sparsity aspects include:

- **K-space sparsity:** In the k-space domain, higher-order samples, which are associated with finer image details, generally contain low energy. They contribute less to the image and can, therefore, be considered less valuable in some cases.
- **Wavelet sparsity:** When reconstructing MRI in the wavelet domain, most of the image's energy is found in only a few coefficients, while many other coefficients are either zero or negligible.
- **Gradient sparsity:** In anatomical MRI, the spatial gradients of the image are generally sparse because most tissues in the human body are relatively homogeneous. The boundaries between tissues constitute only a small fraction of the entire image.

These sparsity characteristics are intrinsic to MRI applications, particularly in imaging, reconstruction, and enhancement processes.

2.7.1 Compressed Sensing (CS) in MRI

An MRI technique called **compressed sensing (CS)** is used to generate high-quality images from sparse k-space data. CS significantly reduces the time required for MRI scans by leveraging sparsity and acquiring fewer samples, making the process faster and more efficient. CS techniques use optimization problems that minimize a cost function to reconstruct sparse signals:

$$\min_x \left(\|S - FX\|_2^2 + \lambda \|\Psi x\|_1 \right) \quad (2.10)$$

where:

- x represents the reconstructed image.
- S is the measured k-space data.
- F is the Fourier transform.
- Ψ is the sparsifying transform (e.g., wavelet transform).
- λ is a weighting factor that controls the relative importance of the two terms.
- $\|\cdot\|_1$ refers to the sum of absolute values.
- $\|\cdot\|_2$ refers to the sum of squared values.

2.7.2 Data Compression

Sparse representations are highly effective in compressing MRI data, both in terms of storage and transmission efficiency. The most common approach is **wavelet-based compression**, which retains the most significant image information by examining wavelet coefficients and removing non-significant ones. This allows for significant reductions in file size while maintaining image quality.

2.7.3 Disturbance Suppression and Image Quality

These sparsity-based methods are also applied for **denoising** and **image enhancement** in MRI. The goal is to eliminate dominant noise components while preserving critical image details. One such method is **wavelet thresholding**, which effectively reduces high-frequency noise while keeping the image sparse, allowing important structures to be preserved while minimizing artifacts.

2.7.4 Sparse Dictionary Learning

Another technique used in MRI is **sparse dictionary learning**, which involves determining a set of basic functions (or “atoms”). Using these basis functions, MR images can be reconstructed with fewer coefficients, resulting in a more compact representation. Sparse coding is particularly useful in applications requiring enhanced image quality or specific feature extraction, such as:

- Super-resolution
- Artifact removal
- Image segmentation

Chapter 3

Literature Review

3.1 Definition of Image Segmentation

One of the fundamental procedures in image analysis for object recognition is image segmentation. Image segmentation is defined as dividing an image into distinct parts based on predetermined criteria for subsequent processing which has been used in many image processing and computer vision applications (İşin, Direkoğlu, & Şah, 2016). The objective is to identify homogeneous areas of an image as distinct and behave differently. The concern for locating an object is not important during the segmentation stage. They can later be given labels (Sachin, 2014). Many medical applications require the ability to segment medical images since medical image segmentation algorithms can aid doctors in improving disease diagnosis and creating more effective treatment methods (Lei & Nandi, 2022). One of these is the segmentation of brain images, which is difficult and complex but important in accurately identifying tumors, edema, and necrotic regions. In diagnos-

tic systems, accurate detection of these tissues is essential. Additionally, Magnetic Resonance Imaging (MRI) is a crucial imaging method for early detection of aberrant alterations in many areas of the brain. High contrast images of the brain are frequently obtained by MR imaging. MRI acquisition parameters can be changed to produce distinct grey levels for various forms of neuropathology and different tissues. Contrast in MRI is better than in Computer Tomography (CT). As a result, MR images are used the most often in studies on medical image segmentation (Balafar et al., 2010). The aim of image segmentation is to find a segmentation map S such that:

$$S(\text{input data}) = \text{useful output segmentation.} \quad (3.1)$$

The map S must have the ability to extract important features from the input linked to the desired output.

Figure 3.1 visually presents the multimodal MRI input data and the corresponding ground truth segmentation mask. As depicted, the input consists of three distinct MRI sequences: FLAIR, T1ce, and T2, each offering unique tissue contrast. For instance, the FLAIR image clearly suppresses CSF signal, while T1ce highlights enhancing tumor regions (K. C. Kim et al., 2023). The Mask image on the right illustrates the desired output, where different colors as detailed in Table 3.1 delineate specific tumor sub-regions (Necrotic and Non-Enhancing Tumor, Peritumoral Edema, Enhancing Tumor) and healthy background. This visual representation underscores the complexity of the segmentation task, requiring the model to differentiate subtle intensity variations across modalities to accurately map these distinct anatomical and pathological areas.

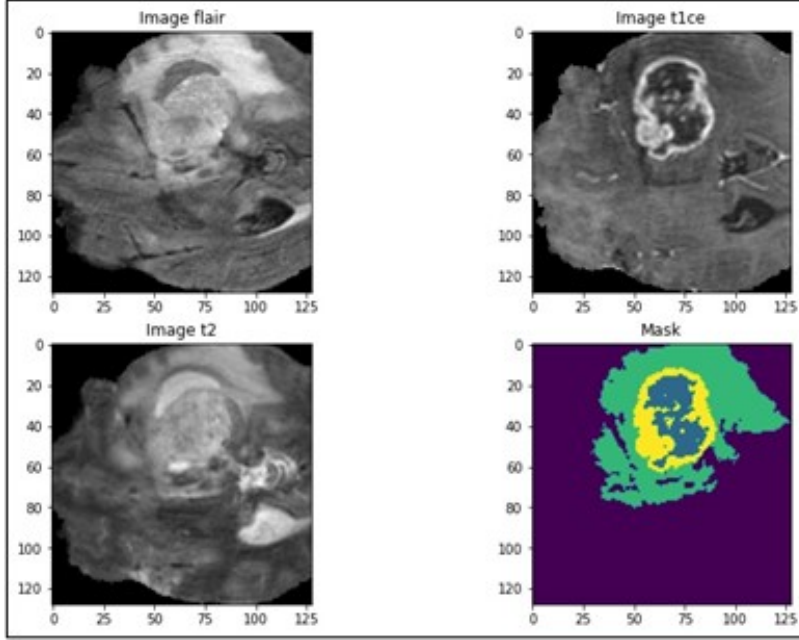


Figure 3.1: The types of input MRI of brain (FLAIR, T1CE, T2) and the output segmentation image (Mask)

Table 3.1: Class Values and Meanings

Class Value	Class Color	Class Meaning
0	Purple	Not tumor (healthy zone or image background)
1	Blue	Necrotic and non-enhancing tumor
2	Green	Peritumoral Edema
3	Yellow	Enhancing Tumor

The notations for image segmentation that we employ in this chapter are as follows:

(Input image B) Let B denote a 2D (or 3D) medical image with pixel grid Δ .

For a 2D image:

$$\Delta = \{X = (x_1, x_2) : x_i \in \{1, 2, \dots, n_{x_i}\}, i = 1, 2\} \quad (3.2)$$

where $B : \Delta \rightarrow \mathbb{R}$ and $M(X) \in \mathbb{R}$ for $X \in \Delta$.

For a 3D image:

$$\Delta = \{X = (x_1, x_2, x_3) : x_i \in \{1, 2, \dots, n_{x_i}\}, i = 1, 2, 3\}. \quad (3.3)$$

where $B : \Delta \rightarrow \mathbb{R}$ and $M(X) \in \mathbb{R}$ for $X \in \Delta$.

(Output S) The output S is a segmentation map derived from B . It assigns a label to each pixel or voxel in Δ :

For binary segmentation, the mapping is defined as:

$$S : \Delta \rightarrow \{0, 1\}$$

In this case, $S(X) = 1$ indicates that the pixel or voxel belongs to the tumor region, while $S(X) = 0$ indicates it does not.

For multi-class segmentation, the mapping is defined as:

$$S : \Delta \rightarrow \{0, 1, 2, 3\}$$

The labels correspond to the following categories:

- 0: Not Tumor (healthy tissue or background)
- 1: Necrotic and Non-Enhancing Tumor (NCR + NET)
- 2: Peritumoral Edema (ED)
- 3: Enhancing Tumor (ET)

the output U is a vector representing classifications probabilities for each pixel/voxel and it defined as:

$$U : \Delta \rightarrow [0, 1]^C$$

where:

C is the number of classes (e.g., $C = 4$ for the multi-class problem). Each component $U_c(X)$ represents the probability that the pixel or voxel X belongs to class c .

3.2 Traditional Approaches for Medical Image Segmentation

Digital image processing and optimization methods were the initial sources of image segmentation. These early algorithms used techniques like region expanding and the snake’s algorithm, which involved setting up initial regions and comparing pixel values to get a notion of the segment map. These techniques viewed an image’s features from a local viewpoint and focused on pixel gradients and local variations. Algorithms that took a global view of the input image came much later on with methods like thresholding methods, Region-Growing method and Active Contour Method being proposed among classical image processing methods.

3.2.1 Thresholding Methods

One of the common image segmentation methods is the threshold method. The threshold segmentation method—typically utilized as a fundamental segmentation technique—can effectively segment images (Wang, Wang, Jiang, Zhang, & Wang, 2016) Thresholding method sets a threshold for categorizing pixels into two classes. When a value is more than the threshold value, the pixel is set to 1, and when it is less than the threshold value, it is set to 0 (K. C. Kim et al., 2023).

The thresholding of B with a threshold value χ , denoted by M_χ , is defined by:

$$M_{\chi} = \begin{cases} 0 & \text{if } M < \chi \\ 1 & \text{otherwise} \end{cases}. \quad (3.4)$$

The threshold segmentation approach is susceptible to noise, though, because it only takes the gradation value of each pixel into account and ignores its spatial features. The threshold segmentation method's calculation speed is slow since it uses a logarithmic operation, making it unsuitable for real-time image analysis (Al-Amri & Kalyankar, 2010). The most challenging and important issue with the threshold segmentation approach is how to establish the threshold value (Wang et al., 2016). One of these methods which used to determine the thresholding value χ is Otsu's thresholding method, named after Nobuyuki Otsu (K. C. Kim et al., 2023).

The Otsu thresholding approach complies with the linear discriminant requirement, which assumes that the image only contains objects foreground and background with ignoring of the heterogeneity and diversity of the background (Yousefi, 2011).

The algorithm of Otsu's method to select the optimal threshold value χ^* is achieved by the following process (Wang et al., 2016).

Let an image B with gray values $1, 2, \dots, m$, total m levels, where the number of pixels with gray value i is n_i . The total number of pixels in an image is as shown in Equation (3.5):

$$N = \sum_{i=1}^m n_i. \quad (3.5)$$

The probability of each gray scale value can be calculated by Equation (3.6):

$$P_i = \frac{n_i}{N} \quad \text{where } i = 1, 2, 3, \dots, m. \quad (3.6)$$

Then, all the gray values of the image are divided into two groups by

using gray level r :

$$Q_0 = \{1, 2, 3, \dots, r\} \quad \text{and} \quad Q_1 = \{r+1, r+2, \dots, m\} \quad (3.7)$$

where r is called the image segmentation threshold. Q_0 is the foreground class which contains all gray values less than pixel r and the total pixels is:

$$S_0 = \sum_{i=1}^r n_i \quad (3.8)$$

And Q_1 is called the background class which contains all the pixel gray values greater than r and the sum of all pixels is:

$$S_1 = \sum_{i=r+1}^m n_i. \quad (3.9)$$

The probabilities of Q_0 and Q_1 respectively ϕ_0 and ϕ_1 are calculated by Equation (3.10):

$$\phi_0 = \frac{S_0}{N} = \sum_{i=1}^r P_i = \phi(r) \quad \text{and} \quad \phi_1 = \frac{S_1}{N} = \sum_{i=r+1}^m P_i = 1 - \phi(r). \quad (3.10)$$

Let μ_0 and μ_1 be the average gray values of Q_0 and Q_1 which describe the cluster centers of the two classes and are calculated by Equation (3.11):

$$\mu_0 = \frac{\sum_{i=1}^r (n_i P_i)}{\phi_0} = \frac{\sum_{i=1}^r (n_i P_i)}{\phi(r)} \quad \text{and} \quad \mu_1 = \frac{\sum_{i=r+1}^m (n_i P_i)}{\phi_1} = \frac{\sum_{i=r+1}^m (n_i P_i)}{(1 - \phi(r))}. \quad (3.11)$$

The average gray value μ of the whole image is calculated by Equation (3.12):

$$\mu = \phi_0 \mu_0 + \phi_1 \mu_1 \quad (3.12)$$

The between-class variance describing class distance can be calculated

by Equation (3.13):

$$\sigma^2(r) = \phi_0(\mu_0 - \mu)^2 + \phi_1(\mu_1 - \mu)^2. \quad (3.13)$$

Substituting Equation (3.12) into Equation (3.13) we obtain:

$$\sigma^2(r) = \phi_0\phi_1(\mu_0 - \mu_1)^2. \quad (3.14)$$

Finally, the optimal threshold value is shown in Equation (3.15):

$$\chi^* = \text{Arg max } \sigma^2(r) \quad \text{where } 0 < r \leq m. \quad (3.15)$$

The best segmentation can be obtained when the maximum threshold value χ^* is used to segment an image.

Figure 3.2 (Amin et al., 2022) provides an example of brain tumor segmentation achieved using the Otsu thresholding method. Figure 3.2(a) displays the original MRI image. In Figure 3.2(b), the result after applying Otsu thresholding is shown. Observe how this method attempts to separate the tumor region from the background based on intensity. While it provides a basic segmentation, it often struggles with the heterogeneous intensity distributions within brain tumors and may not accurately capture irregular tumor shapes or diffuse boundaries, as evident by potential fragmentation or incomplete delineation in the segmented output.

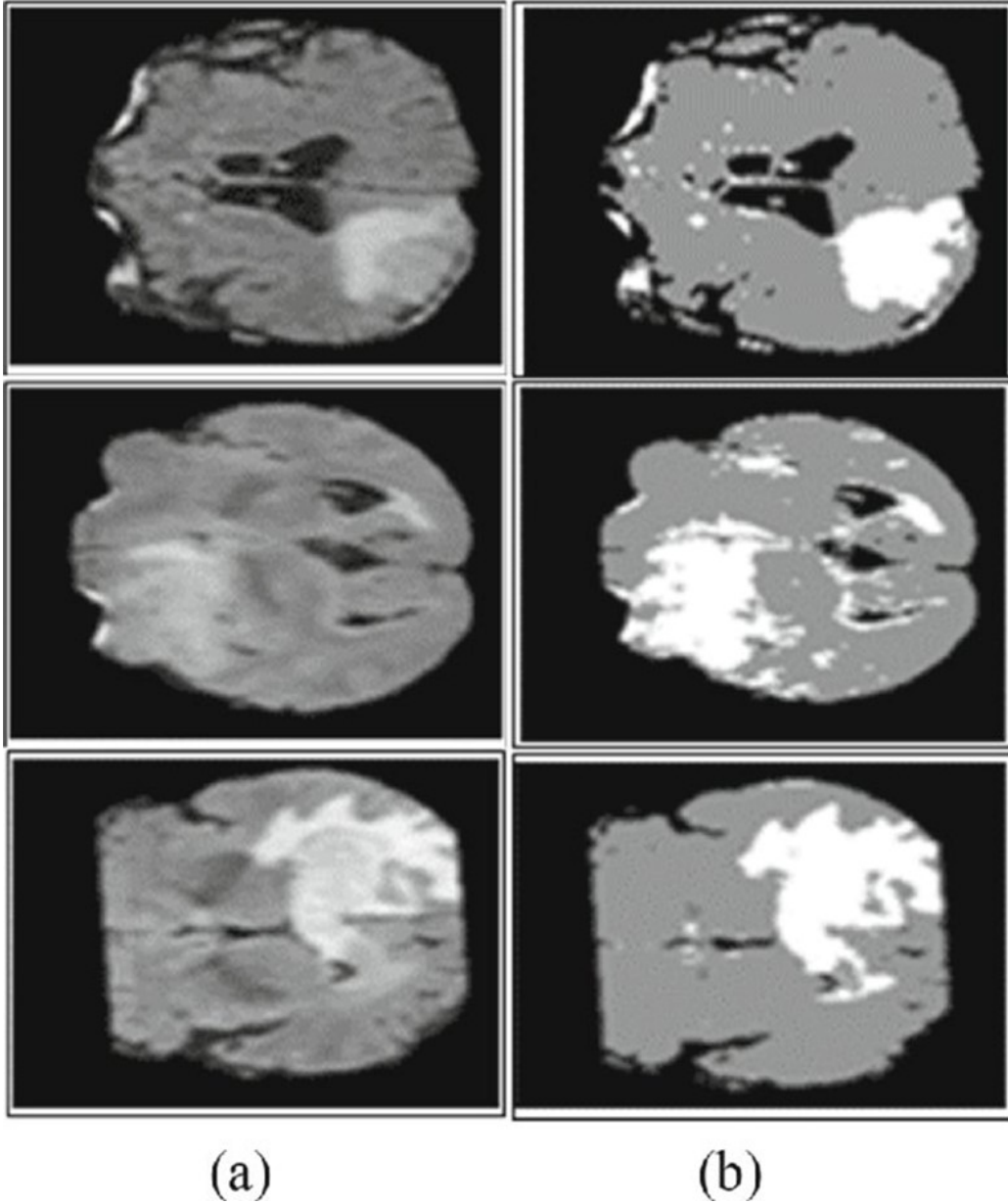


Figure 3.2: (a)original images, (b)Otsu thresholding (Amin et al., 2022).

3.2.2 Region-Growing Method

The region-growing approach, which produces connected regions by growing them from initial seed points, is a pixel-based image segmentation methodology. This technique begins by extracting nearby pixels from the first seed locations by comparing them of each nearby pixel's intensity values with the initial seed point(Balafar et al., 2010). To identify the final re-

gion, this procedure is performed in the same manner. Due to the requirement for manually selecting the initial seed sites, this approach may not be automatic. This method, like thresholding, is sensitive to different elements that affect image quality, such as noise, artifacts, and weak borders. Essentially, this method is local and does not consider the overall structure of images (K. C. Kim et al., 2023). Algorithm 1 contains implementation information for the region-growing approach (Seed Region Growing-SRG) (Shrivastava & Bharti, 2020).

Algorithm 1 SRG

Step 1: Initial seed points are set as $P_1, P_2, \dots, P_n$ that are segmented into $R_1, R_2, \dots, R_n$.

Step 2: Threshold value set as χ .

Step 3: If $|P_i - \text{neighbor pixel value}| \leq \chi$, then the neighboring point could be classified into R_i , where $i = 1, 2, \dots, n$.

Step 4: The new boundary of R_i is evaluated as the new seed points namely P_{is} . The mean pixel intensity values of segment R_i must be recomputed, respectively. Repeat Steps 3 and 4 until the point where every pixel in the image specifies to one of the segments.

Figure 3.3 shows the original image transformed into a pixel matrix as seen in Figure 3.3 (a); three seed points—1, 2, and 7—are then randomly chosen and displayed in a variety of colors in Figure 3.3(b). In this case, zero is chosen at random as the threshold value. The area expands from the original seed sites shown in Figure 3.3(c)–(e). The pixel value 8 does not lie in any of the sections, so the original seed points are no longer sufficient. As a result, the seed points are now 1, 2, 7, and 8, and the ending segments are shown in Figure 3.3 (f) (Shrivastava & Bharti, 2020).

SRG is an effective fast image segmentation method that is helpful for medical images. It can be used in a wide range of contexts, including medical image segmentation can be utilized for a variety of purposes, including the diagnosis of brain tumors from Magnetic Resonance Imaging (MRI), the detection and classification of breast tumors from breast images, and the use of mammograms. It plays a significant role in image analysis, computer vision, geographic image segmentation, object tracking in video surveillance,

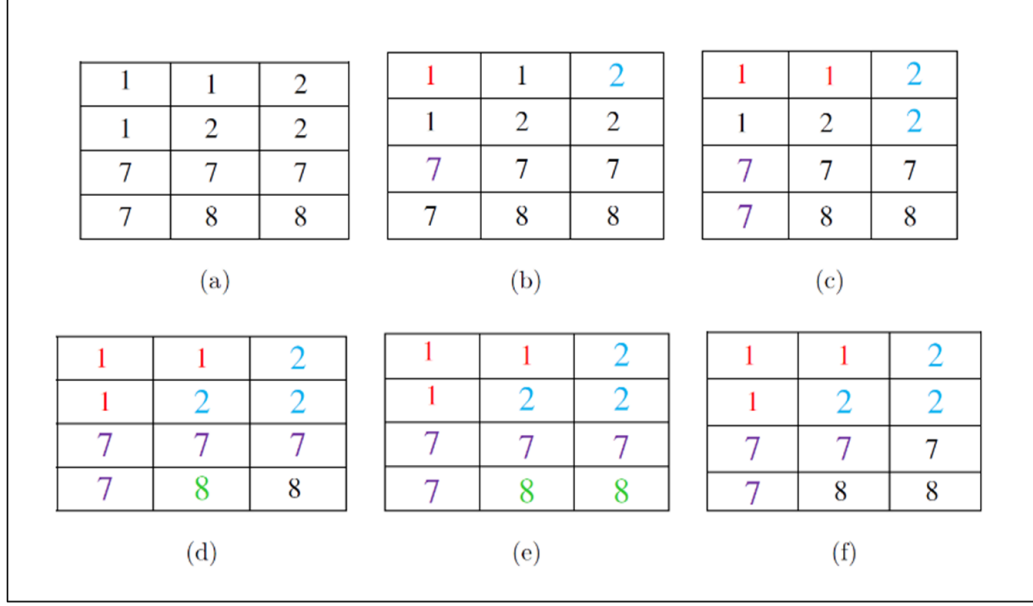


Figure 3.3: (a) original image, (b) three manually selected seed points, and (c–f) region grows with checking neighbor pixels (Shrivastava & Bharti, 2020).

and other fields. Figure 3.4 illustrates an application of the Seed Region Growing (SRG) approach for brain tumor detection in MRI (Biratu et al., 2021). Figures 3.4(a,c,e,g) show skull-removed original brain images with manually or semi-automatically generated potential seed points. The corresponding Figures 3.4(b,d,f,h) display the final segmentation outputs. Visually, the SRG method, guided by these seed points, expands to delineate tumor regions. However, as discussed, the quality of these segmentations is highly dependent on the initial seed point selection and threshold values, which can be a limitation for fully automated and consistent results across diverse tumor presentations.

However, the selection of the initial seed through trial-and-error method does not deliver the expected results. The results are exhibited in a wrong way if a seed point falls on a noisy point, or the user makes an error in the choice of seed points. Different seed points and threshold values create different segmentation results. So, an algorithm is required that gives efficiently the defined automatic selection of seed points and threshold values (Shrivastava & Bharti, 2020).

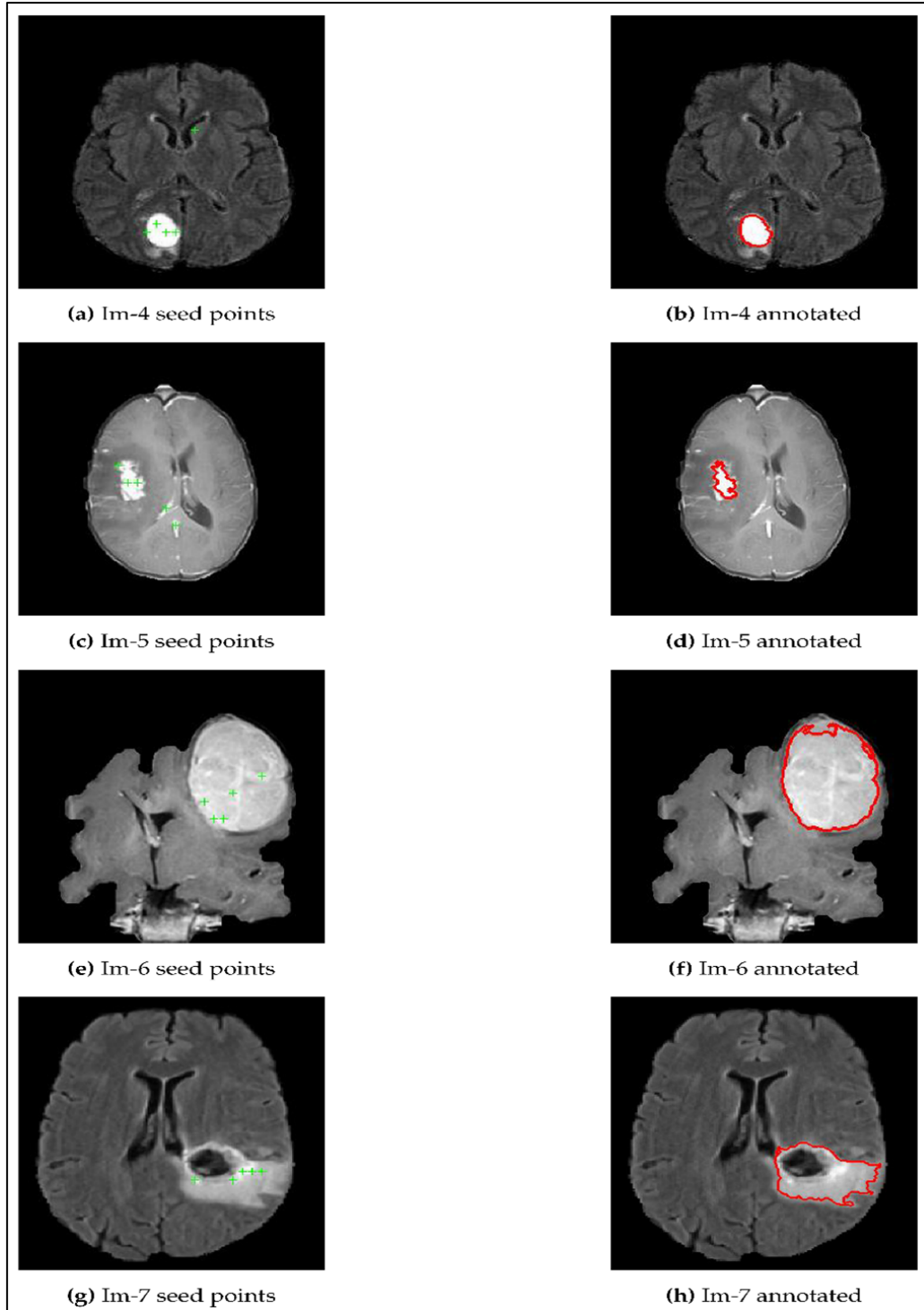


Figure 3.4: Generated possible seed points and annotations using the proposed approach. (a,c,e,g) represent skull-removed original brain images with five potential seed points; (b,d,f,h) show the final segmentation output of each respective (Biratu et al., 2021).

3.2.3 Active Contour Method

Deformable models are an energy-based segmentation technique that drives a closed curve specified within an image area G toward the target segmentation by minimizing an energy comprising of internal energy defined on the curve and external energy from the surrounding area of an image B (K. C. Kim et al., 2023).

Several application-specific energy functionals have been created to form an active contour in the direction of the target region's boundary. The purpose of the energy functional is to find a local minimum by a process of regular shape development that minimizes energy.

Therefore, to end the iterative procedure at the desired boundary, a sufficient initial contour and ending criterion are needed. The typical format of energy-based segmentation methods is as follows, where $\mathcal{P}$ is the contour:

$$E = E_{\text{internal}} + E_{\text{external}}. \quad (3.16)$$

$$\mathcal{P} = \arg \min_{\mathcal{P}} (\text{Fit}(\mathcal{P}) + \lambda \text{Reg}(\mathcal{P})) \quad (3.17)$$

Where $\text{Fit}(\mathcal{P})$ is a fitting term for attracting the contour $\mathcal{P}$ toward the target object in the image, while $\text{Reg}(\mathcal{P})$ is a regularization term to reflect prior knowledge of contour structure, and λ is a tunable fitting parameter that controls the balance between these two terms.

The active contour method (or snakes) is a type of method that uses an energy functional $\Phi(\mathcal{P})$ to evolve an active contour $\mathcal{P}$ toward the boundary of the target region. For a medical image $B(x, y)$, our goal for segmentation is to find a closed curve $\mathcal{P}$ that minimizes the function $\Phi(\mathcal{P})$, which is expressed as follows:

$$\Phi(\mathcal{P}) = \text{Fit}(\mathcal{P}) + \lambda \text{Reg}(\mathcal{P}). \quad (3.18)$$

We start with the simplest form of the energy function $\Phi(\mathcal{P})$:

$$\Phi(\mathcal{P}) := \int_{\mathcal{P}} \frac{1}{1 + |\nabla B|^2} ds. \quad (3.19)$$

For the computation of a local minimum $\mathcal{P}$ of the functional, we may start from an initial contour $\mathcal{P}^0$ and get a minimizing sequence $\mathcal{P}^1, \mathcal{P}^2, \dots$ which converges to the local minimum $\mathcal{P}$:

$$\Phi(\mathcal{P}^0) \geq \Phi(\mathcal{P}^1) \geq \Phi(\mathcal{P}^2) \geq \dots \geq \Phi(\mathcal{P}^n) \geq \Phi(\mathcal{P}^{n+1}) \geq \Phi(\mathcal{P}). \quad (3.20)$$

To calculate the next contour $\mathcal{P}^{(n+1)}$ from $\mathcal{P}^n$, the gradient descent method based on $\nabla\Phi(\mathcal{P}^n)$ is widely used:

$$\mathcal{P}^{(n+1)} = \mathcal{P}^n - \alpha \nabla\Phi(\mathcal{P}^n) \quad (3.21)$$

where $-\alpha \nabla\Phi(\mathcal{P}^n)$ is the steepest descent direction.

What does $\nabla\Phi(\mathcal{P}^n)$ mean? It is more convenient to explain $\mathcal{P}^n$ using the time-varying $\mathcal{P}^t$ rather than the sequence $\mathcal{P}^n$. Then the Equation (3.21) can be expressed as:

$$\mathcal{P}^{(t+\Delta t)} = \mathcal{P}^t - \alpha \Delta t \nabla\Phi(\mathcal{P}^t). \quad (3.22)$$

To calculate $\mathcal{P}^{(t+\Delta t)}$, we parameterize $\mathcal{P}^t$ by

$$r(s, t) = x(s, t)(1, 0) + y(s, t)(0, 1) \quad \text{where} \quad 0 < s < 1. \quad (3.23)$$

In addition, Equation (3.23) represents a continuous parametric curve evolving over time, where (x,y) define boundary points.

Thus, we have:

$$\mathcal{P}^t = \{r(s, t) = x(s, t)(1, 0) + y(s, t)(0, 1) \mid 0 < s < 1\}. \quad (3.24)$$

Setting $\Phi(t) = \Phi(\mathcal{P}^t)$, we have:

$$\Phi(t) := \int_{\mathcal{P}^t} \frac{1}{1 + |\nabla B|^2} ds = \int_0^1 g_B(r(s, t)) |r_s(s, t)| ds \quad (3.25)$$

where $g_B(r) = \frac{1}{1 + |\nabla B|^2}$.

3.3 Deep Learning for Medical Image Segmentation

3.3.1 Introduction

Segmentation of Medical images is a crucial component of a diagnosis, therapy development, and research. Most traditional segmentation approaches are expensive to memory and time. Also, some of these techniques fail in producing the boundary of the images. This reason is that it allows for missing and disjointed edges in separating the area of interest and background. In addition, most of these techniques are complex and don't overlook spatial details, which it is important in medical Image segmentation (Abdulateef & Salman, 2021). Currently, there are several medical imaging methods that are improved in computer vision especially for tumors locating such as MRI, Computer Tomography (CT scan) and Positron Emission Tomography (PET screening)(Lei & Nandi, 2022). Due to the complexity of the brain organ, MRI suffers from a tumor localization in images. Besides, MR images may contain noise or with blur(Aloysius & Geetha, 2017) . Deep learning has emerged as a leading machine learning technique due to its remarkable suc-

cesses across various domains. deep learning significantly enhances segmentation tasks through its generalization abilities, depth of architecture, handling of high-dimensional information, and efficient optimization strategies. This leads to improved accuracy and efficiency in practical applications like medical imaging and object detection in various domains(Berner, Grohs, Kutyniok, & Petersen, 2021). One of these deep learning methods is convolutional neural network (CNN) which is successfully achieving hierarchical feature representation of images, making it one of the most prominent research topics in image processing and computer vision(Lei & Nandi, 2022). Since convolutional neural networks (CNNs) used for feature learning tend to be resilient to image noise, blur, contrast variations, and similar distortions, they deliver highly effective segmentation results for medical images.

3.3.2 Convolutional Neural Networks (CNNs)

The three fundamental principles of convolutional neural networks (CNNs) are local receptive fields, parameter sharing, and equivariant representations (Berner et al., 2021). In this section, the mathematical formulation of convolutional neural networks is presented as described in (Berner et al., 2021).

A convolutional neural network is composed of multiple convolutional blocks, which are specialized layers designed to effectively extract features from input data, particularly in high-dimensional spaces like images. For a group G , which is typically either $[I] \cong \mathbb{Z}/(d\mathbb{Z})$ or $[I]^2 \cong (\mathbb{Z}/(d\mathbb{Z}))^2$ for $I \in \mathbb{N}$, depending on whether we are dealing with one-dimensional or two-dimensional data, where $[I]$ and $[I]^2$ are cyclic groups. The convolution of two vectors $f, g \in \mathbb{R}^G$ is defined as:

$$(f * g)_i = \sum_{j \in G} f_j g_{j-1}, \quad i \in G. \quad (3.26)$$

The definition of a convolutional block is as follows: let $\tilde{G}$ be a subgroup

of G , p be the pooling operator defined as $p : G \rightarrow \tilde{G}$, and C be the number of channels. Then, for a sequence of kernels $h_i \in \mathbb{R}^G$, where $i \in [C]$, the output of a convolutional block is given by:

$$\mathbb{R}^G \ni x \mapsto \tilde{x} := (p(x * h_i))_{i=1}^C \in (\mathbb{R}^{\tilde{G}})^C \quad (3.27)$$

where x represents the pixel values.

These operations subsequently either calculate the average or determine the maximum across patches of comparable size. The convolutional kernels align with the previously mentioned receptive fields and can be considered localized when they have small supports, meaning only a few nonzero entries. However, Convolutional Neural Networks (CNNs) have recently gained significant attention in the field of medical image analysis. Segmentation plays a crucial role in medical applications, including brain segmentation, body organ segmentation, and cardiac ventricle segmentation (de Brebisson & Montana, 2015). Brain image segmentation through deep learning employed a convolutional neural network (CNN) model specifically to segment brain tumors, particularly gliomas. The deep learning CNN algorithm demonstrated greater efficiency compared to traditional machine learning approaches, especially in scenarios with larger datasets, achieving higher accuracy levels. However, Convolutional Neural Network continues to face the following limitations (Jiang, Wang, Liu, Li, & Liu, 2019): First, when employing a standard multi-layer perception, in which all layers are fully connected, (CNNs) can encounter significant computational challenges. This is primarily due to the high dimensionality of image data, which imposes substantial demands on processing and resource efficiency. Second, CNNs lack available information regarding spatial dimensions. Utilizing the pooling layer can lead to significantly reduced efficiency. This is because only the most active neurons can be relayed between layers during neuron transfer, resulting in a considerable loss of information within the pooling layer and diminishing spatial resolution. Consequently, CNNs may

struggle to differentiate between variations in postures and other characteristics. A potential solution to this issue is extensive training across all imaginable angles, although this approach typically demands additional time and computational resources. Third, traditional convolutional neural networks struggle to effectively identify highly complex field-of-view images characterized by significant overlap, mutual masking, and varying backgrounds.

Due to these constraints, the accuracy of its classification results is insufficient. Fully Convolutional Networks (FCNs) are proposed (Qamar, Jin, Zheng, Ahmad, & Usama, 2020). Fully Convolutional Networks (FCNs) are deep learning models primarily utilized for pixel-level classification tasks instead of from image-level classification as in CNN. That means rather than predicting a single label for the entire image, FCNs forecast a label for each pixel in the image. Also, (CNNs) typically end with fully connected layers, which require a fixed input size while FCNs replace these with convolutional layers, allowing them to handle inputs of varying sizes. In addition, (FCNs) integrate deconvolution, also referred to as up sampling layers, enabling the reconstruction of the spatial dimensions of the input image after it has undergone processing in the convolutional layers. This approach facilitates the mapping of features extracted at various hierarchical levels back to their original spatial configuration. They frequently utilize skip connections to connect layers in the encoder, which focuses on feature extraction, with those in the decoder, responsible for up sampling. This approach merges low-level features that retain fine details with high-level features that provide contextual understanding, leading to more precise pixel-level predictions. Fully Convolutional Networks (FCNs) address the loss of information and detail often caused by pooling operations in Convolutional Neural Networks (CNNs), enhancing both accuracy and output relevance in tasks like semantic segmentation (Jiang et al., 2019). Fundamentally, FCNs address the challenges of pixel-level classification and contextual understanding in images through a fully convolutional architecture.

They leverage deconvolution for efficient up sampling and utilize skip connections to improve feature representation. This approach enables FCNs to overcome the limitations of traditional CNNs, where pooling-induced information loss often leads to less precise predictions in tasks demanding high resolution and intricate detail, like image segmentation (Qamar et al., 2020).

Definition 1 (FC Neural Network): Let B be an input space and S be an output space. A fully connected neural network (FCNN) is a function $F : B \rightarrow S$ composed of a sequence of L layers, where each layer $l = 1, 2, \dots, L$ transforms an input $\mathbf{x}^{(l-1)} \in \mathbb{R}^{n_{l-1}}$ from the previous layer into an output $\mathbf{x}^l \in \mathbb{R}^{n_l}$ via a linear transformation followed by a nonlinear activation function. Mathematically, we have:

$$\mathbf{x}^l = \phi_l \left(\mathbf{W}^{(l)} \mathbf{x}^{(l-1)} + \mathbf{b}^{(l)} \right) \quad (3.28)$$

where: $\mathbf{W}^{(l)} \in \mathbb{R}^{n_l \times n_{l-1}}$ represents the weight matrix for layer l , $\mathbf{b}^{(l)} \in \mathbb{R}^{n_l}$ denotes the bias vector for layer l , ϕ_l is a nonlinear activation function (e.g., ReLU, sigmoid, etc.), and $\mathbf{x}^{(0)}$ is the initial input to the network.

Given that convolution is a linear operation, depending on the chosen pooling method, a convolutional block, as previously described, can be reformulated as a fully connected neural network (FCNN). Thus, the convolutional block is defined as:

$$\mathbf{x} \mapsto \mathbf{W}^{(l)} \mathbf{x}^{(l)} + \mathbf{b}^{(l)}. \quad (3.29)$$

Figure 3.5 provides a conceptual illustration of a fully connected neural network incorporating two-dimensional convolutional blocks. The diagram depicts layers of interconnected nodes, where each node in a layer is connected to every node in the subsequent layer, characteristic of a fully connected network. Within this structure, the convolutional blocks (represented by distinct colored sections) perform feature extraction, while the subsequent 2×2 subsampling (pooling) operations are visually indicated by a reduction in the spatial dimensions of the feature maps, demonstrating

how the network progressively learns more abstract representations from the input data (Berner et al., 2021).

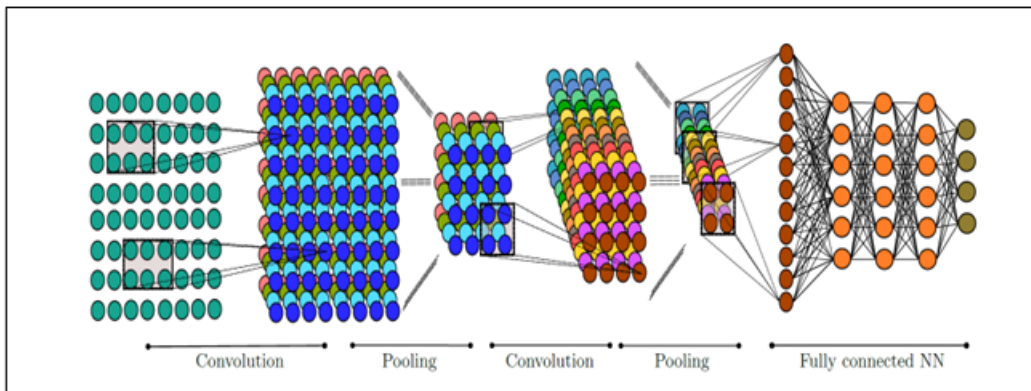


Figure 3.5: featuring two-dimensional convolutional blocks paired with 2×2 subsampling used as the pooling (Berner et al., 2021).

3.3.2.1 Residual Neural Networks

In a standard fully connected neural network (NN), the transformation from one layer to the next is mathematically expressed as:

$$\Phi^{(l)}(x, \theta) = f^{(l)}(\mathbf{W}^{(l)}x + \mathbf{b}^{(l)}) \quad (3.30)$$

where $\mathbf{W}^{(l)}$ and $\mathbf{b}^{(l)}$ are the weights and biases for the l -th layer, and $f^{(l)}$ represents the activation function.

In contrast, a Residual Neural Network (ResNet) modifies this structure by incorporating the input of a residual block into its output. Specifically, the output from a residual block is defined as:

$$\bar{\Phi}^{(l)}(x, \theta) = f^{(l)}(\mathbf{W}^{(l)}x + \mathbf{b}^{(l)}) + \bar{\Phi}^{(l-1)}(x, \theta). \quad (3.31)$$

where the term $\bar{\Phi}^{(l-1)}(x, \theta)$ corresponds to the output of the preceding layer (or the original input in the case of the first layer). This approach, which introduces a "shortcut" or "skip connection" facilitates gradient flow across layers during training. By mitigating vanishing gradient issues, this design enables more effective training in deep networks (Biratu et al., 2021).

3.3.2.2 Batch Normalization

Batch normalization (BN) is an essential method for enhancing the training efficiency and overall performance of deep neural networks. It achieves this by mitigating challenges caused by internal co variate shift (Berner et al., 2021).

Batch normalization is applied to the output of a layer before its activation function. For a mini-batch of inputs $b = (y^{(i)})_{i=1}^m \in \mathbb{R}^n$, where m represents the batch size and $y^{(i)}$ corresponds to the outputs from the preceding layer, the batch normalization process involves the following steps:

1. Compute the Mean and Variance For the given mini-batch, the mean

μ_b and variance σ_b^2 are calculated as:

$$\mu_b = \frac{1}{m} \sum_{i=1}^m y^{(i)}. \quad (3.32)$$

$$\sigma_b^2 = \frac{1}{m} \sum_{i=1}^m (y^{(i)} - \mu_b)^2. \quad (3.33)$$

2. Normalization The next step is to normalize each input $y^{(i)}$ using the computed mean and variance:

$$\hat{y}^{(i)} = \frac{y^{(i)} - \mu_b}{\sqrt{\sigma_b^2 + \varepsilon}} \quad (3.34)$$

where ε denotes a small constant added to avoid division by zero.

3. Scale and Shift Finally, the normalized values are scaled and shifted using learnable parameters γ (scale factor) and β (shift factor):

$$BN_{\beta, \gamma}(y^{(i)}) = \gamma \hat{y}^{(i)} + \beta. \quad (3.35)$$

Both β and γ are trainable parameters that adapt during model training.

Batch normalization normalizes the inputs of each layer by calculating the mini-batch mean and variance, followed by scaling and shifting the normalized outputs using learnable parameters. This process helps stabilize learning, making optimization more efficient and smoother. As a result, it has become a staple in many deep learning architectures due to its profound influence on training dynamics (Biratu et al., 2021).

3.3.2.3 The Encoder-Decoder Structure

The encoder-decoder structure based on deep learning is applied due to its superior performance. Notable methods such as U-Net, and V-Net are prominent examples of FCN. In these architectures, the encoder typically

focuses on extracting image features, while the decoder works to reconstruct these features back to the original image dimensions, ultimately producing the final segmentation output (Teow, 2021).

To understand the convolutional autoencoder algorithms' autoencoding process, this section introduces a mathematical abstraction to model the autoencoding computation in a convolutional autoencoder image segmentation (Teow, 2021).

Suppose the convolutional autoencoder A_ω is under a complete generative feature learning function designed for image classification, denoising, and localization, and consists of two computational networks: the encoder network A_ϕ and the decoder network A_ψ .

The autoencoder A_ω is trained to segment from its input image B to reconstruct a learned approximated clean image at its output $\tilde{B}$.

Figure 3.6 visually outlines the structure of a convolutional autoencoder, denoted as A_ω . The diagram clearly shows two main components: the encoder network A_ϕ on the left and the decoder network A_ψ on the right. The encoder path is depicted as progressively reducing the spatial dimensions of the input image B while increasing feature complexity, leading to a compact latent space Z in the middle. Conversely, the decoder path is shown reconstructing the image from this latent space, gradually increasing spatial dimensions to produce the learned approximated clean image $\tilde{B}$ at the output. This bottleneck architecture is fundamental to autoencoders for learning efficient data representations.

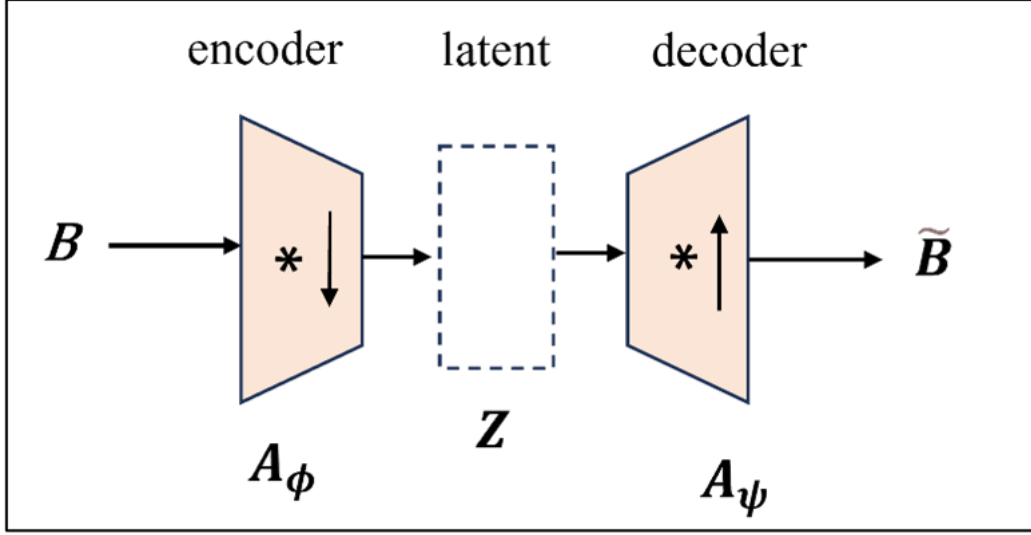


Figure 3.6: The convolutional autoencoder's structure

At the beginning, the encoder network A_ϕ processes the high-dimensional input space B , which consists of unclear images b_n , where $B = \{b_1, \dots, b_N\}, b_N \in \mathbb{R}^D$, into a low-dimensional latent space Z that encapsulates compact latent variables z_n of B , where $Z = \{z_1, \dots, z_N\}, z_n \in \mathbb{R}^M$. In addition, the mapping of A_ϕ is given by:

$$A_\phi : X \longrightarrow Z. \quad (3.36)$$

The network A_ϕ is designed in a feedforward ladder structure, utilizing the convolution $*$ and dyadic decimation $\{\downarrow\}$ operators. The convolution operator $*$ is employed for visual feature extraction and is mathematically defined as a two-dimensional discrete integral. This involves reflecting, shifting, and multiplying the input B by a set of convolution kernels $k = \{k_1, \dots, k_N\}$ to generate the convolution output $Y = \{y_1, \dots, y_N\}$, where $y_n \in \mathbb{R}^D$. The operation $y = k * b$ is performed with the assumption that a rectified linear activation function (ReLU) is applied immediately after the convolution step (Alias, Shah, Abdullah, & Khalid, 2020), which can be expressed as:

$$y(n_1, n_2) = \sum_{h_1} \sum_{h_2} k(h_1, h_2) b(n_1 - h_1, n_2 - h_2) \quad (3.37)$$

where h_1, h_2 defines the position of a kernel (filter) that is applied to the image and n_1, n_2 are the coordinates of the output pixel on the feature map y . The filter slides across the image, and each position is defined by h_1 and h_2 . The output of the convolution is the sum of all the filtered outputs at all positions.

The dyadic decimation operator $\{\downarrow\}$ reduces the sampling rate of y by a factor of two, as follows:

Let

$$y = \begin{bmatrix} y(0) \\ y(1) \\ y(2) \\ y(3) \\ y(4) \\ \vdots \end{bmatrix} \quad (3.38)$$

then

$$\lfloor \downarrow y = \begin{bmatrix} y(0) \\ y(2) \\ y(4) \\ \vdots \end{bmatrix}, \quad (3.39)$$

Equation (3.39) serves as a method for dimensionality reduction, enabling the generation of the latent space Z .

The latent space Z is a low-dimensional feature space that encapsulates a high-level compact learned representation of the input space B . It achieves this by compressing the high-dimensional input space into a smaller space, retaining essential information and discarding noise. This compression pro-

cess reduces the dimensionality of the data, simplifying its structure and making it easier to process and analyze. It acts as a compact representation of the input data, capturing key characteristics and relationships. This representation can then be used for various tasks like image denoising, image reconstruction, or other machine learning applications.

In addition, the process of learning this latent representation, driven by A_ϕ , involves examining B and identifying a set of underlying conceptual patterns that can approximate the input data as closely as possible through A_ψ .

At its conclusion, the decoder network A_ψ produces the learned high-dimensional output space $\tilde{B}$, which contains clear images $\tilde{b}$ derived from Z . This output space is represented as

$$\tilde{B} = \{\tilde{b}_1, \dots, \tilde{b}_N\}, \quad \text{where } \tilde{b}_n \in \mathbb{R}^D. \quad (3.40)$$

The mathematical definition of A_ψ involves generating $\tilde{B}$ from Z and is expressed as:

$$A_\psi : Z \longrightarrow \tilde{B} \quad (3.41)$$

The network A_ψ is designed using the convolution operator $*$ and the dyadic interpolation operator $\{\uparrow\}$ within a feedforward ladder structure. The convolution operator $*$ is defined by Equation (3.26). Based on Equation (3.39), the dyadic interpolation operator $\{\uparrow\}$ performs upsampling of $\{\downarrow\}z$ by a factor of two, under the assumption that $\{\downarrow\}z$ corresponds one-to-one with $\{\downarrow\}y$ in their respective sampling subspaces. This process enables dimensionality expansion to generate $\tilde{B}$ by transforming the convoluted Z as follows:

$$\{\uparrow\}\{\downarrow\}z = \begin{bmatrix} z(0) \\ 0 \\ z(2) \\ 0 \\ z(4) \\ \vdots \end{bmatrix}. \quad (3.42)$$

To clarify Equation (3.42), we should point out Zero Padding, which means the formula starts with a vector of Z , where the first and last elements are 0. This is because the dyadic interpolation operator effectively adds zeroes before and after the original values.

Then, in the upsampling stage, the $\{\downarrow\}$ operator doubles the number of elements in the vector. Since the original vector was padded with zeros, the new elements are also zeros. This results in the repeated appearance of the initial $z(0)$ and the newly added 0.

In the end, we arrive at a convolutional autoencoder model designed for clear images, represented as:

$$A_{\omega} : B \longrightarrow \tilde{B}. \quad (3.43)$$

Both B and $\tilde{B}$ are bounded, finite, and equal in their spatial dimensions.

3.3.2.4 The Theoretical Framework of Simple 3D U-Net Model

The U-Net architecture, originally developed by (Ronneberger et al., 2015) for applications in biomedical image segmentation, incorporates a total of 23 convolutional layers, as detailed in reference (Ronneberger et al., 2015). This model is structured into two primary components: the encoder and the decoder. The encoder is responsible for feature extraction from medical images, employing convolutional layers, Rectified Linear Unit (ReLU) activation functions, and pooling operations. In contrast, the decoder reconstructs and refines the extracted features through up sampling layers. This

reconstruction process facilitates precise localization of features within the image, thereby enabling highly accurate and finely detailed segmentation, as outlined in (Ronneberger et al., 2015). This study introduces the 3D U-Net model, a powerful deep learning architecture specifically designed for brain tumor MRI segmentation. Unlike conventional 2D convolutional neural networks, the 3D U-Net operates on three-dimensional data, making it particularly effective for applications such as MRI or CT scan analysis. By capturing spatial features across height, width, and depth simultaneously, the model significantly improves the detection of complex structures within medical imaging data like brain (Ateş, Günlü, Ekinçi, & Garip, 2024). In this study, we are using a 3D U-Net architecture specifically designed for the segmentation of brain tumors in magnetic resonance imaging (MRI) data. The model is built upon the original U-Net framework, extending its capabilities to handle volumetric data characteristic of medical imaging. Our 3D U-Net comprises a total of 23 convolutional layers organized into an encoder-decoder structure that efficiently captures spatial features across three dimensions. The encoder path is constructed with multiple convolutional blocks that perform feature extraction at increasing levels of abstraction. Each block consists of two 3D convolutional layers followed by a dropout layer to mitigate overfitting, with max pooling operations that progressively downsample the feature maps. The number of filters in each block increases (starting from 16 and going up to 256) to capture more complex features as the resolution decreases, establishing a robust hierarchical representation of the input data. Transitioning into the bottleneck layer, the model maximizes feature comprehension at its deepest point, where the most complex features are identified. The decoder path employs transposed convolutions for upsampling, reinstating the spatial dimensions of the feature maps while simultaneously concatenating features from the corresponding encoder blocks. This skip connection mechanism retains crucial spatial information, facilitating accurate localization in the final segmentation out-

put. The model's output is achieved through a $1 \times 1 \times 1$ convolutional layer, which maps the learned features to the desired output classes, applying a SoftMax activation function to generate class probabilities for each voxel, thus enabling detailed segmentation of brain tumors. The Simple 3D U-Net model can be described mathematically through a series of operations that define its architecture. For the input layer, let $B \in \mathbb{R}^{H \times W \times D \times C}$ where: - H is the height of the input volume, - W is the width of the input volume, - D is the depth of the input volume, and - C is the number of channels.

The contraction path consists of several blocks, each containing convolutional layers followed by max pooling. The operations can be represented as follows:

Let O_i be the block for the encoder structure for $i = 1, 2, 3, 4$. The operations in each block are represented as follows:

1. For the first convolutional layer:

$$L_i = \text{ReLU}(W_i * B_{i-1} + b_i) \quad (3.44)$$

where $*$ denotes the 3D convolution operation, W_i are the weights, B_{i-1} is the input feature map from the previous layer and b_i are the biases.

2. For Dropout:

$$D_i = \text{Dropout}(L_i, p) \quad (3.45)$$

where p is the dropout rate.

3. For the second convolutional layer:

$$L_{i,2} = \text{ReLU}(W_{i,2} * D_i + b_{i,2}) \quad (3.46)$$

where $W_{i,2}$ and $b_{i,2}$ are the weights and biases for the second 3D convolutional layer in block i . 4. For the last operation in this block, max pooling is defined as:

$$P_i = \text{MaxPool}(L_{i,2}, (2, 2, 2)) \quad (3.47)$$

where (2, 2, 2) is the pooling window size and stride for max pooling.

The bottleneck layer is the deepest part of the network, expressed as:

$$L_5 = \text{ReLU}(W_5 * P_4 + b_5) \quad (3.48)$$

where W_5 and b_5 are the weights and biases for the first 3D convolutional layer in the bottleneck.

$$D_5 = \text{Dropout}(L_5, 0.3) \quad (3.49)$$

where 0.3 is the dropout rate.

$$L_{5,2} = \text{ReLU}(W_{5,2} * D_5 + b_{5,2}) \quad (3.50)$$

where $W_{5,2}$ and $b_{5,2}$ are the weights and biases for the second 3D convolutional layer in the bottleneck.

The decoder consists of upsampling and concatenation with corresponding feature maps from the contraction path for block O_j for $j = 6, 7, 8, 9$:

$$O_j = \text{Conv3DTranspose}(C_{j-1}, W_j, \text{stride} = (2, 2, 2)) \quad (3.51)$$

where C_{j-1} is the input feature map to the transposed convolutional layer from the previous block, W_j represents the kernel weights for the 3D transposed convolution and (2,2,2) is the stride for the transposed convolution, effectively doubling spatial dimensions.

For concatenation:

$$C_j = \text{concat}(O_j, C_{5-j}). \quad (3.52)$$

However, C_{5-j} is the corresponding feature map from the encoder path (skip connection) that is concatenated.

3.3.2.5 The Theoretical Framework of V-Net Model

V-Net is a type of fully convolutional neural network (FCN) specifically designed for volumetric medical image segmentation, particularly in the context of MRI scans. The Architecture of V-Net as proposed in (Milletari, Navab, & Ahmadi, 2016) employs a residual network structure, where convolutional operations replace traditional pooling layers as in 3D U-Net. This approach helps to reduce the dimensionality of the feature maps while doubling the number of channels at each stage, which enhances the network's capacity to capture intricate details in the images.

The used V-Net model consists of 23 convolutional layers arranged in a hierarchical structure that includes both contracting and expanding paths. Each convolutional block is composed of two 3D convolutional layers followed by batch normalization and ReLU activation functions, allowing the model to learn a rich set of features at multiple scales. Architecture incorporates residual connections to enhance gradient flow during training and strengthen the model's ability to learn complex features without degradation. The core of the V-Net is the Convolutional Blocks function, which encapsulates the convolutional operation along with batch normalization (BN) to stabilize the learning process. The usage of ReLU activation facilitates the model's non-linear learning capabilities. Additionally, dropout layers are integrated into the model at various points to mitigate the risk of overfitting, particularly in scenarios where training data may be limited.

The architecture of V-Net, as proposed in (Milletari et al., 2016), employs a residual network structure, where convolutional operations replace traditional pooling layers as in 3D U-Net. The V-Net model can be described mathematically through a series of operations that define its architecture. For the input layer, let $B \in \mathbb{R}^{H \times W \times D \times C}$ where: - H is the height of the input volume, - W is the width of the input volume, - D is the depth of the input volume, and - C is the number of channels.

The contraction path consists of several blocks, each containing convo-

lutional layers followed by batch normalization, residual connections, and dropout. The operations can be represented as follows:

Let O_i be the block for the encoder structure for $i = 1, 2, 3, 4$. The operations in each block are represented as follows:

1. For the first convolutional layer:

$$L_i = \text{ReLU}(\text{BatchNorm}(W_i * B_{i-1} + b_i)) \quad (3.53)$$

where $*$ denotes the 3D convolution operation, W_i are the weights, and b_i are the biases.

2. For Dropout:

$$D_i = \text{Dropout}(L_i, p) \quad (3.54)$$

where p is the dropout rate.

3. For the second convolutional layer with residual connection:

$$L_{i,2} = \text{ReLU}(\text{BatchNorm}(W_{i,2} * D_i + b_{i,2})) + L_i. \quad (3.55)$$

where L_i is the output of the first convolutional layer in block i , which forms the residual connection.

4. The traditional pooling layers are replaced by convolutional operations, which can be expressed as:

$$P_i = \text{Conv}(L_{i,2}, W_{pool}, \text{stride} = (2, 2, 2)). \quad (3.56)$$

However, W_{pool} represents the kernel weights for the 3D convolutional operation used for downsampling (pooling). The bottleneck layer is the deepest part of the network, expressed as:

$$L_5 = \text{ReLU}(\text{BatchNorm}(W_5 * P_4 + b_5)). \quad (3.57)$$

$$D_5 = \text{Dropout}(L_5, 0.3). \quad (3.58)$$

$$L_{5,2} = \text{ReLU}(\text{BatchNorm}(W_{5,2} * D_5 + b_{5,2})) + L_5. \quad (3.59)$$

The decoder consists of upsampling and concatenation with corresponding feature maps from the contraction path for block O_j for $j = 6, 7, 8, 9$:

$$O_j = \text{Conv3DTranspose}(C_{j-1}, W_j, \text{stride} = (2, 2, 2)) \quad (3.60)$$

where C_{j-1} is the input feature map to the transposed convolutional layer from the previous decoder block, W_j represents the kernel weights for the 3D transposed convolution. For concatenation:

$$C_j = \text{concat}(O_j, C_{5-j}) \quad (3.61)$$

where C_{5-j} is the corresponding feature map of the encoder path (skip connection) that is concatenated.

In V-Net, the output layer is defined as:

$$\tilde{B} = \text{SoftMax}(W_{out} * C_9 + b_{out}). \quad (3.62)$$

However, (W_{out} and b_{out} are the weights and biases for the final 3D convolutional output layer, while C_9 is the feature map output from the last decoder block.

Chapter 4

Numerical Implementations

4.1 Chapter Overview

This chapter discusses the numerical implementation in some details. All experiments were conducted using Spyder platform, which is an open-source (IDE) designed for scientific programming in the Python language (Team, 2024) using a high ram and GPU-enabled personal computer. The Python code for these implementations was developed, in part, by adapting resources from the 'Python for Microscopists' repository (Sreenivas, 2023). The chapter has five sections. The first section describes the dataset used to implement the experiment of this study. The second section describes the pre-processing of the used dataset. The third section describes the architecture of the models used. In addition, the evaluation metrics are discussed in the fourth section. Finally, the training process is explained in the last section.

4.2 Dataset Used

The BraTS dataset is a widely recognized benchmark in medical image segmentation research. Introduced in 2012 as part of the Brain Tumor Segmentation (BraTS) Challenge, it has undergone regular updates, with versions released in 2012, 2015, 2017, 2018, and 2020 (Ateş et al., 2024). It consists of MRI scans of brain tumors that have been annotated and segmented by expert radiologists and clinicians. The dataset has been widely adopted for tasks such as brain tumor detection, computer-aided diagnosis, and automated segmentation systems. This study utilizes the BraTS 2020 dataset, originally released by the BraTS Challenge organizers. The version used in this work was obtained from an unofficial mirror hosted on the Kaggle platform (Awsaf49, 2020), which replicates the official dataset. The dataset includes a total of 494 MRI scans, split into 369 training samples and 125 validation samples. Each scan is accompanied by a corresponding segmentation mask that identifies tumor regions at the voxel level.

Figure 4.1 presents representative examples from the BraTS 2020 dataset, showcasing the diverse appearance of brain tumors across different MRI modalities. Each row typically displays the FLAIR, T1, T1ce, and T2 sequences, followed by the corresponding ground truth segmentation mask. For instance, observe how the enhancing tumor (often bright in T1ce) and peritumoral edema (bright in FLAIR and T2) are visually distinct across modalities, and how these regions are precisely delineated in the final segmentation mask, providing the target for our models (Wang et al., 2016).

The dataset includes images that provide insights into brain tissue. FLAIR images are used to identify regions of edema, whereas T1ce images enhance tumor boundaries by highlighting bright signals caused by substances accumulated in active cell areas. T1 images focus on depicting healthy tissues, while T2 images are particularly effective in showcasing bright signals associated with edema. Each MRI modality contains 155 slices, which, when combined, form a 3D data structure for the brain. This enables thorough

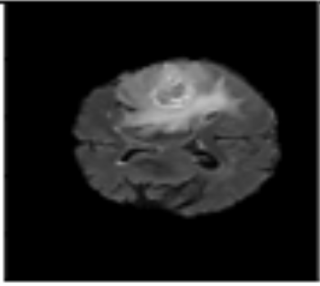
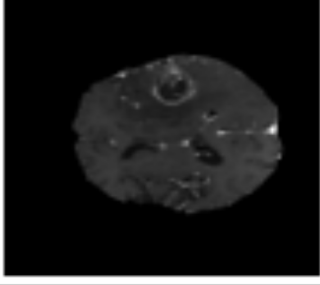
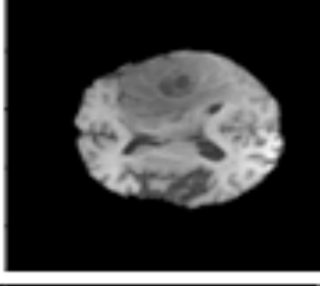
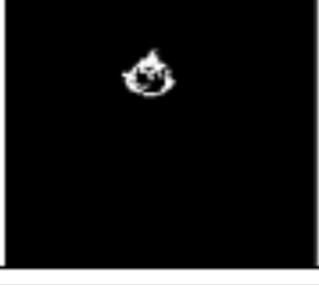
	Image	Mask
Flair		
T1ce		
T1		
T2		

Figure 4.1: Examples from BraTS 2020 dataset

examination and analysis of each modality and its corresponding mask in three dimensions (Ateş et al., 2024).

4.3 Dataset Pre-processing

MRI Modalities and Data Format

BraTS 2020 includes four MRI sequences: FLAIR, T1, T1ce, and T2. These are provided in NIfTI format (.nii), which stands for Neuroimaging Informatics Technology Initiative — a widely used file format for storing and sharing neuroimaging and medical imaging data (Cox, 2004). Each MRI volume is a 3D image and contains rich structural information about the brain.

Preprocessing and Normalization

To ensure consistency across the dataset, each MRI modality is normalized to the range $[0,1]$ using the MinMaxScaler, based on the maximum intensity value in each volume. This normalization helps stabilize and accelerate model training by standardizing pixel intensity distributions across patients. The segmentation masks consist of integer values: 0, 1, 2, and 4, representing background and different tumor subregions. For simplification, the label 4 is remapped to 3, effectively reducing the segmentation task to three tumor classes.

Modality Selection and Input Formatting

In this study, the T1 modality was excluded, as it offers lower contrast for tumor regions compared to FLAIR and T1ce, which provide more diagnostic relevance. The selected modalities — FLAIR, T1ce, and T2 — are combined into a single 4D NumPy array with shape (width, height, depth, channels), where the final dimension represents the number of channels (i.e., 3 for the selected modalities).

Cropping and Patch Preparation

To make the dataset suitable for training, both the combined MRI volumes and their corresponding masks are cropped to fixed-size 3D patches of $64 \times$

64×64 voxels. This step ensures uniformity in input size, which is essential for batch processing and deep learning frameworks. Figure 4.2 provides a visual overview of the dataset preprocessing pipeline. The top row displays raw samples of the FLAIR, T1ce, T2 modalities, and their corresponding mask. The subsequent rows illustrate the effect of preprocessing steps, such as normalization, which standardizes intensity ranges and makes images appear more uniform, and cropping to fixed-size 3D patches of $64 \times 64 \times 64$ voxels. This visual progression from the original full-size scans to the standardized patches highlights how the data is prepared for efficient input into the deep learning models, ensuring consistency and reducing computational load.

Dataset Split

After preprocessing, the dataset is split into training and validation sets. A standard 80/20 split is applied, allocating 80 percent of the samples for training and 20 percent for validation. This ensures the model is evaluated on unseen data while preserving enough data for effective training.

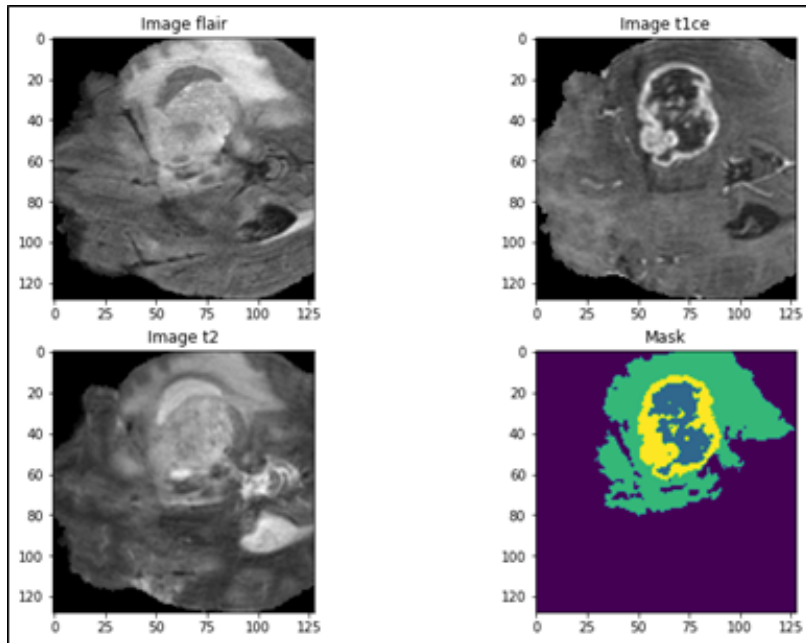


Figure 4.2: Samples of the flair, t1ce,t2 and mask

4.4 The Composition of Used Models

4.4.1 3D U-Net Model

The first model used in this study is a 3D U-Net, an extension of the original U-Net introduced in Chapter 3. It extends U-Net’s capabilities to handle 3D volumetric data, which is common in medical imaging. Figure 4.3 provides a detailed layer-by-layer representation of our 3D U-Net architecture. The left side depicts the contracting path (encoder), where each block typically consists of two 3D convolutional layers followed by a ReLU activation and a dropout layer, with max pooling operations indicated by down arrows progressively reducing spatial dimensions while increasing feature channels. The right side shows the expansive path (decoder), where transposed convolutions (up arrows) upsample the feature maps, which are then concatenated with corresponding feature maps from the encoder via skip connections (horizontal lines). This visual layout clearly demonstrates the symmetric U-shape, emphasizing the flow of information and the role of each layer type in feature extraction and spatial reconstruction. . This design effectively captures spatial features across three dimensions.

Figure 4.4 simplifies the 3D U-Net architecture into a block diagram, highlighting the logical flow of data. The left side represents the encoder, with each block performing feature extraction and downsampling. The central Bottleneck or latent block processes the most abstract features. The right side illustrates the decoder, where upsampling blocks reconstruct the spatial resolution. The crucial skip connections are visually represented by horizontal arrows, showing how high-resolution features from the encoder are directly passed to the decoder, enabling precise localization of tumor boundaries. The encoder path comprises 4 main blocks, each performing feature extraction at increasing levels of abstraction. Every block includes two 3D convolutional layers followed by a dropout layer to reduce overfitting, and a max pooling layer to progressively downsample the feature maps. The

number of filters increases with each block (starting from 16 up to 256), enabling the network to learn increasingly complex representations as the spatial resolution decreases.

At the bottleneck layer, the network captures the most abstract and complex features from the input data.

The decoder path mirrors the encoder with 4 blocks. Each decoder block consists of a transposed convolutional layer (to upsample the feature maps), a concatenation with skip connections from the encoder (to retain critical spatial information), followed by two 3D convolutional layers and a dropout layer. This design helps the model achieve precise localization in the segmentation output.

The final output is generated through a $1 \times 1 \times 1$ convolutional layer, which maps the learned features to the target classes. A SoftMax activation function is applied to produce class probabilities for each voxel. SoftMax turns the final layer's numbers into probabilities, allowing the model to express how likely each voxel belongs to a specific tumor class, thus enabling detailed multi-class segmentation of brain tumors.

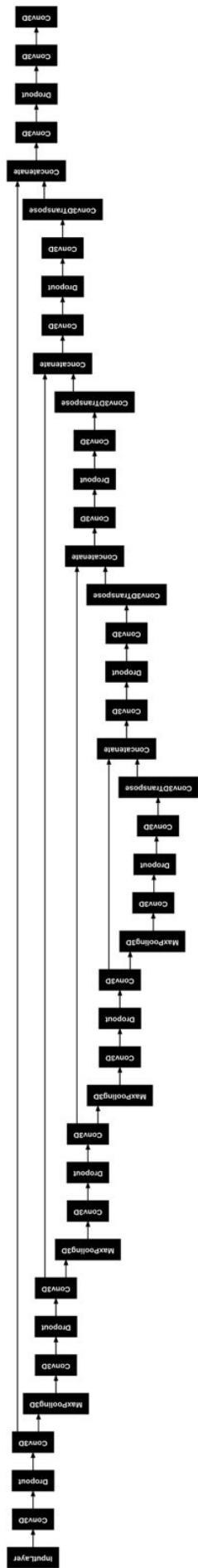


Figure 4.3: 3D U-Net Architecture by layers

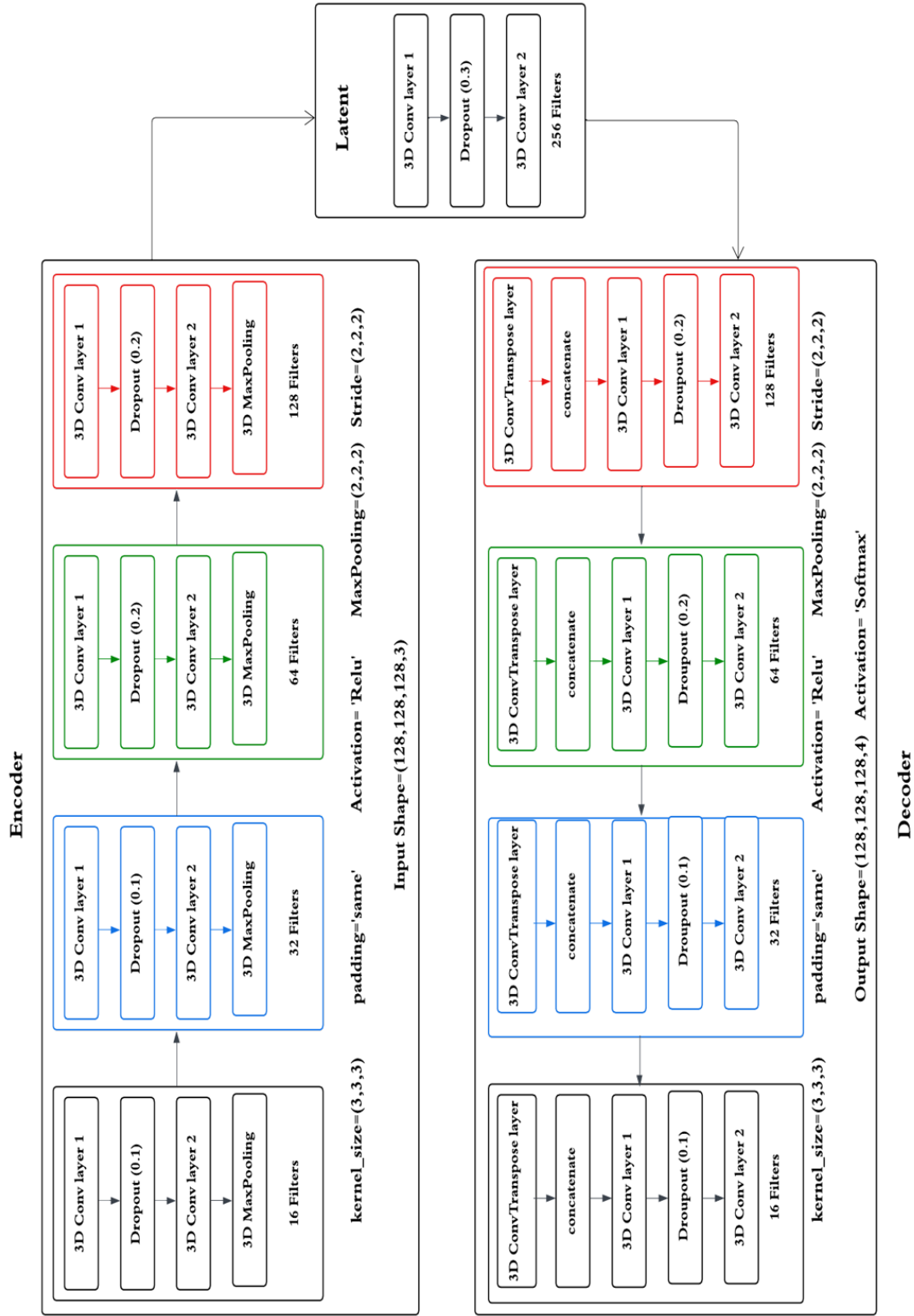


Figure 4.4: 3D U-Net Architecture by block diagram

All convolutional and transposed convolutional layers use a kernel size of (3, 3, 3). The default stride is (1, 1, 1) for convolutional layers, while for transposed convolutional layers it is set to (2, 2, 2), effectively doubling the spatial dimensions during upsampling. ‘Same’ padding is used in all convolutional layers to preserve the input dimensions when the stride is 1. The ReLU (Rectified Linear Unit) activation function is used in all convolutional layers, except for the output layer, which uses SoftMax to enable multi-class probability predictions for each voxel.

4.4.2 V-Net Model

The second model used in this study is V-Net, which utilizes 3D convolutions combined with residual connections in the feature encoder stage. These design choices increase the network’s depth and contribute to enhanced segmentation performance compared to the 3D U-Net (Milletari et al., 2016).

V-Net features an encoder-decoder architecture that allows hierarchical feature extraction from volumetric input data while maintaining spatial information through skip connections. The architecture begins with an initial convolutional block, consisting of a single 3D convolutional layer followed by a ReLU activation function. This block captures the essential low-level features of the input volume.

Following the initial block are four encoder blocks, each incorporating 3D convolutions with a kernel size of (3, 3, 3) and a stride of (2, 2, 2). These blocks reduce spatial resolution while increasing the number of feature maps. Each encoder block is accompanied by a custom convolutional sub-block, which includes two additional 3D convolutional layers, batch normalization, and ReLU activation. This design improves feature extraction while ensuring smooth gradient flow during backpropagation, thus enhancing training efficiency.

The bottleneck section plays a critical role in capturing complex and

abstract features at the network's deepest level. It consists of two convolutional layers with a kernel size of (3, 3, 3), stride (1, 1, 1), and 'same' padding to preserve the spatial context.

The decoder path of the V-Net includes four decoder blocks. Each block begins with a transposed convolution using a kernel size of (3, 3, 3) and a stride of (2, 2, 2) to upsample the feature maps. These upsampled features are concatenated with corresponding encoder features through skip connections, which help retain spatial details lost during downsampling. This is followed by further convolutional layers that integrate both high-level and low-level information to refine the segmentation output.

Notably, the first convolutional block in V-Net uses a larger kernel size of (5, 5, 5), a stride of (1, 1, 1), and 'same' padding, allowing it to capture broader spatial context from the input. All convolutional operations use the ReLU activation function, which introduces non-linearity and improves the model's ability to learn complex patterns.

The final output layer of V-Net is a single 3D convolutional layer with a kernel size of (1×1×1) and a stride of (1, 1, 1), followed by a SoftMax activation function. SoftMax turns the final layer's numbers into probabilities, enabling the model to estimate how likely each voxel belongs to a specific tumor class.

Figure 4.5 provides a block-level visualization of the V-Net architecture. Similar to U-Net, it features a contracting path (encoder) and an expansive path (decoder), forming a 'V' shape. Each block within the encoder and decoder is depicted, showing the sequence of 3D convolutional layers, batch normalization, and ReLU activations. A key visual element is the explicit representation of residual connections, which allow the input of a block to be added to its output, facilitating gradient flow and enabling deeper networks. The diagram clearly illustrates how spatial dimensions are reduced in the encoder and restored in the decoder, with feature channels increasing and then decreasing, respectively.



Figure 4.5: V-Net Architecture by block diagram

Overall, V-Net’s design enables robust learning from volumetric medical data, making it a strong candidate for accurate brain tumor segmentation.

4.5 Evaluation Metrics

The evaluation metric serves as a tool to assess and quantify the performance and effectiveness of our model. Within the field of machine learning, a variety of measurement techniques are available, each tailored to specific applications such as classification, detection, or segmentation. In our work, we relied on some of the most widely recognized metrics, which include:

4.5.1 Dice Coefficient

The Dice coefficient is a widely used statistical measure in image segmentation. It is regarded as one of the most used validation tools in the domain (Laouamer & Drid, 2023). Essentially, it measures the overlap between the predicted segmentation and the ground truth, specifically focusing on comparing pixels from the ground truth with those from the predicted segmentation. The Dice coefficient is calculated by taking twice the intersection value of the predicted mask and the ground truth, dividing it by the sum of their respective areas as shown in equation 4.4.1. The resulting Dice value ranges between zero and one, where a value of one indicates an excellent prediction, while zero signifies a very poor prediction.

The Dice coefficient D is defined as:

$$D = \frac{2 \times |X \cap Y|}{|X| + |Y|} \quad (4.1)$$

Where:

- $|X|$ is the number of pixels in the predicted segmentation.
- $|Y|$ is the number of pixels in the ground truth segmentation.

- $|X \cap Y|$ is the number of pixels that overlap between predicted and ground-truth segmentation.

As an illustrative example, Figure 4.6 visually demonstrates the Dice coefficient. Suppose the predicted segmentation (blue box) and the ground truth (red box) each cover an area of 16 pixels. The overlapping region (green area) between the two covers 4 pixels. Applying the Dice coefficient formula:

$$D = \frac{2 \times |X \cap Y|}{|X| + |Y|} = \frac{2 \times 4}{16 + 16} = \frac{8}{32} = 0.25$$

This result indicates that the predicted mask overlaps with the ground truth by only 25%, reflecting a relatively poor segmentation result. A Dice score of 1.0 would imply perfect overlap, while a score closer to 0 indicates minimal agreement.

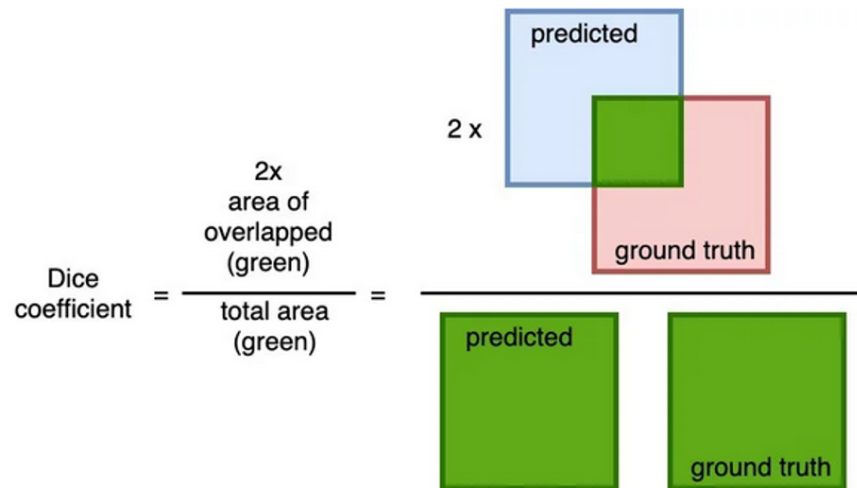


Figure 4.6: Visual representation of the Dice coefficient showing predicted, ground truth, and overlapping (green) areas.

4.5.2 Total Loss

In this study, the total loss function is designed to improve segmentation accuracy by combining two complementary loss functions. Dice Loss and Focal Loss. This combination balances the need for precise overlap with robustness to class imbalance.

4.5.2.1 Dice Loss

The Dice Loss is derived from the Dice coefficient, which measures the overlap between the predicted and ground truth segmentation. It is mathematically defined as:

$$\text{Dice Loss} = 1 - D. \quad (4.2)$$

where D is the Dice coefficient.

With class weights incorporated, the weighted Dice loss can be expressed as:

$$\text{Weighted Dice Loss} = 1 - \sum_{i=0}^{C-1} w_i \cdot \frac{2 \times |X_i \cap Y_i|}{|X_i| + |Y_i|}. \quad (4.3)$$

where:

- C is the number of classes.
- w_i represents the weight for class i . In our case, we utilize $w_0 = w_1 = w_2 = w_3 = 0.25$.

4.5.2.2 Focal Loss

To address class imbalance and improve learning from hard-to-classify voxels, we incorporate Focal Loss. The categorical focal loss for class c can be defined as:

$$\text{Focal Loss} = -\alpha (1 - p_t)^\gamma \log(p_t). \quad (4.4)$$

where:

- p_t is the estimated probability of the model for each class.
- α is a weighting factor for the loss of positive/negative samples, used to balance class importance.
- γ is the focus parameter that reduces the relative loss of well-classified examples, focusing more on misclassified examples.

Finally, the total loss, combining the dice loss and the focal loss, is given by:

$$\text{Total Loss} = \text{Dice Loss} + \lambda \cdot \text{Focal Loss}. \quad (4.5)$$

where λ is a weighting factor for focal loss. In our implementation, we assume that $\lambda = 1$.

4.5.2.3 Combined Dice and Focal Loss Behavior

It is important to clarify a common misconception: the total loss value "should be between 0 and 1" applies only to pure Dice Loss, not to the combined Dice + Focal Loss used in this study. Specifically:

- **Dice Loss** is a similarity measure bounded in $[0, 1]$, quantifying the overlap between predicted and ground-truth masks (Milletari et al., 2016).
- **Focal Loss** is unbounded due to its logarithmic nature, which amplifies the penalty for incorrect predictions with low confidence (Lin, Goyal, Girshick, He, & Dollár, 2017).

- **Combined Loss (Dice + Focal)** inherits Focal Loss’s unbounded characteristics. As a result, total loss values greater than 1.0 are common and expected, especially in the early stages of training.

Empirical studies (Jadon, 2020; Yeung et al., 2022) confirm that combining Dice and Focal losses often results in loss values exceeding 1.0, without indicating instability or error.

4.5.3 Intersection over Union (IoU)

IoU, also known as the Jaccard index, is another standard metric used to assess the precision of segmentation (Laouamer & Drid, 2023). Measures the overlap ratio between the ground truth (GT) and the predicted segments. This value is computed by dividing the intersection of the two regions by their union, as illustrated in Equation 4.6. However, The Mean IoU, calculated over a batch of images, provides an aggregate measure of the model’s performance across multiple classes, offering insights into its robustness and generalization. The resulting value ranges from zero to one, where a value of zero indicates a very poor prediction, and a value of one signifies an excellent prediction. In addition, a threshold can be set to help determine whether a prediction meets the desired accuracy or not.

The Intersection over Union (IoU) is defined as:

$$\text{IoU} = \frac{|X \cap Y|}{|X \cup Y|} \quad (4.6)$$

where:

- $|X \cap Y|$ is the overlap area between the predicted segmentation and the ground truth.
- $|X \cup Y|$ is the area covered by both the prediction and the ground truth.

4.5.4 Accuracy

In machine learning, every task requires evaluation, and the accuracy metric is highly useful for various tasks such as classification, detection, and segmentation (Laouamer & Drid, 2023). This metric assesses how accurate data points are assigned to their correct classes and is calculated by dividing the number of correct predictions by the total number of input samples. While accuracy tends to deliver reliable results, its effectiveness is particularly notable when the classes in the dataset are balanced as shown in Equation 4.7. In addition, the resulting accuracy value ranges between zero and one, where a value of one indicates an excellent classification, while zero signifies a very poor classification. However, in highly imbalanced datasets—such as medical images where tumor regions may be sparse, accuracy alone may not fully capture the model’s performance. Thus, it is used alongside other metrics such as Dice and IoU for comprehensive evaluation.

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN} \quad (4.7)$$

where:

- TP (True Positives): correctly predicted positive samples (e.g., tumor).
- TN (True Negatives): correctly predicted negative samples (e.g., non-tumor).
- FP (False Positives): non-tumor samples misclassified as tumor.
- FN (False Negatives): tumor samples misclassified as non-tumor.

4.5.5 The Training Process

The training process for a segmentation model designed to medical imaging tasks, such as brain tumor segmentation, involves several carefully designed steps and parameters to ensure effective learning and generalization. The process begins with defining the model architecture using 3D U-Net and V-Net models which train separately. Input dimensions are set to $(128 \times 128 \times 128)$ with three channels, each representing different imaging modalities (e.g., FLAIR, T1ce, T2). For optimization, the model employs an Adam optimizer with a learning rate of 0.0001, allowing for adaptive adjustments during training. A hybrid loss function is used, combining Dice Loss with Categorical Focal Loss to handle class imbalances effectively. Equal class weights of 0.25 are applied across the four classes for Dice Loss, ensuring balanced optimization contributions. Training is conducted over 35 epochs with a batch size of 2 to process multiple samples in parallel. The number of steps per epoch is derived by dividing the total number of training images by the batch size, enabling efficient dataset utilization. Validation runs concurrently using a separate validation dataset. Validation steps are similarly calculated based on the total number of validation images and batch size. During training, metrics such as accuracy and Intersection over Union (IoU) are continuously monitored to track model performance. Additionally, training history, including loss and accuracy trends for both training and validation datasets, is visualized to analyze learning progression and detect potential overfitting. Once training is complete, the model is saved for future use—either for inference or further fine-tuning. This ensures the preserved learned parameters can be leveraged for clinical applications requiring robust segmentation capabilities.

Chapter 5

Result Analysis and Comparison

5.1 Result Analysis

In our study, we evaluated the performance of two models for brain tumor segmentation, U-Net and V-Net, each trained for 35 epochs. Table 5.1 shows the results for which both models achieved competitive performance metrics. Specifically, the U-Net model recorded a loss of 0.8103, an accuracy of 0.9826, and an Intersection over Union (IoU) score of 0.6792. In comparison, the V-Net model demonstrated slightly improved results with a loss of 0.7979, an accuracy of 0.9865, and an IoU of 0.7330. These findings suggest that both models are effective for the task at hand, while V-Net achieves significantly better training performance (loss:0.7979 vs. 0.8103; IoU:+7.38%).

Our analysis of the 3D U-Net and V-Net models revealed notable differences in performance trends when evaluating training and validation metrics, especially with regard to loss and accuracy.

Table 5.1: Results for 3D U-Net and V-Net Models

Metric	3D U-Net Model	V-Net Model
Training Loss	0.8103	0.7979
Validation Loss	0.8263	0.8586
Training Accuracy	0.9826	0.9865
Validation Accuracy	0.9750	0.9715
Training IoU	0.6792	0.7330
Validation IoU	0.6292	0.5376

Figure 5.1 shows the training and validation accuracy curves for the 3D U-Net model over 35 epochs. The training accuracy shows a sharp and consistent increase, reaching 0.9826, indicating effective learning on the training data. The validation accuracy also increases, but it plateaus and remains slightly below the training accuracy 0.9750. This gap, where validation accuracy is lower than training accuracy, suggests that while the model learns efficiently, there might be a slight degree of overfitting to the training data, although the performance remains robust.

At the same time, in Figure 5.2 illustrates the training and validation loss curves for the 3D U-Net model. The training loss demonstrates a steady and significant decline throughout the epochs, indicating that the model is effectively minimizing errors on the training dataset. The validation loss also decreases, but it consistently remains slightly higher than the training loss (0.8263 vs 0.8103). This divergence, where validation loss is higher, further supports the observation of a minor degree of overfitting, suggesting that the model's generalization to unseen data is good but could potentially be improved with further regularization or hyperparameter tuning.

On the other hand, Figure 5.3 presents the training and validation accuracy curves for the V-Net model. The training accuracy shows a rapid and strong increase, reaching a high of 0.9865, indicating efficient learning. The validation accuracy also rises, but it exhibits more fluctuations and ultimately settles at 0.9715, which is slightly lower than the training accuracy. This pattern, where validation accuracy is lower and less stable than training ac-

curacy, suggests a degree of overfitting, although the model still achieves high overall performance on unseen data.

Although Figure 5.4 displays the training and validation loss curves for the V-Net model. The training loss shows a steeper and more pronounced decline compared to the 3D U-Net, reaching a lower final value of 0.7979, indicating very effective error reduction on the training data. However, the validation loss shows a more significant divergence from the training loss, ending at 0.8586, which is notably higher than the training loss. This larger gap between training and validation loss, coupled with a lower validation IoU 5.1, suggests a more pronounced overfitting issue for V-Net compared to 3D U-Net, despite its superior training performance.

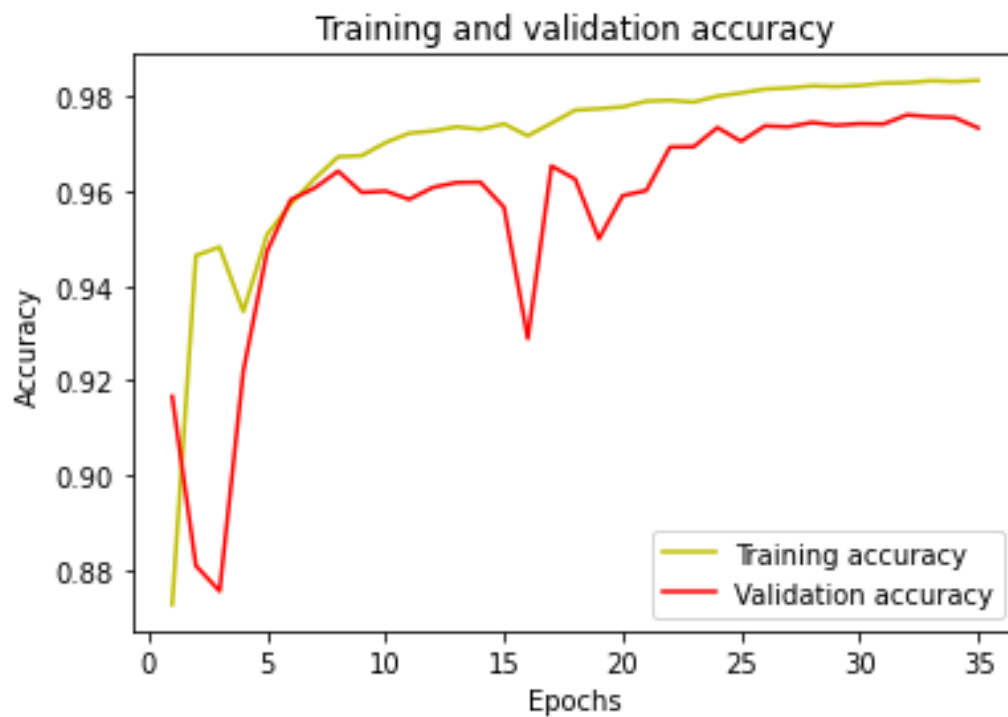


Figure 5.1: The Training vs Validation Accuracy Graph of 3D U-Net model

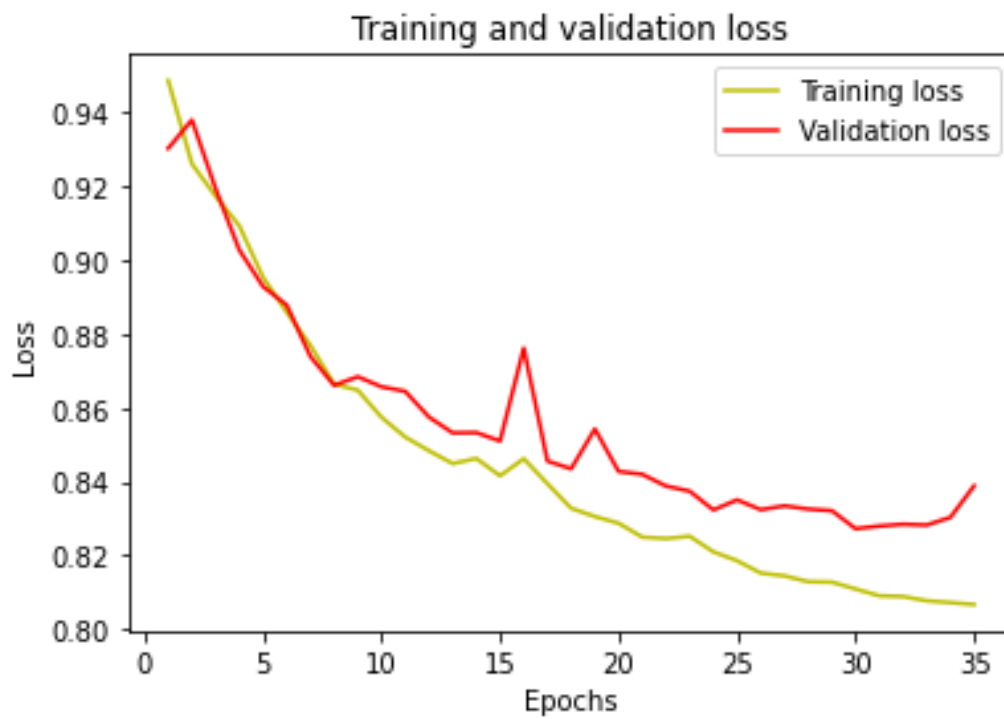


Figure 5.2: The Training vs Validation Loss Graph of 3D U-Net model

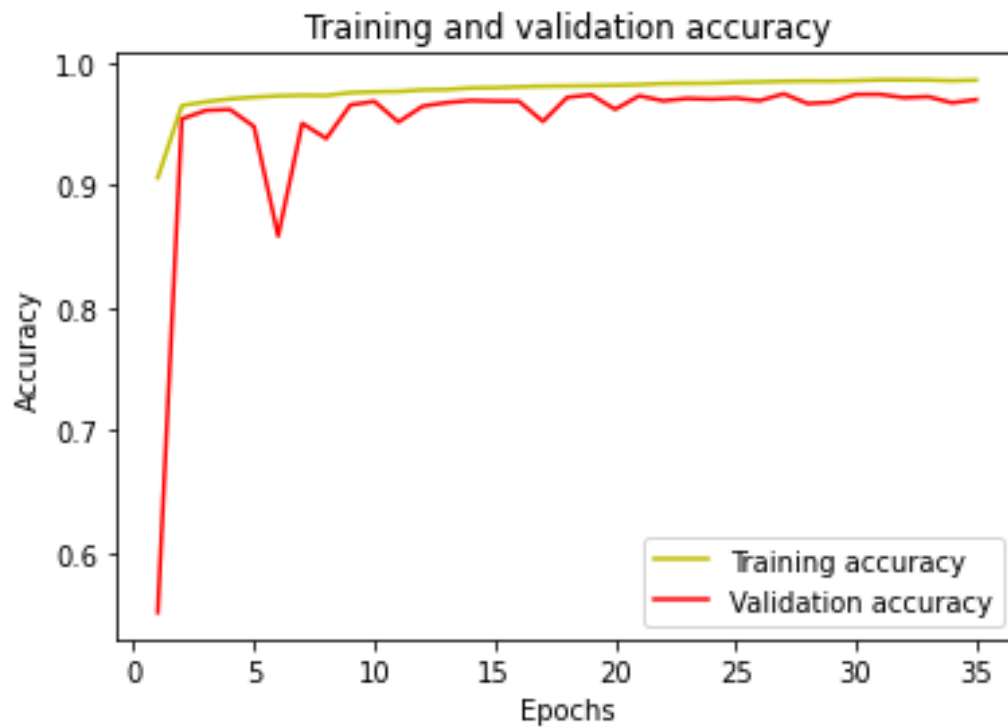


Figure 5.3: The Training vs Validation Accuracy Graph of V-Net model

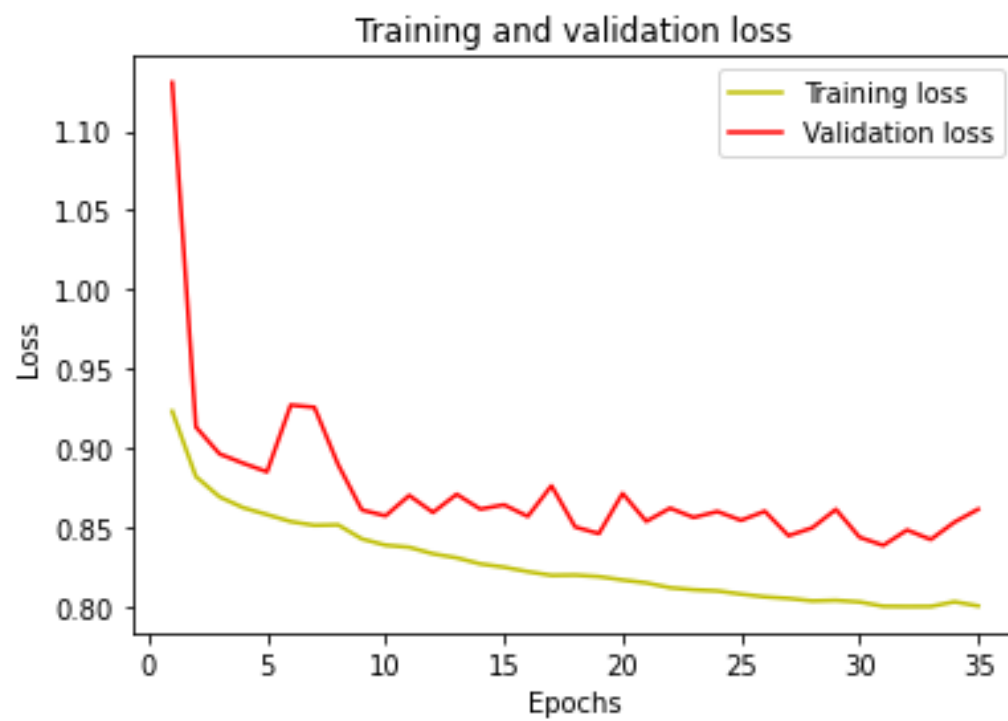


Figure 5.4: The Training vs Validation Loss Graph of V-Net model

5.2 Observation

Our approach placed significant emphasis on minimizing data loss while aiming to build a more efficient model. We have successfully met a substantial portion of our objective, surpassing the performance of many previous models, developed a highly effective solution for both training and validation. Below, we present several plotted prediction accuracies using the 3D U-Net model and the V-net model for specific image files, alongside comparative analysis.

Figure 5.5 provides visual examples of segmentation results from the 3D U-Net model. In the first row, observe how the model successfully delineates the main tumor body. However, in the second row, particularly around the edges of the enhancing tumor (yellow) and edema (green), minor misclassifications or less precise boundaries can be seen where healthy tissue and tumor regions closely overlap. This illustrates U-Net’s general effectiveness but also its challenges in accurately distinguishing between very similar or adjacent tumor sub-regions, especially in complex anatomical contexts.

In other hand, Figure 5.6 showcases visual examples of segmentation results from the V-Net model, demonstrating its superior performance compared to 3D U-Net. Notice in the first row how V-Net provides clearer and more precise tumor boundary definitions, particularly for the irregular shapes of the necrotic core and enhancing tumor. In the second row, the model’s ability to differentiate between neighboring tumor types (e.g., the distinct separation of edema from the enhancing tumor) and healthy tissue is evident, leading to a visibly reduced number of misclassifications and a more accurate overall segmentation map.

To summarize up, both 3D U-Net and V-Net models offer significant contributions to tumor segmentation. However, the V-Net shows greater effectiveness in precisely identifying and outlining tumor regions in brain MRI.

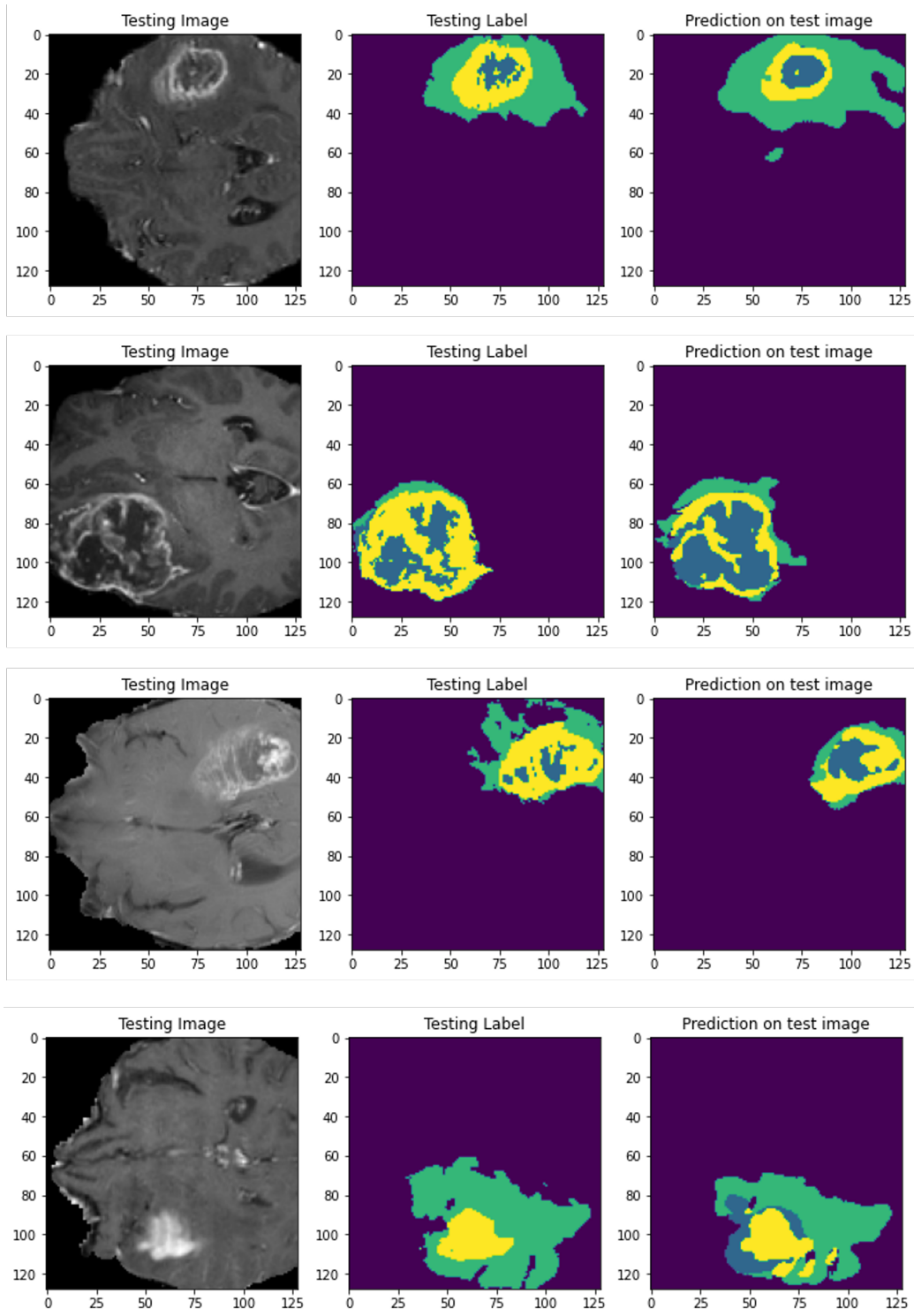


Figure 5.5: Examples of prediction accuracies for some images using 3D U-Net model

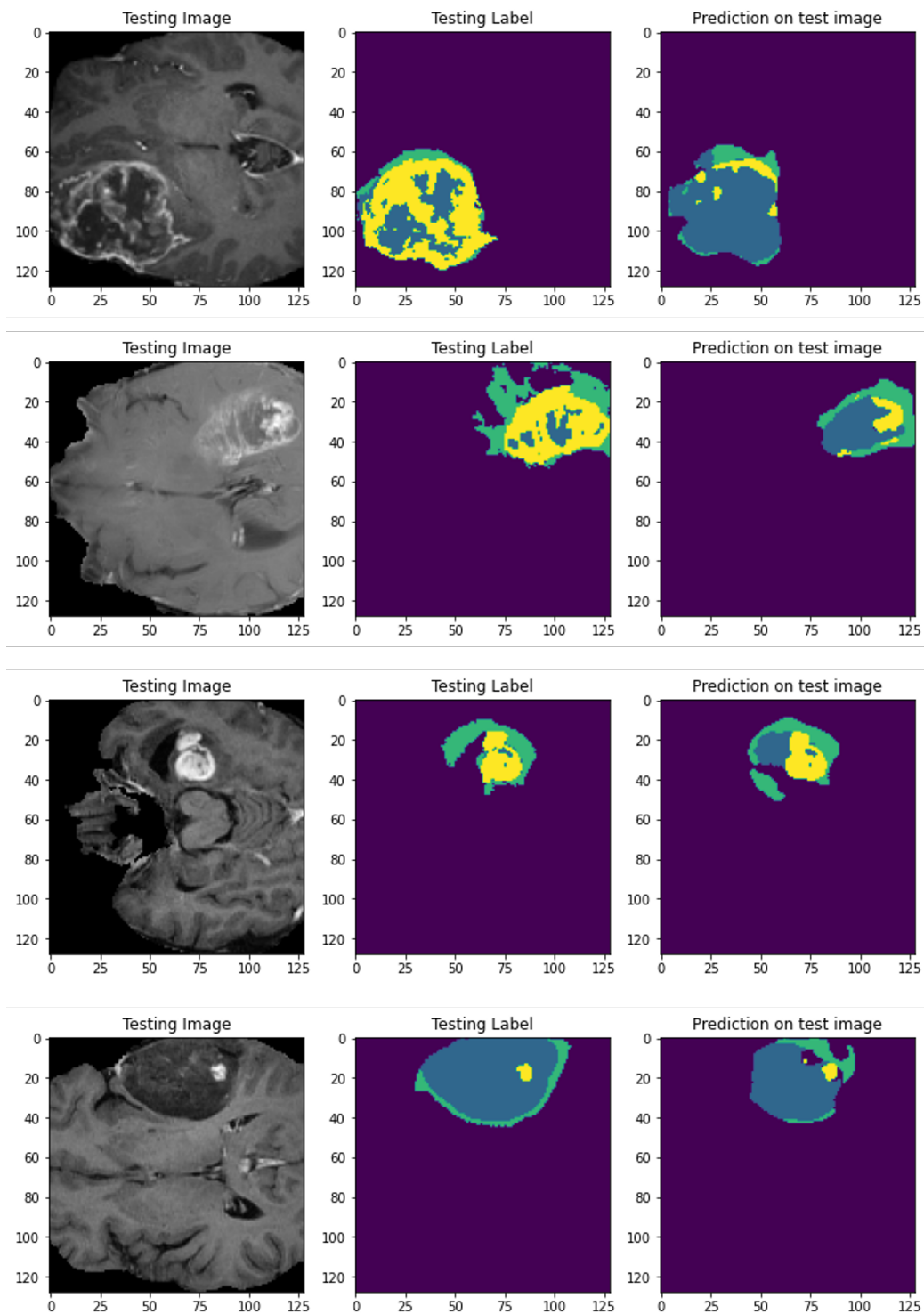


Figure 5.6: Examples of prediction accuracies for some images using V-Net model

5.3 Comparison with Other Models

For the quantitative evaluation, the used approach was assessed against same models which are used in liver tumor segmentation proposed by (Lei & Nandi, 2022) using a validation subject. It used 3D CT scans of liver tumors. U-Net is proposed for 2D instead of 3D and it is composed of 23 layers, including convolutional, pooling, and upsampling operations. Figure 5.7 shows Its architecture which is divided into an encoder and a decoder pathway. The encoder consists of four downsampling blocks, each comprising two convolutional layers with 3x3 kernels activated by ReLU, followed by a max-pooling layer for spatial reduction. The number of feature channels start at 64 in the first block and doubles with each subsequent block, reaching 512 channels in the last one. At the core of the architecture lies the bottleneck, which includes two convolutional layers with 1024 feature channels.

The decoder symmetrically mirrors the encoder, featuring four upsampling blocks. Each block begins with an upsampling operation, followed by a concatenation with the corresponding encoder features and then two convolutional layers. As the decoder reconstructs spatial information, the feature channel count is halved in each block, eventually returning to 64 channels in the final stage. A concluding 1x1 convolutional layer transforms these feature maps into the required number of output classes, effectively enabling binary segmentation tasks.

V-Net, on the other hand, is designed especially for volumetric images composed of 30 layers, integrating convolutional, pooling, and deconvolutional operations, and processes input volumes with dimensions (128×128×64). Figure 5.8 shows the encoder pathway, which is comprised of five downsampling blocks, each containing two 3D convolutional layers with 5x5 x5 kernels activated by ReLU functions. The number of filters progressively doubles from 16 to 128 across these blocks. Each block also includes a downsampling layer with a stride of (2, 2, 2) to reduce spatial dimensions.

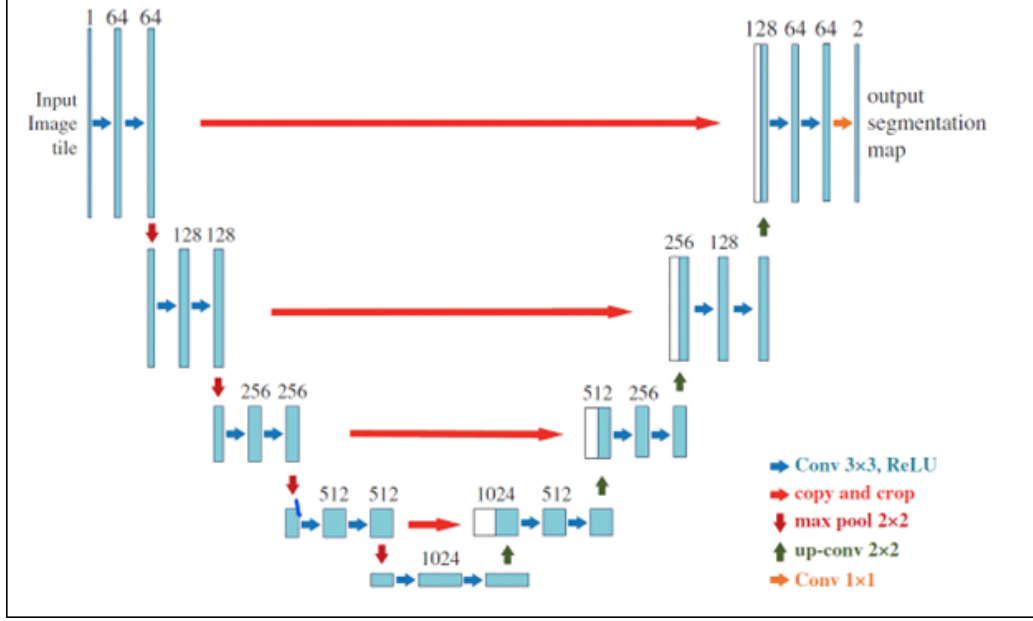


Figure 5.7: The U-Net architecture

At the core of the network lies a bottleneck layer with two 3D convolutional operations and 256 filters, enabling the capture of intricate features.

The decoder pathway mirrors the encoder with five upsampling blocks that employ transposed convolutional layers. These blocks merge features from the encoder via skip connections and are followed by two additional convolutional layers. The output layer is a 3D convolutional layer with a (1, 1, 1) kernel employing SoftMax activation to generate a pixel-wise classification map. To address challenges such as the vanishing gradient problem and enhance feature flow, V-Net integrates residual connections. Dropout layers are also strategically placed to minimize overfitting during training on limited datasets .

Table 5.2: Comparison the results of models used in this study and models used in (Lei & Nandi, 2022).

Model	Loss	Accuracy	IoU
3D U-Net (our model) – 35 epochs	0.8103	0.9826	0.6792
V-Net (our model) – 35 epochs	0.7979	0.9865	0.7330
2D U-Net (Lei & Nandi, 2022) – 50 epochs	0.2000	0.9300	0.8500
V-Net (Lei & Nandi, 2022) – 50 epochs	0.1500	0.9400	0.8800

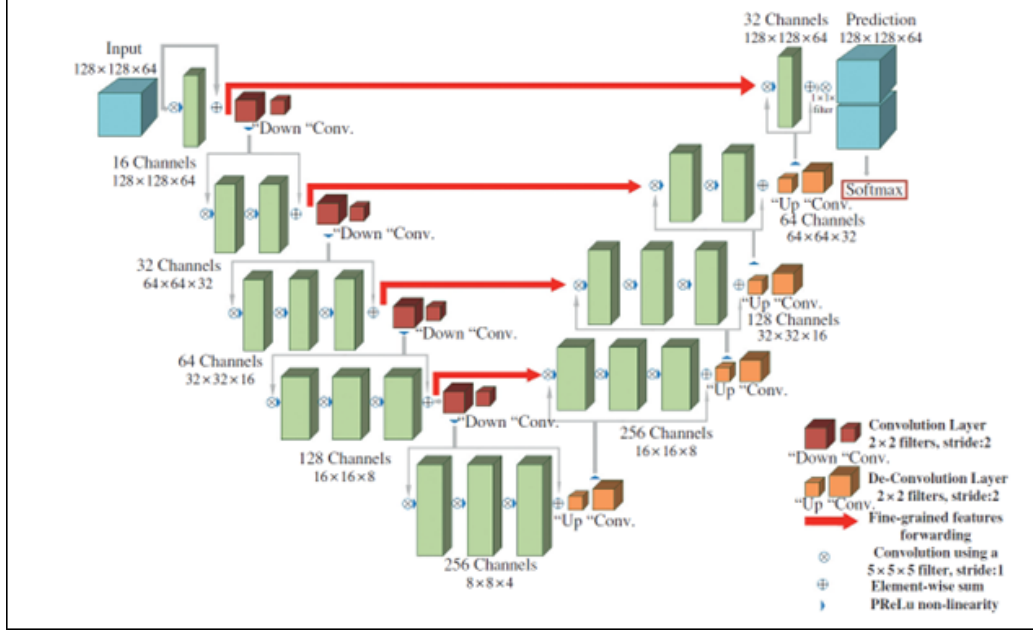


Figure 5.8: The V-Net architecture

Table 5.2 shows the comparison focused on image segmentation in terms of accuracy, loss, and IoU using different numbers of epochs. The models used in this study for brain tumor segmentation while the other models (Lei & Nandi, 2022) used for liver tumor segmentation. When evaluating segmentation performance for tumor identification, U-Net and V-Net were specifically tested on MRI brain tumor segmentation in our study, while baseline comparison models focused on CT liver tumor segmentation used in (Lei & Nandi, 2022). The 3D U-Net model in our work, trained for 35 epochs, achieved an accuracy of 0.9826, a loss of 0.8103, and an Intersection over Union (IoU) of 0.6792. Meanwhile, the V-Net model in our work, also trained for 35 epochs, outperformed U-Net with an accuracy of 0.9865, a lower loss of 0.7979, and a higher IoU of 0.7330. When Comparing 3D U-Net model in our work with 2D U-Net in (Lei & Nandi, 2022), It can be observed that the model used in this study outperforms the one used in the latter study using lower number of epochs (35 epochs) than 2D U-Net (50 epochs). The results show higher accuracy (0.9826) for 3D U-Net model than the 2D U-Net accuracy (0.9300), even our models has higher loss and

IoU because of the low number of epochs. In our study, the performance of the V-Net model is compared to the V-Net model presented in (Lei & Nandi, 2022) . It is evident that the V-Net employed in this research achieves superior results with fewer training epochs (35 epochs) compared to the 50 epochs required by the V-Net in latter study. Notably, our model attains a higher accuracy of 0.9865, surpassing the 0.9400 accuracy reported for the V-Net model used in (Lei & Nandi, 2022). Although our model exhibits higher loss and lower IoU, this can be attributed to the reduced number of training epochs.

This suggests that both 3D U-Net and V-Net in our work, particularly V-Net, are better suited for MRI brain tumor segmentation tasks and can deliver superior results even with fewer training epochs compared to less optimized models trained longer. These findings underscore the effectiveness of the proposed models in clinical applications, emphasizing their potential to enhance diagnostic accuracy in medical imaging, especially for brain tumor analysis.

5.4 Discussion

This section systematically addresses the four research questions through analysis of the 3D U-Net and V-Net performance on the BraTS2020 dataset, with explicit connections to the study’s predefined objectives.

RQ1:How do deep learning-based segmentation principles (e.g., feature extraction, spatial consistency) apply to MRI brain tumor segmentation?

The study aimed to implement and evaluate two 3D deep learning models (U-Net and V-Net) for brain tumor segmentation using the BraTS2020 dataset. Deep learning models, particularly Convolutional Neural Networks (CNNs), excel in feature extraction by learning hierarchical representations directly from the MRI data (Section 3.3.2). The 3D U-Net and V-Net architectures effectively capture spatial consistency across slices, which is crucial for accurately delineating tumor boundaries (Section 4.4.1 and 4.4.2). These models utilize 3D convolutions to maintain volumetric information, allowing for better contextual understanding of tumor structures compared to traditional 2D methods (Section 2.3.3).

RQ2:How do the segmentation performance (e.g., Dice score, IoU) of 3D U-Net and V-Net differ statistically when evaluated on the BraTS2020 dataset?

The objective was to quantitatively compare model performance using segmentation metrics (Accuracy, IoU, Dice, and Focal loss). The evaluation results indicate that V-Net outperformed the 3D U-Net in terms of segmentation accuracy and Intersection over Union (IoU). Specifically, V-Net achieved an accuracy of 98.65% and an IoU of 0.7330, while the 3D U-Net recorded an accuracy of 98.26% and an IoU of 0.6792 (Table 5.1). This statistical difference highlights V-Net’s enhanced capability in addressing tumor boundary uncertainties through its residual connections and advanced feature learning abilities (Section 3.3.2.1).

RQ3:What are the correlations between the size, shape, and localization of brain tumors achieved by U-Net and V-Net models?

The study aimed to qualitatively analyze differences in tumor boundary delineation and localization between U-Net and V-Net. Both models demonstrated the ability to accurately segment tumors of varying sizes and shapes (Section 5.2). However, V-Net showed superior performance in localizing tumors, particularly in complex scenarios where tumor types were in proximity. The skip connections in V-Net facilitated better retention of spatial information, leading to more precise delineation of tumor boundaries (Section 4.4.2).

RQ4:Which model (U-Net or V-Net) better generalizes to rare tumor sub-types in clinical settings?

The objective was to identify clinical implications of each model's performance for diagnosis and treatment planning. Based on the findings, V-Net appears to generalize better to rare tumor sub-types due to its robust architecture that incorporates residual connections and advanced feature extraction techniques (Section 3.3.2.1). This model's ability to learn from a diverse dataset and maintain spatial consistency makes it a more reliable choice for clinical applications, particularly in identifying less common tumor presentations (Section 5.3).

In addition, the findings from this study underscore the significant impact of deep learning in neuro-oncology imaging, particularly in the context of MRI brain tumor segmentation. The comparative analysis of the 3D U-Net and V-Net models reveals that while both architectures are effective, V-Net offers notable advantages in accuracy and reliability, making it a promising tool for clinical decision-making in brain tumor diagnosis and treatment planning. This aligns with the study's objectives of enhancing segmentation accuracy and providing reliable tools for clinical use.

Chapter 6

Conclusion and Future works

6.1 Conclusion

In our study, we utilized an optimized Fully Convolutional Network (FCN) model for semantic segmentation, concentrating on 3D U-Net and V-Net architectures. Segmenting 3D images poses challenges due to organ structure variability and the complexity of 3D models. To enhance accuracy and optimization, we incorporated a bounding box algorithm into the FCN framework. Building on data from previous experiments in liver tumor segmentation, we redirected our efforts towards brain tumor segmentation. For validation, the MICCAI BraTS2020 dataset was employed, leveraging advanced GPU capabilities to boost model performance. We made notable progress, though further refinements are necessary to fully tap into our model's potential. Our findings indicated that both the 3D U-Net and V-Net models excelled in MRI brain tumor segmentation, achieving accuracies of 0.9826 and 0.9865, respectively, with the V-Net model outperforming the

3D U-Net. In conclusion, our research highlights the effectiveness of these models in diagnosing brain tumors, especially emphasizing the benefits of the V-Net model.

6.2 Future Works and Recommendations

In our future work, we aim to develop a more efficient version of the proposed models that reduces data loss while achieving higher segmentation accuracy. Enhancing the training process is a key priority, with particular focus on maximizing validation accuracy and minimizing validation loss through improved optimization strategies.

We also plan to investigate GPU memory reduction techniques during 3D data segmentation. This includes using larger patch sizes, optimizing memory utilization, and expanding the current architecture. In particular, we aim to enhance the existing 3D U-Net and V-Net models by integrating them with other state-of-the-art architectures to further improve segmentation performance.

Additionally, we intend to conduct further experiments on 3D medical imaging datasets to refine segmentation methods. We recommend applying our models to various organs and extending evaluation across diverse datasets such as the LGG segmentation dataset and the PROMISE2012 dataset.

Future extensions of this work should also explore more advanced models like LV-Net, Residual U-Net, and MobileNet U-Net to assess their performance and efficiency in clinical settings.

Moreover, we propose a more comprehensive comparison between U-Net and V-Net architectures in segmenting tumors with varying anatomical characteristics. This includes:

- Tumor volume — evaluating model accuracy across different tumor sizes,

- Tumor shape — using the sphericity index to assess performance on irregular vs. regular geometries,
- Tumor location — computing centroid distance between predicted and ground-truth regions,
- Sensitivity — measuring true positive rate to better understand detection performance, especially for small or diffuse tumors.

As part of regularization during training, we propose evaluating the effect of key hyperparameters. Specifically, we suggest exploring:

- Dropout rates in the range of 0.3 to 0.5,
- Early stopping with a patience of 10 to 15 epochs,

These techniques will help to mitigate overfitting and improve generalization across different segmentation tasks.

References

- Abdulateef, S. K., & Salman, M. D. (2021, 12). A comprehensive review of image segmentation techniques. *Iraqi Journal for Electrical and Electronic Engineering*, 17, 166-175. doi: 10.37917/ijeee.17.2.18
- Al-Amri, S. S., & Kalyankar, N. V. (2010). Image segmentation by using threshold techniques. *arXiv preprint arXiv:1005.4020*.
- Alias, A., Shah, P. M., Abdullah, N. D., & Khalid, N. A. C. (2020). Adapting mobile learning via cyberpneuership course. *Published 2020 Printed in Malaysia*, 33.
- Aloysius, N., & Geetha, M. (2017, April). A review on deep convolutional neural networks. In *2017 international conference on communication and signal processing (iccsp)* (pp. 588–592).
- Amin, J., Sharif, M., Haldorai, A., Yasmin, M., & Nayak, R. S. (2022, 8). Brain tumor detection and classification using machine learning: a comprehensive survey. *Complex and Intelligent Systems*, 8, 3161-3183. doi: 10.1007/s40747-021-00563-y
- Ateş, M. U., Günlü, R. T., Ekinçi, E., & Garip, Z. (2024, 9). Using 3d u-net for brain tumour segmentation from magnetic resonance images. *Academic Platform Journal of Engineering and Smart Systems*, 12, 81-87. Retrieved from <http://dergipark.org.tr/en/doi/10.21541/apjess.1508913> doi: 10.21541/apjess.1508913
- Awsaf49. (2020). *Brats 2020 training data*. Retrieved from <https://www.kaggle.com/datasets/awsaf49/brats2020-training-data>

(Accessed: 2023-10-01)

- Balafar, M. A., Ramli, A. R., Saripan, M. I., & Mashohor, S. (2010, 3). *Review of brain mri image segmentation methods* (Vol. 33). doi: 10.1007/s10462-010-9155-0
- Berner, J., Grohs, P., Kutyniok, G., & Petersen, P. (2021). The modern mathematics of deep learning. *arXiv preprint arXiv:2105.04026*, 78.
- Bhatt, D., Patel, C., Talsania, H., Patel, J., Vaghela, R., Pandya, S., & Ghayvat, H. (2021). Cnn variants for computer vision: History, architecture, application, challenges and future scope. *Electronics*, 10(20), 2470.
- Biratu, E. S., Schwenker, F., Debelee, T. G., Kebede, S. R., Negera, W. G., & Molla, H. T. (2021, 2). Enhanced region growing for brain tumor mr image segmentation. *Journal of Imaging*, 7. doi: 10.3390/jimaging7020022
- Brar, D. S., Aggarwal, A. K., Nanda, V., Saxena, S., & Gautam, S. (2024). Ai and cv based 2d-cnn algorithm: botanical authentication of indian honey. *Sustainable Food Technology*, 2(2), 373–385.
- Buchholz, T. O., & Jug, F. (2022). Fourier image transformer. In *Proceedings of the ieee/cvf conference on computer vision and pattern recognition* (pp. 1846–1854).
- Cox, R. (2004). Nifti-1: A (sort of) new image format for neuroimaging. *NeuroImage*, 22(1), 1–2.
- de Brebisson, A., & Montana, G. (2015). Deep neural networks for anatomical brain segmentation. In *Proceedings of the ieee conference on computer vision and pattern recognition workshops* (pp. 20–28).
- Du, G., Cao, X., Liang, J., Chen, X., & Zhan, Y. (2020). Medical image segmentation based on u-net: A review. *Journal of Imaging Science & Technology*, 64(2).
- Hussain, S., Mubeen, I., Ullah, N., Shah, S. S. U. D., Khan, B. A., Zahoor, M., & Sultan, M. A. (2022). Modern diagnostic imaging technique

- applications and risk factors in the medical field: a review. *BioMed Research International*, 2022(1), 5164970.
- Işin, A., Direkoğlu, C., & Şah, M. (2016). Review of mri-based brain tumor image segmentation using deep learning methods. In *Procedia computer science* (Vol. 102, p. 317-324). Elsevier B.V. doi: 10.1016/j.procs.2016.09.407
- Jadon, S. (2020). A survey of loss functions for semantic segmentation. *Image and Vision Computing*, 102, 103964.
- Jalab, H. A., & Hasan, A. M. (2019). Magnetic resonance imaging segmentation techniques of brain tumors: A review. *Archives of Neuroscience*, 6(Brain Mapping).
- Jiang, X., Wang, Y., Liu, W., Li, S., & Liu, J. (2019). Capsnet, cnn, fcn: Comparative performance evaluation for image classification. *International Journal of Machine Learning and Computing*, 9, 840-848. doi: 10.18178/ijmlc.2019.9.6.881
- Kim, K. C., Jang, T. J., & Seo, J. K. (2023). Deep learning techniques for medical image segmentation and object recognition. In *Deep learning and medical applications* (pp. 51–99). Singapore: Springer Nature Singapore.
- Kim, Y. (n.d.). Convolutional neural networks for sentence classification. In *Proceedings of the 2014 conference on empirical methods in natural language processing (emnlp)* (pp. 1746–1751).
- Kiranyaz, S., Avci, O., Abdeljaber, O., Ince, T., Gabbouj, M., & Inman, D. J. (2021). 1d convolutional neural networks and applications: A survey. *Mechanical Systems and Signal Processing*, 151, 107398.
- Laouamer, I., & Drid, K. (2023). *Algerian democratic and popular republic ministry of higher education and scientific research kasdi merbah university ouar gla faculty of new technologies of information and communication department of computer science and information technology academic master thesis domain: Computer science and*

- information technology field: Computer science speciality: Artificial intelligence and data science* (Tech. Rep.). Kasdi Merbah University Ouar gla.
- Lei, T., & Nandi, A. K. (2022). *Image segmentation: principles, techniques, and applications*. John Wiley & Sons.
- Lin, T.-Y., Goyal, P., Girshick, R., He, K., & Dollár, P. (2017). Focal loss for dense object detection. In *Proceedings of the ieee international conference on computer vision (iccv)* (pp. 2980–2988).
- Milletari, F., Navab, N., & Ahmadi, S. A. (2016, 12). V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *Proceedings - 2016 4th international conference on 3d vision, 3dv 2016* (p. 565-571). Institute of Electrical and Electronics Engineers Inc. doi: 10.1109/3DV.2016.79
- Pinaya, W. H. L., Vieira, S., Garcia-Dias, R., & Mechelli, A. (2020). Convolutional neural networks. In *Machine learning* (pp. 173–191). Academic Press.
- Qamar, S., Jin, H., Zheng, R., Ahmad, P., & Usama, M. (2020, 7). A variant form of 3d-unet for infant brain segmentation. *Future Generation Computer Systems*, 108, 613-623. doi: 10.1016/j.future.2019.11.021
- Ronneberger, O., Fischer, P., & Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *Lecture notes in computer science (including subseries lecture notes in artificial intelligence and lecture notes in bioinformatics)* (Vol. 9351, p. 234-241). Springer Verlag. doi: 10.1007/978-3-319-24574-4_28
- Sachin, N. (2014). *Brain tumor detection based on mathematical analysis and symmetry information* (Vol. 4; Tech. Rep.). Retrieved from www.ijera.com
- Shrivastava, N., & Bharti, J. (2020, 7). Automatic seeded region growing image segmentation for medical image segmentation: A brief review. *International Journal of Image and Graphics*, 20. doi:

10.1142/S0219467820500187

- Sreenivas, B. N. (2023). *Python for microscopists*. https://github.com/bnsreenu/python_for_microscopists.
- Talo, M., Baloglu, U. B., Yildirim, , & Acharya, U. R. (2019). Application of deep transfer learning for automated brain abnormality classification using mr images. *Cognitive Systems Research*, 54, 176–188.
- Team, S. D. (2024). *Spyder ide*. <https://www.spyder-ide.org/>. (Accessed: 2025-06-21)
- Teow, M. Y. (2021, 9). Convolutional autoencoder for image denoising: A compositional subspace representation perspective. In *3rd ieee international conference on artificial intelligence in engineering and technology, iicaiet 2021*. Institute of Electrical and Electronics Engineers Inc. doi: 10.1109/IICAiet51634.2021.9573657
- Wang, Z., Wang, Y., Jiang, L., Zhang, C., & Wang, P. (2016). An image segmentation method using automatic threshold based on improved genetic selecting algorithm. *Automatic Control and Computer Sciences*, 50, 432–440.
- Xu, X., Dehghani, A., Corrigan, D., Caulfield, S., & Moloney, D. (2016, July). Convolutional neural network for 3d object recognition using volumetric representation. In *2016 first international workshop on sensing, processing and learning for intelligent machines (spline)* (pp. 1–5).
- Yeung, M., et al. (2022). A unified loss function for medical image segmentation: Combining dice and focal loss. *Journal of Imaging*, 8(5), 122.
- Yousefi, J. (2011). *Image binarization using otsu thresholding algorithm* (Vol. 10; Tech. Rep. No. 9). Ontario, Canada: University of Guelph.

الملخص

يعد التقسيم الدقيق لأورام الدماغ في التصوير بالرنين المغناطيسي (MRI) أمراً بالغ الأهمية للتشخيص السريري، وتخطيط العلاج، ومتابعة المرضى. تواجه الطرق التقليدية للتقسيم - مثل العتبة، وخوارزميات نمو المناطق، ونماذج الكنتور النشط - قيوداً كبيرة في تحليل أورام الدماغ. غالباً ما تتطلب هذه الأساليب تدخلاً يدوياً، وتعاني من عدم تجانس الشدة في صور الرنين المغناطيسي، وتفشل في التقاط التعقيدات الشكلية للأورام، خاصة الأشكال غير المنتظمة والحدود غير الواضحة. علاوة على ذلك، تتجاهل طرق المعالجة التقليدية ثنائية الأبعاد (شريحة بشريحة) العلاقات المكانية ثلاثية الأبعاد الأساسية، مما يؤثر على دقة التقييم الحجي.

أحدث التعلم العميق ثورة في تقسيم الصور الطبية من خلال معالجة هذه التحديات. أظهرت الشبكات العصبية التلافيفية (CNNs)، وخاصة البنى التلافيفية بالكامل مثل U-Net و V-Net، نجاحاً ملحوظاً في أتمتة تقسيم أورام الدماغ. تتميز هذه النماذج بقدرتها على تعلم الميزات الهرمية مباشرة من البيانات، ومعالجة السياقات الحجمية ثلاثية الأبعاد، والحفاظ على الاتساق المكاني عبر الشرائح - وهي قدرات أساسية للتقسيم القوي للأورام.

تقدم هذه الدراسة مقارنة شاملة بين بنيتين متقدمتين للتعلم العميق لتقسيم أورام الدماغ: شبكة U-Net ثلاثية الأبعاد وشبكة V-Net. عند التقييم على مجموعة بيانات BraTS 2020، حقق تنفيذنا لشبكة U-Net ثلاثية الأبعاد دقة تقسيم بلغت 98.62% IoU و 0.6792، بينما تفوقت شبكة V-Net بأداء بلغ 98.65% IoU و 0.7330، تؤكد النتائج مزايا شبكة

V-Net في التعامل مع غموض حدود الورم من خلال اتصالاتها المتبقية وقدراتها المتقدمة في تعلم الميزات. تفوق كلا النموذجين بشكل كبير على الطرق التقليدية من حيث الكفاءة الحسابية، والقابلية للتكاثر، والقدرة على نمذجة السمات المعقدة للورم دون معالجة مسبقة يدوية.

تسلط هذه التطورات الضوء على الإمكانيات التحويلية للتعلم العميق في طب أورام الأعصاب، حيث توفر لأطباء الأشعة والأطباء أدوات موثوقة وآلية لتحليل الأورام بدقة عالية.

الكلمات المفتاحية: تقسيم أورام الدماغ، التصوير بالرنين المغناطيسي، طرق التقسيم التقليدية، التعلم العميق،

شبكة U-Net ثلاثية الأبعاد، شبكة V-Net، الشبكات العصبية التلافيفية، مجموعة بيانات BraTS

جامعة البحرين

كلية العلوم



طرق الشبكات العصبية التلافيفية الحسابة والحاسوبية

لتجزئة التصوير بالرنين المغناطيسي

أطروحة مقدمة كجزء من متطلبات الحصول على
درجة الماجستير في علوم الرياضيات

إعداد

هدى جعفر آل سعيد

٢٠٠٧١١٩٧

إشراف

د. عبدالله عيد

(أستاذ مشارك)

د.أنور عبدالحسين محمد

(أستاذ مساعد)

جامعة البحرين

مملكة البحرين

يوليو ٢٠٢٥