3D U-Net: Fully Convolutional Neural Network for Automatic Brain Tumor Detection and Segmentation

A thesis

Submitted in partial fulfillment of the requirements for the Degree of

Bachelor of Science in Computer Science and Engineering

Submitted by

Syed Fahim Ahmed	**150104051**
Tasmia Tabassum Arony	**150104079**
Md. Tariqul Islam Bhuiyan	**150104081**
Fairuz Shezuti Rahman	**150104098**

Supervised by

Emam Hossain

Assistant Professor

Department of Computer Science and Engineering

Ahsanullah University of Science and Technology

Department of Computer Science and Engineering

Ahsanullah University of Science and Technology

Dhaka, Bangladesh

June 2019

CANDIDATES' DECLARATION

We, hereby, declare that the thesis presented in this report is the outcome of the investigation performed by us under the supervision of Mr. Emam Hossain, Assistant Professor, Department of Computer Science and Engineering, Ahsanullah University of Science and Technology, Dhaka, Bangladesh. The work was spread over two final year courses, CSE4100: Project and Thesis-I and CSE4250: Project and Thesis-II, in accordance with the course curriculum of the Department for the Bachelor of Science in Computer Science and Engineering program.

It is also declared that neither this thesis nor any part thereof has been submitted anywhere else for the award of any degree, diploma or other qualifications.

Syed Fahim Ahmed
150104051

Tasmia Tabassum Arony
150104079

Md. Tariqul Islam Bhuiyan
150104081

Fairuz Shezuti Rahman
150104098

CERTIFICATION

This thesis titled, **"3D U-Net: Fully Convolutional Neural Network for Automatic Brain Tumor Detection and Segmentation"**, submitted by the group as mentioned below has been accepted as satisfactory in partial fulfillment of the requirements for the degree B.Sc. in Computer Science and Engineering in June 2019.

Group Members:

Syed Fahim Ahmed	**150104051**
Tasmia Tabassum Arony	**150104079**
Md. Tariqul Islam Bhuiyan	**150104081**
Fairuz Shezuti Rahman	**150104098**

Emam Hossain
Assistant Professor & Supervisor
Department of Computer Science and Engineering
Ahsanullah University of Science and Technology

Prof. Dr. Kazi A Kalpoma
Professor & Head
Department of Computer Science and Engineering
Ahsanullah University of Science and Technology

ACKNOWLEDGEMENT

In the name of Allah, the most gracious and the most merciful. Firstly, Alhamdulillah, all praises to Allah for the strengths and His blessing in completing this thesis. Secondly, we would like to express our deep gratitude to our respected supervisor Mr. Emam Hossain for his valuable and constructive guidance during the planning and development of our undergraduate thesis and research study. His guidance helped and motivated us all the time and he always encouraged us to gather knowledge. We could not imagine having any other supervisor and mentor as better as him for our undergraduate thesis study.

Special thanks to Dr. Kazi A Kolpoma, head of the department, CSE, AUST for her professional suggestions, valuable support and useful recommendations on this thesis. We would also like to extend our gratitude to all our respected teachers of the department of CSE, AUST for their help in offering us some useful resources and precious advice without them we could not complete this research study.

We would like to thank our families: our parents and siblings, for their constant support, blessings and unconditional love which helped us to progress, and our other family members for always there for us. We cannot express our deepest feeling and high appreciation through this acknowledgment since they deserve much more. Last but not the least, We would like to pay our warmest tribute to our classmates and friends because, during this thesis, they helped us time to time regarding different problems, technical things, and mental support.

Dhaka
June 2019

Syed Fahim Ahmed

Tasmia Tabassum Arony

Md. Tariqul Islam Bhuiyan

Fairuz Shezuti Rahman

ABSTRACT

As medical science today is making impacts on every aspect regarding healthcare, neurology also shares the same motive for parenting every possibility of the safe detection and examination of the threatening diseases like brain tumor. A principal problem in brain tumor-related diagnosis, monitoring and treatment is the assessment of the tumor threat level and growth. Automatic segmentation is attractive in this context, as it grants us a faster approach and potentially more accurate description of tumor parameters. In this paper, we have proposed a semantic segmentation technique to 3D MRI images based on popular U-net architecture which is a Fully Convolutional Network (FCN) and robust algorithm. Our presented network is evaluated on Multimodal Brain Tumor Image Segmentation (BRATS 2015) datasets, which contains 274 people's brain MRI images. We have used soft dice loss function to subsist with class disparities and augmented the data to prevent overfitting. Our implemented CNN architecture for detection is validated by obtaining 74.30% accuracy on the classification between normal and tumor affected MRI images as well as 69.25% accuracy on differentiating between high-grade glioma (HGG) & low-grade glioma (LGG) affected patients' MRI images. Cross-validation of tumor segmentation has also displayed that our model architecture can get promising segmentation efficiently than other popular architectures with a dice score of 0.79 for the whole tumor.

Contents

CANDIDATES' DECLARATION i

CERTIFICATION ii

ACKNOWLEDGEMENT iii

ABSTRACT iv

List of Figures viii

List of Tables ix

1 Introduction 1

1.1 Motivation 2

1.2 Objective 3

2 Literature Review 8

3 Methodology 15

3.1 Method of Tumor Detection 15

3.1.1 Pre-processing 16

3.1.1.1 Feature Scaling 17

3.1.1.2 Image Augmentation 17

3.1.2 Convolutional Neural Network 18

3.1.2.1 Convolutional Layer 19

3.1.2.2 Activation Function 21

3.1.2.3 Pooling 23

3.1.2.4 Flattening 24

3.1.2.5 Full Connection 25

3.1.2.6 Softmax Function 27

3.1.2.7 Cross-Entropy 27

3.1.3 Model Architecture of Convolutional Neural Network 28

3.2 Method of Tumor Segmentation 31

3.2.1 Pre-processing 32

3.2.1.1 Re-sizing Image . . . 32
3.2.1.2 Image Augmentation . . . 32
3.2.2 Semantic Segmentation . . . 33
3.2.3 3D U-net Based Deep Convolutional Networks . . . 35
3.2.3.1 Convolution Operation . . . 36
3.2.3.2 Max Pooling Operation . . . 36
3.2.3.3 Up-sampling Operation . . . 37
3.2.3.4 Transposed Convolution Operation . . . 37
3.2.3.5 Activation Function . . . 37
3.2.4 Model Architecture of Segmentation . . . 38
3.2.4.1 Training and Optimization . . . 39
3.2.4.2 Performance Evaluation . . . 39
3.3 Used Dataset . . . 40

4 Result & Discussion **42**
4.1 Detection . . . 42
4.1.0.1 Pre-Processing . . . 43
4.1.0.2 Image Augmentation . . . 43
4.1.0.3 Deep Architecture . . . 43
4.1.1 Normal Brain vs. Tumor Affected Brain . . . 44
4.1.1.1 Accuracy of Normal vs. Tumor affected . . . 44
4.1.1.2 Loss function of Normal vs. Tumor affected . . . 45
4.1.2 High Grade Gliomas vs. Low Grade Gliomas . . . 46
4.1.2.1 Accuracy of HGG vs. LGG . . . 46
4.1.2.2 Loss function of HGG vs. LGG . . . 47
4.1.3 Comparison of Different Methods of Tumor Detection . . . 48
4.2 Segmentation . . . 49
4.2.1 Segmentation Result for BRATS 2015 . . . 49
4.2.2 Performance of the Model for Tumor segmentation . . . 50
4.2.2.1 Accuracy for Tumor Segmentation . . . 51
4.2.2.2 Loss Function for Tumor Segmentation . . . 51
4.2.3 Comparison of Different Methods of Tumor Segmentation . . . 52

5 Future Work **54**
5.1 Limitations . . . 54
5.2 Possible Future Approaches . . . 55

6 Conclusion **57**

References **58**

A 3D Data View 61

B Code for Tumor Detection 63

C Code for Tumor Segmentation 66

List of Figures

3.1 Flowchart of Tumor Detection Method 16
3.2 Convolution Neural Network 19
3.3 Convolution Layer 20
3.4 Rectifier Linear Unit function 22
3.5 Sigmoid Function 22
3.6 Max Pooling 23
3.7 Flattening 24
3.8 Full Connection 25
3.9 Full Connection for detecting HGG and LGG 26
3.10 Full Connection for detecting Normal brain and Tumor Brain 26
3.11 Flowchart of Tumor Segmentation Method 31
3.12 Semantic Segmentation 34
3.13 U-Net Architecture of our Model 38
3.14 3D image of Brain 41
3.15 MRI images of HGG. From left to right: Flair, T1, T1c and T2 41
3.16 MRI images of LGG. From left to right: Flair, T1, T1c and T2 41
3.17 MRI images of normal brain. From left to right: Flair, T1, T1c and T2 41

4.1 Accuracy graph for detecting Normal brain vs effected brain 44
4.2 Loss function graph for detecting Normal brain vs effected brain 45
4.3 Accuracy graph for HGG vs. LGG 46
4.4 Loss Function graph for the HGG vs. LGG 47
4.5 Chart of comparison between different method of tumor detection 48
4.6 Segmentation for T1 MRI 49
4.7 Segmentation for T1c MRI 50
4.8 Accuracy graph for Tumor Segmentation Model 51
4.9 Loss Function graph for Tumor Segmentation Model 52
4.10 Chart of comparison between different method of tumor segmentation 53

List of Tables

3.1 Architecture of the CNN 30
3.2 Applied Data Augmentation Methods 32
3.3 Parameters of U-Net Architecture 39

4.1 Accuracy & Loss Function value for Normal vs. Tumor affected 44
4.2 Accuracy & Loss Function value for HGG vs. LGG 46
4.3 Comparison of methods in respect of accuracy 48
4.4 Accuracy & Loss Function value for Tumor Segmentation 50
4.5 Comparison of different methods in respect of DSC 52
4.6 Comparison of different methods in respect of time for prediction 53

Chapter 1

Introduction

Brain tumor is the most dangerous and vulnerable disease of the time. Medical science is approaching with every perspective for the variable and constant result for this issue. As the law of nature, the tumor especially the ones inflected in brains do not follow any perpetual rules that's why identifying them is consolidated.

Human body is created with many types of cells. Each type of cell has a unique function. When a cell loses control over its growth, they divide frequently without any order. The cells forming a mass of tissue which is extra is called a tumor. Brain tumors are created by abnormal and uncontrolled cell division in the brain that is the deed of the brain. Generally, if the growth of this mass tissue becomes more than 50%, then the patient may not be able to recover or might say incurable. Hence, detection of a brain tumor at its early stage with its proper diagnosis is very important. For the identification of the tumor, there are tests like CT and MRI.

The threat level depends on a combination of factors like the type of tumor, its location, its size and its state of development. Brain tumors are divided into two parts, cancerous (malignant) or non-cancerous (benign). Benign brain tumors are low grade. They are non-cancerous brain tumors that grow slowly and push aside normal tissue but do not spread on the surrounding normal tissue. They are homogeneous, separated, well defined and are known as non- metastatic tumors because they do not form any secondary tumor. Whereas, malignant brain tumors are cancerous brain tumors, which grow rapidly and invade the surrounding normal tissue. They are heterogeneous, not well defined, grow in a disorganized manner and are known as metastatic tumors because they initiate the growth of similar tumors in distant organs. Malignant brain tumors or cancerous brain tumors can be counted as the most dangerous and deadly diseases.

Previously, the process was doctor operated, expensive and the percentage of redemption was low. Even the semi automated procedures lacked the efficiency and valid scheme.

For this reason, Magnetic resonance imaging (MRI) has become a widely-used method of

high-quality medical imaging, especially in brain imaging where MRI's soft tissue contrast and non invasiveness are clear advantages. An important use of MRI data is tracking the size of the brain tumor as it responds (or does not) to treatment. Therefore, this is an automatic and reliable element for segmenting tumor.
The major investigations in developed countries demonstrate that the number of people who develop brain tumors and departed this life is not short in number. That's why functioning a better approach to detect brain tumors consummately is our target.

1.1 Motivation

Around 250,000 people are fighting with primary brain tumor globally and of all primary CNS (Central Nervous System) tumors 85% to 90% transform into brain tumors. This is the 10th leading cause of death for women. The 5-year survival rate for people with cancerous brain or CNS tumors is 34% for men and 36% for women [1]. Overlooking these recent statistics, this paper is formed to work for the automated brain tumor detection with MRI. So that, the medical expense will be lessened as well as the method will be prevailing to people because of the availability and time complexity.
Our whole personality, movement, higher cognitive process is controlled by our brain. Several symptoms can be located due to brain tumors for example, patients face an unanticipated personality change such as confusion, anxiety or mood swings, memory difficulties, communication difficulties, depression etc. Sight problems are faced by 28% of patient where they complained to acquiesce facial palsy/ dry eyes, visual field loss, double vision (diplopia), sensitivity to light (photophobia), double vision (diplopia), abnormal eye movements, aphasia (language and speech problems). Physically almost every patient has to adapt a solute feeling of fatigue which is the most common side effect of brain tumor. Up to 60% of patients will conversance at least one seizure. Children who has faced this vicious disease might have a learning problem [2]. Due to the natural reaction of diagnosis and treatment, most of the people have to undergo this kind of problems but not everyone faces this kind of personality or physical change that would be classed as problematic.
Magnetic resonance imaging (MRI) is a noninvasive method that is used for creating three dimensional (3D) tomographic images of the human body. For the detection of tumors, lesions, and other abnormalities in soft tissues, such as the brain MRI is most often used. Clinically, radiologists qualitatively analyze films produced by MRI scanners.
Formerly, the treatment process was primer and manual where the doctors would have to construct a dreadful operation based on their prior knowledge. Not only the process was expensive, but also the successive rate was quite low. As the problem is sensitive enough where the subject was referred to the brain later, it was determined to call out modern procedures to deal with the detection of tumor.

Recently, computer-aided techniques have been investigated in a great number for analyzing and visualizing magnetic resonance (MR) images. Many researchers have researched on detecting and quantifying abnormalities in the brain. For this reason, identifying the tumor in MR images of the brain is an important step in this process. Another focused step for computer-aided analysis is data quality and image quality assurance. Most of the MR images contain unwanted intensity variations due to imperfections in MRI scanners. The accuracy of automated analysis can improve to remove or reduce these kinds of variations and for these reasons we are motivated to endeavor in this field.
Even with these new age methodologies, the accuracy level of splotch the brain tumors are not that much prominent. The main reason behind this might be that, the biomedical nature doesn't follow any certain or practical rule. Most of the patient effectuate a uniquely positioned and catalyzed tumor. Even if the network can successively detect the tumor, to classify it requires a lot more exertion. It can be easily assumed that even a trained machine can't be completely convicted with this. That's when the deep neural network came in hand as the machine is trained with similar data and examined over multiple times. The network can explore prior knowledge and make a decision. Earlier the networks weren't that much impeccable but with time strong and efficient algorithms have been generated and the different techniques and approaches were invoked and thus the validation has been escalated. While these advanced neural networks, support vector machine (SVM) and rectifier classification are being used in this field the performance rate is not that up to the mark till now. The precision and delicacy are not top notches compared to current technology. With these subjects in mind, we are intended to work with this topic to make a well-designed network that can generate an immaculate outcome.

1.2 Objective

CNS, the central nervous control system is formed with the brain and spinal column by which all the main functions are being controlled in our cellular system, and these functions include thought, speech and body movements. It can be accounted that a person's operations, motions, performances can be affected if a tumor grows in the CNS. Brain tumors are initially segmented into two ways, Primary tumors are those whose initial positions are in the brain then they spread in other sections of the body. Primary tumors are further classified into two parts, high grade, and low grade. A high-grade tumor grows rapidly, on the other hand, the low-grade generally grow slowly but it can also be converted into a high grade. If the tumor starts in another part of the body, such as lung, colon or breast and then they spread into the brain they are called secondary brain tumors or brain metastases that are also cancerous. There are many types of primary brain tumors but our main concern is gliomas. The gliomas are assumed to glean from glial cells or glial precursor cells. They are

assigned to grade by which their behavior is measured, low-grade glioma (LGG) (grades I and II) and high-grade glioma (grade III anaplastic glioma and grade IV glioblastoma). The low-grade glioma has a dormant growth and insinuates of white matter with subtle neuro-cognitive deficit and seizures but in some cases, it can adapt some regional changes and be transferred into a more malignant variant. High-grade glioma may present from the beginning (primary) glioblastoma or as a transformation of a lower-grade tumor (e.g. secondary glioblastoma). Gliomas typically grow within white matter and exhibit irregular growth patterns along white matter fibers, infiltrating the surrounding brain. Thus, they present devious boundaries that may be visually intangible on conventional MRI images. A fixed measurement of the tumor boundary and appraisal of tumor size is needed for patient management in terms of planning the accurate treatment and monitoring treatment response.

For diagnosing brain tumors, medical imaging techniques such as Computed Tomography (CT), Positron Emission Tomography (PET) and Magnetic Resonance Imaging (MRI) are used. CT exercises the radioactive rays to penetrate the human body, and the imaging is based on the different characteristics reflecting the rays of different tissues. PET, injects with radioactive drugs to the human body, and the drugs flow to all the cells, tissues and organs with the blood in the whole body. In the MRI process, there are neither instruments inserted nor any medication injected into the human body. The process is considered safer as the human body is not facing any radiation damage. In addition, MRI has high resolution and accurate positioning of soft tissues and is sensitive to the characteristics of diseases, that's why it is easy to detect the tumor and tumor progress modeling process and treatment module. Besides, the MRI image produces more information about the given medical image than the CT or ultrasound image. It also states detailed facts about brain structure and anomaly detection in brain tissue. Unlike from foregoing time it is a smooth way to use the automated methods for brain tumor detect and type cataloging using brain MRI images by just scanning and freighting medical images to the computer, thus it is especially suitable for the diagnosis of brain diseases.

Our primary concern was to classify the gliomas but later we changed our approach and focused on segmenting the tumor rather than classifying because before applying any treatment module, it is important to segment the tumor in order to protect healthy tissues while damaging and destroying tumor cells during the therapy. Brain tumor segmentation involves diagnosing, delineating and separating tumor tissues, such as active cells, necrotic core, and edema from normal brain tissues including Gray Matter (GM), White Matter (WM) and Cerebrospinal Fluid (CSF).

An image is split into regions with connate properties such as color, brightness, texture, grey level and contrast and this process is called segmentation. In medical segmentation the scheme is to Identify Region of Interest (ROI), examine anatomical structure i.e. spot tumor, scratch other irregular characters, expedient tissue volume to expedient growth of

tumor (also reduce in size of tumor with treatment), innovate a premium plan for treatment prior to radiation therapy in radiation dose calculation. Segmentation of medical images is challenging due to their complex nature, poor image contrast, artifacts that result in missing or diffuse tissue boundaries and rarely have any simple linear feature. Due to the presence of artifacts, intensity inhomogeneity, partial volume effect and closeness in the gray level of different soft tissue the result of segmentation can be affected. With appropriate filtering technique (noise remove, low contrast features brightening, detecting the non-sharp edge), proper image restoration approach (motion artifacts) and lastly by applying the requisite algorithm for measuring partial volume and intensity inhomogeneity the MRI images can be evolved.

Previously, a number of MRI images has to go through the manual annotation and segmentation where the radiologist slices the image in multiple parts, detect the tumor and determine the classification. The decision will have to encounter a large intra and inter-rater variability which makes the process more time-consuming. However, to evaluate the results of semi-automatic and fully automatic methods this manual segmentations are vividly used till now, functionally known as ground truth. Later in the Semi-automatic methods, the process became a little handier as it required less manual interaction. Although there were certain methods to ensue. Initialization, to define the region of interest (ROI), measure the approximate tumor region for which the algorithm will work. Pre-processing, to adjust the measurements of input images. In addition, these algorithms can also supersede if in response they are fed right adjustments. Users can also modify and repeat the process if the result meets deficiency.

In recent times, Neural Networks (NN) and Support Vector Machine (SVM) is the usually dominant methods for their good enactment. However freshly, Deep Learning (DL) models established a gripping trend in machine learning as the subterranean architecture can efficiently represent complex relationships without needing a large number of nodes like in the superficial architectures e.g. K-Nearest Neighbor (KNN) and Support Vector Machine (SVM). Consequently, they grew quickly to become the most advanced level to the health informatics areas, for example, medical image analysis, medical informatics, and bioinformatics. The implementation and usage of a neural network are paradigmatic of the human brain. Vector quantization, approximation, data clustering, pattern matching, optimization functions, and classification techniques are the important countenance for brain tumor detection and they are done by the neural network. Based on their interconnections three types of neural networks are feedback, feedforward, and recurrent network. The Feed Forward Neural network is further classified into a single layer network and multilayer network. In the single layer network, the hidden layer is not present. But it contains only the input and output layer. However, the multilayer exists of the input layer, hidden layer, and output layer. The closed-loop based feedback network is called a recurrent network. In the normal neural network, images are not scalable. But in convolution neural network, images

are scalable (i.e) it will take 3D input volume to 3D output volume (length, width, height). The Convolution Neural Network (CNN) consists of an input layer, convolution layer, Rectified Linear Unit (ReLU) layer, pooling layer, and fully connected layer. In the convolution layer, the given input image is separated into various small regions. Element-wise activation function is prosecuted in ReLU layer. Pooling layer is usually optional. However, the pooling layer is mainly used for downsampling. In the final layer (i.e) fully connected layer is used to generate the class score or label score value based on the probability in-between 0 to 1. Training and testing are two periods for the CNN based brain tumor classification. Multiple labels are issued for different categories. In the training phase, preprocessing, feature exaction and classification with Loss function is performed to make a prediction model. In the preprocessing image resizing is applied to change the size of the image. The brain image dataset is normally taken from various sources. If the training emerges from the starting layer, it has to train the entire layer up to ending layer hence time consumption is higher which will strike the performance.

To avoid this kind of problem, in our proposed method, for the detection part, we want to feature the maps with a particular dimension in the convolutional layer. With a finical number of convolutional layers, the activation function, ReLU will be used for input and sigmoid function for the output. Max pooling will be there for dimension reduction as well as for the collective feedbacks flattening part will have to be passed down. Later in the full connection layer, the input and output layer will be determined to connect. Further, as binary entropy, a loss function will be applied.

The convolution architecture that has accustomed in this paper for segmentation is called 3D U-Net. This architecture was initially preferred on small datasets as it generates larger features even from the smaller datasets. Later it was improved in Olaf et al's [3] where they have replaced the pooling layer with upsampling operators. This particular operation has increased the resolution of the output. To localize the tumor the up samples are combined with the high-resolution features from the contracting path. The foremost feature of this method is that it allows the network to construct a symmetric path by propagating the context information to higher resolution layers as it contains a large number of feature channels, therefore yielding a u-shaped architecture.

Our method is designed to volunteer a threefold output, from the given image firstly it will generate a difference between normal and abnormal brain image, it will distinguish between tumor and non-tumor MRI images. Secondly, it can classify between high-grade gliomas and low-grade gliomas. Thirdly, it will predict the result and compare the output with the ground truth to expedient the accuracy. Here it has to be remembered that the ground truth is the container of four images of that particular MRI based on the different type of tumor (T1, T2, Flair & MPR), generated by the dataset which already consists the exact verified result. In that way, the accuracy and dice score which we get is indubitable and authentic. We contemplate that this method will reduce the time consumption by a

great factor and help to detect the tumor with absolute accuracy.
Further in this book, Chapter 2 provides the literature description and related works according to our study. Chapter 3 describes the necessary methods for our thesis study. Chapter 4 has the result of our work model which is described briefly. Lastly, in Chapter 5 & Chapter 6, we have discussed about the limitation of our work and future plan with it & the conclusion.

Chapter 2

Literature Review

The number of publications devoted to automated brain tumor segmentation has grown exponentially in the last several decades. In the field of brain tumor segmentation & detection, there are various types of methods and algorithms previously where some of them used neural networks and some of them used other machine learning techniques. Other related papers have used morphological operations to solve this problem. But from the study, we have able to find out that neural network based especially convolutional neural network based methods work more efficiently than other types of methods regarding segmentation problem. Based on various types of dataset whether it is a 3D or 2D the pre-processing techniques differ from each other. To calculate the performance of these methods, many of them used accuracy, sensitivity or dice score as performance measurement metrics. Among all these methods, the state of the art methods dice score remains 85%.

As stated before that CNN works more efficiently than other methods in the case of brain tumor segmentation. Sergio Pereira et al. [4] and Mohammad Havaei et al. [5] implemented this neural network for their proposed architecture. Sergio Pereira et al. [4] have used deep architectures with small convolutional kernels for the segmentation of gliomas in MRI images. To correct the intensity of the same tissues to varying across the image N4ITK method is used. They have also applied intensity normalization method on each sequence. Erroneously classified tumors are dealt with volumetric constraints by removing clusters in the segmentation obtained by the CNN that are smaller than a predefined threshold value. The proposed architecture obtained the second position with a DSC score of 0.78, 0.65 and 0.75 in complete, core and enhanced regions, respectively in BRATS 2015 challenge dataset. Similarly, Mohammad Havaei et al. [5] proposed a fully automatic brain tumor segmentation method based on Deep Neural Networks (DNNs). They also had used CNN which implements a novel two-pathway architecture that learns about the local details of the brain as well as the larger context. They also employed a novel cascaded architecture as an efficient and conceptually clean alternative to popular structured output methods. Both

of the CNN based structured methods give higher performance metrics in case of dice score as well as reduced the computational time to predict the segmented images of the tumors. Tustison's method takes 100 min to compute predictions per brain as reported in Menze et al. (2014) [6], while the InputCascadeCNN takes 3 min, thanks to the fully convolutional architecture and the GPU implementation, which is over 30 times faster. In Javeria Amin et al. [7], Deep Neural Network (DNN) architecture is used to detect brain tumors, as this is a very popular section of research from magnetic resonance images (MRI). This method is used to segment the tumor. Also for classification, 7 layers are used including 03 convolutional, 03 ReLU and a softmax layer. Multiple patches are made using CNN architecture so that it could be used by DNN. Using 8 benchmark dataset containing BRATS 2012, 2013, 2014, 2015 and ISLES 2015 and 2017, results are verified by accuracy (ACC), sensitivity (SE), specificity (SP), Dice Similarity Coefficient (DSC), precision, false positive rate (FPR), true positive rate (TPR) and Jaccard similarity index (JSI). 99.8% DSC on Flair, 100% results on DWI, 98.0% on T2, 97.4% on T1 and 95.4% on T1-contrast modalities are achieved as a result.

Twenty state-of-the-art tumor segmentation algorithms were implemented to a set of 65 multi-contrast MR scans of low and high-grade glioma patients in Menze et al. [6]. After several numbers of evaluation of different types, it was published that no algorithms were able to detect 100% of segmentation and different algorithms worked best for different subregions of the brain. While the dice score of segmenting core and the active region is 70% and 60% but on average 80% of the whole regions can be endowed by these algorithms.

In a similar fashion, Ali Isin et al. [8] presented reviews of the forefront methods based on deep learning and a brief overview of traditional techniques. While the paper has concise the traditional methods it also focused on the CNN (Convolutional Neural Network) and it is one of the best based on high performance, excellence in complex features for both brain tissue and tumor tissues directly from MRI images. Later the decision was made that future improvements and modifications in CNN architectures and addition of complementary information from other imaging modalities such as Positron Emission Tomography (PET), Magnetic Resonance Spectroscopy (MRS) and Diffusion Tensor Imaging (DTI) may improve the current methods also for better diagnosis and the development of clinically acceptable automatic glioma segmentation methods it will eventually lead the path.

Biomedical images usually contain detailed patterns of the imaged object and brain tumor is one of them. The edge of the object is variable and to cope with the segmentation for the objects with detailed patterns, Long et al. [9] proposed an architecture that combined the high-level representation from deep decoding layers. The appearance representation from shallow encoding layers produces detailed segmentation. This method has demonstrated promising results on natural images [10] and can also be applied to the biomedical images [11]. Ronneberger et al. [3] introduced the U-Net, which employed the

skip-architecture, by solving the cell tracking problem. Hao Dong et al. [12] have narrated down flaws of the traditional way of segmentation of brain tumors from MRI images and using the BRATS-2015 which contains both LGG and HGG patient's MRI as a dataset they have formatted a new approach 2D fully convoluted segmentation network based on U-Net architecture for overcoming those frailties. To boost the segmentation accuracy, data is augmented using various techniques. Besides, soft dice based loss function has been exercised for adapting the unbalanced samples. The network has been implemented using the TensorFlow and the TensorLayer libraries. Compared with manually delineated ground truth, this fully automatic method has obtained promising results. Also compared to other leading methods, this paper has achieved comparable results for delineating the complete tumor regions, and superior segmentation for the core tumor regions. There are still some boundaries of the current work like this segmentation method has been evaluated using a five-fold cross-validation scheme also several parameters need to be carefully tuned in this network. Moreover, the framework is less successful in segmenting enhancing tumor regions of the LGG cohort but as a solution, they suggested to stack all the multimodal MRI channels and performing joint training with HGG datasets. This method achieved 86% DSC for the combined complete tumor segmentation.

Except for CNN, there are other neural networks had been employed in various other biomedical image segmentation. El-Sayed A. El-Dahshan et al. [13] compared different segmentation and classification techniques. By studying the other methods, they proposed a new hybrid technique for computer-aided diagnosis of human brain tumors through MRI. The median filter was used to remove the noise without disturbing the edges. They have used feedback pulse-coupled neural network (FPCNN) which is a modification of PCNN for this stage. To extract features from MRI, the discrete wavelet transform was used which gives information about the signal both in frequency and time domains. The principal component analysis was implemented to decrease the data dimensions while retaining as much as possible of the variation present in the data set to process the data faster and effective. The back-propagation neural network (BPNN) helped the proposed approach to classify MRI into the set of target categories (normal or abnormal) based on feature selection parameters. Confusion matrix which contains information about actual and predicted classifications was used to evaluate the performance of the CAD system. This hybrid technique reached 99% accuracy, 92.8% specificity as well as 100% sensitivity from the dataset they used. In Kailash D.Kharat et al. [14], two Neural Network techniques for the classification of the magnetic resonance human brain images are proposed. Using discrete wavelet transformation (DWT) and principles component analysis (PCA) the features of MRI images were extracted and the dimensions were reduced respectively. Feedforward artificial neural network (FF-ANN) and the second classifier based on Back Propagation Neural Network (BP-ANN) was used for classifying subjects as normal or abnormal MRI brain images.

Detection and classification of brain tumor modified image segmentation techniques were applied on MRI scan images by Dina Aboul Dahab et al. [15]. Gaussian smoothing filter is used to remove noise from the images. The most edges are found out by the Canny algorithm. A modified version of the conventional PNN method helps to classify the brain tumor. 64 images were tested by their proposed system and got 100% accuracy when the spread value is equal to 1. Similarly Pauline John [16] used PNN algorithm for segmentation. Wavelet, a powerful mathematical tool is used for feature extraction. The statistical features from MR images are acquired by using Gray Level Co-occurrence Matrix (GLCM). PNN method is used for implementing an automatic MR image classification of brain tumors into normal, benign and malignant. The accuracy of classification is found to be nearly 100% when the spread value is set to 1.

Clustering algorithms have always been popular in the case of image processing especially on the detection of a contour. An advanced method of segmentation has been processed down by Eman Abdel-Maksoud et al. [17]. In the paper, they have used the K-means clustering technique integrated with the Fuzzy-C-means algorithm. Although the K-mean algorithm can detect a brain tumor faster than Fuzzy C-means. Fuzzy C-means gives a more detailed result of the brain tumor. But the original Fuzzy C-means algorithm has a weakness of segmenting the image corrupted by noise, outliers, and other imaging artifacts. That's why their developed approach integrates the K-means clustering algorithm with the Fuzzy C-means algorithm to detect brain tumor accurately and in minimal execution time. The framework is divided into four parts, pre-processing (de-noising by the median filter and skull removal by BSE (Brain Surface Extractor), clustering (integration of K-means and Fuzzy C-means), extraction and contouring (thresholding and level set), and validation stages. The method is implemented on three types of dataset, Digital Imaging and Communications in Medicine (DICOM) data set with no ground truth, Brain Web data set and BRATS database from Multimodal Brain Tumor Segmentation which is included of ground truth. The results of this experiment clarified our hybrid clustering method (KIFCM) can detect a tumor that cannot be detected by Fuzzy C-means with less execution time rather than both FCM (Fuzzy C Means) and Expectation Maximization (EM). But it takes longer time than K-means (KM) and Mean Shift (MS). In contrast, [18] of Matthew C. Clark et al. built an automated system in which segment and level tumors of human brain MRI images. The initial image segmentation is done by applying unsupervised clustering algorithms. With an integrated knowledge-based technique containing system, these images are processed. These MRIs are T1-weighted, proton density, and T2-weighted images. A total of 120 brain slices containing radiologist diagnosed glioblastoma-multiform tumor was there for processing the system. In this system, three-volume data sets for training and thirteen unseen volume data sets for the test are used. Results of ground truth are pretty well. Per slice basis and tumor volume tracking, both are corresponded to ground truth.

One of the popular method proposed by Stefan Bauer et al. [19] has also been a clustering based algorithm. MRI images are need to be analyzed for studding brain tumor to gaining knowledge because of the increasing number of patients. Automated methods like region or edge-based method, classification and clustering method, segmentation and tumor detection algorithm are implemented with different directions. These methods are applied on standard clinical images protocol. Segmentation method results in reasonable computational time. Tumor detection algorithm detects the growth of tumor to develop a health chart and pathologic patients.

Previously, machine algorithms like SVM, KNN, random forest classifier, etc. were used before the implementation of neural network based methods in the field of brain tumor detection and segmentation. B. Devkota et al. [20] approached for early-stage detection of brain tumor and initially, it has followed the method of MSFPS abbreviated as Image pre-processing has been done by using the median filter. Segmentation of the pre-processed image using mathematical morphological operations, feature extraction and reduction (extraction of first statistical features and textural features), Features reduction using principal component analysis, Classification using Support Vector Machines with GRB kernel and lastly Evaluation of the classification. While they have followed all the methods of this particular process they have changed the segmentation process Spatial Fuzzy C-Means (SFCM) by another method of high peak signal to noise ratio (PSNR). They have established a new algorithm for segmentation and gathered the accuracy of 92% accuracy for detecting cancer which is higher than the current model and classifier has an accuracy of 86.6% and as a limitation this proposed method does not classify into different stages as stage I, II, III, IV of cancer, has not been tested up to the evaluation stage, to compare with the current best solution and they have not been tested on a large set of data.
Stefan Bauer et al.'s [21] another paper on an automatic method has been proposed for the segmentation of brain tissue using support vector machine classification. This classification is based on hierarchical conditional random field regularization which is more effective than other machine learning classifications. Initially, the images undergo a pre-processing pipeline. First-order textures can be completed fast and easily from small patches around each voxel in all for modalities. Classification is done using a soft margin SVM classifier which is based on the LibSVM implementation. Regularization is done in two different stages using a Conditional Random Field method. The gross tumor volume for intra-patient regularized and inter-patient regularized are 0.84 and 0.77 respectively.
Discrete wavelet-based GA (generic algorithm) was implemented in [22] to detect the MRI brain images. Following the process of the optimal or sub-optimal data partition, they have implemented k-Means unsupervised clustering methods into Genetic Algorithms for guiding the last Evolutionary Algorithm but as there was the problem of a non-trivial search because of its intrinsic NP-complete nature. The paper solves using soft threshold DWT for enhance-

ment and genetic algorithms for image segmentation. The developed method uses the GA, to solve optimization problems with a large search space (label of each pixel of an image). The developed method can also integrate some prior knowledge such as local ground truth to work more efficiently. The system shows two types of result, first, it establishes that the method achieved SNR value from 20 to 44 and secondly the segmentation accuracy was increased from 82% to 97%.

A segmentation method in medical image analysis can be revolutionary and provide a better diagnosis to the patients. In the year 2010, [23] was published which implemented a Hierarchical self-organizing map (HSom) to segment images in the two-phase segmentation method which was giving high performance in glioma diagnosis. In the first phase, the data of MRI images are collected from parents' database. The unwanted distortions are erased in this phase. Then HSom is used to segment the image. By using real-life doctor given dataset, results are achieved in different higher level pixels and quick execution time. The Weight vector for Hsom is 3×3 is 14, 5×5 is 8, 7×7 is 15, 9×9 is 23 and 11×11 is 32. The Execution time in Hsom of 3×3 is 13.76, 5×5 is 14.96, 7×7 is 15.20, 9×9 is 11.05 and 11×11 is 11.53. This paper has vast information, knowledge, and data about brain tumor detection.

In Sudipta Roy et al. [24], there are two segmentation processes- Threshold segmentation, Water-shade segmentation which were used to detect the tumor, segment it and calculate the area of the tumor. 6 sets of MRI imaged has been chosen for the operation and some fundamental image enhancement and noise lessening procedure are applied after that the mentioned algorithms were applied where first one for detecting the brain tumor where the inputs MRI of brain image and Output is Tumor portion of the image and the second algorithm is for Area Calculation where the Input is Tumor portion of the image and output is area of the tumor.

By diagnosing the brain cancer using MRI images, the proposed system by Umit Ilhana et al. [25] developed a unique segmentation technique. The dataset is collected from the Cancer Imaging Archive (TCIA). The proposed approach consists of the steps, pre-processing (morphological operations, pixel subtraction), segmentation (threshold based segmentation, proposed segmentation), and filtering (median). It has a 94.28% recognition rate on the images with a tumor and a 100% recognition rate on the ones without a tumor. The overall success rate of this system is 96% but as the process is not entirely automated it has several limitations and time-consuming issues. K. SUDHARANIa et al. [26] represented a new approach for the automatic segmentation of tumors from T2 weighted MRI images. The algorithm works in several steps- brightness adjustment, re-sampling, color plane extraction, histogram processing, tumor measurements, thresholding, Fourier transform, lookup table, advanced morphology, and practical analysis. It works better including ground truth images and low-intensity regions that can be detected with this methodology. The result is shown

in four firewalls: Sensitivity, Specificity, Accuracy and Similarity and they present a satisfactory result.
Detection of the tumor region and to calculate its area based on morphological operation, is another efficient way which was implemented by Madhumantee Naskar et al. [27]. The image segmentation process is done by threshold segmentation methods. The median filter is used to remove noise. The proposed method is time-consuming but generates errors.

Chapter 3

Methodology

Methodology represents the working procedure of a thesis study. After research on the particular field, simplest and feasible methodology is chosen to apply on the dataset. Its a step by step procedure.
In this chapter, the method or working process of our proposed system will be discussed, how the research was conducted and the reason for following the methods. In our model, convolutional neural network is used to detect the tumor and U-Net is used for segmentation.
The basic of the thesis is Deep Neural Network approach. Using Artificial Neural Network's part Convolutional Neural Network or CNN, layer architecture is created and used to give decision of the presence of tumor. Also semantic segmentation is applied to segment the tumor. A structure of semantic segmentation is U-Net Structure which is used in our model. The MRI image of the input is being pre-processed and the feature is extracted which is used to detect tumor using CNN.
For segmentation, input image is also pre-processed and pre-processed images are applied to create blocks in U-net architecture. This architecture is created for segmenting the tumor.

3.1 Method of Tumor Detection

Tumor detection is one of the purpose of our thesis. Detecting tumor means giving the decision of the existence of the tumor. Using our detection method, we can find out the accuracy of our system. By training the data after pre-processing, system will be able to give a prediction about tumor. It will also produce the prediction of having high grade gliomas(HGG) and low grade gliomas(LGG).
To detect the tumor, Convolutional Neural Network is used after pre-processing to give the prediction as CNN's last layer is full connection layer which judge the features and produce

the probability of having tumor or the tumor's nature.

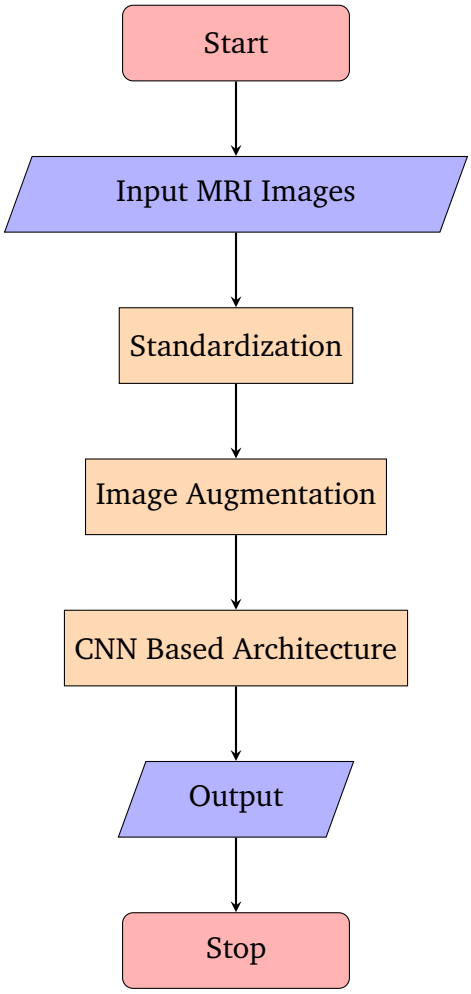

Figure 3.1: Flowchart of Tumor Detection Method

3.1.1 Pre-processing

Pre-processing is an important part of image processing based system. A model's working ability directly depends on the pre-processing of the data. Pre-processing means process the data before feeding it into the model as a set of data can contain much unnecessary information which can only increase the delay of the output. Pre-processing of the data can skip this information and extracts only the integral data. In other words, the raw data of the source dataset is converted into clean data in more understandable way for the architecture. Different model uses different types of data. Not all the model can support every dataset. That's why pre-processing is used.

MRI image is the dataset for our model. These images may have different intensity level, scaling, and many distortions. For the feature to properly extract, we need feature scaling. To scale the data, we used standardization. Also, image augmentation is needed for an enriched dataset to be used.

3.1.1.1 Feature Scaling

The dataset that is used in a model varies most of the time in values. Some data contains a high value and some data contains a very low value. This variation can distort the result of the process because the low values get neglected and only high magnitude values are used. As the machine learning model uses Euclidean distance, these low value and high-value distance is a huge problem which creates suppression.
There are Four widely used way for feature scaling which are,

1. Standardization
2. Mean Normalization
3. Min-Max Scaling
4. Unit Vector

To prevent the domination of any independent larger value over smaller value, we used feature scaling in our model. MRI image has a different intensity level which needs to be standardized. To fix this matter we have used standard scaling in mathematical term.
Standard scaling or standardization means we are putting our variable in the same range on the same scale so that no variable is dominated by others. For each observation and features, we withdraw the mean value of all the values of the feature and divide it by the standard deviation. Our motive is to get comparing measurement of the values so that the value of the feature weighted around 0 to 1 and no value is ignored. Standardization is calculated by equation 3.1,

$$X_{stand} = \frac{x - mean(x)}{standardDeviation(x)} \tag{3.1}$$

In our model, Pixel of MRI ranges between 0 and 255. Here we can convert the different pixel of the MRI images in the range between 0 and 1 by standardization and use it without any problem. Because of the scaling the result of the model is more feasible.

3.1.1.2 Image Augmentation

Machine learning increases performance when a great amount of data is given to it. Data Augmentation is a way to create new artificial data by modifying existing data. Data augmentation improved the accuracy of Deep learning model as the training data is now increased. Image augmentation is one of the data augmentation processes which is obtained by transformation of the images.
Image augmentation is needed to prevent overfitting of the images. If it is not done then

the result of the training set will be more accurate but the test set result will be much lower accurate due to an overfit on the training set.
Which image is needed to augment is important as some data can't give desire accuracy. So the image augmentation is first applied in an isolated method and measure the accuracy than in the main method.
To train a model with images we need huge data to correlate with each other. It might get a chance to correlate with the training set but the problem happens when there comes new data. Now we are working with 960 images, 768 is the training set. This is not enough data to work with. To prevent this problem, data augmentation is used.
Data augmentation will create many batches and each batch will apply some random transformations on a random selection of our image just like rotating, flipping, shifting or even shearing. With this, we can get a lot more material for training. So this is a technique that allows us to enrich our dataset, our training sets without adding more images.
In our method, we have used shear range, zoom range, and horizontal flip. Shear range and zoom range both values are 0.2. We have flipped the image horizontally by 90 degrees. This creates new data without adding any image and extends the dataset. The model will get more data and trains itself with different possible data so that the testing dataset gives more accuracy despite different data.

3.1.2 Convolutional Neural Network

CNN or Convolutional Neural Network is an advanced and feed-forward artificial neural network of deep learning to analyze visual images. It is a basic part of deep learning where the system learns more properly and gives different activities of the human brain. CNN is a version of multilayer perceptron which is an architecture for supervised learning of a classifier and gives the decision of a particular value whether it belongs to a class or not. But perceptron is not regularized because of data over-fitting. CNN regularizes data by adding wights to the loss function.
We have used Convolutional Neural Network to detect tumor from MRI images. Most of the related work that we have studied do not uses CNN directly as this method is more complex. CNN is a developed image classification algorithm than others. The traditional algorithm mostly gives a handmade decision where CNN is a neural filtered algorithm that is self-taught. CNN works more like a human brain with neuron. Traditional Neural Network needs to reduce the resolution of the input image which is not ideal. This problem is reduced by CNN as it works with layers and the neural layers are arranged in such way which gathers the whole data ignoring the problem.
An input MRI is given. This MRI image is a multichannel image. So we worked with volume of the image. The tumor is mentioned as the feature detector. After going through a number

of layers, the CNN gives the decision. There are 11 layers in our convolutional network. In these layers, we have used following concepts.

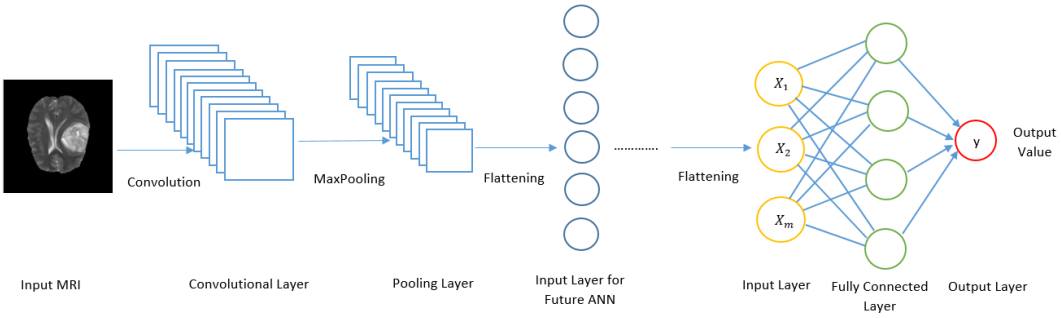

Figure 3.2: Convolution Neural Network

These 11 layers lead us in a probability of having a brain tumor and every layer works with a new input, processed from the previous layer. The 3 main layers that we have used here are,

1. Convolutional Layer
2. Pooling Layer
3. Full Connection Layer

3.1.2.1 Convolutional Layer

The convolutional layer is the main part or core part of Convolutional Neural Network. This part of the CNN makes the input image converted into feature maps.
The convolution layer is applied to get a feature map from the MRI images. The feature map is connected to the previous layer with the weight of the kernel. So, if we have 3×3 size of filter, and depth of 3, then the convolution layer has $3 \times 3 \times 3 = 27$ weight connected with input volume. The convolution layer has fewer weights to train than the FC dense layer of CNN. Which makes this layer easier.
The image of MRI is taken as input into the model. The feature detector slides the image to detect the match. When the match is found, the value of that particular cell of feature map is higher. We run the convolution operation to get a feature map with a higher possibility of tumor. In our convolutional layer, we created 32 feature maps with the dimension 3×3. The connectivity of each neuron in the input volume is discussed. The arrangement of the neuron in output volume depends on three hyperparameter: depth, stride and zero padding. This arrangement is called spatial arrangement.

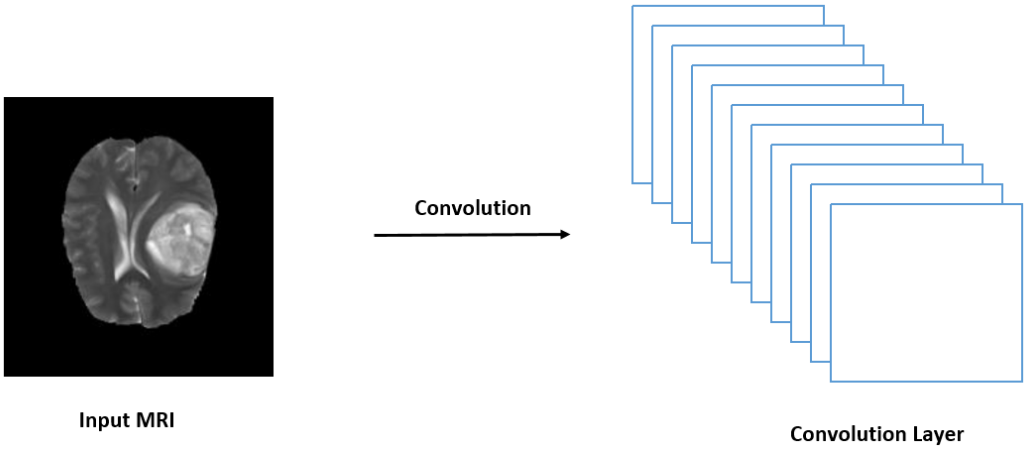

Figure 3.3: Convolution Layer

1. **Depth:** Depth of a output volume corresponds to the number of filters that we wanna use. When an image is taken as input, the raw images' color channel is the depth in first convolutional layer. Next convolutional layers just uses the filters as their depth.
2. **Stride:** Stride should be specified as this is needed for the slide of the filter. When stride is 1×1 then the filter moves by 1 pixel at a time.
3. **Zero-Padding:** Sometimes this is needed to maintain the size of the input image. But not always is used.

These spatial hyperparameter can give the formula for calculating the output volume or spatial size of the volume in each convolutional layer,

- **Input Size:** $D_1 \times W_1 \times H_1$, where D_1 is the depth or color channel, W_1 is the width of the input and H_1 is the height of the input
- **HyperParameters:**
  - stride S
  - receptive field size or filter size F
  - the number of zero-padding P
  - number of filters K
- **Output Size:** Produced volume size is $D_2 \times W_2 \times H_2$ where,
  - output depth $D_2 =$ K
  - output width $W_2 =$ (W_1 - F + 2P) / S + 1
  - output height $H_2 =$ (H_1 - F + 2P) / S + 1

- weight per filter, F . F . D_1 and total of (F . F . D_1) . K weights with K biases
- In the output, the depth slice of d is the result of performing a valid convolution operation of the d-th filter over the input with S stride, and then offset by d-th bias.

3.1.2.2 Activation Function

Basically activation function decides whether a neuron should work or not. It is really important as it helps CNN to learn new data. This function activates the neuron related to the information given with high weight and ignores the rest. Activation function could be both linear and non-linear.
The activation function is very important as without it CNN works as a linear regression model that cannot solve a complex problem. So many complicated works will not possible if there is not any activation function. It also makes back-propagation possible as back-propagation needs weight.
There are 4 types of activation function which are,

1. Sigmoid or Logistic Activation Function
2. Tanh or Hyperbolic Tangent Activation Function
3. ReLU (Rectified Linear Unit) Activation Function
4. Leaky ReLU

Activation fiction has an important role in our CNN. It decides which node of a layer is going to work next. By activating the node of the layer it works. Among the popular activation function, we used ReLU and Sigmoid Function.

Rectifier Function or ReLU:
Rectifier function increases non-linearity in our network. Images are highly non-linear. Different border, different color, different elements are which an image is made of. When we run the convolution network and create a feature map, we risk that images real feature or can create something linear. So we need to break the linearity. That's why we need this function.
ReLU gives max value for all positive values and zeroes for all negative values. For this reason, ReLU is really simple as avoids complex values. In CNN, ReLU is most commonly used. Equation used by ReLU function is,

$$\varnothing(x) = max(x, 0) \tag{3.2}$$

ReLU is most widely used over other activation functions because it doesn't activate all the neurons at a particular time. As we can see, ReLU gives zero when it is negative which

makes the neuron inactivate and the network is more efficient.
We used the ReLU function in our input. It gives value 0 for all the negative values and 1 for otherwise. The curve of the function is,

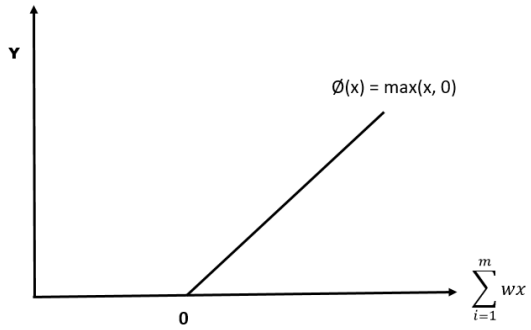

Figure 3.4: Rectifier Linear Unit function

Sigmoid Function:
Sigmoid activation function is one of the commonly used activation function. It is an S-shaped function that is non-linear. This function is continuously differentiable which makes the graph of the function smooth. The equation of the function is,

$$f(x) = \frac{1}{1 + e^{-\beta x}} \tag{3.3}$$

Sigmoid curve represents that a small change in x of the curved region can make a huge change in y. This feature of the function helps to classify a value in the proper class.
Sigmoid Function is used in our output to predict the probability of the data weight. Like ReLU function, sigmoid also gives the value between 0 and 1. So we have used it in our output layer as for the output layer we need binary function.

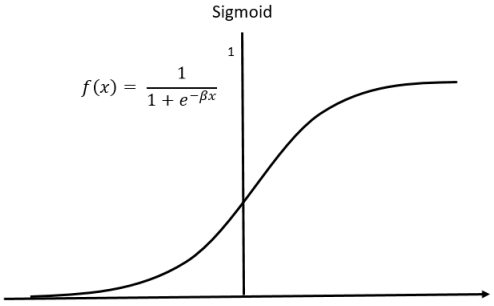

Figure 3.5: Sigmoid Function

3.1.2.3 Pooling

Pooling is one of the building layers of Convolutional Neural Network. It reduces the spatial size of the data and minimizes parameter and computational hazard in the network.
If the feature itself is a bit distorted, our neural network has to have some level of flexibility to able to still find that feature and that is what pooling is all about. It pulls out the feature that is needed from the image.
There are many types of pooling. Widely used pooling are,

1. MIN pooling
2. MAX pooling
3. SUM pooling

As max pooling is most commonly used in CNN, we have used 2×2 max pooling in our model. This maximizes the feature of the data for further use.
The maximum numbers in the feature map represent where the closest similarity of a feature is found. The point of pooling is to still be able to preserve the features and moreover, account for their possible special or textural or other kinds of distorting. Reducing feature helps us to process better.

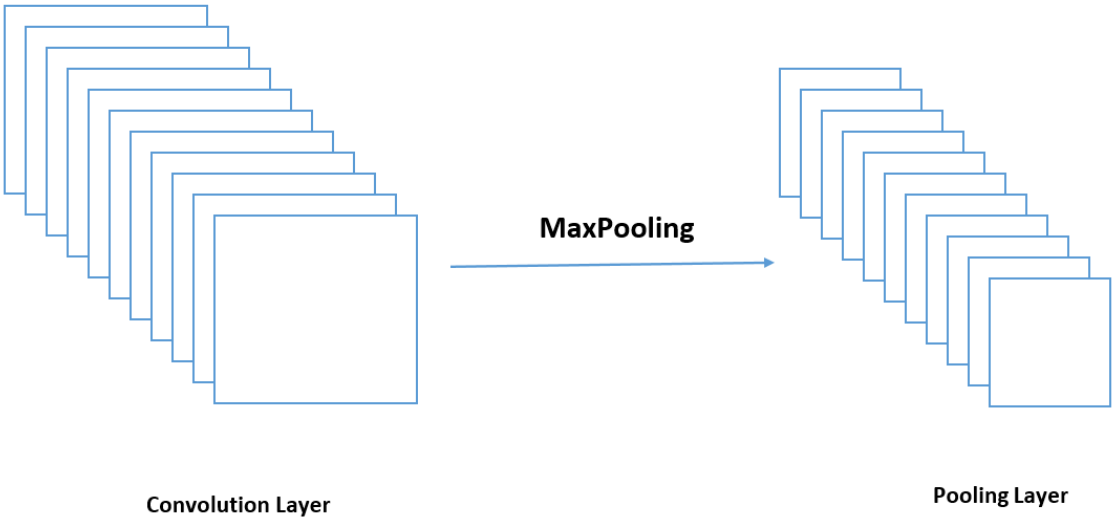

Figure 3.6: Max Pooling

Figure 3.6 is the visual representation of pooling layer. So the pooling layer is,

- **Input volume:** $D_1 \times W_1 \times H_1$, Where D_1 is the depth, W_1 and H_1 is respectively width and height

- **Two Parameter:**
  - spatial extent F
  - stride S
- **Output volume:** $D_2 \times W_2 \times H_2$ where,
  - $D_2 = D_1$
  - $W_2 = (W_1 - F) / S + 1$
  - $H_2 = (H_1 - F) / S + 1$

3.1.2.4 Flattening

Flattening is the process that converts two-dimensional arrays into one-dimensional array or single linear vector of a long chain. This step is needed in CNN as it can provide input for the fully connected layer. In Convolutional Neural Network, convolution gives a series of filters which is then max-pooled. This gives a grid-shaped two-dimensional array. A fully connected layer doesn't work with 2-dimensional input. Flattening basically breaks the spatial structure of the data from a tridimensional tensor into a monodimensional tensor which makes the structure understandable and unstructured as a fully connected layer takes a vector as input. So flattening step is important in Convolutional Neural Network as it works with complex data and that data is needed to be normalized.
The pooled feature image is being flattened into a column of a one-dimensional vector in our model to get special information around the images' pixel. Our MRI pooled image is flattened for the fully connected layer to get the important feature for detection.

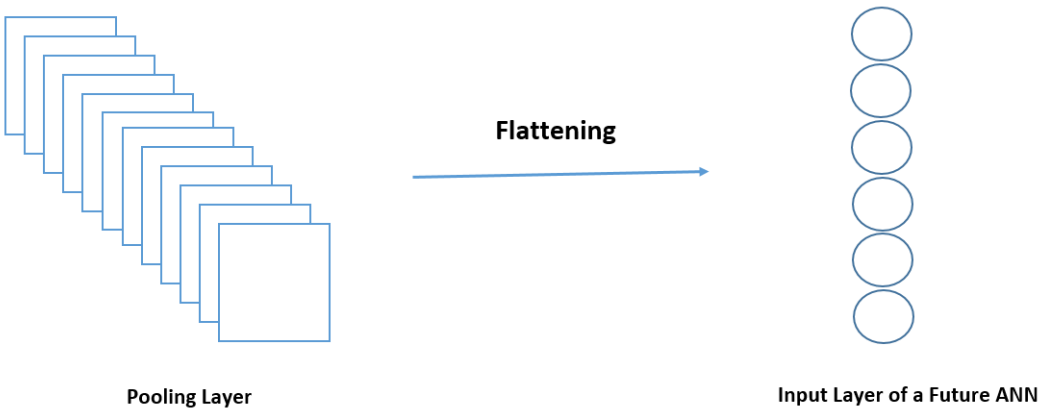

Figure 3.7: Flattening

3.1.2.5 Full Connection

The main purpose of Neural Network is to predict more accurately by combining the features in such a way that can give a more rational prediction. Fully connected layer provides the feature combination that gives the result.
A whole Artificial Neural Network is added to Convolutional Neural Network. Full Connection layer is the hidden layer of CNN. The difference is that in ANN the hidden layers are not connected with the input layer. But in CNN, Full Connection layer is fully connected to the input layer or all the previous layers.
Fully connected layer is used to get the feature combined. This combination of features makes the prediction more accurate. Full connection works with the entire input. Whenever a data is needed to be classified, it's features are stored in the neuron of the full connection and it is connected with the output classes. When a feature is matched that neuron is activated and the class gets more prediction value.

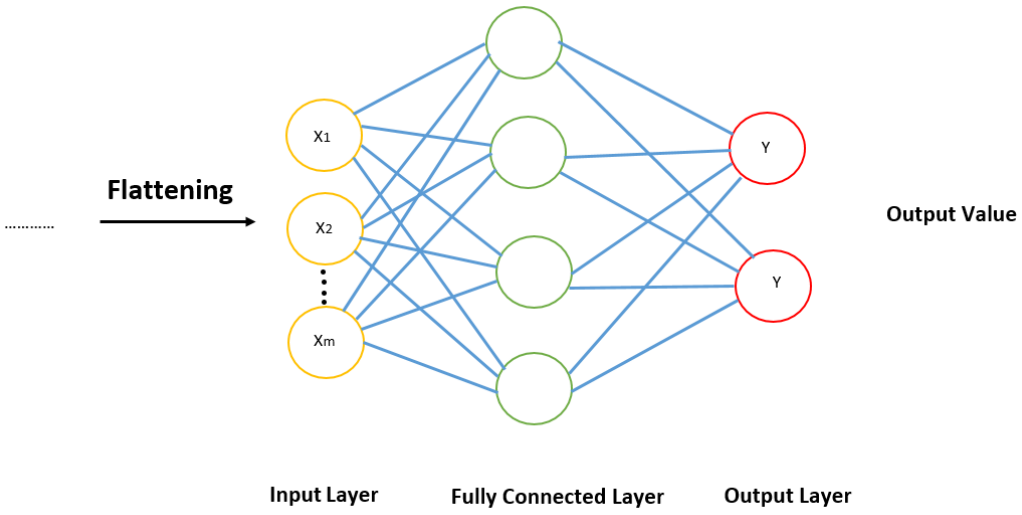

Figure 3.8: Full Connection

Full connection layer uses the output of the flattening layer as input and works with the vector value. The weight of every feature is considered. The weight value of the feature linked with the classes that we need to predict is observed to find out more relevant features of the class. Practically what Full Connection does is,

- A certain feature of the MRI image is detected by the neuron of the fully connected layer.
- The value of this feature is stored.
- This value is searched in both classes that we need to identify.

- Both classes give the output weight that decides whether this image has a tumor or not and which kind of tumor does the image has.

As input node of full connection or hidden layer, we need to take a value between the input layer and the output layer. So we have used 128 nodes in the layer as input node and also used ReLU as the activation function. But in the output layer, we used only 2 nodes and sigmoid function because the output is in the binary value.

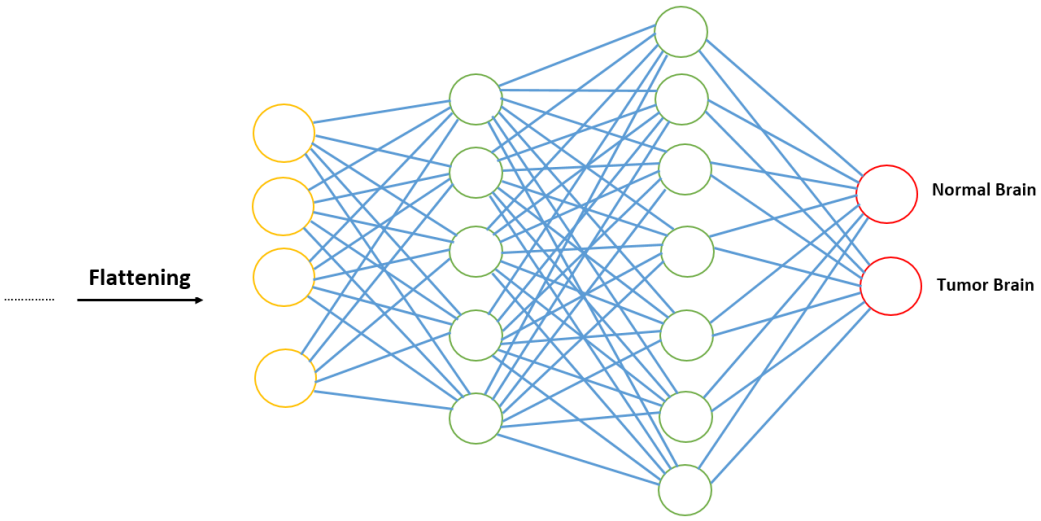

Figure 3.9: Full Connection for detecting HGG and LGG

Figure 3.9 is the visual representation of Full connection to detect whether the brain has tumor or it is just a normal brain.

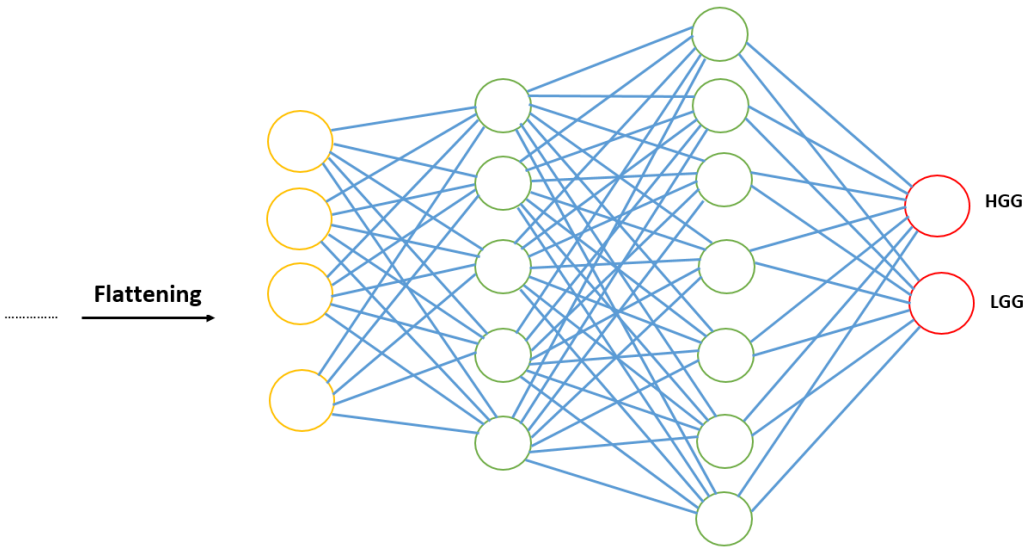

Figure 3.10: Full Connection for detecting Normal brain and Tumor Brain

Figure 3.10 is the visual representation of Full connection to classify the state of the tumor as High Grade Gliomas or Low Grade Gliomas.

3.1.2.6 Softmax Function

The result of the probability of being the particular class is coherence by Softmax Function. Without this function the result we get from the network will not be added from both classes and the final value 1 will not be produced. The equation of the function is,

$$f_j(z) = \frac{e^{z_j}}{\sum_k e^{z_k}} \tag{3.4}$$

3.1.2.7 Cross-Entropy

Cross-Entropy is a measurement of the uncertainty of a result. This is commonly known as loss function. It is what we want to minimize to maximize the performance of the network. The less the loss function value is, the better the performance of the neural network is. In CNN, the cost function is called loss function. To find the loss function, cross-entropy is used.

The whole purpose of the loss function is to detect the goodness of a CNN model. Binary cross-entropy is used when there are only 2 classes. For cross-entropy, the equation 3.5 is used,

$$H(p,q) = -\sum_x p(x) \log q(x) \tag{3.5}$$

Where q is the value of prediction (incorrect) and p is the probability of x. Cross-entropy is always greater than entropy which is determined when there is optimal encoding. The difference between this two-term gives a measurement of the distance between the predicted value and probable value. When the cross-entropy is minimized, the predicted value comes closer to the true value.

In our model, we have used binary cross-entropy as our model has binary classifier. The equation we have used to measure the loss function in binary entropy is,

$$H_p(q) = -\frac{1}{N}\sum_{i=1}^{N} y_i . \log(p(y_i)) + (1 - y_i) . \log(1 - p(y_i)) \tag{3.6}$$

Where y is the level of 2 classes which is 0 or 1 and p is the predicted probability of the feature being tumor effected at a certain point. The formula presents that, whenever y is 1, it adds log(p(y)) with the loss function and log probability of the predicted class being tumor effected. In inverse, when log(1 - p(y)) is added, it refers to the other class with log probability.

3.1.3 Model Architecture of Convolutional Neural Network

In our custom convolutional model architecture, we have used 11 layers consisting 3 kinds of layers. They are convolutional layer, pooling layer & finally fully connected layer. In all convolutional layers, we have used 3×3 kernel for each filter with the activation function rectifier linear unit (ReLU) & for the pooling layer we have used 2×2 pool size with 2×2 stride. For the output, we have used sigmoid function as the activation function. The architecture is thoroughly described below.

- **Layer 1:** It's a convolutional layer, described in the previous section, in which we have used an input of $3 \times 64 \times 64$ where 64×64 are the size of the input image & 3 is the number of color channels or depth. The number of filters used here is 32 as we want to bring variation in the output but don't want the time to be consumed. There is no padding in the layer. Stride S = 1 and F = 3.
  So from input volume, we get the output size by,
  - output depth, D_2 = Number of filters = 32
  - output weight, W_2 = (64 - 3 + 2 . 0) / 1 + 1 = 62 and
  - output Height, H_2 = (64 - 3 + 2 . 0) / 1 + 1 = 62
  - weight per filter, 3 . 3 . 3 = 27 and total weight is 27 . 32

  So the output volume is $32 \times 62 \times 62$.

- **Layer 2:** Layer 2 is also a convolutional layer with an input of the previous layers output which is $32 \times 62 \times 62$. It also has the same number of filters like the previous one and works exactly like layer 1 with same value of K, F, S and P.
  So the output size or spatial arrangement for this layer is $32 \times 60 \times 60$ by calculating with,
  - output depth, D_2 = 32
  - output weight, W_2 = (62 - 3 + 2 . 0) / 1 + 1 = 60 and
  - output Height, H_2 = (62 - 3 + 2 . 0) / 1 + 1 = 60
  - weight per filter, 3 . 3 . 32 = 288 and total weight is 288 . 32

  Besides, there are 288 weight in the convolutional layer connected with input volume.

- **Layer 3:** Also convolutional layer like last 2 layers with same hyperparameter, but input size of $32 \times 60 \times 60$.
  The output volume is obtain by using spatial arrangement's formula which is $32 \times 58 \times 58$ and 288 weight are connected with input volume as like layer 2.

- **Layer 4:** We have used a pooling layer to re-size the input & extract the important pixel values. Max pooling technique is applied here. Input volume is $32 \times 58 \times 58$. For pooling we need the value for spatial extend, F and stride, S. We have used $F = 2$ and $S = 2$ which is very commonly used values.
  So output size is calculated as,
  - depth, $D_2 = 32$
  - Width, $W_2 = (58 - 2) / 2 + 1 = 29$ and
  - Height, $H_2 = (58 - 2) / 2 + 1 = 29$

  Which means, $32 \times 29 \times 29$ is the output volume. This volume is then used for next layer which is convolutional layer.

- **Layer 5:** It is a convolutional layer but this time the number of filters we have chosen is 64 as the more the number of filter is, the more good the value will be. Input is the output of max pooling layer which is $32 \times 29 \times 29$. Hyperparameter of this layer is like previous layers only with $K = 64$. So the output image is $64 \times 27 \times 27$ calculating from,
  - output depth, $D_2 = 64$
  - output weight, $W_2 = (29 - 3 + 2 . 0) / 1 + 1 = 27$ and
  - output Height, $H_2 = (29 - 3 + 2 . 0) / 1 + 1 = 27$
  - weight per filter, $3 . 3 . 32 = 288$ and total weight is $288 . 64$

  As the number of filter is 64, then the total weight connection will be different from previous convolutional layers and weight per connection with input is 288.

- **Layer 6:** Another convolutional layer with input size of $64 \times 27 \times 27$. Just like layer 5 with same parameter. So output arrangement is,
  - output depth, $D_2 = 64$
  - output weight, $W_2 = (27 - 3 + 2 . 0) / 1 + 1 = 25$ and
  - output Height, $H_2 = (27 - 3 + 2 . 0) / 1 + 1 = 25$
  - weight per filter, $3 . 3 . 64 = 576$ and total weight is $576 . 64$

  576 is the weight that is connected and new input or output for the next layer is $64 \times 25 \times 25$

- **Layer 7:** Like previous convolutional layer, this layer also takes input of the last layer and gives output of $64 \times 23 \times 23$ and the weight connection is also same as last layer. This is the last convolutional layer in our model.

- **Layer 8:** Another pooling layer with 2×2 pool size with max pooling technique. It is like layer 4 but here output volume is changed as it takes input of size $64 \times 23 \times 23$. $F = 3$ and $S = 2$ as for the next layer which is fully connected, overlapping pooling is needed and this value refers to overlapping pooling. So the output size we get is,
  - depth, $D_2 = 64$
  - Width, $W_2 = (23 - 3) / 2 + 1 = 11$ and
  - Height, $H_2 = (23 - 3) / 2 + 1 = 11$

  Calculated output size is $64 \times 11 \times 11$. This volume is used by fully connected layer as weight value.

- **Layer 9:** After getting the output from previous layer, we have converted it into input of this layer by using flatten function to convert the matrix into one input layer for full connected layer. For this reason, The full connection layer used input size of (64 . 11 . 11) = 7744. In this layer, full connected layer used 128 output dimension. As we set the filter size exactly the size of input volume so we get the output (128 . 1 . 1) = 128.

- **Layer 10:** Same as layer 9 with the input of 128 and filter of 128. It gives output of 128.

- **Layer 11:** This is our output layer with an activation function as sigmoid function turn it into 1 output neuron as a output of classification.

Table 3.1: Architecture of the CNN

Layers	Type	Filter Size	Stride	filters	FC units	Input
Layer 1	Convolutional Layer	3x3	1x1	32	-	3x64x64
Layer 2	Convolutional Layer	3x3	1x1	32	-	32x62x62
Layer 3	Convolutional Layer	3x3	1x1	32	-	32x60x60
Layer 4	Max-pooling	2x2	2x2	-	-	32x58x58
Layer 5	Convolutional Layer	3x3	1x1	64	-	32x29x29
Layer 6	Convolutional Layer	3x3	1x1	64	-	64x27x27
Layer 7	Convolutional Layer	3x3	1x1	64	-	64x25x25
Layer 8	Max-pooling	2x2	2x2	-	-	64x23x23
Layer 9	Full Connection	-	-	-	128	7744
Layer 10	Full Connection	-	-	-	128	128
Layer 11	Full Connection	-	-	-	1	128

3.2 Method of Tumor Segmentation

Segmentation of an image means segmenting the parts of the image with similar values. Tumor segmentation is also one of the part of our research as this can separate the tumor from the background of brain and show the location of that tumor.
For tumor segmentation we have created an architecture of U-net which is a part of semantic segmentation. Before feeding the data into our system we have done pre-processing. Processed data is used in U-net architecture and by using the fully convolutional network, we can get our desire output of a segmented tumor image.

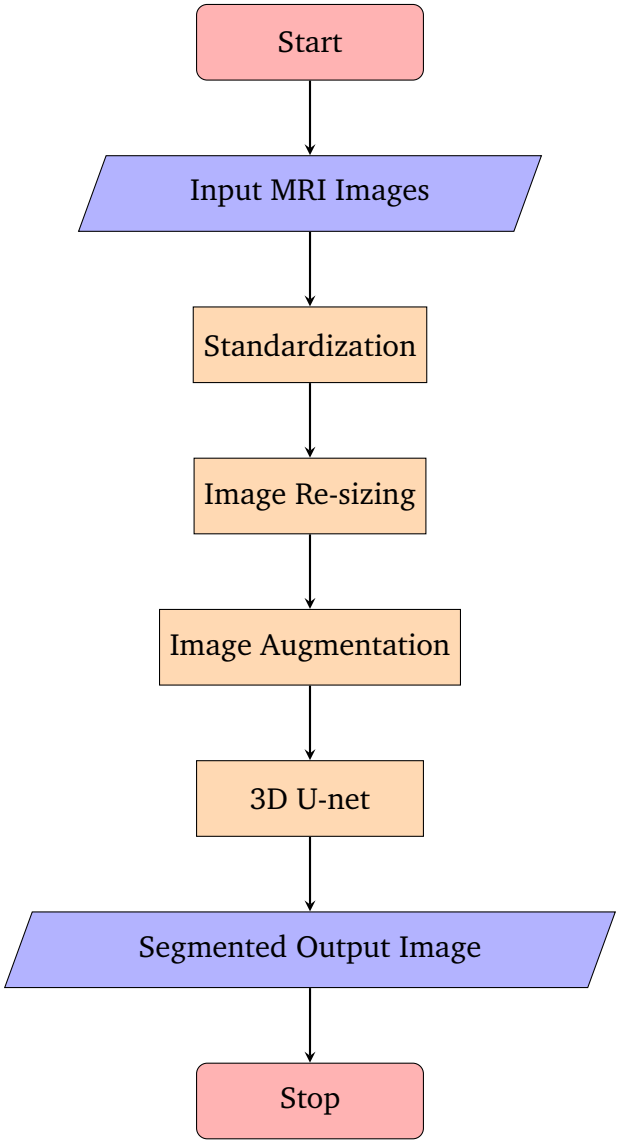

Figure 3.11: Flowchart of Tumor Segmentation Method

3.2.1 Pre-processing

Biomedical images usually contain various important information like imaged objects. To segment the object with detailed information, pre-processing is an essential section that directly influences the model architecture. Our model architecture for brain tumor segmentation was evaluated on BRATS 2015 dataset. But before segmentation, pre-processing was executed on all the MRI images except for the ground truth images. The data were pre-processed in a similar fashion which was done in the detection part. We used standardization on the data which means each pixel values were subtracted by mean pixel values, then divided by their standard deviation. So, the images were conditioned in a manner that no lower values can be dominated by the higher values.

3.2.1.1 Re-sizing Image

Our network architecture, which is based on the U-net, needs re-sized images. First, we re-size all 4 kinds of the 3D MRI images from $240 \times 240 \times 155$ to a $32 \times 32 \times 32$ uniform size. After that, we concatenate all 4 kinds of images (T1, T2, T1c and Flair) into numerical arrays for all 274 patients (220 HGG and 54 LGG). The MRI images were re-sized because if we take the original size of the data it will be for the computer to feed the data into the network as the converted data would be 782 GB which is very huge in terms of size. That's why we re-size the data into the uniform size of $32 \times 32 \times 32$ to make our model more efficient in terms of time consumption. By executing this step, the data reduce to 2.87 GB instead of 782 GB for training and the ground truth data becomes 712 MB for the corresponding data.

3.2.1.2 Image Augmentation

The image augmentation process creates more data from the original one which improves the network performance. For segmenting our dataset we applied a set of data augmentation techniques which is given in table 3.2.

Table 3.2: Applied Data Augmentation Methods

Methods	**Range**
Flip horizontally	50% probability
Flip vertically	50% probability
Rotation	$\pm 20^{\circ}$
Shift	10% on both horizontal and vertical direction
Shear	20% on horizontal direction
Zoom	$\pm 10\%$
Brightness	$\gamma = 0.8 \sim 1.2$

Some easy and uncomplicated transformations such as rotation, zooming, flipping, shifting were implemented which can produce in displacement to images. These operations will not produce samples with extremely different shapes. Operations like shearing can deform the shape of the tumor on horizontal direction. Generating training samples by varying the brightness of the images in all aspects, can make data more effective for the model network. There are some similar operations like we did in the detection part but to improve the metrics of the U-net based model, all these augmentation methods are very essential.

3.2.2 Semantic Segmentation

We, humans, are extremely adept at watching any image and understanding the content within it. It only takes a fraction of a second to analyze it. But it is completely different for the machines. In the last couple of decades to make machines like computers smarter at this kind of understanding various techniques are attempted and finally, the scientists might achieve it by the use of deep learning methods. In computer vision, semantic segmentation is getting popular day by day to improve the perception of the machines. It is the challenging task of partitioning the image into different regions. This makes it very easier to analyze the input image. It is quite similar to grouping each pixel of the image based on specific characteristics. Every pixel in the image belongs to a particular class and the similar pixels are all assigned a single color which segments the image quite efficiently.
A lot of methods have been built to handle computer vision problems ranging from human-computer interaction to robotics, medical science, and agriculture research and so on. But semantic segmentation works like a miracle in this sort of problems in computer vision. Specifically, the purpose of semantic segmentation is to label every pixel of a picture with a corresponding class of what is being entitled with. In medical diagnosis, machines can augment analysis performed by pathologists, greatly decreasing the time required to run diagnostic tests.
Deep Learning has enabled the field of computer vision to next level rapidly in the last couple of decades. When one is using deep convolutional networks, the first few layers tend to learn low-level concepts while deeper layers develop more high-level feature mappings. To maintain efficiency, we typically need to increase the number of feature maps as we get deeper into the network. This didn't necessarily pose a problem for the task of image classification, because for that task we only care about what the image contains. Thus, we could alleviate the computational burden by periodically down-sampling our feature maps through pooling or strided convolutions (ie. compressing the spatial resolution) without concern.
One of the popular approaches for image segmentation models is to maintain an encoder/decoder structure where one have to down-sample the spatial resolution of the input, develop-

ing lower-resolution feature mappings. It is learned to be highly efficient at discriminating between classes, and the up-sample of the feature representations into a full-resolution segmentation map.

The approach of using a "fully convolutional" network trained end-to-end and pixels-to-pixels for image segmentation was introduced by Long et al. [9] in 2014. The paper's writers propose adapting existing, well-structured image classification networks to serve as the encoder module of the network. By appending a decoder module with transpose convolutional layers to up-sample the coarse feature maps into a full-resolution segmentation map, this network model was built.

By using this encoder-decoder model architecture, the U-net method was built which is consisted of a series of convolution operation for each block in the model. Like common convolutional network architectures, there exist several more advanced blocks that can be substituted in for stacked convolutional layers. U-net architecture basically only takes important information from the images and make into smaller part in the encoding module then padding with similar information and make large the smallest convoluted information in decoding module. Lastly, the output image is the same size as the input image but with significant changes which are semantic segmented images. That's why we are using U-net architecture as the main theme of our proposed method and a demonstration is given in Figure 3.12 that how an image can be turned into a semantic segmented image (with U-net architecture).

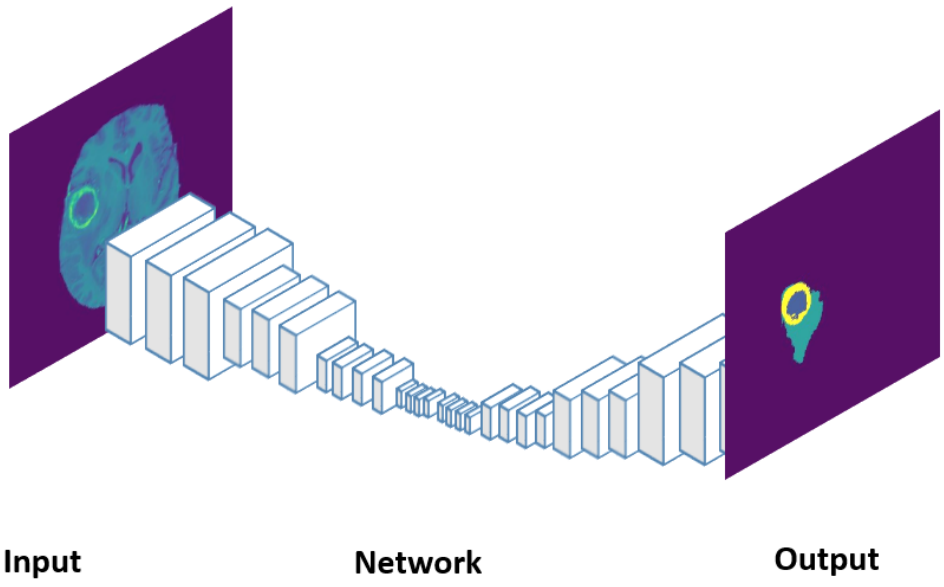

Figure 3.12: Semantic Segmentation

But most approaches of semantic segmentation, are only able to process 2D images while most medical data used in clinical practice consists of 3D volumes. That's why [10] proposed a V-net model for 3D medical imaging to segment the images into more meaningful data.

3.2.3 3D U-net Based Deep Convolutional Networks

There are various levels of granularity in which the machines can gain an understanding of images. For each of these levels, there is a problem stated in the computer vision. When it comes to segment images into different regions, semantic segmentation works accordingly which is a part of the computer vision domain. The input images were segmented in various ways in semantic segmentation like- line segments, curve segments, circles, regions, structures, etc. Most of the time we need to process the images with filters which provide color information, gradient information, etc. It helps computers in recognizing, understanding what's in the image in pixel level as well as most of the traditional methods cannot tell different objects to the computers but it solves the problems quite nicely. Long et al. [9] introduced an architecture that combined the deep level encoding and decoding. This network demonstrated natural images and gave satisfactory results. By using this model, [11] showed us it can also be applied in biomedical images. After that, the U-net was developed by Olaf Ronneberger et al. [3], based on this same decoding and encoding architecture for biomedical image segmentation. The model has two paths. The first path is the construction path which is also named as the encoder part that is used to capture the context in the image. The encoder is just a traditional stack of convolution layers as well as max-pooling layers. The second path is the symmetric expanding path which is also named as the decoder. It is implemented to enable precise localization using transposing convolution layers. Thus it is an end-to-end FCN (fully convolutional network) which only have convolutional layers and does not have any dense layer as it can accept an image of any size.
For those who are familiar with traditional convolutional neural network layers will be familiar with the first part of U-net architecture denoted as contraction path of the architecture. In contraction path (encoder), where we apply regular convolution layers and max-pooling layers previously stated. In the Encoder, the size of the image gradually reduces while it keeps gradually increasing the depth. This means the network learns the "WHAT" information from each image, though it has lost the "WHERE" information from the images. In expansion path (Decoder), where we apply transposed convolution layers along with regular convolution layers. The size of the image gradually increases and the depth gradually decreases in the decoder part. Eventually, the decoder part recovers the "WHERE" information (precise localization) by continuously applying up-sampling. To acquire better precise locations, at each step of the expansion path we use skip connections by concatenating the output of the transposed convolution layers with the feature detectors from the encoder at the same level. After every concatenation, we once more apply two consecutive regular convolutions so that the architecture can learn to congregate a more precise output. The architecture is a symmetric U-shape and that is why it is named U-net architecture. Before we dive into our model 3D U-net, it is important to understand the different operations that are typically used in a fully convolutional network.

3.2.3.1 Convolution Operation

It is the same as the convolution operation done in the detection part. The only difference is that this convolution is done on 3D images. There are three inputs to this convolutional operation.

1. **Input Size:** A 4D volume (input image) with the size of $D_1 \times H_1 \times W_1 \times C_1$, where D_1 is the depth, H_1 is the height, W_1 is the width and C_1 is the color channel of the input image.
2. **Filters:** The dimensionality of the output space that means if there is 'k' kernels or filters the number of output will increase to 'k' from unit input size.
3. **Kernel Size:** $D_2 \times H_2 \times W_2$, specifying the depth, height, and width of the 3D convolution window.

Output Size: A 4D volume which is also called an output image or feature map with the size of $D_3 \times H_3 \times W_3 \times C_3$.
One important term used frequently is named as the Receptive field. This is nothing but the region in the input volume that a particular filter is looking at which is also sometimes called as the context. By putting in very simple terms, the receptive field (context) is the area of the input image that the filter covers at any given point of time.

3.2.3.2 Max Pooling Operation

In simple words, the function of pooling is to reduce the size of the feature detector so that we have fewer parameters in the network architecture. Same as the detection section, here we are using max pooling operation for our network. Basically from every $2 \times 2 \times 2$ block of the input feature detector, we select the maximum pixel value and thus obtain a pooled feature map. A very important point to note here is that both convolution operation and especially the pooling operation reduce the size of the image which is called as down-sampling. All the parameters for this operation is given below.

1. **Input Size:** A 5D tensor with the shape of $B_1 \times D_1 \times H_1 \times W_1 \times C_1$, where B_1 is the batch size, D_1 is the depth, H_1 is the height, W_1 is the width and C_1 is the color channel of the input tensor.
2. **Pool Size:** $D_2 \times H_2 \times W_2$, the mask size for pooling.
3. **Strides:** $D_3 \times H_3 \times W_3$, a tuple of three integers which defines how much the mask has to leap for each step.

Output Size: A 5D volume which is also called an output image of $B_4 \times D_4 \times H_4 \times W_4 \times C_4$.

3.2.3.3 Up-sampling Operation

As stated previously, the output of semantic segmentation is not just a class label or some bounding box parameters, the outputs are complete high-resolution images in which all the pixels are classified. Thus if we use a regular convolutional network with pooling layers and dense layers, we will lose the "WHERE" information and only get the "WHAT" information. This is not what we want. In the case of segmentation, we need both "WHAT" as well as "WHERE" information which will make segmentation more meaningful.
Hence, there is a need to up-sample the image, i.e. convert a low-resolution image to a high-resolution image to regain the "WHERE" information.

1. **Input Size:** A 5D tensor with the shape of $B_1 \times D_1 \times H_1 \times W_1 \times C_1$, where B_1 is the batch size, D_1 is the depth, H_1 is the height, W_1 is the width and C_1 is the color channel of the input tensor.

2. **Size:** $D_2 \times H_2 \times W_2$, The up-sampling factors for depth, height and width which means how much it going to up sample the data.

Output Size: A 5D volume which is also called an output image of $B_3 \times D_3 \times H_3 \times W_3 \times C_3$.

3.2.3.4 Transposed Convolution Operation

Transposed convolution, which is sometimes also called as deconvolution, is a technique to perform up-sampling of an image with learnable parameters. Transposed convolution is exactly the opposite process of a normal convolution operation which practically means the input volume which is a lower resolution image converted into a higher resolution image. We use transposed convolution operations by the same 3D convolution operation just by increasing the filters and all the parameters remain the same as convolution operation stated in the earlier section.

3.2.3.5 Activation Function

As we know that activation function helps CNN based model to learn new data by activating the neuron related to the information given with high weight and it ignores the rest, we have used ReLU (Rectified Linear Unit) Activation Function in this model. In the detection part of our research, we also have used ReLU which is a popular function among all activation function. It provides max value for all positive values and zeroes for all negative values in computation and it is very simple to avoid complex values. We used the ReLU in all normal convolution operation as well as in all transposed convolution operation except for the last

one. The last transposed operation was done with the help of the sigmoid function to get the main output semantic segmented image.

3.2.4 Model Architecture of Segmentation

Our network is inspired by the U-Net architecture of Olaf Ronneberger et al. [3] who also made the architecture for biomedical images. The dataset of BRATS 2015 were consists of 274 patients and each of the 4 images' (T1, T1c, T2 and Flair) original size were $240 \times 240 \times 155$. For the computation efficiency, we resized the data into $32 \times 32 \times 32$ and designed the model to process those large 3D input images.
Like [9], this network architecture uses the skip-architecture that combined the down-sampling (encoding) path and an up-sampling (decoding) path as shown in Figure 3.13.

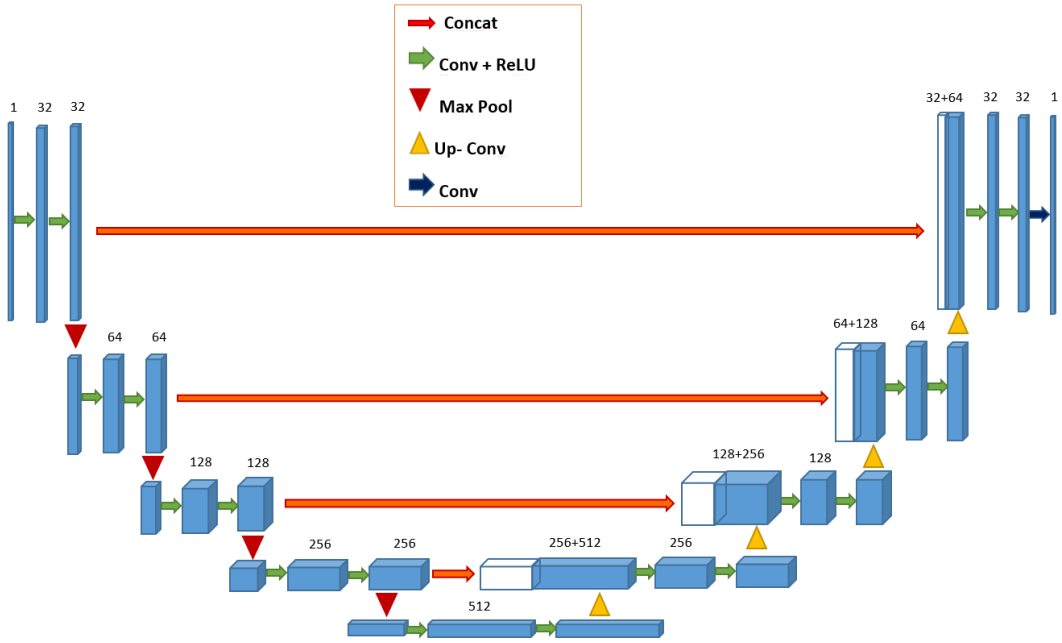

Figure 3.13: U-Net Architecture of our Model

This model has the depth of 5 which means it has 5 convolutional blocks for down-sampling path and each block has two convolutional layers with a filter size of 3×3, a stride of 1 in all three directions and ReLU function as the activation function. It increases the number of feature detectors from 1 to 512. In the case of down-sampling, max pooling with stride 1 in all three dimensions with a pool size of $2 \times 2 \times 2$ is used. It is applied to the end of every block except the last block. The size of the feature detectors decreases from $32 \times 32 \times 32$ to $2 \times 2 \times 2$. In the up-sampling path, each level block starts with an up-sampling layer with filter size of $2 \times 2 \times 2$ which doubles the size of the feature detectors in three dimensions but decreases the number of feature maps by two, so the size of the feature detectors increases

from $2 \times 2 \times 2$ to $32 \times 32 \times 32$. In each up-sampling block, two convolutional layers reduce the number of the feature detectors of the concatenation of up-sampling (deconvolution) feature detectors and the feature maps from the encoding path. Different from the original U-net architecture built by Ronneberger et al. [3], we use zero paddings to keep the output dimension for all the convolutional layers of both down-sampling and decoding. At last, a $1 \times 1 \times 1$ convolutional layer is built to reduce the number of the feature detectors to two that reflect the foreground and background segmentation respectively. No fully connected layer is used in the network architecture.
The parameters of this network architecture remain the number of convolutional and the number of deconvolutional blocks which can vary a little bit by changing it. The number of blocks remains in the range of 4-6 in tuning the network according to the 3D U-net. We didn't use any regularization like- L1, L2, Dropout as it doesn't make any changes in achieving desired performance metrics. The parameters of this model architecture are given in Table 3.3.

Table 3.3: Parameters of U-Net Architecture

Parameters	**Value**
Number of convolutional blocks	[4, 5, 6]
Number of deconvolutional blocks	[4, 5, 6]

3.2.4.1 Training and Optimization

During the training process, the Soft Dice metric described in [10] was used to evaluate our network model's performance as a cost function rather than the cross-entropy based or the quadratic cost function. Soft Dice can be considered as a differentiable form of the original Dice Similarity Coefficient (DSC) as we get it from subtracting the obtained DSC from 1 [10]. Training deep neural networks requires stochastic gradient-based optimization which minimizes the cost function concerning its parameters and that is why we implemented the adaptive moment estimator (Adam) [28] to estimate the parameters. In general, Adam employs the first and second moments of gradients for updating and correcting the moving average of the current gradients. The parameters of our Adam optimizer were set as learning rate $= 0.0001$ and the maximum number of epochs $= 300$. All weights were assigned by normal distribution with a mean of 0 and a standard deviation of 0.01, and all biases were assigned as 0.

3.2.4.2 Performance Evaluation

The evaluation of our network has been done using a four-fold cross-validation method for the HGG and LGG combined data, respectively. The performance evaluation was done on

the complete tumor region of the brains. The segmentations have been evaluated using the dice soft coefficient, and the accuracy was also calculated. The DSC provides the overlap measurement between the brain tumor and the segmentation results of our fully automatic method that is,

$$DSC = \frac{2TP}{FP + 2TP + FN} \tag{3.7}$$

In this equation, the true positive, false positive and false negative are denoted as TP, FP, FN measurements, respectively. Another performance metrics, accuracy is used to evaluate the whole data and that is,

$$Accuracy = \frac{TP + TN}{TP + TN + FP + FN} \tag{3.8}$$

where, TP = True positive; FP = False positive; TN = True negative; FN = False negative

3.3 Used Dataset

We have used BRATS 2015 dataset in our project where we have 220 HGG or High Grade Glioma patients' data and 54 LGG or Low Grade Glioma patients' data for training. For testing we have 110 patients' data. Also we have data for normal brain. All of the MRI images are of 4 types: Flair, T1, T1c and T2.

1. **Flair:** Using a long inversion time and a long effective echo time, the signal is made null from bright cerebrospinal fluid (CSF) of brain in flair image. High CSF is suppressed by long inversion time and the small periventricular lesions are visualized easily.

2. **T1:** Images normally is dark in T1 image. T1 image is identified easily by looking for fluid filled space in body. When an MRI sequence is set to produce T1- weighted image, the short T1 valued tissue produce highest magnetization and appear the brightest in image.

3. **T1c:** In T1-weighted MRI, many tumors show signal enhancement after administration of contrast media. This type of T1 images are T1c -weighted image.

4. **T2:** Image contrast is based predominantly on the T2 (transverse) relaxation time of tissue. Tissue with long T2 relaxation time appears brighter (hyperintense). A T2-weighted image produces T2 contrast by de-emphasizing the T1-weighted image. Fluids are bright here.

Figure 3.14 is the visual representation of the 3D brain data that we worked with.
The type of dataset we used are given below.

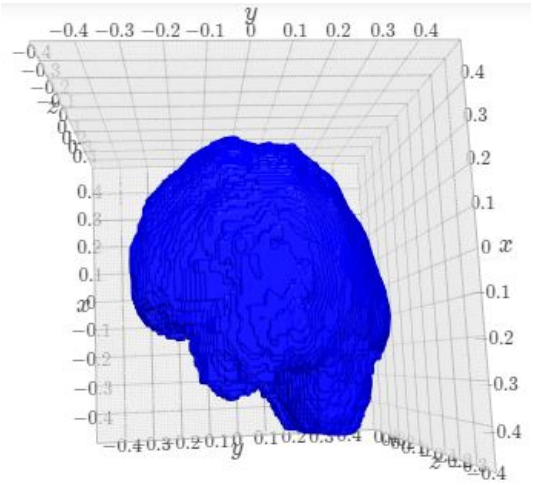

Figure 3.14: 3D image of Brain

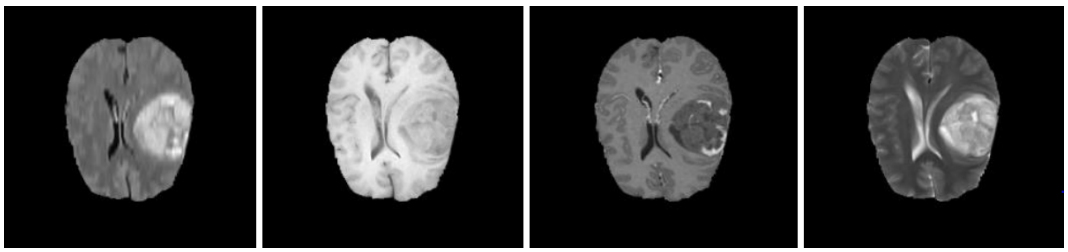

Figure 3.15: MRI images of HGG. From left to right: Flair, T1, T1c and T2

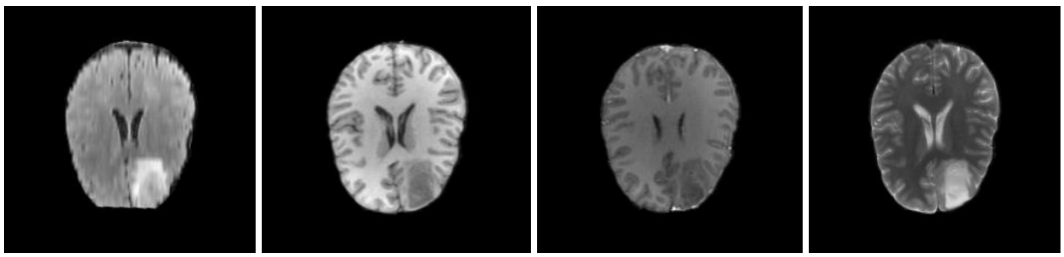

Figure 3.16: MRI images of LGG. From left to right: Flair, T1, T1c and T2

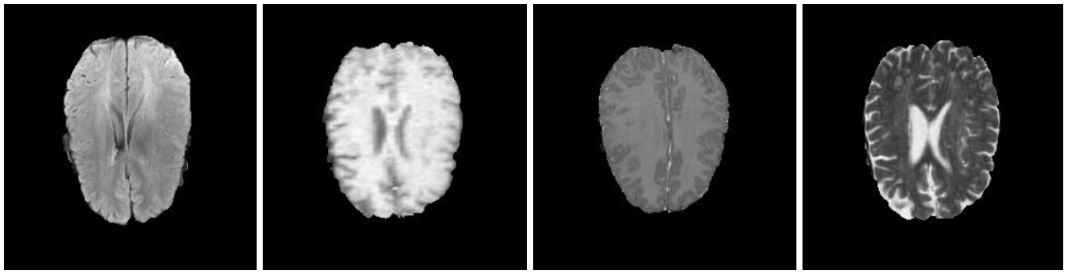

Figure 3.17: MRI images of normal brain. From left to right: Flair, T1, T1c and T2

Chapter 4

Result & Discussion

In this chapter, we analyze the key component's effect on the result & the performance of the implemented methods and the comparison of the result of other's methods. We exemplify the effect of the experiments in the classification of normal brain and brain with tumor as well as the classification of tumors in two type patients (HGG and LGG). In the experiments of tumor detection, we maintained the architecture of CNN presented in Table 3.1 to conduct the classifications correctly. Also, only data of the training dataset are augmented to train the system model properly. In the section of tumor segmentation experiment, we applied U-net architecture of semantic segmentation mentioned in Figure 3.13 which was developed by Olaf Ronneberger et al. [3] to get the feature from downsampling in the first path and enable precise localization from upsampling in the second path. This separates the tumor from the brain creating a region around the tumor.
We used validation testing to test the system accuracy after training the data. Validation test is checked to test our data with a ground truth before we test our system as it is held back from the training data to estimate model skill.

4.1 Detection

The result of detecting the tumor of our model is quite satisfying compared to our resources. BRATS 2015 dataset was used and this dataset has a good number of patient cases than other used paper's dataset.
The predicted probability of having brain tumor and the type of tumor is given by the detection model. This result gives the accuracy graph and loss function graph with a percentage. We trained our data and done validation testing to measure the performance of the tumor detection system. Also, some key component works as a prominent fact behind the result.

4.1.0.1 Pre-Processing

The effect of our pre-processing method on the classification was evaluated by comparing with Gaussian Method. We choose this method because it is also utilized in a CNN based brain tumor classification method. As the pixel's min value is 0 and max value is 255, we divided each pixel by 255 to scale them between 0 and 1 which make scaled images for the CNN.

4.1.0.2 Image Augmentation

Artificial image augmentation is a common procedure in the context of CNN when the dataset is relatively small. In the system proposed by Havaei et al. [5], they considered its application, but found to be ineffective. We implemented 4 types of image augmentation. The effect of the image augmentation by increasing the number of samples using rotations, horizontal flips, sheering & zooming. The rotation angle was 90 degree. The sheer & zoom range were 20%. Every change in each image increases the dataset of training and the accuracy is increased by that. Also extended dataset gives increased accuracy in validation testing and elevates the model's performance.

4.1.0.3 Deep Architecture

In our convolutional neural network, we used 3×3 kernels in convolutional layers to get the advantage of maintaining the same effective receptive field of bigger kernels, while reducing the number of weights. To evaluate the real impact of this technique on brain tumor classification, we changed the convolutional layers before each max pooling of the used architecture. By one layer with larger kernels with the equivalent effective, receptive field. We change the group of layers 1, 2, 3 and 5, 6, 7 (Table 3.1) by one convolutional layer. Using this architecture, we experimented to variants for classifications:

1. We maintained 32 feature maps in the first convolutional layer group and 64 in the second.

2. We increase the capacity of the CNN by using wider layers, namely, 64 feature maps in the first convolutional layer and 128 in the second.

The result that we got from the convolutional layer is enhanced by this approach. Also the main purpose of increasing the performance of the model is fulfilled.

4.1.1 Normal Brain vs. Tumor Affected Brain

We used our neural network architecture to differentiate between normal and tumor affected patient using their MRI images. This architecture then gives a measurement of performance to show the accuracy and loss function for our training and testing dataset.
After the classification we achieved 84.07% max accuracy of trainning data and 74.30% accuracy in validation testing by epoching the data 60 times. Which means our system can detect tumor effected and normal brain in training with 84.07% accurately and in validation testing 74.30% accurately. We used 768 MRI image per epoch and used batch size 32.

Table 4.1: Accuracy & Loss Function value for Normal vs. Tumor affected

Data Type	Accuracy	Loss Function
Training	84.07%	38.89%
Validation Testing	74.30%	47.79%

4.1.1.1 Accuracy of Normal vs. Tumor affected

Figure 4.1 is obtained from our experiment to indicate the accuracy graph that we got.

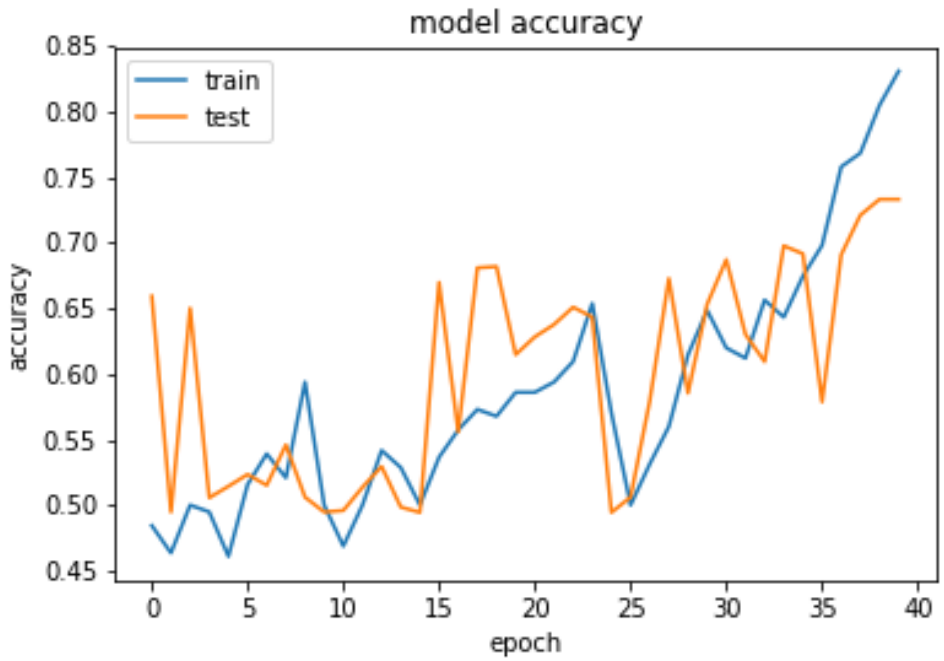

Figure 4.1: Accuracy graph for detecting Normal brain vs effected brain

In the figure, we can see that from the beginning of the experiment the training accuracy was around 0.48 & until 14 epochs it didn't improve much. But from the 15th epoch, the training, as well as the validation testing accuracy, was exponentially getting higher. From

the 30th to the 40th epoch the accuracy of the training was improving & it was between 0.70 to 0.8407. On the other hand, validation testing accuracy was also increased from the same epoch and achieved max accuracy of 0.7430 at the 38th epoch. That means, by increasing the number of epochs, the accuracy gradually will increase as in one epoch the complete dataset is learned for once and the increasing number of epoch will increase the learning rate. But as we used same data from 60 epochs, the accuracy didn't improve at all. So, to decrease the time delay of training we remained 40 epochs for this case.

4.1.1.2 Loss function of Normal vs. Tumor affected

As we used binary cross entropy for our loss function of normal VS tumor affected, we have achieved Figure 4.2. Binary cross entropy is just a special case of categorical cross entropy. The equation for binary cross entropy loss is the exact equation for categorical cross entropy loss with one output node.

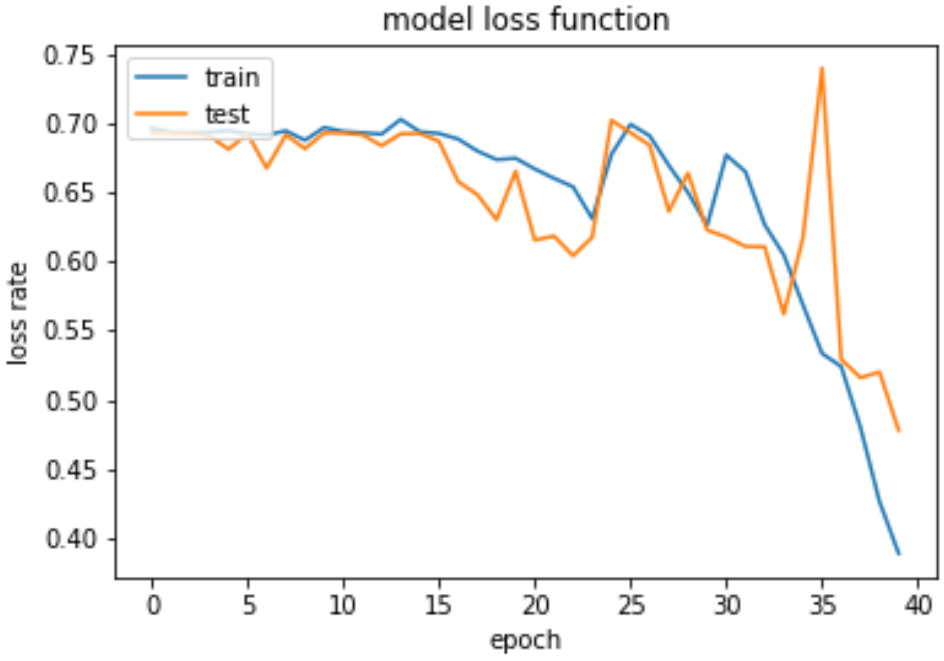

Figure 4.2: Loss function graph for detecting Normal brain vs effected brain

In the figure, we can see that the loss rate was higher at the beginning of the experiment which is 0.74 but it gradually decreased from 30th epoch to 0.37 for training data & 0.47 for the validation testing data. The training graph was exponentially leveling downwards that makes the learning more efficient as the lower the loss rate the lower the error of the training and the accuracy is more higher of the training data.

4.1.2 High Grade Gliomas vs. Low Grade Gliomas

High Grade Gliomas (HGG) and Low Grade Gliomas affected patients can also be detected by using our neural network architecture for tumor detection. From our dataset, we have decided to use a portion of data to train our model and we got the accuracy that indicates our models precision.
For the case, we have used half of the data of the previous classification. 384 & 192 MRI images are used for each type. Here we used higher epochs to achieve higher accuracy as our data was limited. It took 60 epochs to achieve 78.12% max accuracy and 47.04% loss function in training data and 69.25% accuracy and 69.32% loss function in validation testing.

Table 4.2: Accuracy & Loss Function value for HGG vs. LGG

Data Type	Accuracy	Loss Function
Training	78.12%	47.04%
Validation Testing	69.25%	69.32%

4.1.2.1 Accuracy of HGG vs. LGG

The accuracy graph produced by our model can be seen in Figure 4.3 where the measurement of the correct predicted probability is given.

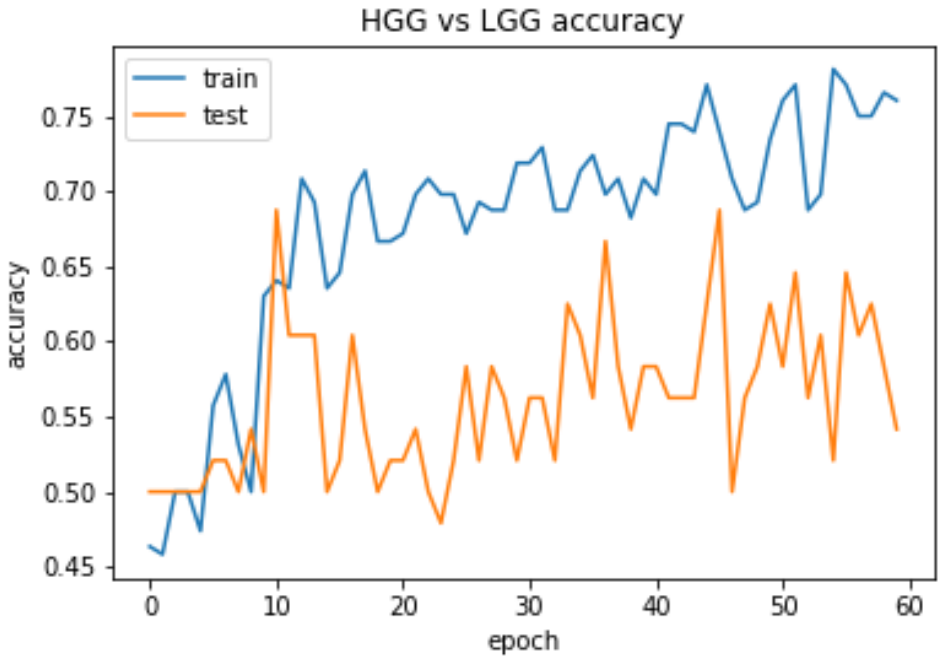

Figure 4.3: Accuracy graph for HGG vs. LGG

The starting of the experiment of the training accuracy was only 0.46 & until 9 epochs, there was no improvement which can be shown by the figure clearly. But from 10th epoch the training as well as the validation testing accuracy was getting higher. At 10th epoch we achieved the max accuracy of 0.6925 for validation testing. After 10th epoch, we couldn't get any higher value. It was always between 0.58 to 0.65 by gradually increasing & decreasing. For training data the accuracy was exponentially increasing from 10th epoch & the accuracy was 0.78 after 60th epoch.

4.1.2.2 Loss function of HGG vs. LGG

Loss or error function of our system is measured by using binary cross entropy. For HGG vs LGG loss function, we have obtained Figure 4.4.

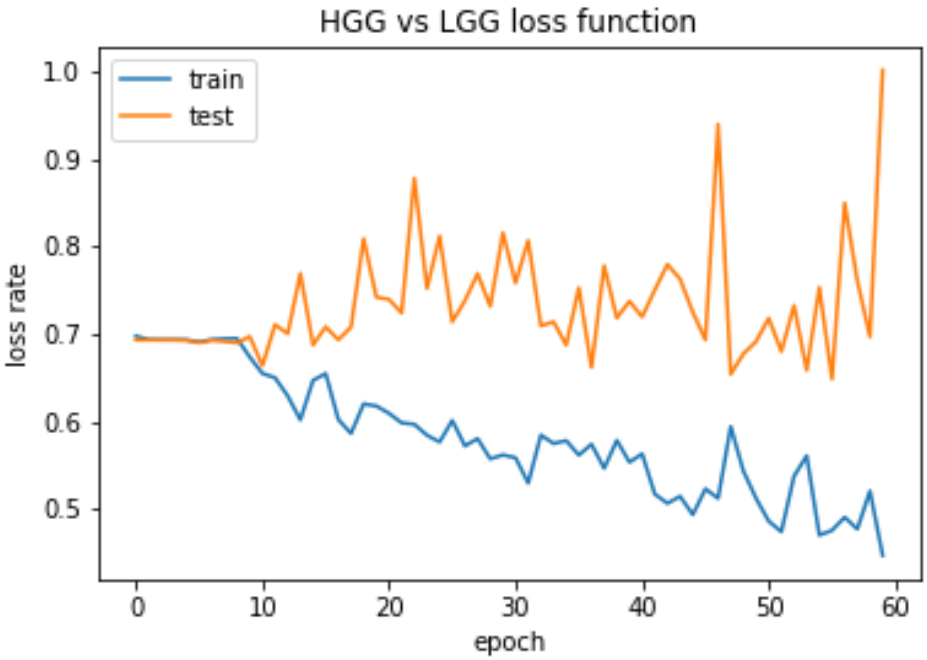

Figure 4.4: Loss Function graph for the HGG vs. LGG

Binary cross entropy is applied when there is binary classifier for prediction. For detecting HGG and LGG tumor, we have used the same equation that is used for detecting normal vs tumor affected which is the exact equation for categorical cross entropy loss with one output node. In this graph, at the start of the classification, the loss rate was 0.70 but gradually by leveling up the epochs, the validation testing loss rate was maintained among 0.75 to 0.80 which was not decreasing at all. But the training graph was exponentially leveling downwards and at the 60th epoch it is 0.4 that makes the learning more efficient as the lower the loss function the higher the accuracy.

4.1.3 Comparison of Different Methods of Tumor Detection

In the comparison of accuracy, we were in the middle with the contrast of other methods in classification of normal vs. tumor affected images. Our detection method gives the accuracy of a proper percentage. It can detect the tumor with the accuracy of 74.30%. If we compare our work with the paper of Pauline John et al. [16] then we can notice that his paper got 70% of accuracy on tumor detection which is lesser then ours. On the other hand [21] have 78% of accuracy which is only 4% greater then our architecture.
In Havaei et al. [5] & Pereira et al. [4], they have accuracy of 83% and 88% accuracy respectively. This accuracy was not impossible for us to achieve as we worked with only 4GB GPU on the same dataset where they work with 12GB. We know, more GPU capacity gives more weight to the network which increases the accuracy and also the time delay.

Table 4.3: Comparison of methods in respect of accuracy

Methods	Accuracy
Proposed Method [table 3.1]	74.30%
Mohammad Havaei et al. [5]	83%
Sergio Pereira et al. [4]	88%
Stefan Bauer et al. [21]	78%
Pauline John et al. [16]	70%

Our accuracy can gradually increase if we increase the number of training & testing data so that, neural network will learn about the pattern of the images & differentiate them more successfully.

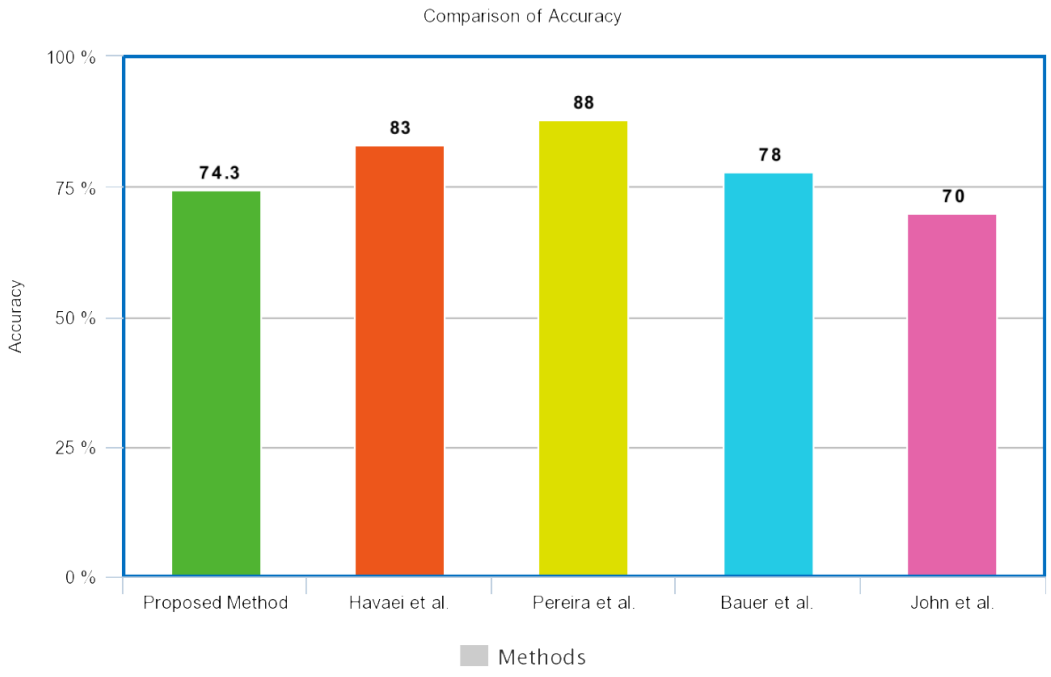

Figure 4.5: Chart of comparison between different method of tumor detection

4.2 Segmentation

Our segmentation model using U-Net architecture gives us a very good accuracy for tumor area localization. This accuracy has not been achieved by any paper with the resources that we have used. It searches the image pixel by pixel to get the tumor value compare with the ground truth. Whichever pixel matches with the ground truth that is counted. After counting 274 patients images, an accuracy is found and we get the tumor segmented area.

4.2.1 Segmentation Result for BRATS 2015

We have applied our architecture of segmentation on the dataset BRATS 2015. This dataset have 4 types of image for each patient. Flair, T1, T1c and T2.
Tumors of T1 and T1c type images are most clearly visible among them. This 2 types of image is segmented and the prediction of our model gives accurate result as compared to ground truth.
T1 images are normally dark image but can easily identified as the body of this image is filled with fluid. The short T1 produce highest magnetization which appears the brightest in images.

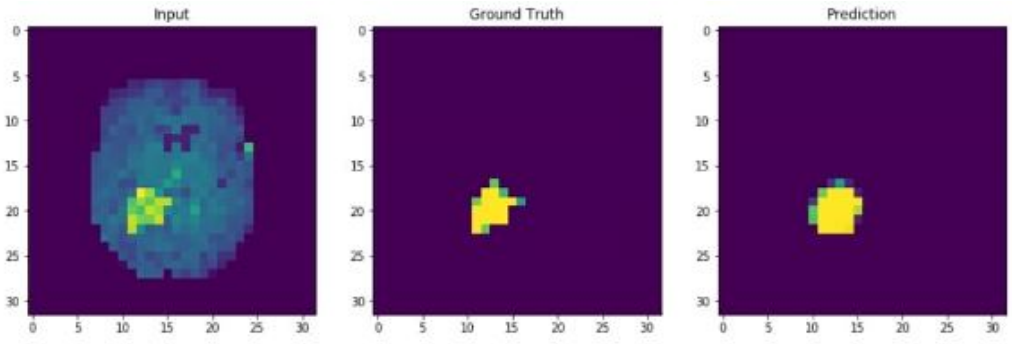

Figure 4.6: Segmentation for T1 MRI

In Figure 4.6, the input image is processed in our model and a segmented predicted image is found. Ground truth is the level image for training the model to give accurate prediction. Our prediction matches the ground truth value very closely. By training more and more this prediction can be more accurate.
In T1-weighted MRI, many tumors show signal enhancement after administration of contrast media. This type of T1 images are T1c -weighted image. It is similar with T1 image and the tumor is more visible. For this reason, this type of image is more accurate for our experiment.

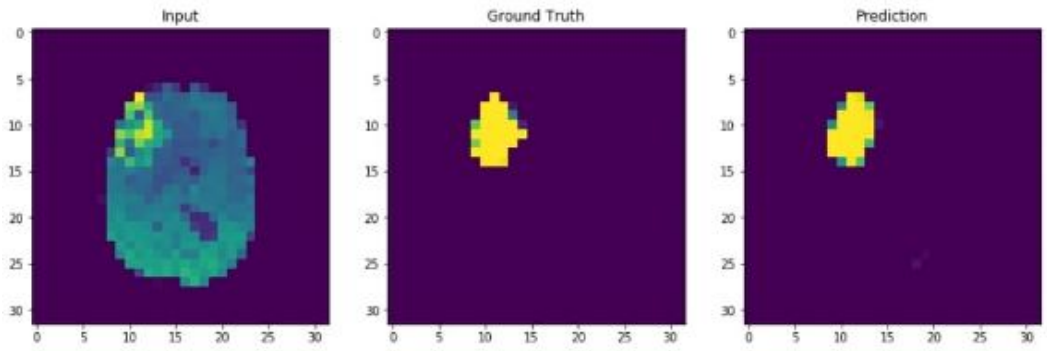

Figure 4.7: Segmentation for T1c MRI

In Figure 4.7, a predicted image is found from T1c type input image by applying our segmentation method. This procedure only highlights the tumor area and all the other pixels are colored as background. This image is quite similar with the ground truth. Which means, the predicted tumor image is accurately segmented.

4.2.2 Performance of the Model for Tumor segmentation

We used our U-Net architecture to segment the tumor image. This segmentation is done by considering the image pixel by pixel with voxel of a 3D image. When the pixel of the image has same value, after segmentation that number is counted. When all the pixel is done then the number of predicted image is compared with the ground truth. If the number matches correctly with the ground truth then the accuracy is 100%. Otherwise the loss function is increased.
The loss function is measured from Dice Soft Loss(DSL). This function counts the unmatched pixels and give the loss or error function value. The equation for loss function is,

$$DSL = 1 - DSC \tag{4.1}$$

The equation is calculated by subtracting dice soft coefficient from whole probability. Dice soft coefficient is needed for accuracy measurement.
From our model, we achieved 93.08% accuracy on training data and 92.09% accuracy on validation testing. The loss function's value is also low with 19% in training data and 21% validation testing data.

Table 4.4: Accuracy & Loss Function value for Tumor Segmentation

Data Type	Accuracy	Loss Function
Training	93.08%	19%
Validation Testing	92.09%	21%

4.2.2.1 Accuracy for Tumor Segmentation

The accuracy curve of the segmentation model is given in Figure 4.8 which shows the accuracy for training and validation testing data.

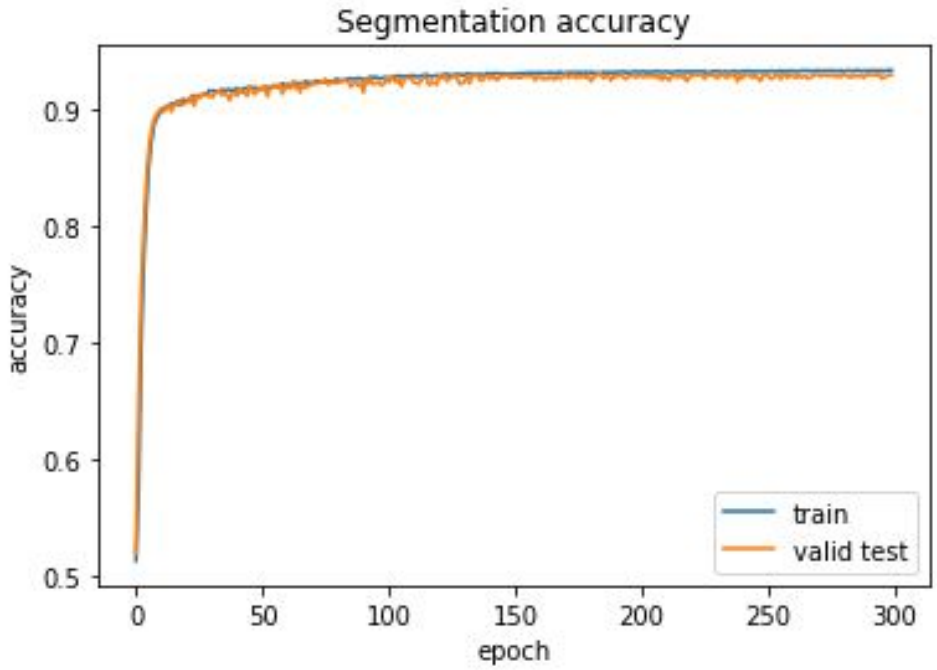

Figure 4.8: Accuracy graph for Tumor Segmentation Model

From the very beginning of the graph, the accuracy have been risen and with the increasing number of epoch, the accuracy increases very slightly. On 1st epoch, the accuracy was 0.9 which is an excellent value. Moving epoch by epoch, the accuracy increases for both training and validation testing data. After our final epoch which is 300, the accuracy for training data is 0.93 and for validation testing data is 0.92.

4.2.2.2 Loss Function for Tumor Segmentation

Figure 4.9 is the loss function of our model which is also giving marvelous result after increasing epoch.

The result, that we obtained from the segmentation loss function is measurement by dice soft loss. At the 1st epoch, the loss value was high for both data which was 0.92. After increasing number of epoch, the loss value is decreased. The gradually downwards curves shows the efficiency of our model. Lastly at the 300th epoch the loss function's value for training data is 0.19 and for validation testing is 0.21. If we increase the epoch number further, then the value for loss function will be decreased.

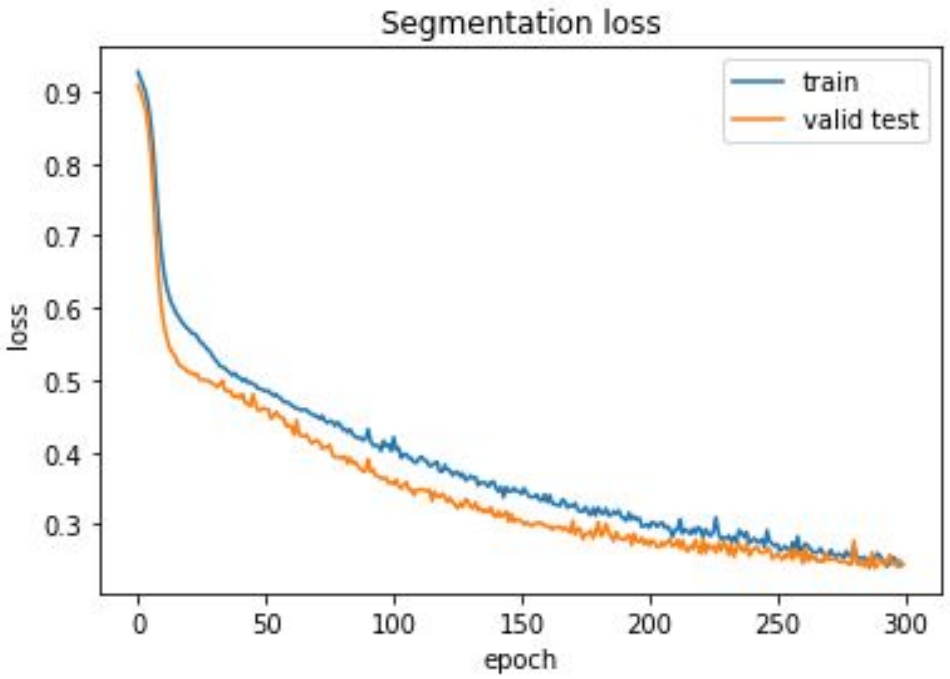

Figure 4.9: Loss Function graph for Tumor Segmentation Model

4.2.3 Comparison of Different Methods of Tumor Segmentation

The novel work for our thesis paper is done in the segmentation part. We have created a model that has a good accuracy and compared to some paper, the result is excellent. We have compared our work accuracy with four other papers who where said to be best papers. Sergio Pereira et al. [4] and Mohammad Havaei et al. [5] has achieved the DSC value of 0.78 but our DSC value is 0.79 which is a good accuracy than these papers.
Konstantinos Kamnitsas et al. [29] has DSC value of 0.85 which is good. They have used 2D dataset for their proposed method. But we have used 3D dataset in our model, counting the image of voxels for segmentation as we have used 4GB GPU. We can't have the whole sized image as the calculation delay would be high. That's why we re-sized our image dataset for the efficiency.

Table 4.5: Comparison of different methods in respect of DSC

Method	Data	DSC
Proposed Method [Figure 3.13]	BRATS 2015	0.79
Sergio Pereira et al. [4]	BRATS 2015	0.78
Mohammad Havaei et al. [5]	BRATS 2015	0.78
Konstantinos Kamnitsas et al. [29]	BRATS 2015	0.85

[12] has used TensorFlow and TensorLayer library but we have used Keras library in our model. Our image resolution was smaller than everyone as we used 3D image. For that reason, [9] took 22 hours 5 minutes for training the data others took above one day but we

took only 12 hours 30minutes.

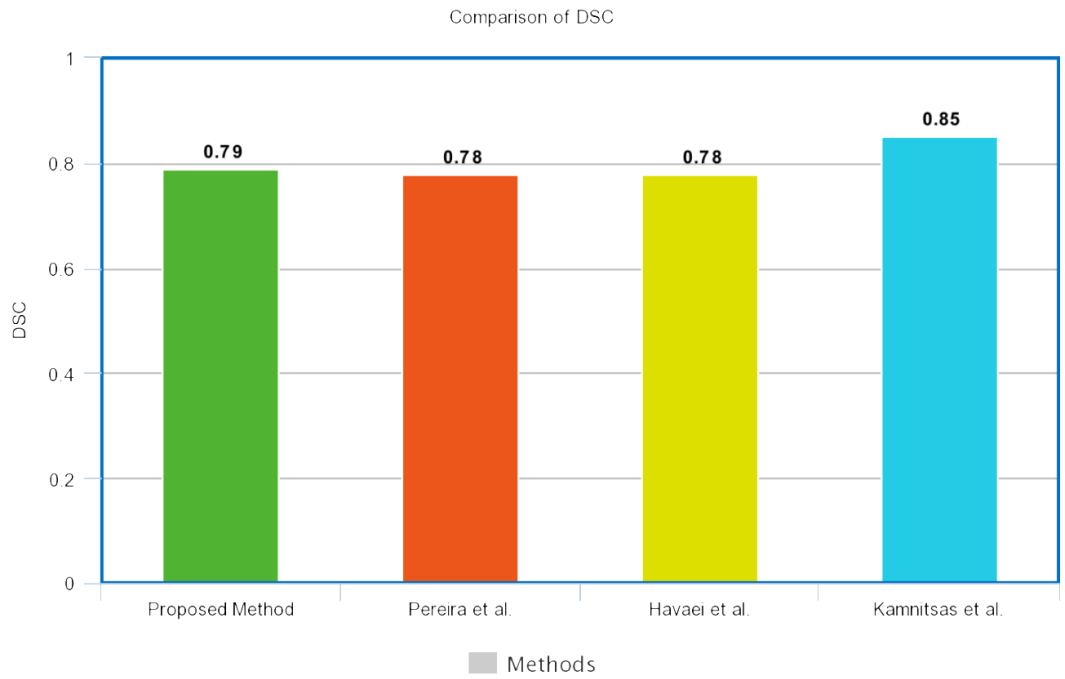

Figure 4.10: Chart of comparison between different method of tumor segmentation

Our model gives the most time optimizing result than most of others. Prediction giving delay is considered as the efficiency measurement scale.
In our proposed method, we have given the prediction in less than 1 sec. It is the lowest time consuming model. Sergio Pereira et al. [4] and Mohammad Havaei et al. [5] have given the prediction in 8 min and $25sec \sim 3min$ respectively which is not as good as our model. Even the lowest time of Hao Dong et al. [12] is $2 \sim 3sec$ which is also higher than our time delay.

Table 4.6: Comparison of different methods in respect of time for prediction

Method	Data	Time
Proposed Method [Figure 3.13]	BRATS 2015	less than $1sec$
Hao Dong et al. [12]	BRATS 2015	$2 \sim 3sec$
Sergio Pereira et al. [4]	BRATS 2015	$8min$
Mohammad Havaei et al. [5]	BRATS 2015	$25sec \sim 3min$

Our result can be increased in accuracy and efficiency, if we have used 12GB GPU like [5], [4] and [12] on same dataset. Still the efficiency measurement of our model is highest.

Chapter 5

Future Work

We tried to implement a novel architecture to detect and segment brain tumor in an easy & efficient way. We have classified the data between normal brain & brain with gliomas (tumors) as well as between high grade & low grade tumor patients using our model Convolutional Neural Network architecture in detection part. We also segmented all 4 kinds of MRI images of both HGG and LGG patients in BRATS 2015 dataset and evaluated our results compared with ground truth images. But there are still some limitations of the current work both in detection and segmentation model.

5.1 Limitations

The classification of the brain, based on tumor grade, is not precise enough. So we should have used more independent data consisted of different kinds of MRI images of many more various patients. It helps the model to learn more about the tumors as well as their prediction of the detection of tumor areas. The method we built for semantic segmentation has been evaluated using a cross-validation scheme, which can provide an unbiased predictor. But by running our model on a separate and independent testing dataset may produce a more effective evaluation. After that, there are several parameters in our architecture which should be carefully tuned. In particular, regularization like L1, L2 or dropout should be counted as a parameter, though [12] used all these regularizations but didn't get any performance upgrade. This may be a fact that an effective image distortion should have been used in our network which was applied during [12] model, and it is difficult to overfit a network with a large amount of training data.
Moreover, our current framework is less successful in term of dice coefficient score which is the performance metrics of all the method which have used BRATS datasets than state-of-the-art [29]. But [29] and other methods have used higher GPU than us and so, in the

future, we will like to evaluate our model in higher GPU than 4 GB in which we are currently executing our work. Less GPU gives less computational help and the model weight that we desire for our network can't be obtained in the current situation. Though we implemented our model in a much lesser GPU than other, we got faster performance in training as well as in predicting tumor regions.

5.2 Possible Future Approaches

To develop our network architecture's performance, we can tune in the hyperparameters of our fully convolutional neural network to examine which parameter is affecting the metrics. Hyperparameters in a CNN based network are sensitive to each other. As all the layers are connected in a U-shaped manner, the tuning should be the same in both the encoding and decoding section.

We can also improve the model by adding or removing more depth from the network which is the number of blocks on both sides of the network. Adding new layers not only makes the network more complicated but also it makes it heavier in all sorts of computational complexities. It will take more time than usual which will decrease the performance of the model in terms of time-consumption. But if we remove layers from the network than the parameters mentions in the network model will surely reduce the performance metrics as well as increase the losses per epoch. As for increasing epochs, the validation accuracy and dice soft coefficient don't improve at all. It will only increase the time consumption of the proposed architecture.

From the above discussion, we can see that adding layers will only affect the performance of the model and so, we should tune the hyperparameters of the neural network architecture to get better DSC. If convert the environment from 4 GB to 12 GB the computation time will reduce gradually and for initializing the model with a better weight of the model architecture, the training will be smoother than ever and give better metrics and reduce the loss per epoch both in training and validation testing.

There is one more way to make the proposed model more efficient which is by feeding the network with more independent data rather than the BRATS 2015 dataset. This introduces the model with a more variable environment and many other variables to deal within neurons. It will make the data more competent and well organized for real-life experiments where computer diagnosis is much needed than manual ones. As the manual segmentation is tedious, automatic segmentation with a mostly accurate method like our network will be handy for the patients.

Finally, in the future, we want to know more models that have been used in biomedics and have an accuracy higher in that fields which can be used in brain tumor segmentation as biomedical images are quite similar. This study will help us to improve the architecture of

our network and diagnosis will be better than ever. We also employ model on survival rate from the MRI images with their statistical survey which can help a network predict the level of threat of the tumors as well as foresee the growth of the tumor and its nature.

Chapter 6

Conclusion

For the time being brain tumor is one of the focal perilous problems in this world. Radiotherapy, chemotherapy, advanced imaging, and surgical procedure are preferred for aid but still, certain cases of a malignant tumor are considered untreatable. The segmentation of MRI scans can proficiently supervise tumor treatment. Operating by the humans, the manual segmentation is considered to be time consuming and labor-intensive. Hence, for a multifold, distributional and longitudinal clinical trial, an effective, objective and propagating solution is egregiously needed.

Deep learning had a propensity for delivering an applicable and functional result for the mentioned vocation. Formed by several convolutional layers, the convolutional Neural Network allows the functioning of a hierarchy of features from the MRI to constitute a dynamic and cogent model. In this paper, we approached a fully automatic brain tumor detection and segmentation method using the 3D U-Net based deep convolution network. By using our CNN architecture, we can classify between an abnormal and normal brain and also differentiate between high-grade gliomas and low-grade gliomas and using the 3D U-Net architecture we can segment the MRI images and the network is trained to distinguish the high-grade and low-grade gliomas.

Our result is mainly divided into two primary fragments. Detection of the tumors have been done by prosecuting our approached CNN architecture where it has achieved 84.07% max accuracy. Classification between low-grade gliomas and high-grade gliomas, where the network has scored 69.25% accuracy. Later by promulgating our U-Net network for segmentation, we had set the accuracy to 93% and dice score to 79%. Along with the detection and segmentation we have barred down the training time by a large scale. Thus the computation time has been decremented to a handsome extent. The proposed method can inaugurate a patient concentrate brain tumor segmentation model without manual inference and this empirical management can be exercised into the diagnosis, treatment, and surgical planning and patient welfare.

References

[1] "Brain tumor: Statistics." https://www.cancer.net/cancer-types/brain-tumor/statistics. Accessed: 2019-03.

[2] "Side effects of a brain tumour." https://www.thebraintumourcharity.org/living-with-a-brain-tumour/side-effects/. Accessed: 2018-09-30.

[3] T. B. Olaf Ronneberger, Philipp Fischer, "U-net: Convolutional networks for biomedical image segmentation," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015*, pp. 234–241, 2015.

[4] V. A. Sergio Pereira, Adriano Pinto and C. A. Silva, "Brain tumor segmentation using convolutional neural networks in mri images," in *IEEE TRANSACTIONS ON MEDICAL IMAGING: Volume 35, Number 5*, pp. 1240–1251, 2016.

[5] D. W.-F. A. B. A. C. Y. B. C. P. P.-M. J. H. L. Mohammad Havaei, Axel Davy, "Brain tumor segmentation with deep neural networks," in *Medical Image Analysis: Volume 35*, pp. 18–31, 2017.

[6] S. B. J. K.-C. K. F. J. K. Y. B.-N. P. J. S. R. W. L. L. E. G. M.-A. W. T. A. B. B. A. N. A. P. B. D. L. C. N. C. J. J. C. A. C. T. D. H. D. Ã. D. C. R. D. M. D. S. D. J. F. F. F. E. G. B. G. P. G. X. G. A. H. K. M. I. R. J. N. M. J. E. K. D. L. J. A. M. R. M. S. P. D. P. S. J. P. T. R. R. S. M. S. R. M. R. D. S. L. S. H.-C. S. J. S. C. A. S. N. S. N. K. S. G. S. T. J. T. O. M. T. N. J. T. G. U. F. V. M. W. D. H. Y. L. Z. B. Z. D. Z. M. P. M. R. Bjoern H. Menze, Andras Jakab and K. V. Leemput, "The multimodal brain tumor image segmentation benchmark (brats)," in *IEEE TRANSACTIONS ON MEDICAL IMAGING: Volume 34, Number 10*, pp. 1993–2024, 2015.

[7] M. Y. S. L. F. Javeria Amin, Muhammad Sharif, "Big data analysis for brain tumor detection: Deep convolutional neural networks," in *Future Generation Computer Systems: Volume 87*, pp. 290–297, 2018.

[8] M. S. Ali Isin, Cem Direkoglu, "Review of mri-based brain tumor image segmentation using deep learning methods," in *Procedia Computer Science: Volume 102*, pp. 317–324, 2016.

[9] T. D. Jonathan Long, Evan Shelhamer, "Fully convolutional networks for semantic segmentation," in *The IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3431–3440, 2015.

[10] S.-A. A. Fausto Milletari, Nassir Navab, "V-net: Fully convolutional neural networks for volumetric medical image segmentation," in *Fourth International Conference on 3D Vision (3DV)*, pp. 1–11, 2016.

[11] G. C. S. K. Michal Drozdzal, Eugene Vorontsov and C. Pal, "The importance of skip connections in biomedical image segmentation," in *Deep Learning and Data Labeling for Medical Applications*, pp. 179–187, 2016.

[12] F. L. Y. M. Y. G. Hao Dong, Guang Yang, "Automatic brain tumor detection and segmentation using u-net based fully convolutional networks," in *Medical Image Understanding and Analysis*, pp. 506–517, 2017.

[13] K. R. A.-B. M. S. El-Sayed A. El-Dahshan, Heba M. Mohsen, "Computer-aided diagnosis of human brain tumor through mri: A survey and a new algorithm," in *Expert Systems with Applications: Volume 41, Issue 11*, pp. 5526–5545, 2014.

[14] M. Kailash D.Kharat, Pradyumna P.Kulkarni, "Brain tumor classification using neural network based methods," in *International Journal of Computer Science and Informatics, ISSN (PRINT) 2231–5292: Volume 1, Issue 2*, 2012.

[15] G. M. S. Dina Aboul Dahab, Samy S. A. Ghoniemy, "Automated brain tumor detection and identification using image processing and probabilistic neural network techniques," in *International Journal of Image Processing and Visual Communication, ISSN (Online) 2319–1724: Volume 1, Issue 2*, 2012.

[16] P. John, "Brain tumor classification using wavelet and texture based neural network," in *International Journal of Scientific & Engineering Research, ISSN 2229–5518: Volume 3, Issue 10*, 2012.

[17] R.-A. EmanAbdel-Maksoud, MohammedElmogy, "Brain tumor segmentation based on a hybrid clustering technique," in *Egyptian Informatics Journa: Volume 16, Issue 1*, pp. 71–81, 2015.

[18] D. B. G. R. V.-F. R. M. Matthew C. Clark, Lawrence O. Hall and M. S. Silbiger, "Automatic tumor segmentation using knowledge-based techniques," in *IEEE TRANSACTIONS ON MEDICAL IMAGING: Volume 17, Issue 2*, pp. 187–201, 1998.

[19] L.-P. N. Stefan Bauer, Roland Wiest and M. Reyes, "A survey of mri-based medical image analysis for brain tumor studies," in *Physics in Medicine & Biology: Volume 58, Number 13*, 2013.

[20] P. P. A. K. S. A. E. B. Devkota, Abeer Alsadoona, "Image segmentation for early stage brain tumor detection using mathematical morphological reconstruction," in *Procedia Computer Science: Volume 125*, pp. 115–123, 2018.

[21] L.-P. N. Stefan Bauer and M. Reyes, "Fully automatic segmentation of brain tumor images using support vector machine classification in combination with hierarchical conditional random field regularization," in *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2011*, pp. 354–361, 2011.

[22] D. R. H. R. G Rajesh Chandra, "Tumor detection in brain using genetic algorithm," in *Procedia Computer Science: Volume 79*, pp. 449–457, 2016.

[23] T. Logeswari and M. Karnan, "An improved implementation of brain tumor detection using segmentation based on soft computing," in *Journal of Cancer Research and Experimental Oncology: Volume 2(1)*, pp. 006–014, 2010.

[24] S. K. B. Sudipta Roy, "Detection and quantification of brain tumor from mri of brain and it's symmetric analysis," in *International Journal of Information and Communication Technology Research, ISSN 2223–4985: Volume 2, Number 6*, 2012.

[25] A. I. Umit Ilhana, "Brain tumor segmentation based on a new threshold approach," in *Procedia Computer Science: Volume 120*, pp. 580–587, 2017.

[26] D. K. S. P. K. SUDHARANI, Dr.T.C. SARMA, "Advanced morphological technique for automatic brain tumor detection and evaluation of statistical parameters," in *Procedia Technology: Volume 24*, pp. 1374–1387, 2016.

[27] M. Naskar, "An automated system for brain tumor detection & segmentation," in *Journal of Engineering Research and Studies: Volume 6, Issues 1*, pp. 03–05, 2015.

[28] J. B. Diederik P. Kingma, "Adam: A method for stochastic optimization," in *3rd International Conference for Learning Representations, San Diego*, 2015.

[29] V. F. J. P. A. D. D. K. D. R. B. G. Konstantinos Kamnitsas, Christian Ledig, "Efficient multi-scale 3d cnn with fully connected crf for accurate brain lesion segmentation," in *Medical Image Analysis: Volume 36*, pp. 61–78, 2017.

Appendix A

3D Data View

```
import matplotlib.pyplot as plt
import os
from medpy.io import load


image_data, image_header = load('D:/Thesis/BRATS2015_Training/HGG/
    brats_2013_pat0001_1/VSD.Brain.XX.O.MR_T1c.54514/VSD.Brain.XX.O.MR_T1c
    .54514.mha')
t1 = image_data
t1 = t1.T


def remove_keymap_conflicts(new_keys_set):
    for prop in plt.rcParams:
        if prop.startswith('keymap.'):
            keys = plt.rcParams[prop]
            remove_list = set(keys) & new_keys_set
            for key in remove_list:
                keys.remove(key)


def multi_slice_viewer(volume):
    remove_keymap_conflicts({'j', 'k'})
    fig, ax = plt.subplots()
    ax.volume = volume
    ax.index = volume.shape[0] // 2
    ax.imshow(volume[ax.index], cmap='gray')
    fig.canvas.mpl_connect('key_press_event', process_key)


def process_key(event):
    fig = event.canvas.figure
```

```
    ax = fig.axes[0]
    if event.key == 'j':
        previous_slice(ax)
    elif event.key == 'k':
        next_slice(ax)
    fig.canvas.draw()


def previous_slice(ax):
    volume = ax.volume
    ax.index = (ax.index - 1) % volume.shape[0]  # wrap around using %
    ax.images[0].set_array(volume[ax.index])


def next_slice(ax):
    volume = ax.volume
    ax.index = (ax.index + 1) % volume.shape[0]
    ax.images[0].set_array(volume[ax.index])


multi_slice_viewer(t1)


import k3d
import numpy as np


fam = np.array(image_data)
plot = k3d.plot()
isosurface = k3d.marching_cubes(fam, level=5)
plot += isosurface
plot.display()
```

Appendix B

Code for Tumor Detection

```
from keras.models import Sequential
from keras.layers import Convolution2D
from keras.layers import MaxPooling2D
from keras.layers import Flatten
from keras.layers import Dense
import matplotlib.pyplot as plt


classifier = Sequential()

classifier.add(Convolution2D(32, 3, 3, input_shape= (64, 64, 3), activation = '
    relu'))
classifier.add(Convolution2D(32, 3, 3, activation = 'relu'))
classifier.add(Convolution2D(32, 3, 3, activation = 'relu'))
classifier.add(MaxPooling2D(pool_size = (2, 2)))


classifier.add(Convolution2D(64, 3, 3, activation = 'relu'))
classifier.add(Convolution2D(64, 3, 3, activation = 'relu'))
classifier.add(Convolution2D(64, 3, 3, activation = 'relu'))
classifier.add(MaxPooling2D(pool_size = (2, 2)))


classifier.add(Flatten())


classifier.add(Dense(output_dim = 128, activation = 'relu'))
classifier.add(Dense(output_dim = 128, activation = 'relu'))
classifier.add(Dense(output_dim = 1, activation = 'sigmoid'))


classifier.compile(optimizer = 'adam', loss = 'binary_crossentropy', metrics =
```

```
    ['accuracy'])


from keras.preprocessing.image import ImageDataGenerator

train_datagen = ImageDataGenerator(rescale = 1./255,
                                   shear_range = 0.2,
                                   zoom_range = 0.2,
                                   horizontal_flip = True)


test_datagen = ImageDataGenerator(rescale = 1./255)

training_set = train_datagen.flow_from_directory('dataset/BRATS2015_Training',
                                                 target_size = (64, 64),
                                                 batch_size = 32,
                                                 class_mode = 'binary')

test_set = test_datagen.flow_from_directory('dataset/Testing',
                                            target_size = (64, 64),
                                            batch_size = 32,
                                            class_mode = 'binary')


from sklearn.cross_validation import StandardScaler
sc_X = StandardScaler()
training_set = sc_X.fit_transform(training_set)
test_set = sc_X.fit_transform(test_set)


history = classifier.fit_generator(training_set,
                         samples_per_epoch = 1096,
                         nb_epoch = 25,
                         validation_data = test_set,
                         nb_val_samples = 274)


classifier.summery()


# list all data in history
print(history.history.keys())
# summarize history for accuracy
plt.plot(history.history['acc'])
plt.plot(history.history['val_acc'])
plt.title('model accuracy')
plt.ylabel('accuracy')
```

```
78 plt.xlabel('epoch')
79 plt.legend(['train', 'test'], loc='upper left')
80 plt.show()
81
82
83 # summarize history for loss
84 plt.plot(history.history['loss'])
85 plt.plot(history.history['val_loss'])
86 plt.title('model loss function')
87 plt.ylabel('loss rate')
88 plt.xlabel('epoch')
89 plt.legend(['train', 'test'], loc='upper left')
90 plt.show()
```

Appendix C

Code for Tumor Segmentation

```
import numpy as np
import pandas as pd
import matplotlib.pyplot as plt
import skimage.io as io
import skimage.transform as trans
import random as r
from keras.optimizers import Adam

import glob

from keras.models import Input, Model
from keras.layers import Conv3D, Concatenate, MaxPooling3D, Reshape
from keras.layers import UpSampling3D, Activation, Permute

import keras.backend as K


def create_data(src, mask, label=False, resize=(128,128,128)):
    files = glob.glob(src + mask, recursive=True)
    imgs = []
    for file in files:
        img = io.imread(file, plugin='simpleitk')
        if label:
            img[img > 0] = 1
            img = img.astype('float32')
        else:
            img = (img-img.mean()) / img.std()
        img = trans.resize(img, resize, mode='constant')
        imgs.append(img)
    name = 'Y_train' if label else 'X_train'
    np.save(name, np.array(imgs)[..., np.newaxis].astype('float32'))
    print('Saved', len(files), 'to', name)
```

```
33
34
35 def level_block_3d(k, dim, depth, factor, acti):
36     if depth > 0:
37         n = Conv3D(dim, 3, activation=acti, padding='same')(k)
38         m = Conv3D(dim, 3, activation=acti, padding='same')(n)
39         m = MaxPooling3D()(n)
40         m = level_block_3d(m, int(factor*dim), depth-1, factor, acti)
41         m = UpSampling3D()(m)
42         m = Concatenate(axis=4)([n, m])
43         m = Conv3D(dim, 3, activation=acti, padding='same')(m)
44     return Conv3D(dim, 3, activation=acti, padding='same')(m)
45
46
47 def UNet_3D(img_shape, n_out=1, dim=64, depth=4, factor=2, acti='elu'):
48     i = Input(shape=img_shape)
49     o = level_block_3d(i, dim, depth, factor, acti)
50     o = Conv3D(n_out, 1, activation='sigmoid')(o)
51     return Model(inputs=i, outputs=o)
52
53
54 def f1_score(y_true, y_pred):
55     y_true_f = K.flatten(y_true)
56     y_pred_f = K.flatten(y_pred)
57     intersection = K.sum(y_true_f * y_pred_f)
58     return (2. * intersection + 1.) / (K.sum(y_true_f) + K.sum(y_pred_f) + 1.)
59
60 def f1_loss(y_true, y_pred):
61     return 1-f1_score(y_true, y_pred)
62
63
64 create_data('BRATS2015_Training/HGG/', '/**/*T1c*.mha', label=False, resize
       =(32,32,32))
65 create_data('BRATS2015_Training/HGG/', '/**/*OT*.mha', label=True, resize
       =(32,32,32))
66
67
68 x = np.load('/content/gdrive/My Drive/X_train.npy')
69 print('x: ', x.shape)
70 y = np.load('/content/gdrive/My Drive/Y_train.npy')
71 print('y:', y.shape)
72
73
74 import random as r
75 i = int(r.random() * x.shape[0])
76 plt.figure(figsize=(10,10))
77 plt.subplot(121)
```

```
plt.imshow(x[i, int(x.shape[1]/2), :, :, 0])
plt.subplot(122)
plt.imshow(y[i, int(y.shape[1]/2), :, :, 0])
plt.show()


model = UNet_3D(x.shape[1:], dim=32, factor=2)
model.compile(optimizer=Adam(lr=0.0001), loss=f1_loss, metrics=['accuracy'])


from sklearn.cross_validation import train_test_split
X_train, X_valid, Y_train, Y_valid = train_test_split(x, y, test_size = 0.2,
    random_state = 0)


from keras.preprocessing.image import ImageDataGenerator


datagen = ImageDataGenerator(
    rotation_range=20,
    width_shift_range=0.1,
    height_shift_range=0.1,
    horizontal_flip=True,
    vertical_flip=True,
    shear_range=0.2,
    zoom_range=0.1,
    brightness_range=[0.8,1.2])


datagen.fit(X_train)


history = model.fit_generator(datagen.flow(X_train, Y_train, batch_size=8),
    validation_data=(X_valid, Y_valid),steps_per_epoch=len(X_train)/8,epochs
    =300)
model.save_weights('demo_weights.h5')
pred = model.predict(x[:60])


import random as r
num = int(x.shape[1]/2)
for n in range(3):
    i = int(r.random() * pred.shape[0])
    plt.figure(figsize=(15,10))

    plt.subplot(131)
    plt.title('Input')
```

```
122     plt.imshow(x[i, num, :, :, 0])
123 
124     plt.subplot(132)
125     plt.title('Ground Truth')
126     plt.imshow(y[i, num, :, :, 0])
127 
128     plt.subplot(133)
129     plt.title('Prediction')
130     plt.imshow(pred[i, num, :, :, 0])
131 
132     plt.show()
133 
134 
135 model.summary()
136 
137 
138 # list all data in history
139 print(history.history.keys())
140 # summarize history for accuracy
141 plt.plot(history.history['acc'])
142 plt.plot(history.history['val_acc'])
143 plt.title('Segmentation accuracy')
144 plt.ylabel('accuracy')
145 plt.xlabel('epoch')
146 plt.legend(['train', 'valid test'], loc='lower right')
147 plt.show()
148 
149 
150 # summarize history for loss
151 plt.plot(history.history['loss'])
152 plt.plot(history.history['val_loss'])
153 plt.title('Segmentation loss')
154 plt.ylabel('loss')
155 plt.xlabel('epoch')
156 plt.legend(['train', 'valid test'], loc='upper right')
157 plt.show()
```

Generated using Undegraduate Thesis LaTeX Template, Version 1.4. Department of Computer Science and Engineering, Ahsanullah University of Science and Technology, Dhaka, Bangladesh.

This thesis was generated on Monday 3rd January, 2022 at 8:20pm.