SEGMENTING BRAIN TUMOR ON MRI IMAGES USING LIGHTWEIGHT SE UNET MODEL

Major project report submitted
in partial fulfillment of the requirement for award of the degree of

Bachelor of Technology
in
Computer Science & Engineering

By

LOGESHWARAN K S (20UECS0530) **(VTU15366)**
KALLURI ANISHA DEVI (20UECS0441) **(VTU17604)**

Under the guidance of
Dr. S. SARAN RAJ, M.E., Ph.D.,
ASSISTANT PROFESSOR

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SCHOOL OF COMPUTING

VEL TECH RANGARAJAN DR. SAGUNTHALA R&D INSTITUTE OF SCIENCE & TECHNOLOGY

(Deemed to be University Estd u/s 3 of UGC Act, 1956)
Accredited by NAAC with A++ Grade
CHENNAI 600 062, TAMILNADU, INDIA

May, 2024

SEGMENTING BRAIN TUMOR ON MRI IMAGES USING LIGHTWEIGHT SE UNET MODEL

Major project report submitted
in partial fulfillment of the requirement for award of the degree of

Bachelor of Technology
in
Computer Science & Engineering

By

LOGESHWARAN K S	(20UECS0530)	**(VTU15366)**
KALLURI ANISHA DEVI	(20UECS0441)	**(VTU17604)**

Under the guidance of
Dr. S. SARAN RAJ, M.E., Ph.D.,
ASSISTANT PROFESSOR

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING
SCHOOL OF COMPUTING

VEL TECH RANGARAJAN DR. SAGUNTHALA R&D INSTITUTE OF SCIENCE & TECHNOLOGY

(Deemed to be University Estd u/s 3 of UGC Act, 1956)
Accredited by NAAC with A++ Grade
CHENNAI 600 062, TAMILNADU, INDIA

May, 2024

CERTIFICATE

It is certified that the work contained in the project report titled “SEGMENTING BRAIN TUMOR ON MRI IMAGES USING LIGHTWEIGHT SE UNET MODEL” by “LOGESHWARAN K S (20UE CS0530), KALLURI ANISHA DEVI (20UECS0441)” has been carried out under my supervision and that this work has not been submitted elsewhere for a degree.

Signature of Supervisor	**Signature of Professor In-charge**
Computer Science & Engineering	**Computer Science & Engineering**
School of Computing	**School of Computing**
Vel Tech Rangarajan Dr. Sagunthala R&D	**Vel Tech Rangarajan Dr. Sagunthala R&D**
Institute of Science & Technology	**Institute of Science & Technology**
May, 2024	**May, 2024**

DECLARATION

We declare that this written submission represents our ideas in our own words and where others' ideas or words have been included, we have adequately cited and referenced the original sources. We also declare that we have adhered to all principles of academic honesty and integrity and have not misrepresented or fabricated or falsified any idea in our submission. We understand that any violation of the above will be cause for disciplinary action by the Institute and can also evoke penal action from the sources which have thus not been properly cited or from whom proper permission has not been taken when needed.

LOGESHWARAN K S

Date: / /

KALLURI ANISHA DEVI

Date: / /

APPROVAL SHEET

This project report entitled SEGMENTING BRAIN TUMOR ON MRI IMAGES USING LIGHTWEIGHT SE UNET MODEL by LOGESHWARAN K S (20UECS0530), KALLURI ANISHA DEVI (20UECS0441) is approved for the degree of B.Tech in Computer Science & Engineering.

Examiners	**Supervisor**
	Dr. S. SARAN RAJ, M.E., Ph.D.,
	ASSISTANT PROFESSOR

Date: / /

Place:

ACKNOWLEDGEMENT

We express our deepest gratitude to our respected **Founder Chancellor and President Col. Prof. Dr. R. RANGARAJAN B.E. (EEE), B.E. (MECH), M.S (AUTO),D.Sc., Foundress President Dr. R. SAGUNTHALA RANGARAJAN M.B.B.S.** Chairperson Managing Trustee and Vice President.

We are very much grateful to our beloved **Vice Chancellor Prof. S. SALIVAHANAN,** for providing us with an environment to complete our project successfully.

We record indebtedness to our **Professor & Dean, Department of Computer Science & Engineering, School of Computing, Dr. V. SRINIVASA RAO, M.Tech., Ph.D.,** for immense care and encouragement towards us throughout the course of this project.

We are thankful to our **Head, Department of Computer Science & Engineering, Dr. M. S. MURALI DHAR, M.E., Ph.D.,** for providing immense support in all our endeavors.

We also take this opportunity to express a deep sense of gratitude to our **Internal Supervisor Dr. S. SARAN RAJ, M.E., Ph.D.,** for his cordial support, valuable information and guidance, he helped us in completing this project through various stages.

A special thanks to our **Project Coordinators Mr. V. ASHOK KUMAR, M.Tech., Ms. C. SHYAMALA KUMARI, M.E.,** for their valuable guidance and support throughout the course of the project.

We thank our department faculty, supporting staff and friends for their help and guidance to complete this project.

LOGESHWARAN K S	**(20UECS0530)**
KALLURI ANISHA DEVI	**(20UECS0441)**

ABSTRACT

All over the world, the leading cause of people's death is cancer. The most common types of cancer are brain, breast, colon, lung, prostate, rectum, and skin. Cancer is nothing but a group of abnormal cells that grow beyond the limited boundaries. It infiltrates and destroys normal body tissue, and makes it hard for people to survive. Brain tumor has the highest death rate compared to other cancer-affected people. In America (2023), 94,390 people have been diagnosed and their relative survival rate is 35.7%. Survival rates depend on the grade (low grade and high grade) and prognostic factors of the tumor. Medical heads are doing their best to cure but there are many unknown reasons behind the formation of tumors. Brain tumors are diagnosed with the help of Magnetic Reasoning Imaging (MRI) and Computed Tomography (CT) scans. There are many automated segmentation and classification techniques available in Deep Learning (DL) with base U-NET architecture but there are some disadvantages that come along with it like more time to train, higher convergence rate, repeated layer, and so on. With the help of MRI data along with its different modalities, we are proposing a 2-dimensional hybrid architecture with a combination of Squeeze-and-Excitation (SE) to suppress the irrelevant feature to make up the issue of redundancy, and U-NET as a base architecture. This hybrid architecture provides significant performance while evaluating the T1 weighted dataset from the Figshare repository. The results obtained through our architecture are 98.4% of accuracy, 1.5% of loss, 81.9% of dice coeff, and 99.72% of specificity are mentioned along with hyper-parameters used and tested with our model.

Keywords: Brain Tumor, Image Pre-processing, Segmentation, Bio-Medical Imaging, Squeeze-and-Excitation, U-NET Architecture

LIST OF FIGURES

LIST OF TABLES

LIST OF ACRONYMS AND ABBREVIATIONS

AI	Artificial Intelligence
ANN	Artificial Neural Networks
API	Application Programming Interface
ASPP	Atrous Spatial Pyramid Pooling
AUC	Area Under the ROC Curve
BIBM	Bioinformatics and Biomedicine
BraTS	Brain Tumor Segmentation
CAD	Computer-Aided Design
CBTRUS	Central Brain Tumor Registry of the United States
CDC	Centers for Disease Control
CNN	Convolutional Neural Network
CNS	Central Nervous System
CRF	Conditional Random Field
CT	Computed Tomography
DCNN	Deep Convolutional Neural Networks
DL	Deep Learning
DS	Dice Score
DT	Decision Tree
FCN	Fully Convolutional Network
FLAIR	Fluid Attenuated Inversion Recovery
GAN	Generative Adversarial Networks
GB	Giga Byte
GPU	Graphics Processing Unit
HGG	High-Grade Glioblastoma

ICAIT	International Conference on Advances in Information Technology
IDE	Integrated Development Environment
IEC	International Electrotechnical Commission
ISO	International Organization for Standardization
KNN	K-Nearest Neighbors
LGG	Low-Grade Glioblastoma
ML	Machine Learning
MRI	Magnetic Resonance Imaging
NCI	National Cancer Institute’s
NPCR	National Program of Cancer Registries
OS	Operating System
PET	Positron Emission Tomography
RAM	Random Access Memory
RF	Radio Frequency
ROC	Receiver Operating Characteristic
SGD	Stochastic Gradient Descent
SE	Squeeze-and-Excitation
SEER	Surveillance, Epidemiology, and End Results
SVM	Support Vector Machine
URL	Uniform Resource Locator
US	United States
XGBoost	Extreme Gradient Boosting

TABLE OF CONTENTS

Chapter 1

INTRODUCTION

1.1 Introduction

The National Center for Health Statistics projects that 1,918,030 instances of cancer will take place in 2022 [9]. Of which 609,360 died as a result of complications related to cancer. In 2023 [10], approximately 94,390 people will be diagnosed and their relative survival rate is 35.7%. Medical imaging systems are the most widely utilized for assisting with decisions that are significant for medical diagnosis and therapy. These instruments are widely used to locate internal lesions, such as malignancies. Techniques for image processing tend to be employed in the construction of medical imaging equipment. In modern times, a system that can recognize malignant tumors and other diseases has become indispensable. Tumors called gliomas seem to have emerged from glial cells, which is why they can infect adjoining healthy tissues. Glioblastoma, the main categorization for these tumors, is split into two variants: high-grade and low-grade.

CT and MRI serve numerous clinical uses, from tumor diagnosis to sickness classification. Nuclear magnetic resonance is a complex method used in MRIs. You may quickly optimize a wide range of MRI settings that generate a variety of resolutions, sharpnesses, and anatomical features. A crucial instrument that will assist decision-makers come to extremely accurate verdicts is the application of segmentation techniques to MRI and CT scans, which are also used in the identification of these brain tumors. As a result, effective glioma segmentation may significantly improve patient survival rates and empower surgeons with every aspect of surgical care, including preoperative and postoperative planning.

One of the primary ways to tackle this challenge is to incorporate data from different MRI techniques, including FLAIR, T1, T1 contrast, and T2. The advancement of automated segmentation algorithms has permitted practitioners to work in this field more passionately and efficiently. Brain tumors can be safely treated by medical experts, especially during surgery, by exploiting the quick and precise segmentation capabilities of imaging technologies. Utilization of 3D convolutions demands an extensive processing cost and memory bandwidth in comparison to the 2D U-NET architecture approach of extracting tumor regions from healthy tissue on the BraTS2020 dataset. To assist physicians by having a computer enhance the effectiveness of the initial scan, it tested with a 2-dimensional U-NET in conjunction with the SE block. It achieves this through two main operations: squeeze, which aggregates the global spatial information of feature maps into a channel descriptor, and excitation, which learns to recalibrate the feature channels by using the descriptor to emphasize informative features while suppressing less useful ones. By integrating SE blocks into traditional convolutional networks, models can achieve significant gains in accuracy and efficiency, particularly in tasks that require a detailed and context-sensitive understanding of input data. It has been compared model against several current Convolutional Neural Networks (CNN) models and assesses the metrics obtained by applying the recommended methodology.

1.2 Aim of the Project

The primary aim of the project is to use a lightweight SE U-NET architecture specifically designed for processing T1-weighted MRI images to create an advanced and effective brain tumor segmentation system. The objective is to precisely identify the areas of the tumor, which will help with the proper diagnosis and efficient treatment planning of patients suffering from brain tumors. Here the power of CNNs reinforced with SE blocks is utilized. By dynamically recalibrating channel-wise feature responses, SE blocks greatly boost the representational capacity of the network.

The model can concentrate more on informative features by incorporating the SE blocks into the U-NET architecture, which improves segmentation accuracy. Large data quantities are handled by the system with efficiency, guaranteeing stable performance regardless of changes in tumor morphology and picture quality. The ultimate goal of this project is to advance medical imaging by offering a dependable tool that, while preserving high levels of accuracy and computational efficiency, improves the interpretation of MRI scans, makes early tumor detection easier, and supports tailored treatment plans.

1.3 Project Domain

The research is in the field of medical image processing, with a particular emphasis on the use of DL methods for MRI brain tumor segmentation. This field combines aspects of computer vision, Artificial Intelligence (AI), and medicine to create algorithms that can decipher complicated medical images. One state-of-the-art method in the field is the use of CNNs, namely the modified U-NET architecture with integrated SE blocks. The objective of this study is to improve the precision and effectiveness of tumor detection and segmentation by utilizing these cutting-edge Machine Learning (ML) algorithms. In this way, it advances the more general objectives of enhancing neurology and oncology patient outcomes, treatment planning, and diagnostic procedures.

1.4 Scope of the Project

The scope of this project is to create, evaluate, and implement a DL system that will separate brain cancers from T1-weighted MRI data. With a lightweight SE U-NET design that uses creative SE blocks to promote feature recalibration and network efficiency, the project's specific goal is to improve segmentation accuracy. The gathering and preparation of MRI data, as well as the thorough training and validation of the neural network model on annotated datasets, constitute a substantial portion of the scope.

In addition, the study seeks to verify that the model satisfies clinical requirements for medical diagnostic tools by assessing its performance using quantitative metrics including accuracy, Dice coefficient, and loss. This lays the groundwork for upcoming improvements and possible integration with real-time medical imaging systems, broadening the scope to encompass clinical real-world applications and ongoing model refinement in response to input and changing needs.

Chapter 2

LITERATURE REVIEW

[1] S Saran Raj et al., (2024) introduced a Gaussian-based lightweight U-NET architecture that yielded promising results. Gaussian smoothing preprocessing is used to cut the edges and blur the surrounding regions by reducing the noise which makes it suitable for segmentation of the BraTS2020 dataset. U-NET is a doubled architecture of the CNN model. The BraTS2020 dataset was utilized for this research, which has multi-modalities like T1-weighted MRI (T1), T1-weighted MRI with contrast-enhanced (Tce), T2-weighted MRI (T2), segmented image, and Fluid-Attenuated Inversion Recovery (FLAIR). The accuracy, loss, and precision obtained by the research work are 0.9941, 0.0249, and 0.9927, respectively.

[2] Hafeez Hafiz et al., (2023) introduced a CNN architecture designed to extract intricate features from brain tumor images, particularly for glioma-type classification. Remarkably, the model comprises only 12 layers and a mere 0.287 million parameters, contrasting starkly with existing CNN models for glioma grading. It's capacity to extract features accurately and its classification performance for glioma grades. In the initial stage, the suggested CNN model serves for feature extraction. Experimental evaluations encompass feature extraction from state-of-the-art CNN models, including the proposed one, followed by classification utilizing SVM.

[3] Renugadevi M et al., (2023) addressed the challenges inherent in tumor segmentation, including issues like low contrast imaging, uncertain tumor location, unclear boundaries, and annotation biases. Employing advanced preprocessing, segmentation, and grading techniques, the study aims to predict tumor lifetimes. Among various encoder-decoder architectures explored, the U-NET++ architecture is selected for brain tumor detection, achieving an impressive 98% accuracy and an Intersection Over Union (IoU) score of 0.7483 during testing. The Stochastic Gradient Descent (SGD) method attains the highest accuracy of 96%, while extreme gradient boosting yields a Mean Square Error (MSE) of 93726.45.

[4] Ozkaya Cagin et al., (2023) presented two new approaches for brain tumor grade classification and segmentation. CNN models were used as the first approach to classify High-Grade Glioma (HGG) and Low-Grade Glioma (LGG) tumors and achieved 99.85% accuracy, 99.85% F1 and 99.92% AUC scores. Experimental results have shown that the proposed algorithm has improved to measure the complete tumor region 15% more compared to the fixed thresholding. The major contributions of the proposed work are given as follows: to obtain higher HGG-LGG classification accuracy, to evaluate the segmentation algorithm with the Dice Similarity (DS) metric for complete tumors to develop a novel adaptive threshold determination approach for MRI binarization

[5] Neamah, Karrar et al., (2023) analyzed DL algorithms for brain tumor diagnosis and classification, aiming to understand their efficacy and guide future research. It examines methodologies, including model types, architecture, and training strategies, to identify trends and areas for improvement. Findings provide insights into DL's benefits and limitations in this context, facilitating further advancements in medical research. The review covers brain tumor identification using DL from 2019 to 2022, demonstrating the algorithms' effectiveness through performance metric evaluation. The methodology involves a meticulous literature review, employing innovative strategies to accelerate model creation without extensive labeled datasets. Overall, the review aims to contribute to the advancement of brain tumor identification in medical research.

[6] Ullah Faizan et al., (2023) proposed an evolutionary lightweight model aimed at detecting brain cancer and classification, starting from the analysis of MRI. The proposed model named lightweight ensemble combines (weighted average and lightweight combines multiple XGBoost decision trees) is the modified version of the recent Multimodal Lightweight XGBoost. They evaluate their proposed model using the BraTS 2020 dataset. The dataset consists of 285 MRI scans of patients diagnosed with gliomas. The simulation results showed that their proposed model achieved 93.0% accuracy, 0.94 precision, 0.93 recall, 0.94 F1 score, and an area under the Receiver Operating Characteristic Curve (AUC-ROC) value of 0.984. The proposed model named lightweight ensemble is the enhancement for the current Multimodal Lightweight XGBoost.

[7] M. Rizwan et al., (2022) customized CNNA is proposed to categorize various grades and types of BT. The system's design is enhanced by utilizing diverse configurations to acquire the most suitable framework. After the 8515th iteration, the accuracy level achieved is almost 100%. Lastly, the best overall precision obtained during the test stage is 96.13%. the loss curve is nearly 0. The presented work has accomplished the most noteworthy accuracy rate of 97.14% and 99.8% through this research. This paper presented a CAD approach for detecting and categorizing BT's radiological images into three kinds(pituitary-tumor, glioma-tumor, and meningioma-tumor). They also classified glioma-tumor into various categories (Grade two, grade three, and grade four) utilizing the GCNN approach.

[8] Tejas Shelatkar et al., (2022) proposed a novel approach for the diagnosis of brain tumors using a lightweight DL model with fine-tuning. The proposed model is based on transfer learning and fine-tuning techniques, which can effectively reduce the time and computational resources required for training the model. The Brain Tumor Segmentation (BraTS) dataset was used to evaluate the performance of the proposed model. The dataset consists of MRI scans of patients with different types of brain images.

[9] Siegel, R. L et al., (2022) estimated the number of new cancer cases and deaths in the United States and compiled the most recent data on population-based cancer occurrence and outcomes. They summarized the progress that stagnated for breast and prostate cancers but strengthened for lung cancer, coinciding with changes in medical practice related to cancer screening and/or treatment. More targeted cancer control interventions and investment in improved early detection and treatment would facilitate reductions in cancer mortality.

[10] Quinn T Ostrom et al., (2022) provided a comprehensive summary of the current descriptive epidemiology of primary brain and other CNS tumors in the US population. Primary brain and other CNS tumors include those tumors that originate from the tissues of the brain or CNS. CBTRUS obtained the latest available population-based data on all the reported newly diagnosed primary brain and other CNS tumors from the Centers for Disease Control and Prevention's (CDC) National Program of Cancer Registries (NPCR), and the National Cancer Institute's (NCI) Surveillance, Epidemiology, and End Results (SEER) Program for diagnosis years 2015-2019.

[11] Nahian Siddique et al., (2021) presented a comprehensive review of U-NET and its variants for medical image segmentation. The paper provides an overview of the theoretical background of U-NET and its various modifications, as well as a detailed discussion of their applications in medical image analysis. Discuss the strengths and weaknesses of different variants of U-NET, such as Nested U-NET, Attention U-NET, and Residual U-NET. They also provide a comparison of U-NET with other popular segmentation models such as Fully Convolutional Networks (FCN) and DeepLab.

[12] Samia Mushtaq et al., (2021) compared the performance of different ML algorithms in detecting brain tumors using MRI data. The authors used data from MRI images of the brain, including healthy individuals and those affected by tumors. They applied various ML algorithms to the data, including SVM, KNN, ANN, DT, and RF. They evaluated the performance of each function based on various parameters such as accuracy, sensitivity, specificity, and area under the Receiver Operating Characteristic (ROC) curve. The results show SVM's highest accuracy of 97%.

[13] P. Khan et al., (2021) provided a comprehensive review of the principles and recent advances in ML and DL for brain diagnostics. The authors emphasize the importance of early detection and diagnosis of brain diseases such as Alzheimer's disease, Parkinson's disease, stroke, and brain tumors. The different types of neuroimaging used for diagnosis, such as MRI, CT, and Positron Emission Tomography (PET).

[14] Aledhari Mohammed et al., (2020) classified and segmented the whole HGG in multi-channel MRI scans using DL techniques. The BraTS'19 labeled dataset was used. The BraTS'19 labeled dataset had folders for both HGG and LGG. For this study, only HGG was focused on.They are each in the dimensions of 240x240x155. The optimizer used to minimize dice loss is ADAM for both the standard 2D-U-NET and 2D-SE-RU-NET. The learning rate is set to 1e-4. The smooth factor was set to 0.01. The 2D-SE-RU-NET is superior to the traditional 2D-U-NET for biomedical image segmentation of 3D images largely due to efficiency. The 2D-SE-RU-NET may be opted for by those who wish to save time or minimize computational costs.

[15] M. Ali et al., (2020) proposed an ensemble of two segmentation networks: a 3D CNN and a U-NET, in a significant yet straightforward combinative technique that results in better and more accurate predictions. The suggested ensemble achieved dice scores of 0.750, 0.906, and 0.846 for enhancing tumor, whole tumor, and tumor core, respectively, on the validation set, The outputs of these networks are segmentation map that differs in terms of segmented tumor sub-regions. The first model used in the ensemble is a 3D CNN - It uses a multifiber unit with weighted dilated convolutions, The second model of their ensemble is a 3D U-NET variant which is different from the classical U-NET architecture; leaky ReLUs replace the ReLU activation function. achieving mean dice scores of 0.750, 0.906, and 0.846 on enhancing tumor, whole tumor, and tumor core, respectively.

[16] N. Feng et al., (2020) combined the deep CNN with the cascading structure and a uniform learning framework is established with the use of a conditional random field. Secondly, the Conditional Random Field (CRF) is used for post-segmentation processing, which effectively solves the contradiction between the segmentation accuracy and the network depth, and the number of pooling times in the traditional convolutional network. The Adam algorithm is an improved algorithm based on the Adadata algorithm. the learning rate is set to 103, the block size is 5, Dice coefficient of the experimental method is superior to the two control methods, indicating that the ASPP structure and CRF can indeed improve the performance of DCNN in MRI image segmentation of brain tumors

[17] B. S. Vittikop et al, (2019) proposed system is designed on U-NET architecture that is a full CNN. It utilizes ”contracting to expand” ways with skip associations. Rather than utilizing a weight map for every pixel The input set is 4 3D scans (240x240x155) with different contrast settings (T1, T2, T1 with FLAIR and contrast enhancement), and the outcome is a labeled 240x240x155 picture where each voxel is segmented as (i) healthy tissue, (ii) necrosis tumor, (iii) edema, (iv) enhancing tumor. Semantic Segmentation to solve this problem used U-NET architecture, which uses a Fully Convolutional Network Model for the task.

Chapter 3

PROJECT DESCRIPTION

3.1 Existing System

Currently, available methods for classifying brain cancers on T1-weighted MRI images use CNN architectures, either standard or modified versions such as U-NET. Because these structures are good at capturing spatial dependencies and extracting useful features from images, they have been frequently used in medical image segmentation tasks. To obtain accurate segmentation findings in the context of brain tumor segmentation, the current method probably uses pre-trained CNN models that have been refined on annotated MRI datasets. Traditional CNN architectures are successful, but they have some drawbacks that affect how well they perform in tasks involving the segmentation of brain tumors. One such drawback is the propensity to ignore minute details or textures connected to tumor areas, particularly when tumors have uneven forms or diverse properties.

Disadvantages of Existing System

- U-NET has a deep architecture with high parameters to extract the feature from the data. The computation cost of the U-NET is high and makes it longer and harder to train the data.
- In each step, it uses two convolution layers in both the encoder and the decoder which serve the same.
- High computational complexity during training and inference, requiring significant resources such as GPUs and memory overhead.

3.2 Proposed System

A lightweight SE U-NET architecture is incorporated into the proposed system, which offers a novel approach to brain tumor segmentation on T1-weighted MRI images. In order to overcome the shortcomings of conventional CNN designs in medical image segmentation applications, this architecture combines the advantages of the U-NET and SE block techniques. The fundamental components of the U-NET design are a symmetric expanding path for accurate localization and a contracting path for context capture. The capacity of this paradigm to capture both local details and global context are both essential for accurate segmentation which led to its widespread adoption in the biomedical picture segmentation field. The incorporation of SE blocks into the U-NET architecture improves the model's accuracy in segmentation by enabling it to capture contextual information and fine-grained details.

Advantages of Proposed System

- The overall parameter of the traditional U-NET model is reduced by half by eliminating one of the convolutional layers.
- To overcome the feature extraction distortion, the SE block makes the features more highlighted for further levels of encoder and decoder in U-NET architecture.
- While training, the model attains a faster convergence rate which reduces the no of epochs required for training.

3.3 Feasibility Study

3.3.1 Economic Feasibility

The economic feasibility of optimized U-NET for brain tumor segmentation depends on several factors, including the cost of implementation, the potential benefits, and the return on investment. The potential benefits of optimized U-NET for brain tumor segmentation could outweigh the implementation costs. By enabling faster and more accurate diagnosis of brain tumors, the model could lead to more timely and effective treatment, ultimately improving patient outcomes and reducing the overall cost of care. Optimized U-NET could also facilitate medical research by providing reliable and accurate segmentation results for brain tumor analysis, potentially leading to new insights into the biology and characteristics of brain tumors and informing the development of new treatments.

It depends on the specific context and needs of healthcare institutions and research facilities. While there may be upfront implementation costs, the potential benefits could ultimately justify the investment, especially for institutions prioritizing innovation and excellence in patient care and research.

3.3.2 Technical Feasibility

The technical feasibility of optimized U-NET for brain tumor segmentation depends on several factors, including the availability and quality of the medical imaging dataset, the computational resources required for training and inference, and the expertise in medical image analysis and DL. Tumor regions typically constitute a small fraction of the overall image, making it difficult for the model to segment them accurately. Addressing this issue requires specialized loss functions, data augmentation techniques, and other optimization strategies. Segmentation requires significant computational resources, including high-performance GPUs and specialized deep-learning software.

This can be a challenge for smaller healthcare institutions or research facilities that may not have the necessary resources or expertise. There are technical challenges associated with developing and optimizing U-NET for brain tumor segmentation, with the right expertise, tools, and resources, it is feasible to develop an accurate and reliable model that can improve the diagnosis and treatment of brain tumors.

3.3.3 Social Feasibility

The effectiveness of the brain segmentation project using the U-NET architecture can be seen from several aspects. From a clinical perspective, this project could have a major impact on patient care and management by providing accurate and efficient tumor segmentation. In turn, this can help doctors make better treatment decisions and improve patient outcomes. Classification of medical images using DL techniques such as U-NET has become more common and accepted by medical professionals.

This project can contribute to the development and improvement of algorithms to increase accuracy and efficiency. It is important to ensure that the use of such algorithms is appropriately regulated and that patient privacy and confidentiality are maintained at all times. Therefore, it is essential to conduct extensive testing and validation of the algorithm and to obtain appropriate approvals from regulatory authorities before implementing it in a clinical setting.

3.4 System Specification

3.4.1 Hardware Specification

- RAM: Minimum 12 GB
- Processor: Intel Core i5
- GPU: NVIDIA Tesla K80
- Storage: 25GB

3.4.2 Software Specification

- Browser: Google Chrome
- Python: 3.7
- IDE: Google Colab
- Framework: TensorFlow/Keras
- Packages: NumPy, OS, Matplotlib, Skimage, Sklearn, h5py

3.4.3 Standards and Policies

Google Colab

Google Colab adheres to Google Cloud Platform's policies and terms of service, ensuring the security and privacy of user data and resources. Google Colab users are expected to comply with Google's Acceptable Use Policy, which prohibits activities such as spamming, distributing malware, and engaging in illegal activities. Users are responsible for ensuring the confidentiality and security of their data and code while using Google Colab. It is recommended to avoid storing sensitive information or proprietary data on Colab instances. Google Colab provides integration with Google Drive for saving and loading files, enabling users to store and access data securely. Google Colab instances have automatic shutdown mechanisms to conserve resources and prevent abuse.

Standard Used: ISO/IEC 27001

Chapter 4

METHODOLOGY

4.1 General Architecture

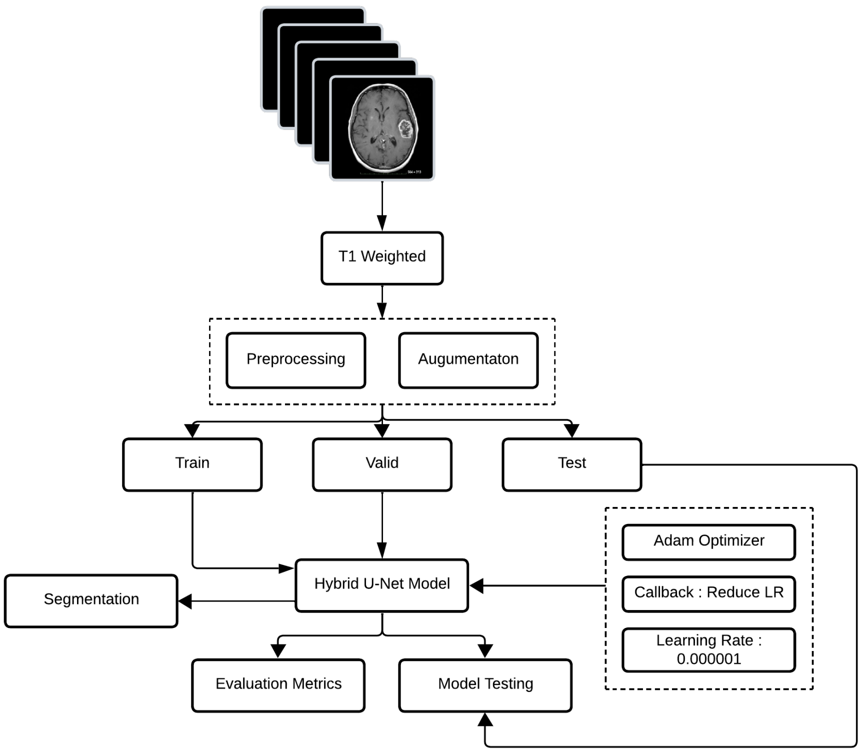


Figure 4.1: General Architecture of the Proposed System

Figure 4.1 depicts a brain tumor segmentation system using a lightweight SE U-NET model on T1-weighted MRI images. First, MRI images undergo preprocessing including skull removal, standardization, and potentially data augmentation. The preprocessed data is then split into training, validation, and testing sets. A single convolutional layer SE U-NET architecture is trained on the training data using an Adam optimizer. A callback function monitors validation loss and adjusts the learning rate to prevent overfitting. Finally, the trained model is evaluated on unseen test data using metrics like accuracy and dice coefficient to assess its effectiveness in segmenting brain tumors in new MRI scans.

4.2 Design Phase

4.2.1 Data Flow Diagram

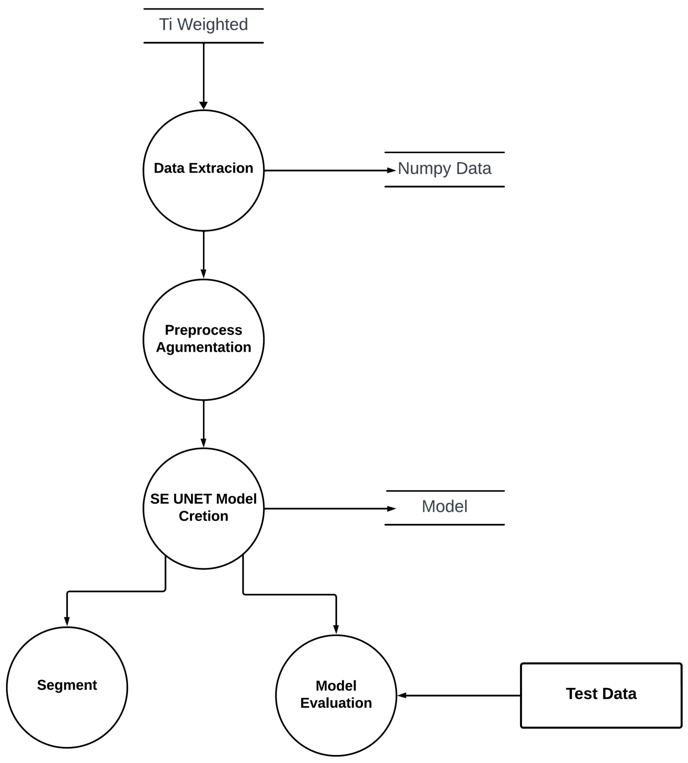


Figure 4.2: Data Flow Diagram of the Proposed System

Figure 4.2 explains the flow of the proposed system. The dataset is T1-weighted MRI images. After data extraction, these images are preprocessed and augmented. This preprocessing step might involve skull stripping and intensity normalization to prepare the data for the model. Data augmentation is employed. Then, the data flows into the SE U-NET model creation process. This involves defining the architecture of the model, which leverages a single convolutional layer for efficient feature extraction. This lightweight design aims to reduce computational complexity compared to traditional U-NET architectures. A training loop optimizes the model by iteratively feeding its image batches and adjusting its internal parameters based on the segmentation errors. Following this, the model is evaluated using unseen test data. Overall, the data follows a path from raw T1-weighted MRI images through preprocessing, model creation with a single convolutional layer SE U-NET architecture, training, and evaluation to achieve brain tumor segmentation. This approach balances accuracy with computational efficiency.

4.3 Algorithm & Pseudo Code

4.3.1 Algorithm

Algorithm 1 Brain Tumor Segmentation

```
1: Download the dataset from Figshare in zip format and extract all the data
2: Load MRI images, label, and corresponding masks
3: Store those data into the drive by converting them into a numpy array
4: Split the data into training and testing sets in the ratio of 80 to 20
5: Define image preprocessing functions (e.g., resizing, normalization)
6: Augment the training data (e.g., flip, zoom, adjust brightness)
7: Define SE block function
8: Construct SE U-NET Model with all parameters
9: Define loss functions (e.g., dice loss, binary cross-entropy + dice loss)
10: Define model training function
11: Train the model using training data
12: Evaluate the model on testing data
13: Threshold the predicted masks to binary
14: Visualize the original images, ground truth masks, and predicted masks for evaluation
```

4.3.2 Pseudo Code

```
1. Download Dataset from Figshare:
   - Use Figshare's official website to download the dataset in zip format.

2. Extract Data:
   - Unzip the downloaded file to extract all the data.

3. Load Data:
   - Load MRI images, labels, and corresponding masks.

4. Convert to Numpy Arrays:
   - Convert the loaded data into numpy arrays: images_array, labels_array, masks_array.

5. Split Data:
   - Split the data into training and testing sets:
    - images_train, images_test, masks_train, masks_test = train_test_split(images_array,
        masks_array, test_size=0.2, random_state=42)

6. Image Preprocessing:
   - Define image preprocessing functions:
     - resize(image, size): Resize images to 128 x 128x 1 size.
     - normalize(image): Normalize pixel values of images.
     - preprocess_image(image): Apply resizing and normalization to an image.

 7. Data Augmentation:
   - Augment the training data:
     - Define augmentation techniques such as flipping, zooming, and brightness adjustment.
     - Apply augmentation techniques to images_train and masks_train.

 8. Define SE Block Function:
   - Define a function to construct the Squeeze-and-Excitation (SE) block:
    - se_block(input_tensor, ratio): Implement the SE block.

 9. Construct SE U-NET Model:
   - Define the architecture of the SE U-NET model:
     - Define the contraction (encoder) and expansion (decoder) paths with SE blocks.
     - Include skip connections to preserve spatial information.
     - Define the output layer with sigmoid activation.

 10. Define Loss Functions:
    - Define loss functions for training the model:
      - dice_loss(y_true, y_pred): Calculate the Dice loss.
      - bce_dice_loss(y_true, y_pred): Calculate the binary cross-entropy + Dice loss.

 11. Define Model Training Function:
    - Define a function to train the SE U-NET model:
      - Compile the model with Adam optimizer and specified loss function.
      - Train the model using the fit method with training data.
      - Use callbacks for early stopping, learning rate reduction, and model checkpointing.

 12. Train the Model:
    - Train the SE U-NET model using the defined training function.

 13. Evaluate the Model:
    - Evaluate the trained model on the testing data:
      - Generate predictions for images_test.
      - Calculate evaluation metrics such as accuracy, dice coefficient, etc.

 14. Threshold Predictions:
    - Threshold the predicted masks to binary:
      - Apply a threshold value to the predicted masks.

 15. Visualization:
    - Visualize the original images, ground truth masks, and predicted masks for evaluation:
      - Display a set of images along with their corresponding ground truth masks and predicted
        masks.
```

4.4 Module Description

4.4.1 Collection of Data & Storing

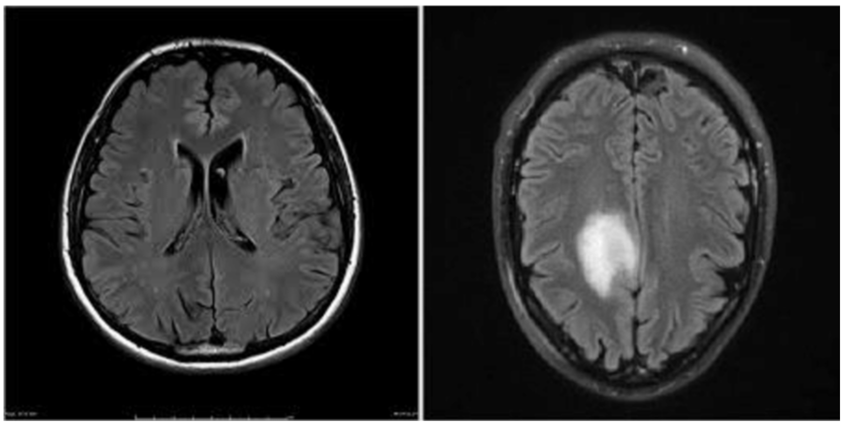


Figure 4.3: T1 Weighted Dataset

The initial step in this module is to collect the information required for segmenting brain tumors. The dataset was obtained from Figshare, a reliable source that is well-known for housing datasets that are openly accessible. It contains labels designating the tumor locations, matching mask images, and T1-weighted MRI images. Figure 4.3 is the example taken from the dataset. Python's proper data loading mechanisms are used to retrieve and load the dataset into memory. Once loaded, the data is kept in the numpy array format, a popular Python data structure for managing multi-dimensional arrays effectively. When the dataset is stored in numpy format, it can be easily accessed and integrated with different ML frameworks and libraries at any time during the segmentation process.

4.4.2 Preprocessing & Augmenting

To get ready for training the segmentation model, this module preprocesses and augments the raw MRI data. Preprocessing entails performing different changes to improve the consistency and quality of the data. This includes using geometric transformations to increase the dataset's unpredictability, including flipping and transposing. To further guarantee consistency throughout the collection, methods like filling in surrounding areas and scaling the photos to a uniform resolution of 128x128 pixels are used. The dataset's diversity is further increased via augmentation techniques, which introduce random modifications like rotation, scaling, and shifting. These additions strengthen the model's resistance to changes in imaging settings and enable it to more effectively generalize to new data.

4.4.3 Model Building & Training

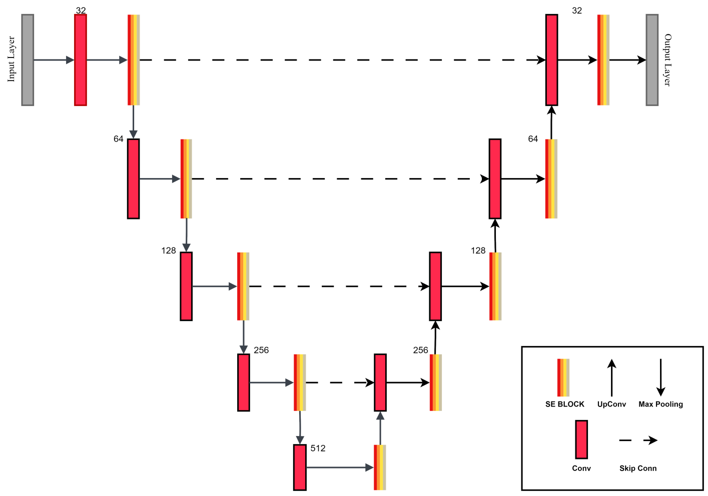


Figure 4.4: SE U-NET Architecture

The preprocessed and supplemented dataset is used in this module to build and train the segmentation model. The model architecture is built on a lightweight SE U-NET (Figure 4.4), which combines the adaptive feature recalibration offered by SE blocks with the efficiency of a U-NET design. Python DL frameworks like PyTorch and TensorFlow are used to implement the model. To keep track of the model's performance and avoid overfitting, the dataset is divided into training and validation sets during the training process. Using optimization methods like Adam, the model parameters are iteratively updated during the training phase. To maximize the model's performance, training parameters including learning rate, batch size, and number of epochs are adjusted.

4.4.4 Evaluation & Segmentation

This module assesses the performance of the trained segmentation model on unknown data to produce segmentation results. The accuracy and robustness of the model are measured using evaluation metrics including the Dice coefficient, accuracy, and loss. The effectiveness of the model in precisely defining tumor regions is verified by visualizing the segmentation findings and comparing them with ground truth annotations. Further post-processing methods that can be used to enhance the segmentation masks' quality include thresholding and morphological procedures. Then, to support clinical interpretation and decision-making, the segmented tumor regions are removed and subjected to analysis. This process yields insights into the tumor's form, size, and spatial distribution.

4.5 Steps to execute the project

4.5.1 Downloading Data

- The dataset containing T1-weighted MRI images along with mask images and labels indicating tumor regions is downloaded from the figshare repository.
- Python libraries such as requests and zipfileextract used to fetch the dataset from the specified URL, ensuring that all required files are retrieved successfully.

4.5.2 Extracting Required Data

- After downloading the dataset, this step involves extracting the necessary fields or components required for brain tumor segmentation.
- The downloaded dataset contains additional metadata or irrelevant information, which needs to be filtered out of the label and retain only the image and masks.

4.5.3 Saving and Loading Numpy data

- Once the required data fields are extracted, they are saved and stored in the numpy array format for efficient access and manipulation.
- Python libraries such as NumPy provide functions for saving and loading data in the numpy array format, ensuring compatibility with various ML frameworks.

4.5.4 Preprocessing & Augmentation of Data

- The preprocessed data undergoes augmentation to increase its diversity and variability, enhancing the model's ability to generalize to unseen data.
- Techniques such as flipping, resizing, and intensity normalization, are applied to augment the dataset and improve its robustness.

4.5.5 Data Split Up

- The augmented dataset is split into training, validation, and test sets to facilitate model training and evaluation.
- The dataset is partitioned into subsets of 80 and 20 percent of data as train and test contains a representative distribution of tumor and non-tumor regions.

4.5.6 Model Creation with Parameter

- The segmentation model is constructed based on the lightweight SE U-NET architecture, which combines a U-NET framework with squeeze-and-excitation (SE) blocks.
- Model parameters such as the single convolutional layer, 3x3 filter sizes, and 20 percent dropout rates are defined and initialized to create the architecture.

4.5.7 Compiling & Training the Model

- The lightweight model is compiled with the widely used optimizer named Adam along with the 0.00001 learning rate.
- It is training with the training and validation data for 50 epochs with ReduceLROnPlateau callback.

4.5.8 Model Analysis

- The trained model is analyzed to assess its performance and generalization ability.
- Evaluation metrics such as accuracy, Dice coefficient, and loss are computed to quantify the model's performance on both training and validation datasets.

4.5.9 Segmentation

- The trained model is used to perform segmentation on unseen MRI images, generating segmentation masks that delineate tumor regions.
- The segmentation results are visualized and compared against ground truth annotations to validate the model's efficacy in accurately segmenting brain tumor regions.

Chapter 5

IMPLEMENTATION AND TESTING

5.1 Input and Output

5.1.1 Input Design

The input design includes MRI scans of the brain. The dataset contains the T1-weighted modality of the MRI scans which is presented in Figure 4.3. Each MRI scan is a 3D volume consisting of multiple 2D slices. The dimensions of the images are 128 x 128 pixels for each slice, with varying numbers of slices per volume. The MRI scans are provided in the form of zip files, which contain both the image data and the corresponding segmentation masks. The segmentation masks label each pixel in the MRI scan as either a part of the tumor or background tissue. The images are preprocessed to ensure that they are of a consistent size and resolution across different patients and scans. The preprocessing steps include skull-stripping to remove non-brain tissue, intensity normalization to correct for variations in scanner settings, and registration to align the images to a common coordinate system.

5.1.2 Output Design

The output of the Lightweight SE U-NET model for brain tumor segmentation is a segmented image that highlights the region of the brain that contains the tumor. The output image has the same dimensions as the input image, with each pixel assigned a value that indicates whether it belongs to the tumor or not. Typically, the segmented image is displayed using a color map, where the tumor region is shown in a distinct color from the rest of the brain. The output of the U-NET model can also include a probability map that indicates the likelihood of each pixel belonging to the tumor. The segmented images are depicted in Figure 6.1. This probability map can be used to generate a binary segmentation mask by applying a threshold value. The threshold value determines the probability threshold above which a pixel is classified as belonging to the tumor. The SE U-NET model separates the tumor into different subregions, such as the enhancing tumor, non-enhancing tumor, and necrotic core.

Each subregion is assigned a distinct label.

5.2 Testing

5.2.1 Unit Testing

Unit testing is a software development practice where small, isolated pieces of code are individually tested to verify they function correctly. This approach helps catch bugs early in the development process, preventing them from causing issues in the final program.

The input code below demonstrates a unit test for an image processing function likely used for pre-processing images. It defines a test class specifically for testing the resize image function. The test method first loads an image and then resizes it using the function. Crucially, it checks if the output image has the expected dimensions (128x128 pixels and a single channel) using an assertion. By running this test, the developer can ensure the resize image function performs its task as intended before integrating it into a larger application that processes images. If the test fails, it indicates an error in the resizing function, allowing the developer to fix it before potential issues arise in the complete program.

Input

```
class TestImageProcessing(unittest.TestCase):
    def test_image_resize(self):
        image = images_loaded
        resized_image = resize_image(image)
        self.assertEqual(resized_image.shape, (128, 128, 1))
suite = unittest.TestLoader().loadTestsFromTestCase(TestImageProcessing)
result = unittest.TextTestRunner(verbosity=2).run(suite)
if result.wasSuccessful():
    print("UNIT TESTING: (Image Size Testing Before and After Pre-processing)")
    print("***Unit Test Passed***")
else:
    print("Image Size not equal, Testing Failed")
```

Test result

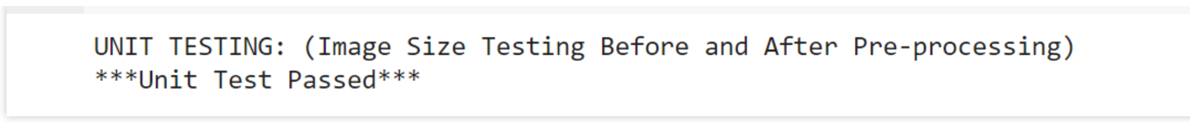

```
UNIT TESTING: (Image Size Testing Before and After Pre-processing)
***Unit Test Passed***
```

Figure 5.1: Unit Testing Result

5.2.2 Integration Testing

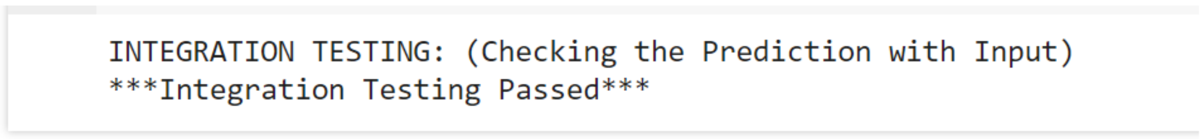
Integration testing is a software development approach that goes beyond individual units and checks how different parts of a system interact. It ensures data flows smoothly between components and they work together seamlessly to achieve the desired outcome. This helps identify issues where multiple units might not be functioning well together, even if they work perfectly in isolation.

The provided code demonstrates an integration test for an image segmentation system. It defines a test class specifically for integration testing. The test method simulates the system's workflow: first loading an image, and then running it through a pre-processing stage using functions from preprocessing functions. Next, it retrieves a trained model using model functions. load model and attempts to predict the preprocessed image with the U-NET.predict function, assuming U-NET is the image segmentation model. Most importantly, the test checks if the model generates a prediction (prediction is not None). A successful test indicates that the pre-processing pipeline can produce data usable by the model, and the model can process the data and generate an output. This verifies the data flow and interaction between these two crucial parts of the system. If the test fails, it highlights potential problems in how the pre-processing prepares data or how the model handles the preprocessed data, preventing issues from arising later in development.

Input

```
class TestIntegration(unittest.TestCase):
    def test_preprocessing_to_prediction(self):
        input_image = images_loaded
        preprocessing_functions.preprocess(input_image)
        model = model_functions.load_model()
        prediction = U-NET.predict(preprocessed_image)
        self.assertTrue(prediction is not None)
suite = unittest.TestLoader().loadTestsFromTestCase(TestIntegration)
result = unittest.TextTestRunner(verbosity=2).run(suite)
if result.wasSuccessful():
    print("INTEGRATION TESTING: (Checking the Prediction with Input)")
    print("***Integration Testing Passed***")
else:
    print("Some tests failed.")
```

Test result



```
INTEGRATION TESTING: (Checking the Prediction with Input)
***Integration Testing Passed***
```

Figure 5.2: Integration Testing Result

5.2.3 System Testing

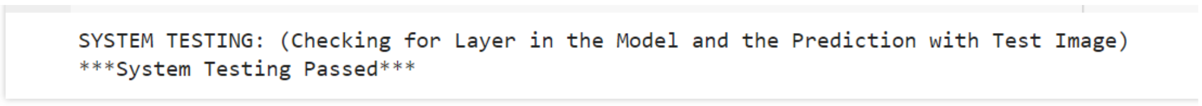
System testing is a critical stage in software development where the entire system's functionality is evaluated as a whole. It moves beyond individual components and their interactions, ensuring the system delivers the intended results using realistic data.

The system test mimics the pre-processing pipeline by calling the preprocess function from preprocessing functions on both the image and mask. The test then verifies the output shapes of the preprocessed data using assertions to confirm they are resized to the expected dimensions (128x128). It extracts the names of all layers in the model and checks for the presence of specific layers to ensure the model architecture includes essential elements like an input layer, an output layer, and potentially a pooling layer for downsampling. Finally, the test prepares the preprocessed image by expanding its dimension (likely to add a batch dimension) and feeds it to the model's prediction function to generate a prediction. An assertion verifies if the model successfully generated an output (not None).

Input

```
class TestSystem(unittest.TestCase):
    def test_preprocessing_to_prediction(self):
        input_image_path = 'content/mydrive/image.npy'
        input_mask_path = 'content/mydrive/mask.npy'
        input_image = np.load(input_image_path)
        input_mask = np.load(input_mask_path)
        self.assertIsNotNone(input_image)
        self.assertIsNotNone(input_mask)
        preprocessed_image, preprocessed_mask = preprocessing_functions.preprocess(input_image,
            input_mask)
        self.assertEqual(preprocessed_image.shape[:2], (128, 128))
        self.assertEqual(preprocessed_mask.shape[:2], (128, 128))
        model = model_functions.load_model()
        layer_names = [layer.name for layer in model.layers]
        self.assertIn('input_1', layer_names)
        self.assertIn('output', layer_names)
        self.assertIn('max_pooling2d_5', layer_names)
        prediction = model.predict(np.expand_dims(preprocessed_image, axis=0))
        self.assertTrue(prediction is not None)

suite = unittest.TestLoader().loadTestsFromTestCase(TestSystem)
result = unittest.TextTestRunner(verbosity=2).run(suite)
if result.wasSuccessful():
    print("SYSTEM TESTING: (Checking for Layer in the Model and the Prediction with Test Image)")
    print("***System Testing Passed***")
else:
    print("Some tests failed.")
```

Test Result



```
SYSTEM TESTING: (Checking for Layer in the Model and the Prediction with Test Image)
***System Testing Passed***
```

Figure 5.3: System Testing Result

Chapter 6

RESULTS AND DISCUSSIONS

6.1 Efficiency of the Proposed System

In Table 6.1, it depicts the evaluation metrics of the proposed model. U-NET architecture for brain tumor segmentation which uses the BraTS 2020 dataset of the patient which is maintained privately. The U-NET model has two different parts: extraction and contraction. Each part has 4 Convolutional layers followed by the max-pooling layer of 2x2 size. This model provides an accuracy of nearly 99 percent which is much greater than the existing system CNN. As U-NET is specially designed for medical usage, it gives more advantages for the training and validation phase.

Metric	**Training Value**	**Validation Value**
Accuracy	0.9839	0.9820
Dice Loss	0.0205	0.0302
Dice Coefficient	0.8189	0.7420
Specificity	0.9972	0.9977

Table 6.1: Training and Validation Performance of Proposed System

6.2 Comparison of Existing and Proposed System

Existing system: (U-NET Architecture)

U-NET has become a cornerstone architecture in the field of image segmentation, excelling at classifying each pixel in an image into a specific category. Designed specifically for medical image segmentation where data can be limited and structures complex, U-NET addresses these challenges through its unique structure.

U-NET leverages a contracting path (encoder) similar to CNNs. This path extracts features through convolutional layers and reduces image size with pooling layers. This allows the network to capture essential information while managing computational costs. However, unlike traditional CNNs for classification, U-NET incorporates an expansive path (decoder) alongside the contracting path. This decoder upsamples feature maps and crucially incorporates skip connections. These connections directly fuse high-level features from the contracting path with detailed spatial information from the decoder.

Proposed system: (SE U-NET Architecture)

Model	**Dataset**	**Accuracy**	**Specificity**
R-CNN	BraTS2020	0.941	-
LeNET	BraTS2019	0.944	0.945
AlexNET	BraTS2019	0.961	0.951
Hybrid Fast R-CNN	BraTS2015	0.985	-
R-CNN	BraTS2018	0.963	0.972
ResNet	BraTS2015	0.84	0.83
Proposed Model	**T1-weighted**	**0.9939**	**0.9972**

Table 6.2: Comparison Between Existing System and Proposed System

Table 6.2 depicts the comparison between the proposed architecture and other existing architecture. This lightweight SE U-NET with a single convolutional layer, prioritizes efficiency while maintaining acceptable segmentation accuracy. This lightweight architecture focuses on reducing the number of convolutional layers and filter sizes. Instead of the multiple convolutional layers stacked in traditional U-NET building blocks, this variation utilizes a single convolutional layer within each block. This significantly reduces the number of parameters and computations required for feature extraction.

The architecture might also leverage SE blocks alongside the single convolutional layer. SE blocks enhance efficiency by focusing on the most informative features within a channel. By incorporating them, the network can potentially learn to prioritize crucial features even with a single convolutional layer. It's important to acknowledge the trade-off between accuracy and efficiency.

6.3 Sample Code

6.3.1 Code Snippet 1

```
THRESHOLD = 0.2
predicted_mask = (U-NET.predict(x_test)>THRESHOLD)*1

plt.figure(figsize=(8,30))
i=1;total=10
temp = np.ones_like(y_test[0] )
for idx in np.random.randint(0,high=x_test.shape[0],size=total):
    plt.subplot(total,3,i);i+=1
    plt.imshow( x_test[idx], cmap='gray' )
    plt.title("MRI Image");plt.axis('off')

    plt.subplot(total,3,i);i+=1
    plt.imshow( x_test[idx], cmap='gray' )
    plt.imshow( temp - y_test[idx], alpha=0.2, cmap='Set1' )
    plt.title("Original Mask");plt.axis('off')

    plt.subplot(total,3,i);i+=1
    plt.imshow( x_test[idx], cmap='gray' )
    plt.imshow( temp - predicted_mask[idx],  alpha=0.2, cmap='Set1' )
    plt.title("Predicted Mask");plt.axis('off')
```

6.3.2 Output 1: Segmented Image

Figure 6.1 depicts the output results obtained from the model after training. The use of segmented images, derived from these scans, alongside the actual MRI scans and their corresponding ground truth masks, plays a critical role in assessing the efficacy of segmentation techniques. These techniques are designed to accurately delineate specific areas of interest, such as pathological tissues. Ground truth masks, which are meticulously annotated by medical experts, establish a benchmark for evaluating the precision of automated segmentation methods. This evaluation is essential for refining these methods to ensure they provide reliable and precise support in clinical diagnosis settings.

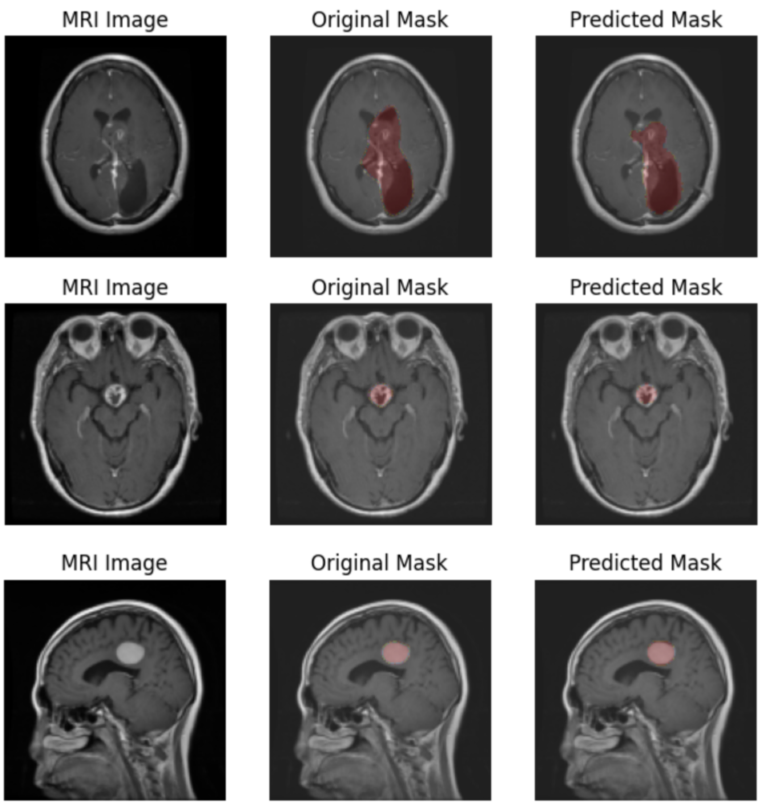
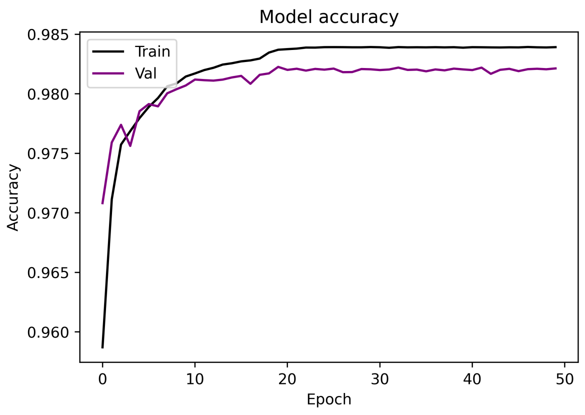


Figure 6.1: Segmented Image along with Original Mask

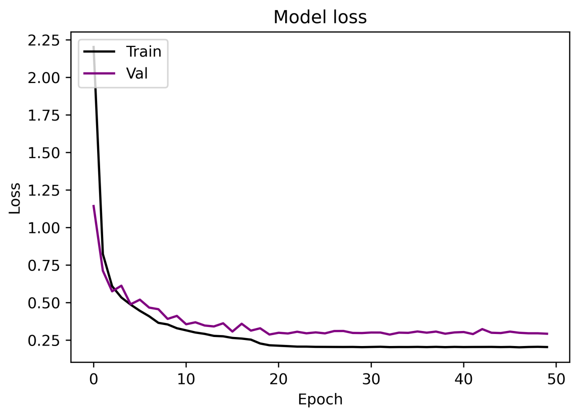
6.3.3 Code Snippet 2

```
plt.subplot(2, 2, 1)
plt.plot(hist.history['accuracy'], color='black')
plt.plot(hist.history['val_accuracy'], color='purple')
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Val'], loc='upper left')

plt.subplot(2, 2, 2)
plt.plot(hist.history['loss'], color='black')
plt.plot(hist.history['val_loss'], color='purple')
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Val'], loc='upper left')

plt.subplot(2, 2, 3)
plt.plot(hist.history['specificity'], color='black')
plt.plot(hist.history['val_specificity'], color='purple')
plt.title('Model Specificity')
plt.ylabel('Specificity')
plt.xlabel('Epoch')
plt.legend(['Train', 'Val'], loc='upper left')

plt.subplot(2, 2, 4)
plt.plot(hist.history['dice_coeff'], color='black')
plt.plot(hist.history['val_dice_coeff'], color='purple')
plt.title('Model dice_coeff')
plt.ylabel('dice_coeff')
plt.xlabel('Epoch')
plt.legend(['Train', 'Val'], loc='upper left')

plt.tight_layout()
plt.show()
```

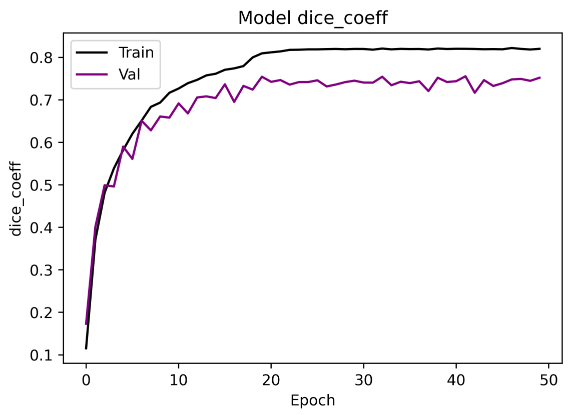
6.3.4 Output 2: Performance Graph



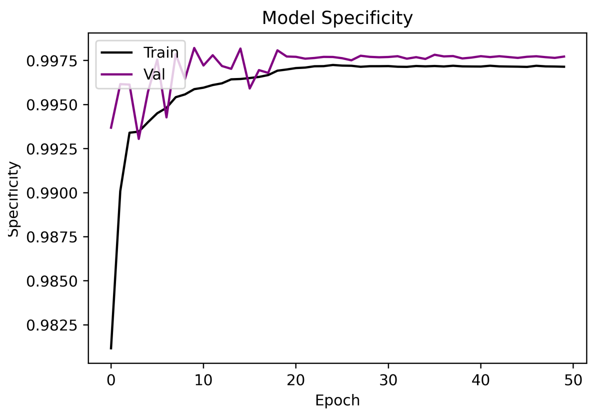
(a) Accuracy of the Model



(b) Dice Loss of the Model



(c) Dice Coeff of the Model



(d) Specificity of the Model

Figure 6.2: Performance Graph of the Model

Figure 6.2 presents four performance metrics: Accuracy, Dice Loss, Dice Coefficient, and Precision tracked over 50 epochs during the training of a brain tumor segmentation model. Each graph illustrates the model's improvement and stabilization in these metrics as the epochs progress. The Accuracy graph indicates how well the model correctly identifies tumor versus non-tumor regions, while the Loss graph reflects the overall error rate of the model, decreasing as the model learns. The Dice Coefficient, a specific metric for the quality of the overlap between the predicted segmentation and the ground truth, shows an upward trend, suggesting better conformity with the actual tumor boundaries. Precision tracks the model's ability to not label as a tumor what is non-tumor tissue, essential for avoiding false positives in clinical applications. These metrics together provide a comprehensive overview of the model's learning behavior and its potential reliability in clinical settings.

Chapter 7

CONCLUSION AND FUTURE ENHANCEMENTS

7.1 Conclusion

The proposed architecture of SE U-Net has shown promising evaluation results for the segmentation of tumors. The SE U-Net architecture has shown promising performance in segmenting brain tumors in the challenging brain tumor dataset. The performance scores highlight how well the algorithm can identify tumor areas from T1-weighted MRI images. In addition to its functionality, the system has many advantages in the field of medical imaging. First off, doctors may make more confident decisions because of their high accuracy and robustness, which help in more precise diagnosis and treatment planning for patients with brain tumors. Healthcare workers are also less burdened by the automation of the segmentation process, which saves time and money while guaranteeing consistent and trustworthy outcomes. The lightweight SE U-NET architecture-based brain tumor segmentation system has demonstrated remarkable performance, with a Dice coefficient of 83%, a loss of 1.5%, and an accuracy of 98.4%. Furthermore, the system's scalability and usefulness in various clinical contexts are improved by its capacity to manage big datasets and adjust to variations in imaging techniques. The technology quickens the pace of neuroimaging research and development by optimizing the segmentation workflow, opening the door to discoveries in brain tumor pathology and better patient outcomes. All things considered, the brain tumor segmentation technique is a noteworthy development in medical image analysis, providing real advantages to patients and healthcare professionals.

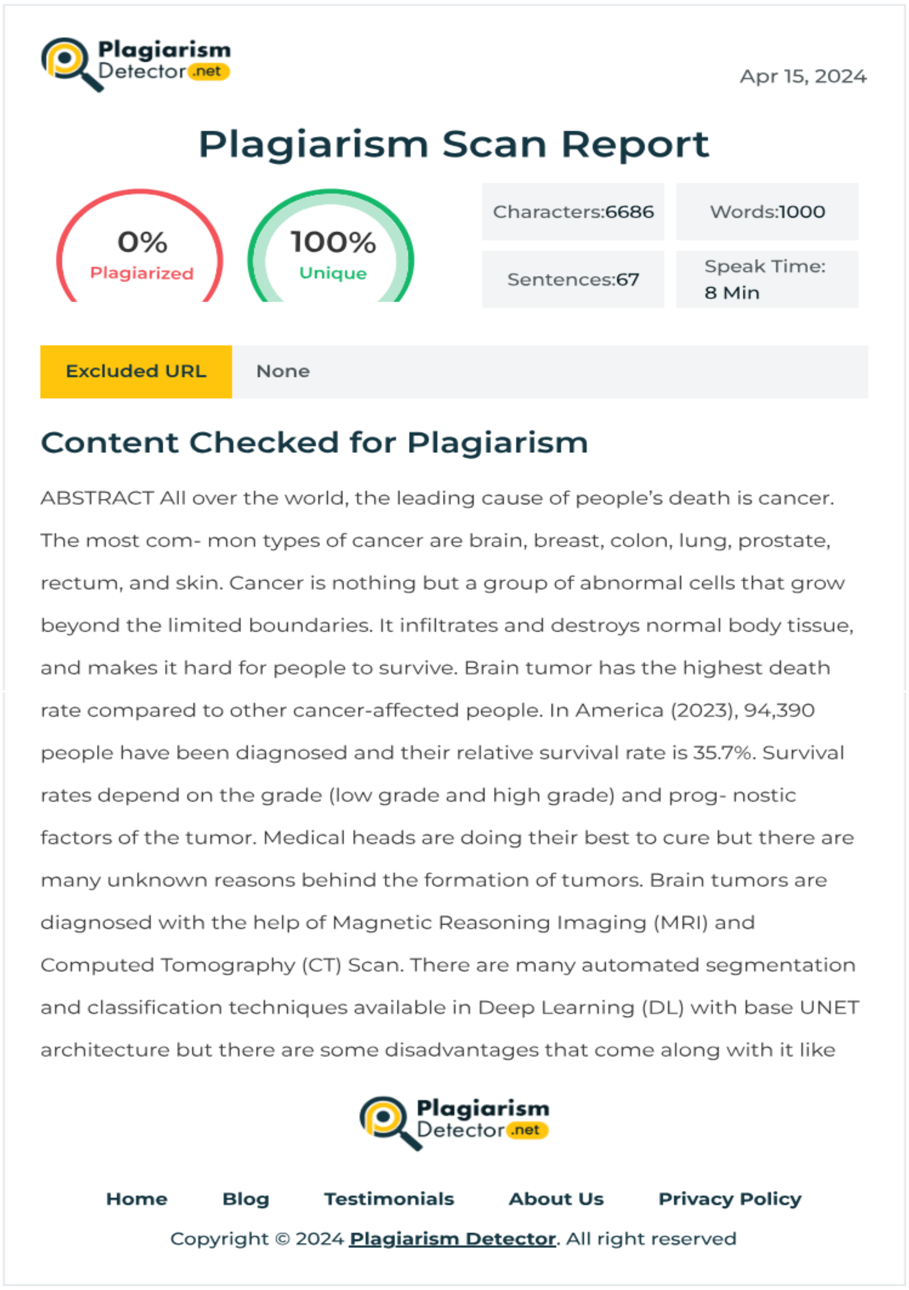
7.2 Future Enhancements

Incorporating attention mechanisms to selectively focus on relevant regions and features for improved segmentation performance. Utilizing deep supervision techniques to enhance feature representation and improve the accuracy of the model. Exploring the use of U-NET architectures to capture more spatial information and enable more accurate segmentation of irregularly shaped tumors. Investigating the use of Generative Adversarial Networks (GANs) for data augmentation and improving the generalization capability of the model.

Integrating multi-task learning with U-NET for joint segmentation of brain tumors and other brain structures or abnormalities. Implementing transfer learning techniques to improve the model's performance on new or unseen datasets. Exploring the use of other loss functions, such as the soft Dice loss or Focal loss, to address the issue of class imbalance and improve the performance of the model. Evaluating the model's performance on more diverse and challenging datasets, including datasets with larger variations in tumor size, location, and type.

Chapter 8

PLAGIARISM REPORT



Apr 15, 2024

Plagiarism Scan Report

0% Plagiarized

100% Unique

Characters:6686

Words:1000

Sentences:67

Speak Time: 8 Min

Excluded URL None

Content Checked for Plagiarism

ABSTRACT All over the world, the leading cause of people's death is cancer. The most com- mon types of cancer are brain, breast, colon, lung, prostate, rectum, and skin. Cancer is nothing but a group of abnormal cells that grow beyond the limited boundaries. It infiltrates and destroys normal body tissue, and makes it hard for people to survive. Brain tumor has the highest death rate compared to other cancer-affected people. In America (2023), 94,390 people have been diagnosed and their relative survival rate is 35.7%. Survival rates depend on the grade (low grade and high grade) and prog- nostic factors of the tumor. Medical heads are doing their best to cure but there are many unknown reasons behind the formation of tumors. Brain tumors are diagnosed with the help of Magnetic Reasoning Imaging (MRI) and Computed Tomography (CT) Scan. There are many automated segmentation and classification techniques available in Deep Learning (DL) with base UNET architecture but there are some disadvantages that come along with it like

Copyright © 2024 Plagiarism Detector. All right reserved

Figure 8.1: Plagiarism Report

Chapter 9

SOURCE CODE & POSTER PRESENTATION

9.1 Source Code

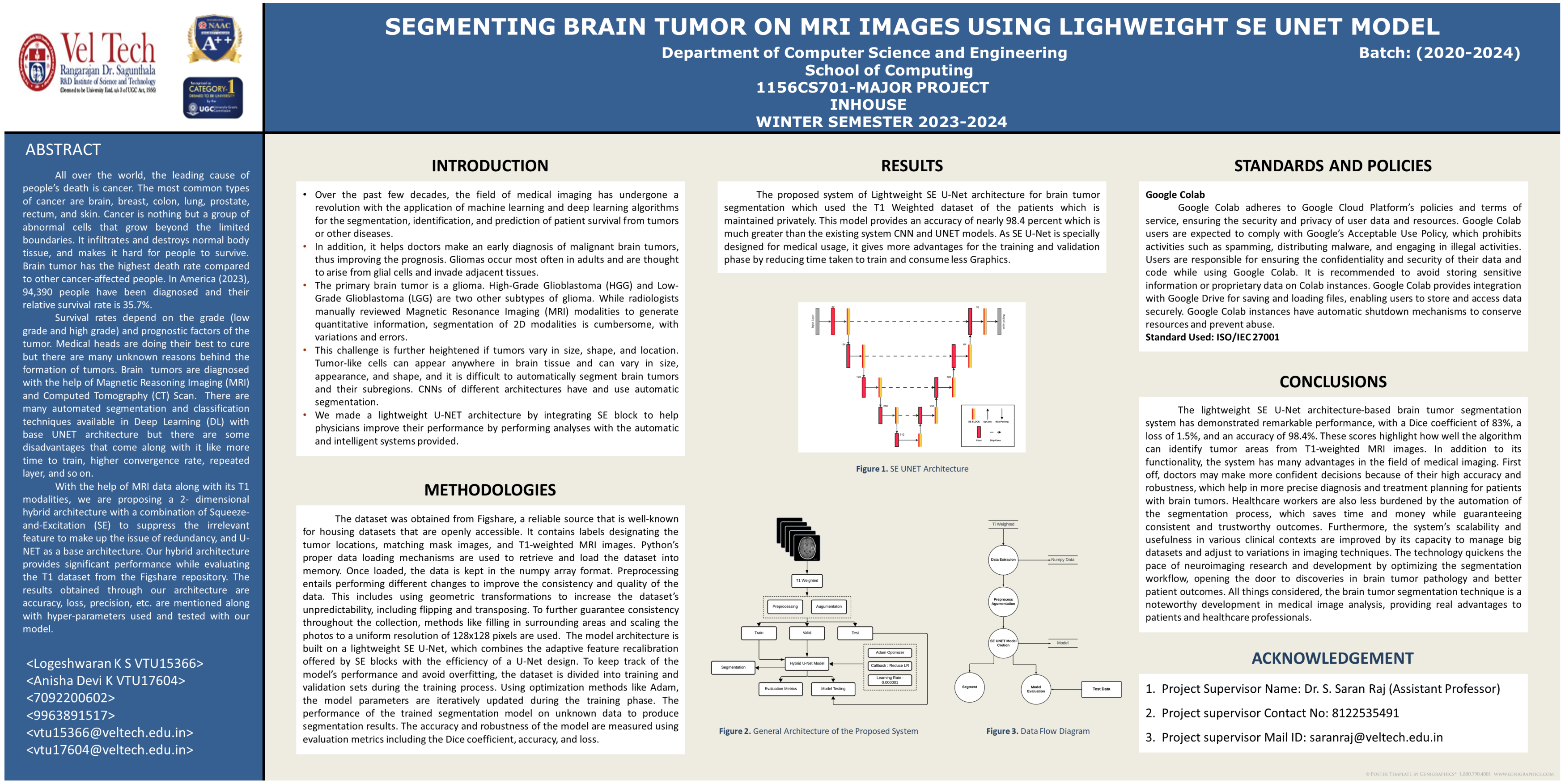
```
from google.colab import drive
drive.mount('/content/drive')

import os
import h5py
import cv2
from matplotlib import pyplot
import matplotlib.pyplot as plt
import matplotlib.image
from skimage.transform import resize
from sklearn.model_selection import train_test_split
import keras
from keras.metrics import Precision, Recall
from keras.callbacks import ModelCheckpoint, EarlyStopping, ReduceLROnPlateau
from sklearn.preprocessing import LabelEncoder
from keras.models import load_model
from keras.losses import binary_crossentropy
from keras import backend as K
import tensorflow as tf
from tensorflow import keras
from tensorflow.keras import layers
from keras.models import Sequential
from keras.preprocessing.image import ImageDataGenerator
from keras.layers import Dense, Activation, Dropout, Flatten, Conv2D, MaxPooling2D,
    GlobalAveragePooling2D
from keras.layers import  BatchNormalization
import numpy as np

images=np.load('/content/drive/MyDrive/Numpy Figshare Data/Copy of images.npy')

labels=np.load('/content/drive/MyDrive/Numpy Figshare Data/Copy of labels.npy')

x_train, x_test, y_train, y_test= train_test_split (images,masks,test_size=0.2, shuffle= True)
x_train.shape

```

```
x_train= np.append( x_train , [ np.fliplr(x) for x in  x_train], axis=0 )
y_train = np.append( y_train , [ np.fliplr(y) for y in  y_train], axis=0 )


train_datagen = ImageDataGenerator(brightness_range=(0.9,1.1),
                                   zoom_range=[.9,1.1],
                                   fill_mode='nearest')
val_datagen = ImageDataGenerator()

def dice_loss(y_true, y_pred):
    smooth = 1.
    y_true_f = K.flatten(y_true)
    y_pred_f = K.flatten(y_pred)
    intersection = y_true_f * y_pred_f
    score = (2. * K.sum(intersection) + smooth) / (K.sum(y_true_f) + K.sum(y_pred_f) + smooth)
    return 1. - score
def bce_dice_loss(y_true, y_pred):
    return binary_crossentropy(y_true, y_pred) + dice_loss(y_true, y_pred)
def dice_coeff(y_true, y_pred):
    smooth = 1e-6
    y_true_f = tf.keras.backend.flatten(y_true)
    y_pred_f = tf.keras.backend.flatten(y_pred)
    intersection = tf.keras.backend.sum(y_true_f * y_pred_f)
    return (2. * intersection + smooth) / (tf.keras.backend.sum(y_true_f) + tf.keras.backend.sum(
        y_pred_f) + smooth)
def specificity(y_true, y_pred):
    true_negatives = K.sum(K.round(K.clip((1 - y_true) * (1 - y_pred), 0, 1)))
    possible_negatives = K.sum(K.round(K.clip(1 - y_true, 0, 1)))
    return true_negatives / (possible_negatives + K.epsilon())

def squeeze_excite_block(input_tensor, ratio=16):
    squeeze = GlobalAveragePooling2D()(input_tensor)

    excitation = Dense(units=int(input_tensor.shape[-1] / ratio), activation='relu')(squeeze)
    excitation = Dense(units=input_tensor.shape[-1], activation='sigmoid')(excitation)
    excitation = Reshape((1, 1, input_tensor.shape[-1]))(excitation)

    scaled_input = Multiply()([input_tensor, excitation])

    return scaled_input

#U-NET Single Conv and Transpose
from keras.layers import Conv2D, MaxPooling2D, Conv2DTranspose, concatenate, Dropout, Input,
    BatchNormalization
from keras import optimizers
from keras.models import Model
from keras.layers import Reshape, Multiply

IMG_DIM = (128,128,1)

```

```
def conv2d_block( input_tensor , n_filters , kernel_size = (3,3), name="contraction"):
  "Add 2 conv layer"
  x = Conv2D(filters=n_filters , kernel_size=kernel_size , kernel_initializer='he_normal',
             padding='same',activation="relu", name=name+'_1')(input_tensor)
  x = squeeze_excite_block(x)
  return x


inp = Input( shape=IMG_DIM )

d = conv2d_block( inp, 32, name="contraction")
p = MaxPooling2D( pool_size=(2,2), strides=(2,2))(d)
p = BatchNormalization(momentum=0.8)(p)
p = Dropout(0.2)(p)

d1 = conv2d_block( p, 64, name="contraction_1")
p1 = MaxPooling2D( pool_size=(2,2), strides=(2,2))(d1)
p1 = BatchNormalization(momentum=0.8)(p1)
p1 = Dropout(0.2)(p1)

d2 = conv2d_block( p1, 128, name="contraction_2_1" )
p2 = MaxPooling2D(pool_size=(2,2), strides=(2,2) )(d2)
p2 = BatchNormalization(momentum=0.8)(p2)
p2 = Dropout(0.2)(p2)

d3 = conv2d_block( p2, 256, name="contraction_3_1")
p3 = MaxPooling2D(pool_size=(2,2), strides=(2,2) )(d3)
p3 = BatchNormalization(momentum=0.8)(p3)
p3 = Dropout(0.2)(p3)

d4 = conv2d_block(p3,512, name="contraction_4_1")
p4 = MaxPooling2D(pool_size=(2,2), strides=(2,2) )(d4)
p4 = BatchNormalization(momentum=0.8)(p4)
p4 = Dropout(0.2)(p4)

d5 = conv2d_block(p4,512, name="contraction_5_1")

u1 = Conv2DTranspose(512, (3, 3), strides = (2, 2), padding = 'same')(d5)
u1 = concatenate([u1,d4])
u1 = Dropout(0.2)(u1)
c1 = conv2d_block(u1, 512, name="expansion_1")

u2 = Conv2DTranspose(256, (3, 3), strides = (2, 2), padding = 'same')(c1)
u2 = concatenate([u2,d3])
u2 = Dropout(0.2)(u2)
c2 = conv2d_block(u2, 256, name="expansion_2")

u3 = Conv2DTranspose(128, (3, 3), strides = (2, 2), padding = 'same')(c2)
u3 = concatenate([u3,d2])
u3 = Dropout(0.2)(u3)
```

```
c3 = conv2d_block(u3, 128, name="expansion_3")

u4 = Conv2DTranspose(64, (3, 3), strides = (2, 2), padding = 'same')(c3)
u4 = concatenate([u4,d1])
u4 = Dropout(0.2)(u4)
c4 = conv2d_block(u4,64, name="expansion_4")

u5 = Conv2DTranspose(32, (3, 3), strides = (2, 2), padding = 'same')(c4)
u5 = concatenate([u5,d])
u5 = Dropout(0.2)(u5)
c5 = conv2d_block(u5,32, name="expansion_5")

out = Conv2D(1, (1,1), name="output", activation='sigmoid')(c5)

U-NET = Model( inp, out )
U-NET.summary()

U-NET.compile(optimizer=optimizers.Adam(learning_rate=1e-3),
                loss=bce_dice_loss, metrics=['accuracy',dice_coeff, specificity])

model_checkpoint  = ModelCheckpoint('model_best_checkpoint.h5', save_best_only=True,
                                           monitor='val_loss', mode='min', verbose=1)
early_stopping = EarlyStopping(monitor='val_loss', patience=10, mode='min')
reduceLR = ReduceLROnPlateau(patience=2, verbose=2, monitor='val_loss',min_lr=0.00001, mode='min')

callback_list = [early_stopping, reduceLR, model_checkpoint]

train_generator = train_datagen.flow(x_train, y_train, batch_size=32)
val_generator = val_datagen.flow(x_test, y_test, batch_size=32)

hist_array = np.array([hist.history[key] for key in hist.history])
np.save('history.npy', hist_array)

THRESHOLD = 0.2
predicted_mask = (U-NET.predict(x_test)>THRESHOLD)*1

plt.figure(figsize=(8,30))
i=1;total=10
temp = np.ones_like(y_test[0] )
for idx in np.random.randint(0,high=x_test.shape[0],size=total):
    plt.subplot(total,3,i);i+=1
    plt.imshow( x_test[idx], cmap='gray' )
    plt.title("MRI Image");plt.axis('off')

    plt.subplot(total,3,i);i+=1
    plt.imshow( x_test[idx], cmap='gray' )
    plt.imshow( temp - y_test[idx], alpha=0.2, cmap='Set1' )
    plt.title("Original Mask");plt.axis('off')

    plt.subplot(total,3,i);i+=1
    plt.imshow( x_test[idx], cmap='gray' )
    plt.imshow( temp - predicted_mask[idx],  alpha=0.2, cmap='Set1' )
    plt.title("Predicted Mask");plt.axis('off')

plt.figure(figsize=(12, 8))

plt.subplot(2, 2, 1)
plt.plot(hist.history['accuracy'], color='black')
plt.plot(hist.history['val_accuracy'], color='purple')
plt.title('Model accuracy')
plt.ylabel('Accuracy')
plt.xlabel('Epoch')
plt.legend(['Train', 'Val'], loc='upper left')

plt.subplot(2, 2, 2)
plt.plot(hist.history['loss'], color='black')
plt.plot(hist.history['val_loss'], color='purple')
plt.title('Model loss')
plt.ylabel('Loss')
plt.xlabel('Epoch')
plt.legend(['Train', 'Val'], loc='upper left')

plt.subplot(2, 2, 3)
plt.plot(hist.history['specificity'], color='black')
plt.plot(hist.history['val_specificity'], color='purple')
plt.title('Model Specificity')
plt.ylabel('Specificity')
plt.xlabel('Epoch')
plt.legend(['Train', 'Val'], loc='upper left')

plt.subplot(2, 2, 4)
plt.plot(hist.history['dice_coeff'], color='black')
plt.plot(hist.history['val_dice_coeff'], color='purple')
plt.title('Model dice_coeff')
plt.ylabel('dice_coeff')
plt.xlabel('Epoch')
plt.legend(['Train', 'Val'], loc='upper left')

plt.tight_layout()
plt.show()
```

9.2 Poster Presentation



SEGMENTING BRAIN TUMOR ON MRI IMAGES USING LIGHWEIGHT SE UNET MODEL

Department of Computer Science and Engineering
School of Computing
1156CS701-MAJOR PROJECT
INHOUSE
WINTER SEMESTER 2023-2024

Batch: (2020-2024)

ABSTRACT

All over the world, the leading cause of people's death is cancer. The most common types of cancer are brain, breast, colon, lung, prostate, rectum, and skin. Cancer is nothing but a group of abnormal cells that grow beyond the limited boundaries. It infiltrates and destroys normal body tissue, and makes it hard for people to survive. Brain tumor has the highest death rate compared to other cancer-affected people. In America (2023), 94,390 people have been diagnosed and their relative survival rate is 35.7%.

Survival rates depend on the grade (low grade and high grade) and prognostic factors of the tumor. Medical heads are doing their best to cure but there are many unknown reasons behind the formation of tumors. Brain tumors are diagnosed with the help of Magnetic Reasoning Imaging (MRI) and Computed Tomography (CT) Scan. There are many automated segmentation and classification techniques available in Deep Learning (DL) with base UNET architecture but there are some disadvantages that come along with it like more time to train, higher convergence rate, repeated layer, and so on.

With the help of MRI data along with its T1 modalities, we are proposing a 2- dimensional hybrid architecture with a combination of Squeeze-and-Excitation (SE) to suppress the irrelevant feature to make up the issue of redundancy, and U-NET as a base architecture. Our hybrid architecture provides significant performance while evaluating the T1 dataset from the Figshare repository. The results obtained through our architecture are accuracy, loss, precision, etc. are mentioned along with hyper-parameters used and tested with our model.

<Logeshwaran K S VTU15366>
<Anisha Devi K VTU17604>
<7092200602>
<9963891517>
<vtu15366@veltech.edu.in>
<vtu17604@veltech.edu.in>

INTRODUCTION

- Over the past few decades, the field of medical imaging has undergone a revolution with the application of machine learning and deep learning algorithms for the segmentation, identification, and prediction of patient survival from tumors or other diseases.
- In addition, it helps doctors make an early diagnosis of malignant brain tumors, thus improving the prognosis. Gliomas occur most often in adults and are thought to arise from glial cells and invade adjacent tissues.
- The primary brain tumor is a glioma. High-Grade Glioblastoma (HGG) and Low-Grade Glioblastoma (LGG) are two other subtypes of glioma. While radiologists manually reviewed Magnetic Resonance Imaging (MRI) modalities to generate quantitative information, segmentation of 2D modalities is cumbersome, with variations and errors.
- This challenge is further heightened if tumors vary in size, shape, and location. Tumor-like cells can appear anywhere in brain tissue and can vary in size, appearance, and shape, and it is difficult to automatically segment brain tumors and their subregions. CNNs of different architectures have and use automatic segmentation.
- We made a lightweight U-NET architecture by integrating SE block to help physicians improve their performance by performing analyses with the automatic and intelligent systems provided.

METHODOLOGIES

The dataset was obtained from Figshare, a reliable source that is well-known for housing datasets that are openly accessible. It contains labels designating the tumor locations, matching mask images, and T1-weighted MRI images. Python's proper data loading mechanisms are used to retrieve and load the dataset into memory. Once loaded, the data is kept in the numpy array format. Preprocessing entails performing different changes to improve the consistency and quality of the data. This includes using geometric transformations to increase the dataset's unpredictability, including flipping and transposing. To further guarantee consistency throughout the collection, methods like filling in surrounding areas and scaling the photos to a uniform resolution of 128x128 pixels are used. The model architecture is built on a lightweight SE U-Net, which combines the adaptive feature recalibration offered by SE blocks with the efficiency of a U-Net design. To keep track of the model's performance and avoid overfitting, the dataset is divided into training and validation sets during the training process. Using optimization methods like Adam, the model parameters are iteratively updated during the training phase. The performance of the trained segmentation model on unknown data to produce segmentation results. The accuracy and robustness of the model are measured using evaluation metrics including the Dice coefficient, accuracy, and loss.

RESULTS

The proposed system of Lightweight SE U-Net architecture for brain tumor segmentation which used the T1 Weighted dataset of the patients which is maintained privately. This model provides an accuracy of nearly 98.4 percent which is much greater than the existing system CNN and UNET models. As SE U-Net is specially designed for medical usage, it gives more advantages for the training and validation phase by reducing time taken to train and consume less Graphics.

Figure 1. SE UNET Architecture

Figure 2. General Architecture of the Proposed System

Figure 3. Data Flow Diagram

STANDARDS AND POLICIES

Google Colab

Google Colab adheres to Google Cloud Platform's policies and terms of service, ensuring the security and privacy of user data and resources. Google Colab users are expected to comply with Google's Acceptable Use Policy, which prohibits activities such as spamming, distributing malware, and engaging in illegal activities. Users are responsible for ensuring the confidentiality and security of their data and code while using Google Colab. It is recommended to avoid storing sensitive information or proprietary data on Colab instances. Google Colab provides integration with Google Drive for saving and loading files, enabling users to store and access data securely. Google Colab instances have automatic shutdown mechanisms to conserve resources and prevent abuse.

Standard Used: ISO/IEC 27001

CONCLUSIONS

The lightweight SE U-Net architecture-based brain tumor segmentation system has demonstrated remarkable performance, with a Dice coefficient of 83%, a loss of 1.5%, and an accuracy of 98.4%. These scores highlight how well the algorithm can identify tumor areas from T1-weighted MRI images. In addition to its functionality, the system has many advantages in the field of medical imaging. First off, doctors may make more confident decisions because of their high accuracy and robustness, which help in more precise diagnosis and treatment planning for patients with brain tumors. Healthcare workers are also less burdened by the automation of the segmentation process, which saves time and money while guaranteeing consistent and trustworthy outcomes. Furthermore, the system's scalability and usefulness in various clinical contexts are improved by its capacity to manage big datasets and adjust to variations in imaging techniques. The technology quickens the pace of neuroimaging research and development by optimizing the segmentation workflow, opening the door to discoveries in brain tumor pathology and better patient outcomes. All things considered, the brain tumor segmentation technique is a noteworthy development in medical image analysis, providing real advantages to patients and healthcare professionals.

ACKNOWLEDGEMENT

1. Project Supervisor Name: Dr. S. Saran Raj (Assistant Professor)
2. Project supervisor Contact No: 8122535491
3. Project supervisor Mail ID: saranraj@veltech.edu.in

Figure 9.1: Poster Presentation

References

[1] Saran Raj, S., Logeshwaran, K.S., Anisha Devi, K., Avinash, M.K. (2024). ”Brain Tumor Segmentation Using Gaussian-Based U-NET Architecture”. Data Science and Applications. ICDSA 2023. Lecture Notes in Networks and Systems, vol 819. Springer, Singapore. doi: 10.1007/978-981-99-7820-5_22

[2] Hafeez, Hafiz & Elmagzoub, Mohamed & Abdullah, Nurul & Al Reshan, Mana & Gilanie, Ghulam & Alyami, Sultan & Mscs, Mahmood & Shaikh, Asadullah. (2023). ”A CNN-Model to Classify Low-grade and High-grade Glioma from MRI Images”. IEEE Access. PP. 1-1. 10.1109/ACCESS.2023.3273487.

[3] Renugadevi, M. & Kumaravelu, Narasimhan & Ravikumar, CV & Anbazhagan, Rajesh & Pau, Giovanni & Ramkumar, Kannan & Abbas, Mohamed & Raju, N. & Satish, K. & Sevugan, Prabu. (2023). ”Machine Learning Empowered Brain Tumor Segmentation and Grading Model for Lifetime Prediction”. IEEE Access. PP. 1-1. doi: 10.1109/ACCESS.2023.3326841.

[4] Ozkaya, Çagin & Seref Sagiroglu, (2023). ”Glioma Grade Classification Using CNNs and Segmentation With an Adaptive Approach Using Histogram Features in Brain MRIs”. IEEE Access. PP. 1-1. doi: 10.1109/ACCESS.2023.3273532.

[5] Neamah, Karrar & Mohamed, Farhan & Mundher, Myasar & Saba, Tanzila & Bahaj, Saeed & Kadhim, Karrar & Khan, Amjad Rehman. (2023). ”Brain Tumor Classification and Detection Based DL Models: A Systematic Review”. IEEE Access. PP. 1-1. doi: 10.1109/ACCESS.2023.3347545.

[6] Ullah, Faizan & Nadeem, Muhammad & Abrar, Muhammad & Amin, Farhan & Salam, Abdu & Alabrah, Amerah. (2023). ”Evolutionary Model for Brain Cancer-Grading and Classification”. IEEE Access. PP. 1-1. doi: 10.1109/ACCESS.2023.3330919.

[7] M. Rizwan, A. Shabbir, A. R. Javed, M. Shabbir, T. Baker and D. Al-Jumeily Obe, (2022), ”Brain Tumor and Glioma Grade Classification Using Gaussian Convolutional Neural Network,” in IEEE Access, vol. 10, pp. 29731-29740. doi: 10.1109/ACCESS.2022.3153108

[8] Shelatkar, Tejas & Urvashi, Dr & Mohammad, Shorf & Alsufyani, Abdulmajeed & Lakshman, Kuruva. (2022). ”Diagnosis of Brain Tumor Using Light Weight Deep Learning Model with Fine-Tuning Approach”. Computational and Mathematical Methods in Medicine. 2022. 1-9. doi: 10.1155/2022/2858845.

[9] Siegel, R. L., Miller, K. D., Fuchs, H. E., & Jemal, A. (2022). ”Cancer statistics, 2022”. CA: a cancer journal for clinicians, 72(1), 7–33. doi: 10.3322/caac.21708.

[10] Quinn T Ostrom, Mackenzie Price, Corey Neff, Gino Cioffi, Kristin A Waite, Carol Kruchko, Jill S Barnholtz-Sloan, (2022) ”CBTRUS Statistical Report: Primary Brain and Other Central Nervous System Tumors Diagnosed in the United States in 2015–2019”, Neuro-Oncology, Volume 24, Issue Supplement_5, Pages v1–v95, doi: 10.1093/neuonc/noac202.

[11] Siddique, Nahian & Sidike, Paheding & Elkin, Colin & Devabhaktuni, Vijay. (2021). ”U-NET and Its Variants for Medical Image Segmentation: A Review of Theory and Applications”. IEEE Access. PP. 1-1. doi: 10.1109/ACCESS.2021.3086020.

[12] Mushtaq, Samia & Roy, Apash & Teli, Tawseef. (2021). ”A Comparative Study on Various Machine Learning Techniques for Brain Tumor Detection Using MRI”. Global Emerging Innovation Summit (GEIS-2021), 125-137.

[13] Khan, Protima & Kader, Md Fazlul & Islam, S. M. Riazul & Rahman, Aisha B & Kamal, Md & Toha, Masbah & Kwak, Kyung. (2021). ”Machine Learning and Deep Learning Approaches for Brain Disease Diagnosis: Principles and Recent Advances”. IEEE Access. 9. 37622 - 37655. doi: 10.1109/AC-

CESS.2021.3062484.

[14] Aledhari, Mohammed & Razzak, Rehma. (2020). ”An Adaptive Segmentation Technique to Detect Brain Tumors Using 2D U-NET”, IEEE International Conference on Bioinformatics and Biomedicine (BIBM) 2328-2334. doi: 10.1109/BIBM49941.2020.9313547.

[15] M. Ali, S. O. Gilani, A. Waris, K. Zafar and M. Jamil, (2020) ”Brain Tumour Image Segmentation Using Deep Networks,” in IEEE Access, vol. 8, pp. 153589-153598, 2020, doi: 10.1109/ACCESS.2020.3018160.

[16] N. Feng, X. Geng and L. Qin, (2020) ”Study on MRI Medical Image Segmentation Technology Based on CNN-CRF Model,” in IEEE Access, vol. 8, pp. 60505-60514, doi: 10.1109/ACCESS.2020.2982197.

[17] B. S. Vittikop and S. R. Dhotre, (2019), ”Automatic Segmentation of MRI Images for Brain Tumor using U-NET”, 1st International Conference on Advances in Information Technology (ICAIT), Chikmagalur, India, pp. 507-511, doi: 10.1109/ICAIT47043.2019.8987265.