HELWAN UNIVERSITY
Faculty of Computers and Artificial Intelligence
MedicalInformaticsProgram

VITA

A graduation project dissertation by:

Somaya Ahmed Sharkawy (20208138)
Eman Ahmed El-daoushy (20208057)
Mohamed Fouad Shabaan (20208204)
Noura Nabil Hosni (20208262)
Hoda Ayman Rashad (20208286)

Submitted in partial fulfilment of the requirements for the degree of Bachelor of Science in Computers & Artificial Intelligence, at Medical Informatics Program, the Faculty of Computers & Artificial Intelligence, Helwan University

Supervised by:

Marwa M. A. El Fattah

June 2023

جامعة حلوان
كلية الحاسبات والذكاء الاصطناعي
برنامج المعلوماتية الطبية

حياة

رسالة مشروع تخرج مقدمة من:

سميه احمد شرقاوي 20208138
ايمان احمد الداعوشى 20208057
محمد فؤاد شعبان 20208204
نورا نبيل حسنى 20208262
هدى ايمن رشاد 20208286

رسالة مقدمة ضمن متطلبات الحصول على درجة البكالوريوس في الحاسبات والذكاء الإصطناعي،
ببرنامج المعلوماتية الطبية كلية الحاسبات والذكاء الإصطناعي، جامعة حلوان

تحت إشراف:

(د/ مروة عبد الفتاح)

يونيو 2023

Acknowledgements:

We begin by expressing our utmost gratitude and thanks to Allah, the Most Merciful and the Most Gracious, for His divine blessings, guidance, and unwavering support throughout the entire duration of this project.

We would like to express our deepest gratitude to Dr. Marwa for her invaluable guidance and supervision throughout the duration of this project, her expertise, dedication, and constant support have been instrumental in the successful completion of this endeavor.

We are truly grateful for Dr. Marwa's mentorship and her willingness to share her knowledge and experience. Her insightful feedback and constructive criticism have significantly improved the quality of our work and shaped our understanding of the subject matter.

We would also like to extend our appreciation to the entire research team for their contributions and collaboration. Their valuable insights, discussions, and assistance have been instrumental in the progression of this project.

Furthermore, we would like to acknowledge the support and encouragement provided by our friends and family. Their unwavering belief in our abilities and their continuous motivation have been vital to keeping us focused and inspired.

Lastly, we would like to express our gratitude to all the resources, references, and institutions that have provided the necessary materials and information for this project. Their contributions have been instrumental in shaping the foundation of this work.

Content

Abstract

In IoT-based healthcare systems, the classification and segmentation of brain tumors (BT) is crucial for the diagnosis of brain cancer (BC). The most common use of artificial intelligence (AI) methods based on computer-aided diagnostic systems (CADS) is the precise identification of brain cancer. However, because artificial diagnostic tools are inaccurate, healthcare providers are not efficiently utilizing them in the detection of brain cancer. To address the problem of low accuracy in current artificial diagnosis systems, we proposed a reliable deep learning (DL) strategy for classifying and segmenting brain tumors in this research study. The suggested method uses brain magnetic resonance (MR) imaging data to classify and segment brain cancers using an upgraded model.

By using data augmentation, transfer learning techniques and segmentation models, the model classification and segmentation performance has increased. The outcomes demonstrated that the model outperformed the baseline models in terms of accuracy. We recommend the suggested approach for brain cancer diagnosis in IoT-healthcare systems based on highly predictive results.

Chapter one: Introduction

1.1 Overview

Brain tumors are caused by changes in the DNA inside cells. Gliomas account for more than 70% of all brain tumors, and of theses, glioblastoma is the most frequent and malignant histologic type (WHO grade IV). There are many reports that indicate lifestyle and environmental factors including some occupations, diet and environmental carcinogens can cause elevated glioma risk. Only about 5% of people with glioblastoma will be alive five years after diagnosis.

Our online medical system is designed to help users which are doctors, medical students or the patients in classifying MRI scans of patients into glioma, meningioma, pituitary tumor or no tumor by applying classification techniques using transfer learning model and Custom CNN architecture. This online medical system will save the effort and time for the doctor in classifying some tumors. It also helps doctors in diagnosing brain tumors by using segmentation as our online medical system views a 3D segmented image for the tumor. It can help the doctor in decision making with choosing the right or ideal treatment for the cases according to the classification and segmentation tools.

It can help medical students in their studies so they can upload the image of MRI scan to predict if the case is healthy or it is from the three tumors that we decided to develop the model to predict them.

The patients can use the online medical system to read the frequently asked questions (FAQs) about brain tumors generally including types of brain tumors, risk factors and information about awake brain surgeries. They can also read the stories of survived patients from the brain tumors. We designed a simple game for patients to play. This game is used as a measurement for their memory to check how much the brain is affected by the tumors or to detect how badly the case is. The patients can also read information about the brain tumors in the Info section or read about treatments.

1.2 Objectives of the project

Classification Objectives:
Develop a deep learning-based system that utilizes 2D MRI images to accurately classify meningioma, pituitary, and glioma tumors.
Achieve high classification accuracy to assist doctors and students in making informed medical decisions regarding tumor identification and diagnosis.

Segmentation Objectives:
Develop a deep learning-based system that utilizes MRI images to accurately segment glioma tumors.

Provide precise tumor boundaries through segmentation to aid doctors and students in treatment planning and monitoring of glioma patients.

Overall, the project aims to leverage deep learning techniques to achieve high accuracies in tumor classification and segmentation, with the primary goal of assisting medical professionals and students in making informed decisions related to brain tumor analysis and improving patient outcomes.

1.3 Purpose

1- The purpose of a brain tumor classification model is to accurately categorize brain tumors into specific types based on their characteristics and features. This can help doctors make more informed decisions about treatment options and prognosis for the patient.
Brain tumors can be classified based on a variety of factors, including their location, size, shape, and biological behavior. Some common types of brain tumors include gliomas, meningiomas, and metastatic brain tumors.

Classification models use machine learning algorithms to analyze medical imaging data, such as MRI or CT scans of the brain, and identify patterns and features that are indicative of a specific type of brain tumor. This can involve analyzing the shape, size, and texture of the tumor, as well as the location within the brain and the extent of surrounding tissue involvement.

Accurate classification of brain tumors is important because different types of tumors may respond differently to different treatments, and some types may be more aggressive than others. For example, certain types of gliomas may be treated with surgery, radiation, or chemotherapy, while other types may require more aggressive treatment or have a poorer prognosis.

2-The purpose of a brain tumor segmentation model is to accurately identify and locate tumors within medical imaging data, such as MRI or CT scans of the brain. By segmenting the tumor from the surrounding healthy brain tissue, doctors can better assess the size, location, and characteristics of the tumor, which is critical for making a diagnosis and planning treatment.

Accurate segmentation of brain tumors can be challenging, as tumors can vary in size, shape, and location, and may have irregular borders that blend into surrounding tissue. Segmentation models use machine learning algorithms to analyze medical images and identify patterns that correspond to tumor tissue, allowing for more precise and consistent tumor identification.

Once the tumor has been segmented, doctors can use the resulting 3D models to plan surgical interventions or radiation treatments that specifically target the tumor while minimizing damage to healthy brain tissue. The segmentation model can also be used to track changes in tumor size and shape over time.

1.4 Scope

1. **Introduction:** This chapter outlines the scope of the project, defining the boundaries and limitations within which the research and development activities will be conducted. It provides clarity on the specific aspects and areas that will be addressed in the project.

2. **Inclusion Criteria:** The scope of this project encompasses the following criteria:

2.1. Tumor Types
The project focuses on the classification and segmentation of three specific brain tumor types: meningioma, pituitary tumors, and gliomas (with an emphasis on gliomas for segmentation). Other brain tumor types and non-tumor structures will not be considered within the scope of this project.

2.2. Data
The project utilizes 2D MRI images for classification and 3D MRI for segmentation tasks. The data used for model development and evaluation will be obtained from publicly available datasets or with proper permissions and adherence to ethical considerations.

2.3. Classification
The project aims to achieve high accuracy in classifying the aforementioned brain tumor types using deep learning techniques.
The classification will be based on the analysis of 2D MRI images, considering the distinctive features and characteristics of each tumor type.

2.4. Segmentation
The project focuses on the accurate segmentation of glioma tumors using MRI images.
The segmentation process aims to delineate the tumor boundaries with precision to aid in treatment planning and monitoring.

3. Exclusion Criteria: The scope of this project excludes the following aspects:

3.1. Other Tumor Types
The project does not encompass the classification and segmentation of brain tumor types beyond meningioma, pituitary tumors, and gliomas.

4. Limitations: It is important to acknowledge the limitations of this project:

4.1. Data Availability

The scope of the project is contingent upon the availability and accessibility of suitable MRI datasets for training and evaluation purposes.

4.2. Generalizability

The performance and generalizability of the developed models may be influenced by variations in imaging protocols, image quality, and patient demographics.

4.3. Computational Resources

The scalability and efficiency of the developed models may be constrained by computational resources, such as processing power and memory limitations.

Chapter two: Working theory and literature review.

2.1 Working theory of our proposed work:

2.1.1 Introduction:

Artificial intelligence (AI) is the development of computer systems that can perform tasks that would typically require human intelligence. AI techniques can be used to analyze large amounts of medical data, such as medical imaging and genomics data, to aid in the diagnosis and treatment of diseases. In the medical field, AI has the potential to improve patient outcomes by aiding in the diagnosis, treatment, and management of diseases.[1]

Machine learning (ML) is a subset of AI that focuses on teaching machines how to learn from data without being explicitly programmed. ML techniques can be used to analyze medical data, such as medical images of brain tumors, to identify patterns and anomalies that are not easily visible to the human eye. In the detection of brain tumors, ML algorithms can be trained on medical imaging data to automatically identify and classify abnormal tissue.
Deep learning (DL) is a subset of ML that uses artificial neural networks to learn from large amounts of data. DL techniques can be used to improve the accuracy of ML algorithms by automatically extracting features from raw data, which would otherwise require manual feature engineering. In the medical field, DL techniques can be used to analyze complex medical data, such as genomics data, to identify disease risk factors and develop personalized treatment plans.
The relationship between AI, ML, and DL is complementary, as each technique builds on the other to achieve more advanced results. By leveraging the power of AI, ML, and DL, medical professionals can improve patient outcomes and provide better care for individuals with brain tumors.

2.1.2 Basic Operation of Neural Networks:

Neural networks are a subset of artificial intelligence [2] that are inspired by the structure and function of the human brain. At a basic level, neural networks consist of layers of interconnected nodes or neurons that perform mathematical operations on input data. The neurons in the input layer receive the input data, which is then processed through one or more hidden layers before reaching the output layer, which produces the final result. Each neuron in a neural network performs a weighted sum of its inputs, which is then passed through an activation function to produce the neuron's output. During the training process, the weights of the neurons are adjusted to minimize the difference between the predicted output and the actual output. By adjusting these weights, neural networks can learn to recognize patterns and make predictions based on input data, making them a powerful tool for a variety of applications, including image and speech recognition, natural language processing, and more.

2.1.3 Convolutional Neural Network:

Convolutional neural networks (CNNs) are a type of neural network [3] that is designed to process data with a grid-like structure, such as images. CNNs consist of multiple layers, each of which performs a specific function in the network:

1. Convolutional layers: In these layers, the network learns a set of filters that are applied to the input image to extract meaningful features. This allows the network to automatically learn important features without the need for manual feature engineering.
2. Pooling layers: These layers reduce the spatial size of the output from the convolutional layers, which helps to decrease the number of parameters in the network and prevent overfitting.
3. Fully connected layers: These layers process the output from the pooling layers to produce the final output, which can be used for classification, regression, or other tasks.

In addition to these layers, CNNs also use activation functions to introduce non-linearity into the network. Some common activation functions include:

1. ReLU (Rectified Linear Unit): This activation function sets all negative values in the input to 0, while leaving positive values unchanged. ReLU is a popular choice for CNNs because it is computationally efficient and has been shown to perform well in practice. (Which we used in this project)
2. Sigmoid: This activation function maps the input to a value between 0 and 1, which can be interpreted as a probability. Sigmoid is often used in binary classification tasks but can suffer from the vanishing gradient problem when used in deep networks.

It also uses optimization functions to adjust the weights and biases in the network during training. Some common optimization functions include:

1. Adam: This optimization function combines the benefits of momentum and adaptive learning rate methods to provide faster convergence and better generalization. (Which we used in this project)
2. Stochastic Gradient Descent (SGD): This optimization function updates the weights and biases of the network based on the gradient of the loss function with respect to the parameters. SGD is a popular choice for CNNs because it is simple and easy to implement.

CNNs also use dropout to prevent overfitting. Dropout is a regularization technique that randomly sets a fraction of the input values to 0 during training. This forces the network to learn more robust features and reduces the impact of individual nodes.

Some benefits of CNN layers, activation functions, optimization functions, and dropout include:

- The convolutional layers learn spatially relevant features, which allows the network to identify objects and patterns in an image more accurately than traditional machine learning algorithms.

- Pooling layers reduce the spatial size of the output, which decreases the number of parameters in the network and helps to prevent overfitting.
- Fully connected layers allow the network to process the output from the pooling layers and make predictions for the task at hand.
- Activation functions introduce non-linearity into the network, which allows the network to learn more complex relationships between the input and output.
- Optimization functions adjust the weights and biases in the network to improve its performance on the training data.
- Dropout prevents overfitting by encouraging the network to learn more robust features and reducing the impact of individual nodes.

CNNs also have some disadvantages:

1. they require a large training data.
2. They require an appropriate model.
3. time consuming
4. They are computationally expensive
5. They can be susceptible to overfitting if the network is too large, or the training data is not representative of the population.
6. While convolutional networks have already existed for a long time, their success was limited due to the size of the considered network.

Solution-Transfer Learning for inadequate data which will replace the last fully connected layer with pre-trained ConvNet with new fully connected layer.

2.1.4 Transfer Learning:

It is a machine learning technique that involves leveraging the knowledge and models learned from one task to improve performance on a different but related task. In transfer learning [4], a pre-trained model is used as the starting point for a new task, and then fine-tuned using a smaller amount of data specific to the new task. By using a pre-trained model, transfer learning can significantly reduce the amount of data required to train a new model, making it a powerful technique for applications where data is limited or expensive to obtain. Transfer learning has been successfully applied to a wide range of applications, including image and speech recognition, natural language processing, and more. In the medical field, transfer learning can be used to improve the accuracy and efficiency of models used for diagnosis and treatment planning by leveraging knowledge learned from large-scale datasets. By reducing the amount of data required to train a new model, it can help accelerate the development of new medical applications, making it a valuable technique for researchers and medical professionals alike.

2.1.5 Brain tumors:

Brain tumors [5] are abnormal growths of cells in the brain that can cause a wide range of symptoms and can be life-threatening. Some of the most frequent types of brain tumors include gliomas, meningiomas, and pituitary adenomas. Magnetic Resonance Imaging (MRI) is a medical imaging technique that uses strong magnetic fields and radio waves to produce detailed images of the brain. MRI can help doctors diagnose different types of brain tumors and determine their size, location, and the extent of their spread.
However, interpreting MRI images to accurately diagnose brain tumors can be a challenging task, even for experienced radiologists. Deep learning models, such as Convolutional Neural Networks (CNN), have shown great promise in classifying medical images, including MRI scans of the brain. By training a CNN on a large dataset of MRI images of different types of brain tumors, the model can learn to distinguish between different types of tumors with a high degree of accuracy. This can help doctors make faster and more accurate diagnoses, leading to earlier detection and more effective treatment. Developing deep learning models for classifying brain tumors from MRI scans also has the potential to improve patient outcomes by providing personalized treatment recommendations based on the specific type of tumor detected. Additionally, these models can assist radiologists in accurately identifying the location and extent of the tumor, which can aid in surgical planning and monitoring the effectiveness of treatment.

2.1.6 Brain tumor classification:

Brain tumor classification [6] is a vital task in medical imaging, aimed at accurately identifying and categorizing different types of brain tumors. It involves the analysis of various imaging modalities, such as MRI, to discern specific tumor characteristics and distinguish between different tumor types, such as meningioma, pituitary tumors, and gliomas. Deep learning techniques, such as Convolutional Neural Networks (CNNs), have shown promising results in automating and improving the accuracy of brain tumor classification. By providing accurate tumor identification, classification algorithms assist healthcare professionals in making informed treatment decisions and developing personalized patient care plans.

2.1.7 Brain tumor segmentation:

Brain tumor segmentation [7] is an important application of segmentation in medical image analysis. It involves identifying and segmenting regions of the brain that are affected by tumors, which can be used for diagnosis, treatment planning, and monitoring of the disease progression.

Deep learning models have shown promising results in brain tumor segmentation tasks, particularly in the context of magnetic resonance imaging (MRI) data. There are several popular deep learning architectures that have been used for brain tumor segmentation, including U-Net, 3D-CNNs, and attention-based models.

In addition to deep learning models, there are also traditional machine learning approaches such as random forests and support vector machines that have been used for brain tumor segmentation. However, these methods generally require manual feature engineering and may not perform as well as deep learning models.

One of the challenges in brain tumor segmentation is dealing with class imbalance, as the tumor regions are often much smaller than the surrounding healthy tissue. This can be addressed using specialized loss functions, such as the weighted cross-entropy loss or the Dice loss.

Overall, brain tumor segmentation is an important area of research in medical image analysis, and deep learning models have shown great promise in this field. By accurately identifying and segmenting tumor regions, these models can help improve the accuracy of diagnosis and treatment planning, leading to better outcomes for patients.

2.1.8 Evaluation matrices:

Evaluate the performance of the developed deep learning models for brain tumor classification using appropriate evaluation metrics.
Utilize standard classification metrics such as accuracy, Sensitivity, Specificity, precision, recall and F1-score.
Accuracy: Measure the overall correctness of the classification predictions by calculating the ratio of correctly classified samples to the total number of samples.
Precision: Determine the proportion of true positive predictions out of all positive predictions, providing insight into the model's ability to minimize false positive errors.
Recall: Calculate the proportion of true positive predictions out of all actual positive samples, indicating the model's ability to capture all positive cases and minimize false negative errors.
Sensitivity: Measure the model's ability to correctly identify positive samples, also known as the true positive rate or recall.
Specificity: Determine the model's ability to correctly identify negative samples, also known as the true negative rate.
F1-Score: Combine precision and recall into a single metric, providing a balanced measure of the model's performance by considering both false positive and false negative errors.

2.2 Literature review

2.2.1 Classification literature review

a- Amin ul Haq, et.al have used a deep CNN model for the classification of tumor types of Meningioma, Glioma, and Pituitary employing brain tumor MR images data. To enhance the predictive capability of the CNN model. [8]

- Try to solve problems that have facing them by using:

1) Data augmentation technique (zooming)

2) using transfer learning

(ResNet-50, VGG-16, Inception V3, DenseNet201, Xception, and MobileNet)
Trained with big ImageNet data set.

- **Results before and after augmentation:**

CNN	Transfer learning	Integrated framework
Before Aug: 97.40% After: 98.56% **With using cross dataset** Before Aug: 97.96% After: 98.97%	**ResNet-50** before: 97.03% After: 98.07% **VGG-16** Before: 94.77% After: 95.97% **Inception V3** Before: 93.23% After: 96.03% **Xception** Before: 93.00% After: 95.60% **MobilNet** Before: 96.76% After: 97.87%	**ResNet50-CNN** Before: 99.10% After: 99.90% **The VGG-16-CNN** Before: 96.78% After: 97.88% **Inception V3-CNN** Before: 97.00% After: 98.02% **DenseNet201-CNN** Before: 97.00% After: 97.90% **Xception-CNN** Before: 98.20% After: 98.97% **MobileNet-CNN** Before: 98.08% After: 98.56%

Table 1: results of previous work

B- Md. Saikat Islam Khan, et.al have introduces two deep learning models for identifying brain abnormalities as well as classifying different tumor grades, including meningioma, glioma, and pituitary. The "proposed 23-layer CNN" architecture is designed to work with a relatively large volume of image data, whereas the "Fine-tuned CNN with VGG16" architecture is designed for a limited amount of image data. A comprehensive data augmentation technique is also conducted to enhance the "Fine-tuned CNN with VGG16" model's performance. These experimental results demonstrated that both models enhance the prediction performance of diagnosis of brain tumors. CNN achieved 97.8% in dataset 1 (Fig share dataset.) and combined model achieve 100% prediction accuracy for dataset 2(Harvard Medical Dataset) [9]

C- Ahmed S. Salama et.al [10] used nine pre-trained TL which are: Inceptionresnetv2, Inceptionv3, Xception, Resnet18, Resnet50, Resnet101, Shufflenet, Densnet201 and Mobilenetv2 as they used a fine-grained classification approach to identify automatically the brain tumors and detect them. Fine-grained classification experiments explains that Inceptionresnetv2 TL algorithm achieves the highest accuracy which is 98.91% in detecting and classifying three brain tumors which are: glioma, meningioma and pituitary brain tumors and Resnet50 achieved the lowest average accuracy which is 67.03% .Variants of Resnet framework achieved different results as Resnet18 achieved a minimum accuracy of 67.03% , Resnet50 achieved an accuracy of 67.03% whereas Resnet101 achieved an accuracy of 74.09% which is the highest among all variants. Inceptionv3 achieved an accuracy of 94.48%, Xception achieved

an accuracy of 98.37%, Shufflenet achieved an accuracy of 68.71%, Densenet201 achieved an accuracy of 68.71% and Mobilenetv2 achieved an accuracy of 82.61%.

They compare inceptionresnetv2 TL algorithm with hybrid approaches as they used convolutional Neural Networks (CNN) for deep feature extraction and Support Vector Machine (SVM) for classification. The hybrid DL approaches used in the experiments are Mobilnetv2, Densenet201, Squeeznet, Alexnet, Googlenet, Inceptionv3, Resnet50, Resnet18, Resnet101, Xception, Inceptionresnetv3, VGG19 and Shufflenet.They selected these models because they are popular with their good performance for image classification.

They used a small dataset of brain MRI images [11] to classify brain tumors into three categorical categories: glioma, pituitary and meningioma. The dataset contains 822 MRI images of meningioma,827 MRI images of pituitary tumors and 826 MRI images of glioma brain tumors in the training folder. The dataset contains 115 images of meningioma, 72 images of pituitary tumors, and 100 images of glioma brain tumors. They used 80% of the dataset as training and 20% was used for testing.
They found that applying CNNs on the medical imaging due to three reasons, the first reason is small datasets as radiologists considered labeling the images is time consuming task, so it is better for them a small number of images. The second reason is training CNN on a small dataset causes overfitting. The third reason is adjusting the hyperparameters of CNN classifiers to achieve better performance requires domain expertise. They solved this problem by data augmentation, using pre-trained models on TL and fine tuning.

The values of the parameters they used in all transfer learning architectures: optimization algorithm is SGDM, Maximum Epochs =14, Learning rate is 0.01, verbose is False, Validation Frequency is 30 and Shuffle is Every epoch.
They performed all experiments with Intel® core™ i5-5200U CPU and 8 GB of RAM and R2020a version of MATLAB for implementation.

D- Abdul Hannan Khan et.al [12] used Internet of medical things (IoMT) as the brain tumor identification system consists of three essential layers: data acquisition, preprocessing & application. In the first layer (data acquisition) collects data in raw form and this layer consists of variables such as input and output. Then it passed to the second layer (preprocessing layer) then was managed, moved and normalized. The data will be sent to the prediction layer (third layer) where CNN was applied .and this third layer computed the precision and miss rate of the problem.
The model has two phases: Training and validation in which they used 2870 images, 87% of the total input for the training phase, and the remaining 13% for validation phase. This dataset is divided into 826 training images of glioma,822 training images of meningioma, 395 training No-tumor and 827 training images of pituitary tumors. The dataset was divided into 100 testing images of glioma, 115 testing images of meningioma, 105 testing images of no-tumor and 74 testing images of pituitary tumors.

The result is: Accuracy in training phase is 94.84% and Miss Rate is 51.6% and the accuracy in testing phase is 92.13% and Miss Rate is 7.87%

E - Hossam H. Sultan, et.al were getting the best validation accuracy obtained by using CNN is 96.1%. While the loss function is less than 0.2.
We must mention that because of using 32 images as a mini-batch size the curve firstly drops sharply with
some fluctuations [13] but these tend to disappear after 10000 iterations for both curves. On the other hand, for
model testing, we use 459 slices, and the model shows the test accuracy of 93.2%.

Tumour type	Precision %	Sensitivity %	Specificity %	Accuracy %
Glioma	93.8 %	98.7 %	94.8 %	96.5 %
Meningioma	96.8 %	89.4 %	96.8 %	96.6 %
Pituitary	99.1 %	97.8 %	99.1 %	99.1 %

Table 2: results of previous work

Model	Accuracy %
J. Cheng et al. [14]	91.28
Paul et al. [15]	91.43
Parian Afshar et al. [16]	90.89
Amin Kabir et al. [17]	94.2
Swati et al. [18]	94.82
Talon Muhammed, et al. [19]	95.23 ± 0.6
Alex Net	82.2
Resnet-50	75.6

Table 3: results of previous work

F -Abhishek Sawner, et al [20] have designed a model to predict the four classes namely.
Glioma, Meningioma, Pituitary or No tumor from the jpeg image. For this model to train they have
used images from Kaggle database and the experiments were performed on google colab.
Keras models used for this work are Dense, Flatten, Conv2D, Maxpool2D, and Dropout.
TensorFlow framework is used to build the model. The Epoch were set at 15 with callback for early stop. The size of the batch is 32.

Table 2. Result of 4 Way Classifications

	Class	Precision	Recall	Accuracy
Resnet50	**Pituitary**	82.05	80.00	73.77
	Glioma	51.16	**78.57**	
	Meningioma	**100.00**	66.66	
	No Tumor	86.66	66.66	
InceptionV3	**Pituitary**	75.00	82.50	77.86
	Glioma	66.66	50.00	
	Meningioma	81.25	**86.66**	
	No Tumor	85.36	89.74	
Our model	**Pituitary**	**91.89**	**85.00**	**85.24**
	Glioma	**67.74**	75.00	
	Meningioma	85.71	80.00	
	No Tumor	**92.50**	**94.87**	

Table 4: results of previous work

G - Osman Özkaraca, et al [21] have used CNN, VGG16 and DenseNet. They modified CNN architecture after discovering the deficiencies in each model and the modified CNN was run on 20 epochs during the training phase .This modified CNN achieved 99% accuracy in the training phase and 95% in the validation (pre-test phase) accuracy rate and they added a k-fold cross validation method with k equals 10 in order to ensure the reliability of the model .At the beginning , they used k=5 to detect how the operation of the system was affected. They observed an increase in the success rates between 97 and 95.

Drawback: There are some deficiencies in Basic CNN in classification as it can't reach the desired success rate due to the small number of layers, so they solved this problem by increasing the number of layers. They observed using the transfer learning method doesn't affect health field's success rate, even if the success rate is high in other areas. The success rate of DenseNet didn't increase to the expected level due to the transfer learning so they modified CNN architecture to maintain the strength points and remove the weakness points. The most important weakness of the proposed CNN model is the long processing time. The late response of the model is due to creating the model using both dense and convolutional layers.

Result: The models were run on the dataset with 10 epochs. The accuracy of CNN is up to 92% during the training phase and the validation accuracy rate (pretest phase) is 91.92%. The accuracy of VGG16 is up to 91.03% and the validation accuracy is 86.63%. The accuracy of DenseNet is up to 90.30% and the validation accuracy rate is 86.08%. Basic CNN achieved average precision, average recall and average f1score which are 0.9225, 0.9 and 0.92 respectively. VGG16Net achieved average precision, average recall and average f1score which are 0.8625, 0.8625 and 0.8575 respectively. DenseNet achieved average precision, average recall and average f1score which are 0.8775, 0.8775 and 0.8475 respectively. Modified CNN achieved average precision, average recall and average f1score which are 0.96, 0.96 and 0.9650 respectively.

1)Performance of CNN model by class:

Brain tumours class	Precision	Recall	F1score
0	0.89	0.90	0.89
1	0.97	0.98	0.97
2	0.88	0.85	0.86
3	0.95	0.97	0.96

Table 5: results of previous work

2) Performance of VGG16 model by class:

Brain tumour class	precision	Recall	F1score
0	0.80	0.89	0.84
1	0.93	0.96	0.94
2	0.85	0.67	0.75
3	0.83	0.94	0.90

Table 6: results of previous work

3) DenseNet Architectural structure performance values by class:

Brain tumours class	precision	Recall	F1score
0	0.75	0.99	0.85
1	0.91	0.99	0.95
2	0.93	0.83	0.88
3	0.92	0.58	0.71

Table 7: results of previous work

4) Performance values by modified CNN model class:

Brain tumours classes	precision	Recall	F1score
0	0.96	0.97	0.97
1	0.97	0.98	0.98
2	0.96	0.91	0.94
3	0.95	0.99	0.97

Table 8: results of previous work

2.2.2 Segmentation literature review

a- K. Alhajeri et al have used a hybrid deep CNN architecture that combines both 2D and 3D convolutions to improve the performance of the network.
The authors evaluated their algorithm on the BraTS 2020 dataset and achieved a mean Dice score of 0.87, a mean sensitivity of 0.86, and a mean specificity of 0.99. The proposed algorithm outperformed several other state-of-the-art methods on the dataset.[22]

Summary of the Algorithm:
The proposed algorithm consists of several key components, including:

1. Hybrid deep CNN: The algorithm uses a hybrid deep CNN architecture that combines both 2D and 3D convolutions to improve the performance of the network. The hybrid CNN includes both 2D and 3D convolutional layers, allowing the network to capture both local and global features.
2. Segmentation: The algorithm uses a SoftMax activation function to perform the segmentation. The segmentation is performed in a voxel-wise manner, with each voxel being classified as either tumor or non-tumor.

The authors also incorporated several other techniques to improve the performance of the algorithm, including data augmentation, batch normalization, and dropout.

b- Y. Wu et al have used a modified U-Net architecture that incorporates both 2D and 3D convolutions to improve the performance of the network.
The authors evaluated their approach on the BraTS 2020 dataset and achieved a mean Dice score of 0.82, a mean sensitivity of 0.82, and a mean specificity of 0.99. The proposed approach outperformed several other state-of-the-art methods on the dataset.[23]

Summary of the Algorithm:
The proposed algorithm consists of several key components, including:

1. U-Net architecture: The algorithm uses a modified U-Net architecture that incorporates both 2D and 3D convolutions to improve the performance of the network. The modified U-Net includes both 2D and 3D convolutional layers, allowing the network to capture both local and global features.
2. Multimodal approach: The algorithm leverages multiple MRI modalities for brain tumor segmentation, including T1-weighted, T1-weighted with contrast, T2-weighted, and FLAIR. The multimodal approach helps to improve the accuracy of the segmentation.
3. Segmentation: The algorithm uses a SoftMax activation function to perform the segmentation. The segmentation is performed in a voxel-wise manner, with each voxel being classified as either tumor or non-tumor.

The authors also incorporated several other techniques to improve the performance of the algorithm, including data augmentation, batch normalization, and dropout.

Chapter three: Planning and Analysis

3.1 Project planning

3.1.1 Feasibility study

Start date:16/6/2022.
Estimated End Date:10/6/2023.

Executive Summary:

The aim of our project is to help doctors in diagnosing 3 brain tumors which are: glioma, meningioma and pituitary tumor and also in diagnosing the healthy case and segmenting the tumors.

Marketing Feasibility:

1- **Market Analysis:**

Conduct a comprehensive analysis of the healthcare industry, specifically the field of brain tumor analysis and diagnosis.
Identify the target market, including hospitals, medical clinics, and research institutions.
Assess the demand for advanced brain tumor classification and segmentation solutions.

2- **Competitor Assessment**:

Identify and evaluate existing deep learning-based systems or software solutions in the market.
Analyze their strengths, weaknesses, pricing models, customer reviews, and market share.
Understand the competitive landscape.

3- **Unique Selling Proposition:**

Define the unique features and advantages of your project's deep learning-based system.
Highlight accuracy, efficiency, user-friendliness, and potential to improve medical decision-making and patient outcomes.

4- **Target Customers:**

Identify primary customer segments, such as healthcare professionals (radiologists, oncologists) and medical students.
Understand their specific needs, pain points, and preferences in brain tumor analysis.

5- **Market Reach and Distribution:**

Determine effective distribution channels to reach target customers.
Consider collaborations with medical institutions, partnerships with healthcare technology providers, or direct marketing to medical professionals and educational institutions.

Technical Feasibility:

We plan to use HTML, CSS, JavaScript, and Bootstrap for frontend, and we also plan to use python and flask framework to handle connection between front and machine models.
We plan to use deep learning models like custom CNN, We plan to use transfer learning models like Xception, InceptionV3, VGG16, VGG19, DenseNet121, ResNet50 and ResNet152V2 , Appling data augmentation to solve the small-sized dataset and Appling fine tuning to give high accuracies in classification. And using segmentation models to doing segmentation process

Financial Feasibility:

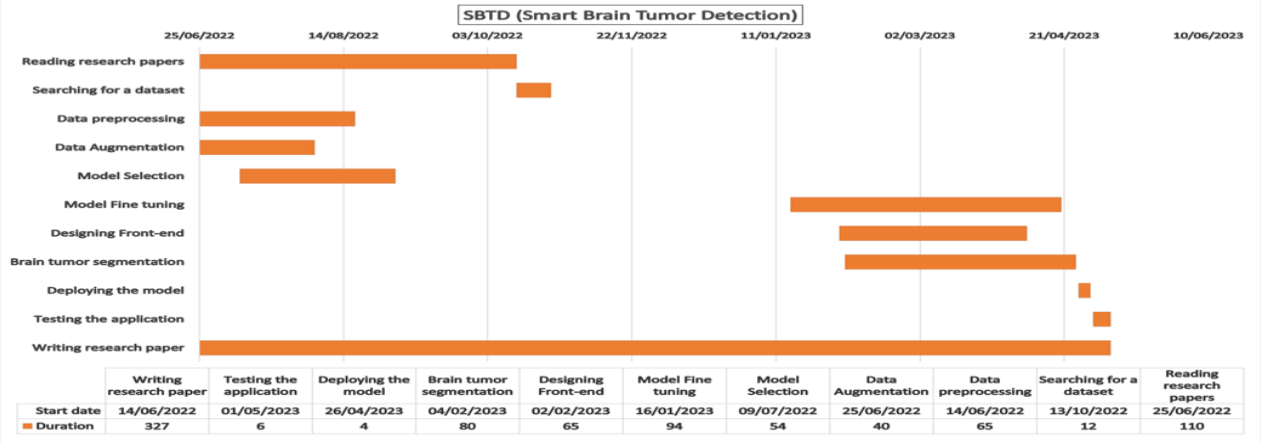
There are no investors or current revenue.

Organizational Feasibility:
The general Structure of the team is:
Team Leader:
Somia Ahmed
Team Members:
Eman Ahmed
Noura Nabil
Muhammed Fouad
Huda Ayman

3.1.2 Estimated cost

There are no fees for a database or in deploying the online medical system but the cost in our project is our effort in reading research papers related to our project, searching for dataset, learning deep learning techniques, fine tuning model with high accuracies, designing the front end, deploying the model, testing the web application, writing our research paper and publishing it.

3.1.3 Gantt chart



	Writing research paper	Testing the application	Deploying the model	Brain tumor segmentation	Designing Front-end	Model Fine tuning	Model Selection	Data Augmentation	Data preprocessing	Searching for a dataset	Reading research papers
Start date	14/06/2022	01/05/2023	26/04/2023	04/02/2023	02/02/2023	16/01/2023	09/07/2022	25/06/2022	14/06/2022	13/10/2022	25/06/2022
Duration	327	6	4	80	65	94	54	40	65	12	110

Fig 1: Gantt chart

3.2 Analysis and Limitations of existing System

Project Background:
The project title is "Brain tumor detection System ". Main features of this software are based on machine learning algorithms and deep learning methods to detect brain tumors from magnetic resonance images with high accuracy. A Convolutional Neural Network (CNN) has been used as the algorithm for feature extraction, classification, and segmentation.

System Analysis:
System analysis is a process of gathering and interpreting facts, diagnosing problems and the information to recommend improvements to the system. System analysis is an important phase of any system development process. The system is viewed as a whole and the input to the system is identified. The outputs from the organizations are traced to the various processes. System analysis is concerned with becoming aware of the problem, identifying the relevant and decisional variables, analyzing and synthesizing the various factors and determining an optimal or at least a satisfactory solution or program of action.

Existing System:
there is a computer-based procedures to detect tumor blocks and classify the type of tumor using Artificial Neural Network Algorithm for MRI images of different patients and there is multiply segmentation tools that are used to detect the area of the tumor, but it is has low accuracy because of small dataset, simple user interface, Time consuming and need more computational power.

Proposed System:
The aim of the proposed system is to develop a system to improve medical services, The proposed system can overcome all the limitations of the existing system. The existing system has several disadvantages and many more difficulties working well. The proposed system tries to eliminate or reduce these difficulties to some extent. It will help the user to reduce the workload and mental conflict.

Expected Advantages of Proposed System:

The system is very simple in design and to implement. The system requires very low system resources, and the system will work in almost all configurations. It has got the following features.
1-Minimum time needed for the various processing.
2-Minimize manual data entry.
3-Higher accuracy
4-Greater efficiency.
5-Better service.

6-User friendliness and interactive.
7-Minimum time required.

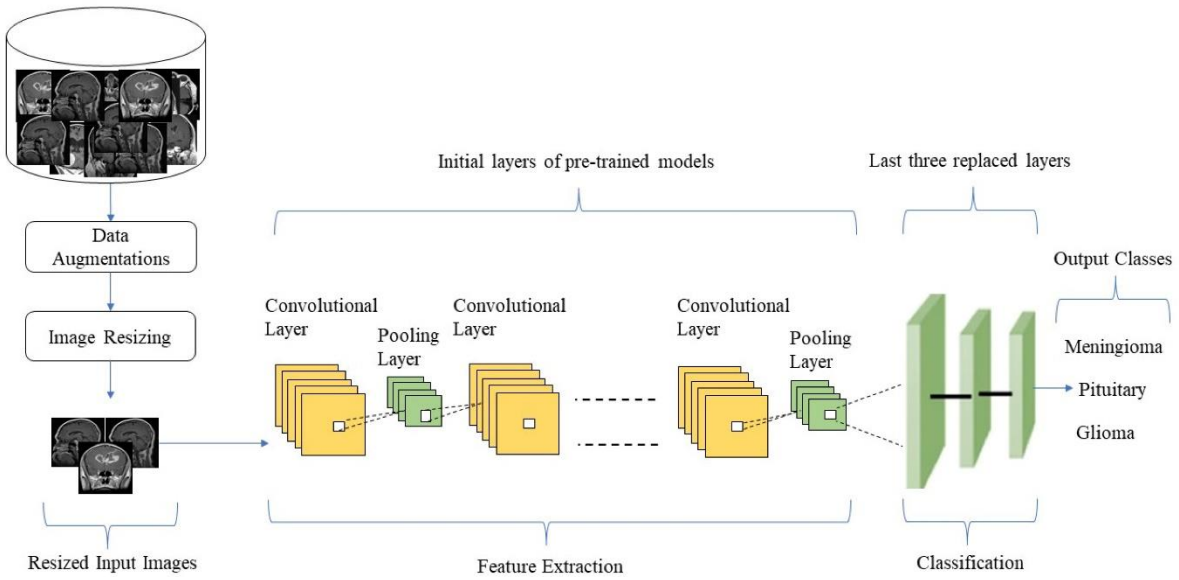


Fig 2: Classification tool workflow [24]

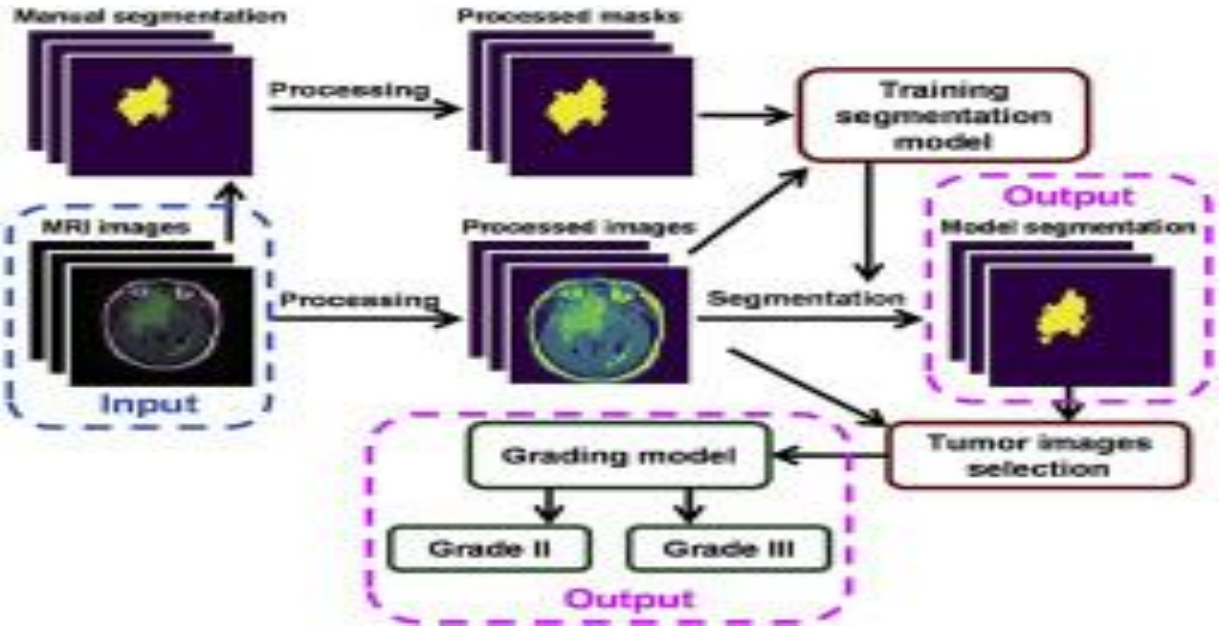


Fig 3: Segmentation tool workflow [25]

3.3 Analysis of the System:

3.3.1: User requirements:

1- It is imperative for users to distinguish between the three types of tumors displayed within the system, to ensure appropriate diagnosis and treatment.

2- The accurate determination of tumor depth and differentiation of health from affected tissue are crucial aspects of tumor management as this knowledge can aid in treatment decisions and improve outcomes.

3- A fundamental knowledge of tumors is essential for individuals, in order to accurately diagnose and treat such conditions.

4- The acquisition of an adequate understanding of common inquiries related to tumors is crucial for the development of a comprehensive comprehension of these conditions.

5- It is essential for practitioners and patients to be apprised of the recent advances and trends in tumor treatment, in order to provide optimal care and improve outcomes.

6- Gaining knowledge of the experiences of afflicted individuals, their struggles, and trials regarding such conditions, can offer valuable insights into the patient experience, ultimately contributing to improved patient care.

7-There should be a part for children and the elderly, so that the information is not harsh and difficult to understand, or there should be an interactive part

3.3.2: System requirements:

1- It is implemented through utilizing the classification tool offered by the application, the user simply needs to navigate to the navigation bar and click on the appropriate option (tools and choose classification). Following this, the user can upload the image that they wish to have classified and then click the submit button. Subsequently, the user will have to wait for some time for the tool to process the image and provide a clearer image. The system will then process the image and provide the user with a clearer image along with the name of the tumor displayed. The user may need to wait for a brief period of time for the processing to take place.

2- From the navigation bar, select Preoperative Tool. Then the user uploads several photos that express the length, width and depth of the disease. And he presses the submit button, where the images are sent to the segmentation model, and he merges the images to show the depth of the tumor in one image. This image provides a clear visual representation of the depth at which the three images are combined.

3- The user can do this by clicking on the information in the navigation bar, where a brief description of the three tumors is available; The system then displays this information along with the prevalence of the three tumors and the survival rates for each disease.
4- This is done by clicking directly on the FAQS from the navigation bar.

5- This is done by clicking directly on the treatment from the navigation bar, as this page contains the latest available treatments and documented sources.

6- This is done by clicking directly on the treatment from the navigation bar, as this page contains the latest available treatments and documented sources.

7- This is done by clicking directly on the memory game from the navigation bar. This game represents the interactive part for patients with illnesses, where the user tries to get two similar cards and has a certain number of attempts, which are 22 attempts. If his attempts ended with a number of cards remaining, a message appears to him that he can try again and play again.

3.3.3: Functional requirements:

User features:

Requirements ID	Functional Requirements	priority	Comments
FR001	Any user should be able to view and use Homepage	Must	
FR002	Any user should be able to view and use tool page	Must	
FR003	Any user should be able to use classification tool from tool page	Must	
FR004	Any user should be able to browse MRI images from the device (as an input to the classification tool)	Must	Classification model should be taking MRI images as input from the user and identify the type of the brain tumor and show the results
FR005	Any user should be able to use segmentation tool from tool page	Must	
FR006	Any user should be able to import 3D MRI file (as an input to the segmentation model)	Must	Segmentation model should be able to use 3D MRI file to identify more information about the tumors (e.g., it's location and extent of the tumor, as well as the segmentation of the tumor into its different components, such as the active tumor, the edema (swelling), and the necrosis (dead tissue).)
FR007	Any user should be able to save the results on his device (e.g., pc, laptop, etc.)	Must	
FR008	Any user should be able to know more information about diagnosis of these brain tumors, treatment, follow up and how to deal with it through the FAQ that will be provided in the Homepage	Must	
FR009	Any user should be able to view stories about patients who have been treated.	Must	

FR010	Any user should be able to play puzzle game to improve memory	Must	
FR011	Any user should be able to view treatment plan page	Must	
FR012	Any user should be able to view project team information	Must	

Table 9: function requirements

3.3.4: Non-functional requirements:

Performance Requirements:

Requirement id	Requirement Description
1.PR	The Application could be used by many users at the same time.
2.PR	The Application must have an active internet connection.
3.PR	The Application should be used with modern browsers.
4.PR	The Application should respond operations within 3 seconds for all users or admin

Table 10: performance requirements

Safety Requirements:

Requirement id	Requirement Description
1.SR	The Application must be prevented from injection attacks to prevent loss or modifying data.

Table 11: safety requirements

Security Requirements:

Requirement Id	Requirement description
1.SR	The Application must use hashing technology to prevent the unauthorized access & provide secure login.

Table 12: security requirements

Look-and-feel requirements:

Requirement Id	Requirement description
1.LR	The product should use blue and white colours.

Table 13: look and feel requirements.

Software Quality Attributes:

1- Usability:

1- The application shall have user friendly interface as it is easy to be used by users on the first.

2- usage without training.

2- Accuracy:

The results that the user will receive should be highly accurate and precise.

3- Portability:

1- The System shall use python and flask framework for backend.

2- The system shall use Html, Css, JavaScript and Bootstrap programming languages for the frontend.

3- The system shall use python and deep learning libraries for classification and using preprocessing techniques.

4- Maintainability:

Application should be able to be modified to have a new class of users. (Admins, Doctors.)

5- Correctness:

Application should not allow users to edit details available on applications.

6- Availability:

1- The availability of the Application will be all days through the week and in case.

2- on system crash (down time), it will be back to be available within hours.

7- Reliability:

After crashing, the online medical system must be restarted within five seconds.

8- Testability:

This attribute refers to the fact that each functional and non-functional requirement can be tested.

3.4 Risk and risk management

3.4.1 Brain tumor classification dataset

There is a lot of risks associated with the dataset as after collecting the dataset we observed its small amount so we begin to research how the model performs using this small dataset so this is a risk on the results of the project any it may lead to overfitting, so we overcome this risk by applying data augmentation on the training set.

Another risk related to the dataset is that no image is preprocessed so we decided to do pre-processing stage as we resized the images, adjust images brightness, resolution and crops the rectangular out of the extreme points on the image, threshold the image, then perform a series of erosions and dilations to remove any small regions of noise and grab the largest contours in thresholder image.

There was another risk which was getting low accuracies in some model, and we solved it by adding cleaned images after preprocessing stage to each type of tumor which belongs to it in the training and validation after doing augmentation.

There is a risk about performance of the model, so we overcame this risk by splitting dataset into training data and validation data.

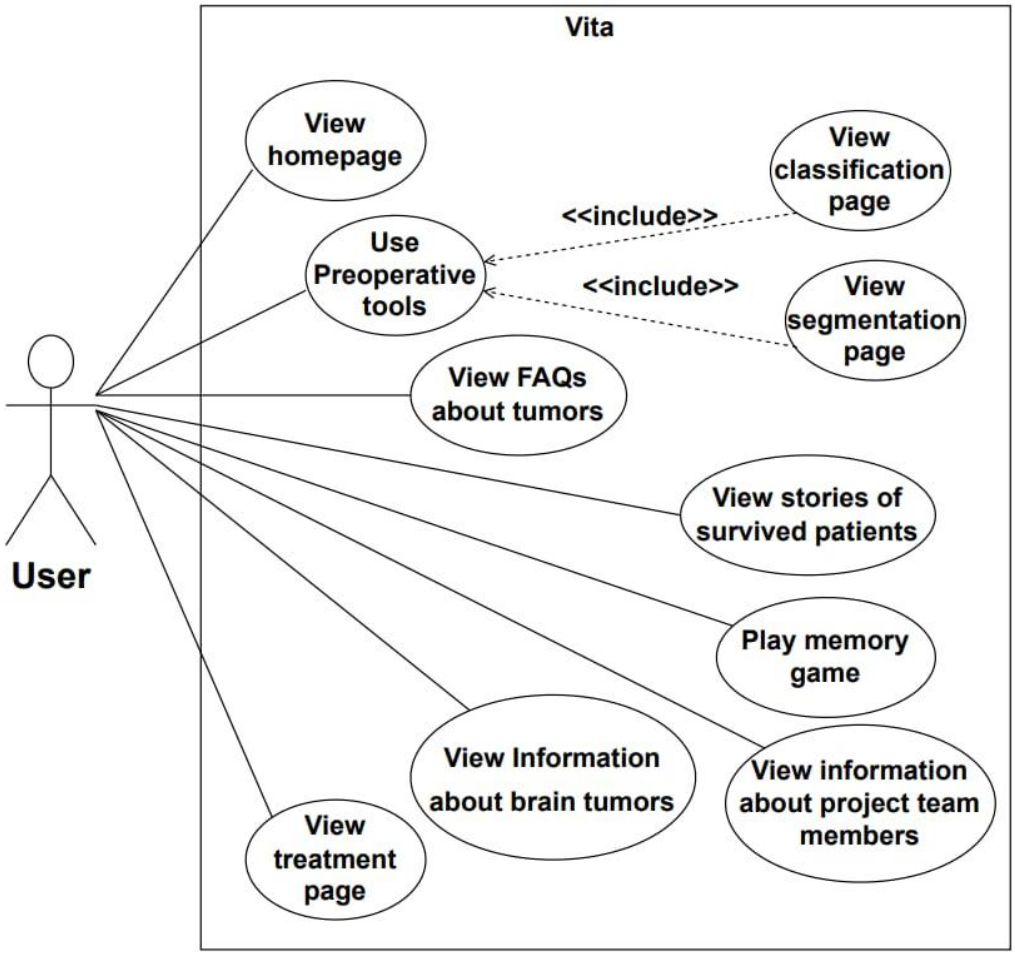
There is a risk of overfitting on training data so we viewed training accuracy, training loss, validation accuracy and validation loss on each epoch and we applied the early stopping in the code with patience=4 so the model will stop training in case of repeating the same accuracies for four times after each other and in some transfer learning models we reduced patience to 2 to overcome rapid overfitting.

3.4.2 brain tumor segmentation dataset

We faced a problem in segmentation the tumor as the number of files of segmentation mask is less than number of cases by one, so we discovered that there is one file of segmentation mask that wasn't named by the name of the file of its case, so we changed its name. The extension of these files was. nii so we convert it by NumPy to deal with the model.
Another problem is that the labels were supposed to be 0123 but the written sequence is 0124 as the label no. (3) was not written and after searching in other datasets we renamed label no. (4) to label no. (3).
Another problem was that there weren’t any contributions in visualizing the result as 3d image, but it was visualized as animation (GIF image), and it was managed as we searched for many ways to visualize the 3d image.

Chapter four: Software design

4.1 Use case diagram:



The USECASE description:

1-View homepage

1. Identifier:	UC1 view homepage
2. Initiator:	The user
3. Goal:	The user views the homepage and shows the tabs of the tabbed navigation menu to select other pages to view from the tabs and the user can back to the homepage again.
4. Precondition:	None.
5. Postcondition:	The homepage will be viewed, and the user can choose from tabs to view other pages.
6. Main success scenario:	1. The user opens our website. 2. The use can select which page to open.

Table 14: homepage description

2-Use preoperative tools

1. Identifier:	UC2 use preoperative tools
2. Initiator:	The user
3. Goal:	The user views the menu of preoperative tools which contains classification and segmentation tools, and the user can choose which tool to use.
4. Precondition:	The user viewed the homepage and clicked on “Tools” tab from the tabbed navigation menu.
5. Postcondition:	Tools tab will list the available tools which are classification tool and segmentation tool.
6. Main success scenario:	1. The user views the homepage. 2. The user clicks on the “Tools” tab on the tabbed navigation menu. 3. The user chooses which tool to use.

Table 15: preoperative tools page description

3-View classification page

1. Identifier:	UC3 view classification page.
2. Initiator:	The user
3. Goal:	The user views the classification page to use the classification tool to classify the tumour of the MRI scan.
4. Precondition:	The user viewed the homepage and selected “Tools” tab then clicked on classification tool.
5. Postcondition:	The user can upload the image of MRI scan to classify the image into glioma, meningioma, pituitary tumor or no tumor.
6. Main success scenario:	1. The user views the homepage. 2. The user clicks on the “Tools” tab on the tabbed navigation menu. 3. The user selects the classification tool and views the classification page. 4. The user uploads the image of MRI scan to classify the tumor and clicks on submit button. 5. The result of the classification (the name of the tumor) will appear on the page.
7. Exceptions:	4.1- The user uploads an image of unavailable extensions 4.2-A message will appear to the user telling him /her to upload image with an appropriate extension.

Table 16: classification page description

4-View Segmentation page:

1. Identifier:	UC4 view segmentation page
2. Initiator:	The user
3. Goal:	The user views the segmentation page to use the preoperative tool to segment the tumor of the MRI scan.
4. Precondition:	The user viewed the homepage and selected "Tools" tab then clicked on preoperative tool.
5. Postcondition:	The user chooses the number of cases then the image of segmentation will appear.
6. Main success scenario:	1. The user views the homepage. 2. The user clicks on the "Preoperative tools" tab on the tabbed navigation menu. 3. The user selects the segmentation tool and views the segmentation page. 4. The user chooses the case to view the segmented image of the tumor then can choose to view. 1.
7. Exceptions:	4.1- When the user enters a wrong number of case, a message will appear telling him/her this number don't exist.

Table 17: segmentation page description

5-View FAQs about tumours

1. Identifier:	UC5 view FAQs about tumors.
2. Initiator:	The user.
3. Goal:	The user can view the page of the frequently asked questions about tumors.
4. Precondition:	The user viewed the homepage and clicked on "FAQs" tab.
5. Postcondition:	The FAQs page will be viewed, and the user can read the questions.
6. Main success scenario:	1. The user views the homepage. 2. The user clicks on the" FAQs "tab on the tabbed navigation menu. 3. The user views the FAQs page and read the questions about the tumors in generally, its types, how to avoid it, It's risk factors and information about awake brain surgeries.

Table 18: FAQ page description

6-View stories of survived patients

1. Identifier:	UC6 View stories for patients who have been survived.
2. Initiator:	The user.
3. Goal:	The user views the page of the stories of patients who have been survived.
4. Precondition:	The user viewed the homepage and clicked on “stories” tab.
5. Postcondition:	The stories page will be viewed, and the user can read the stories of patients.
6. Main success scenario:	1. The user views the homepage. 2. The user clicks on the “Stories” tab on the tabbed navigation menu. 3. The user views the stories page.

Table 19: view stories page description

7-Play memory game

1. Identifier:	UC7 play memory game.
2. Initiator:	The user.
3. Goal:	The user views the page of the puzzle game of 6 X4 image and can play the game.
4. Precondition:	The user viewed the homepage and clicked on “memory game “tab.
5. Postcondition:	The page of the puzzle game will be viewed, and the user can play the game with score.
6. Main success scenario:	1. The user views the homepage. 2. The user clicks on the “Memory game” tab on the tabbed navigation menu. 3. The user views the page of the puzzle game and can play it with a counter of 22 tries counting down by one with every try on turning 2 cards and the 2 similar cards will not be flipped again. 4- A message will appear after finishing the game asking him if he wants to play again after he / she loses

Table 20: memory page description

8-View information about project team members.

1. Identifier:	UC8 view information about project team members.
2. Initiator:	The user.
3. Goal:	The user views the "About Us" page and can read information about our project team members who developed and designed this website.
4. Precondition:	The user viewed the homepage and clicked on "About us" tab.
5. Postcondition:	The "About Us" page will be viewed, and the user can read the information our project team members.
6. Main success scenario:	1. The user views the homepage. 2. The user clicks on the "About Us" tab on the tabbed navigation menu. 3. The user viewed the "About Us" page.

Table 21: information page description

9-View information about brain tumors

1. Identifier:	UC9 view information about brain tumors
2. Initiator:	The user.
3. Goal:	The user views the information (Info) page to read about brain tumors.
4. Precondition:	The user viewed the homepage and clicked on "Info" tab.
5. Postcondition:	The treatment page will be viewed, and the user can read the types of treatments.
6. Main success scenario:	1. The user views the homepage. 2. The user clicks on the "Info" tab on the tabbed navigation menu. 3. The user views the information page and read about tumors.

Table 22: brain tumors information page description

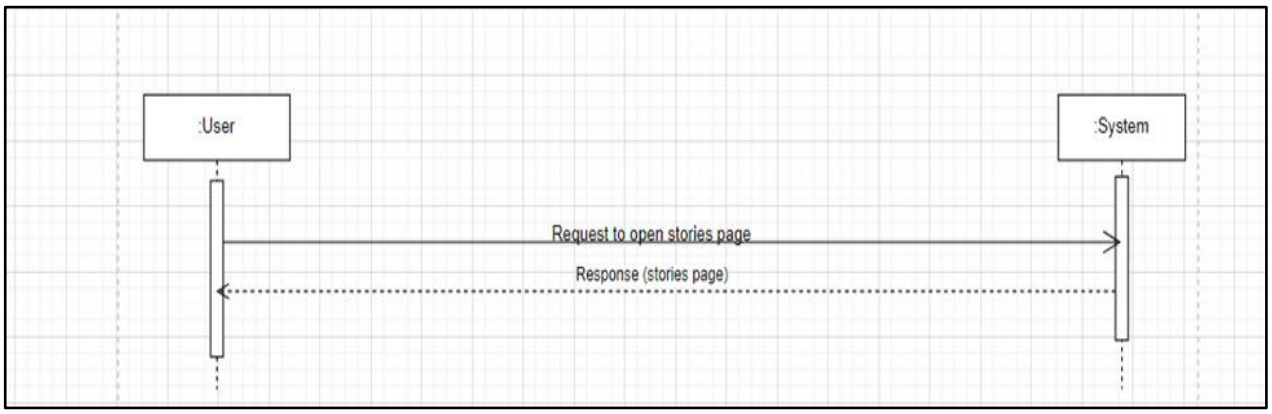
10- View treatment page.

1. Identifier:	UC10 view treatment page
2. Initiator:	The user.
3. Goal:	The user views the treatment page and read the common treatments of brain tumors.
4. Precondition:	The user viewed the homepage and clicked on "Treatment" tab.
5. Postcondition:	The treatment page will be viewed, and the user can read the types of treatments.
6. Main success scenario:	1. The user views the homepage. 2. The user clicks on the "Treatment" tab on the tabbed navigation menu. 3. The user views the treatment page.

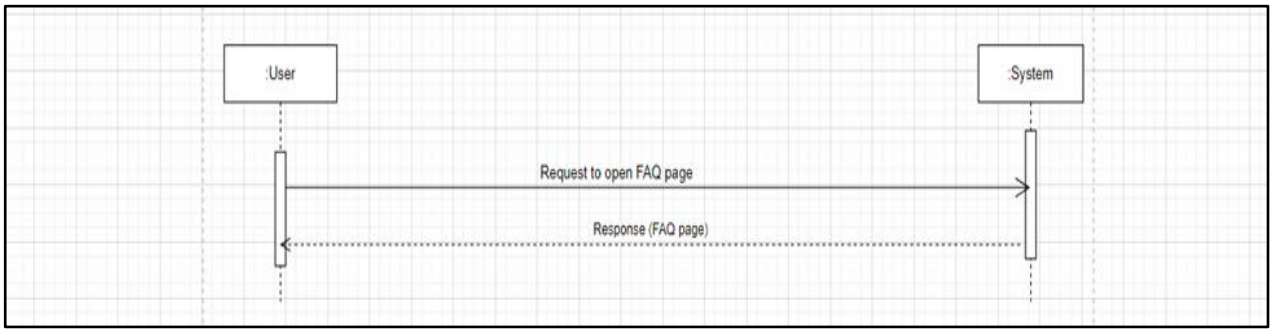
Table 23: treatment page description

4.2 sequence diagram

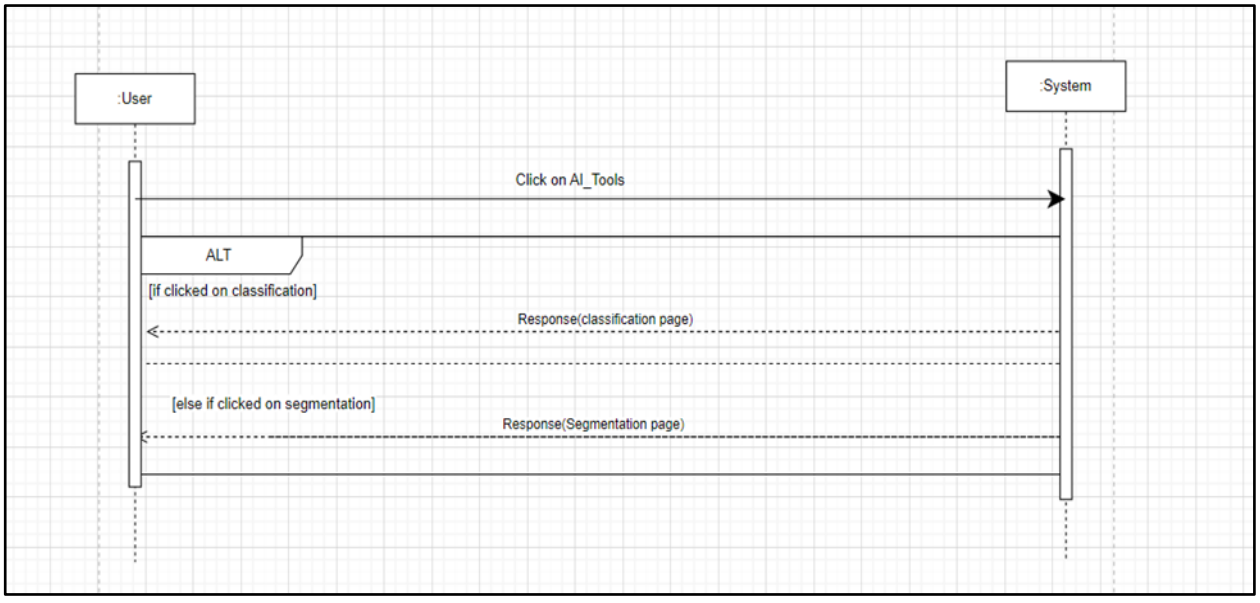
Stories page:



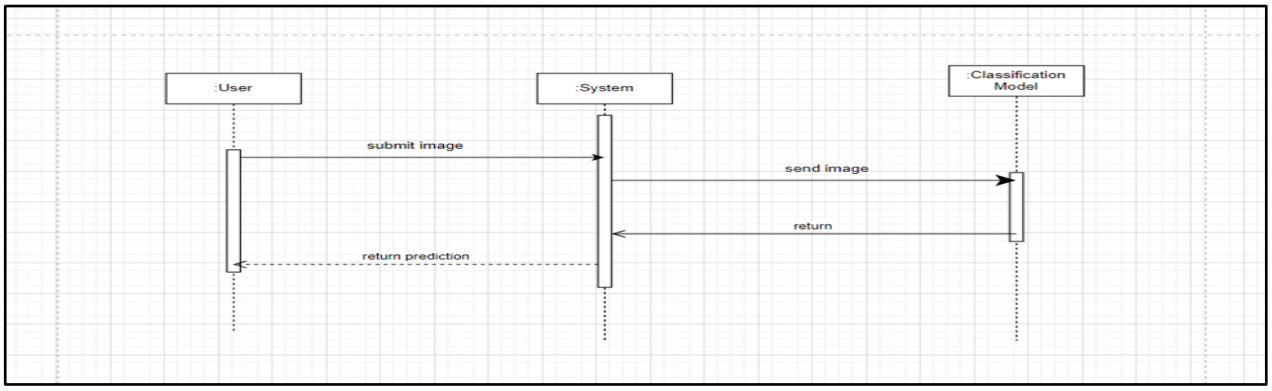
FAQ page:



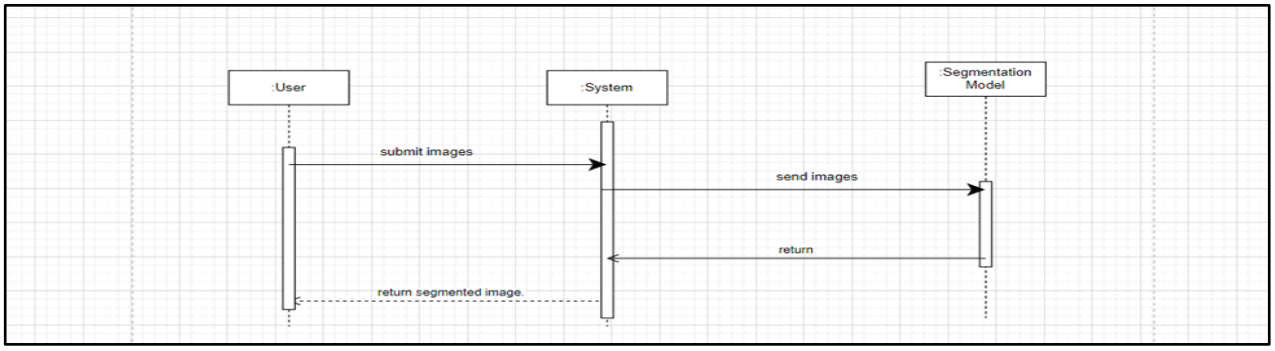
AI_Tools:



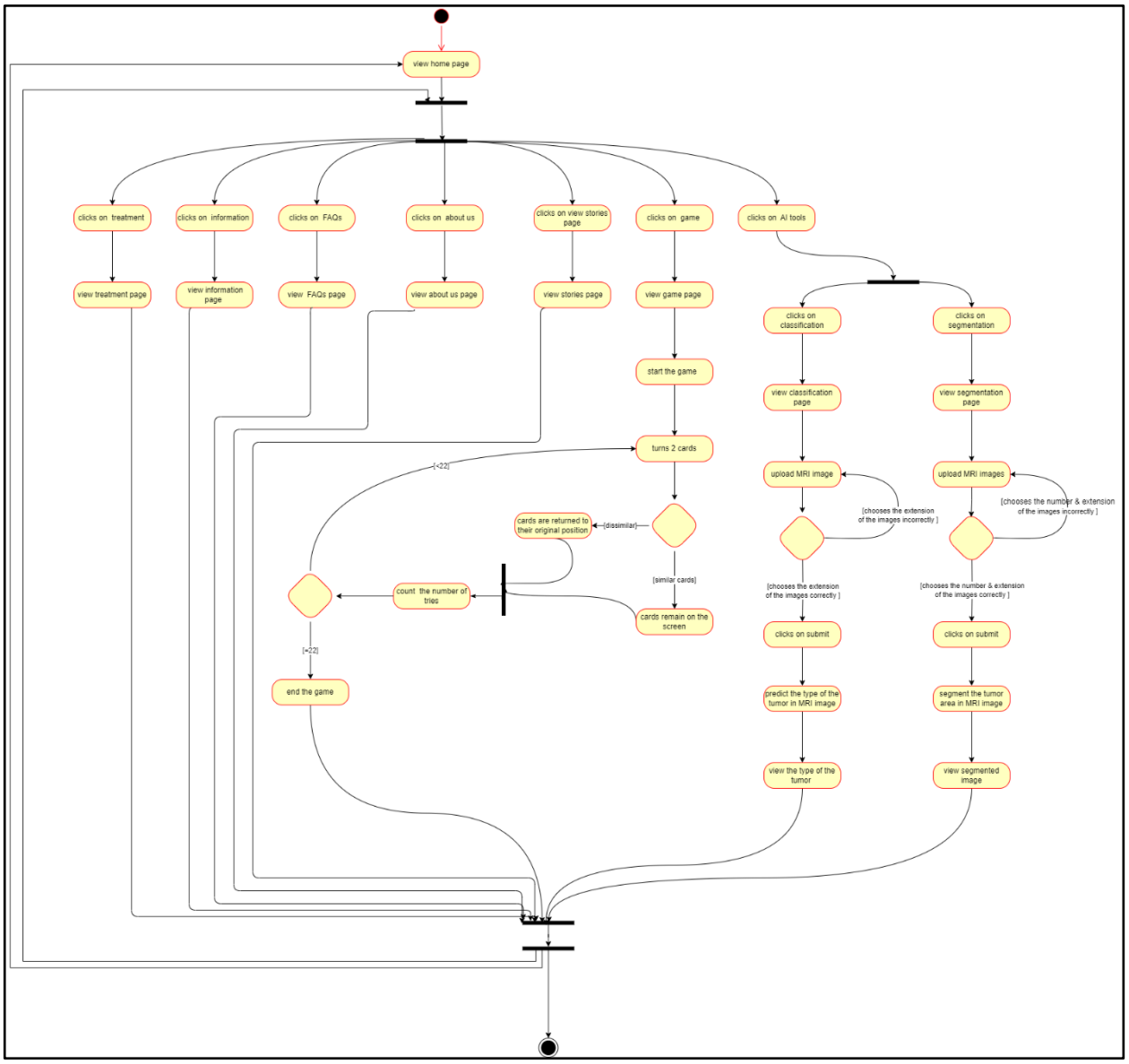
Classification page:



Segmentation page:



4.3 Activity diagram



Chapter five: Our proposed work

5.1 Brain tumor classification

(FLOWCHART FOR DESIGN AND DEVELOPMENT OF PROPOSED PROJECT)

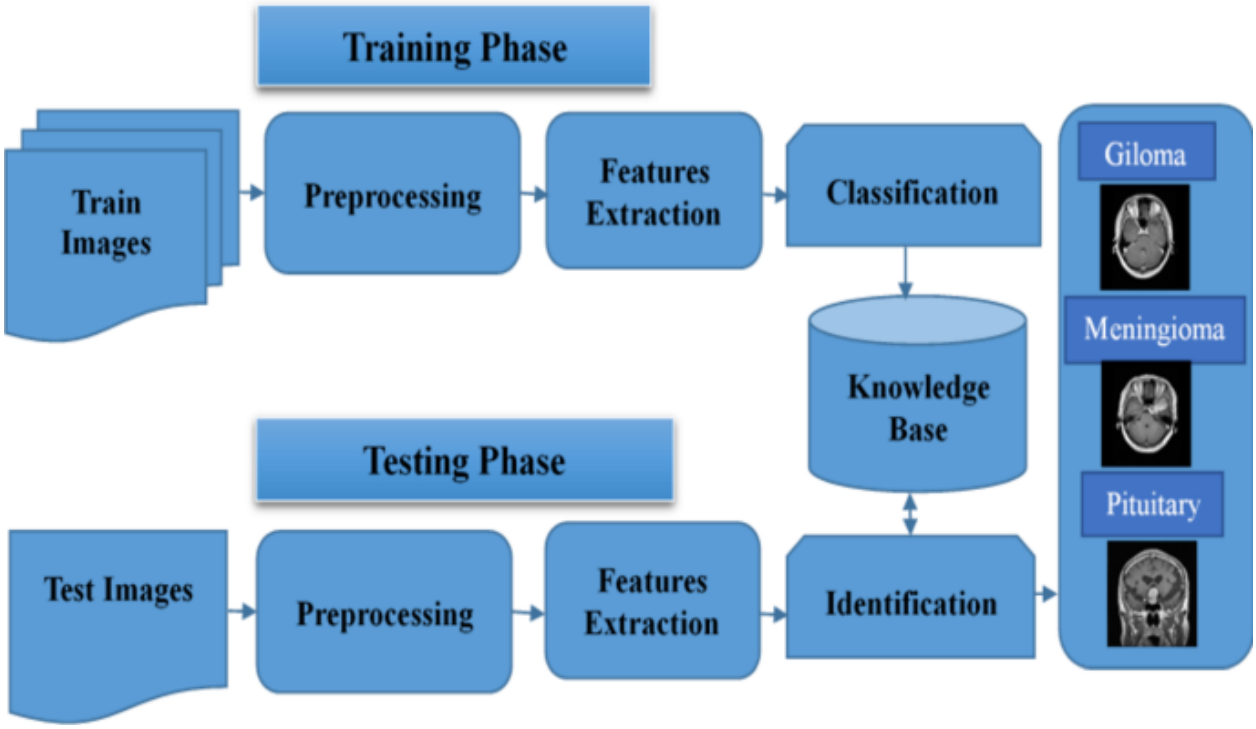


Fig 4: classification flow chart [26]

5.1.1 Software Requirements:

1- Python [version 3.9.12] - Python is chosen as the programming language for this project due to several reasons:

   1. Python code is compact and readable, making it easier to understand and maintain compared to MATLAB.

   2. Python offers superior data structures compared to MATLAB, making it more versatile for handling complex data.

   3. Python is an open-source language with a large community, providing access to a wide range of graphic packages and data sets.

2- Keras (with TensorFlow backend version 2.10.0) - Keras is a popular neural network API that is built on top of TensorFlow, CNTk, Theano, and other deep learning frameworks.

3- Python packages like NumPy, Matplotlib, Pandas - These packages are used for mathematical computation and plotting graphs in the project.

4- Kaggle - Kaggle was used to obtain the online dataset for the project.

5- GitHub and Stackoverflow - These platforms were used for reference in case of programming syntax errors and for finding solutions to coding issues.

6- OpenCV (Open-Source Computer Vision) - OpenCV is a popular library of programming functions used for real-time computer vision, including image processing and operations related to images such as reading and writing images, modifying image quality, noise removal using Gaussian Blur, etc. It supports multiple programming languages including C++, Java, C, and Python, and is commonly used in applications like CamScanner, Instagram, and GitHub.

7- Google Colaboratory (open-source Jupyter Notebook interface with high GPU facility) - Google Colab is a free Jupyter notebook environment that runs on the cloud, requiring no setup. It provides the ability to write and execute code, save and share analyses, and access powerful computing resources for free from a web browser. Jupyter Notebook is a powerful way to iteratively write and run Python code for data analysis, making it a valuable tool in this project. It is built on top of iPython, which is an interactive way of running Python code, and supports multiple languages and markdown for documentation.

5.1.2 Hardware Requirements:

1-Operating System: Windows 10 Pro 64-bit (10.0, BUILD 19044)
2-Minimum CPU or processor speed: Intel(R) Core (TM) i7-1065G7 CPU @ 1.30GHz (8 CPUs), 1.5 GHz
3- Minimum GPU or video memory:3979MB
4-Minimum system memory (RAM):8192 MB RAM
5-Minimum free storage space:8433MB available
6-Audio hardware (sound card, speakers, etc.): Speakers (Realtek® Audio)

5.1.3 Methodology

Our proposed methodology is based on the DNN learning.
The proposed methodology for classifying brain tumors.
is as follows:
Step 1: Brain MRIs Dataset acquisition
Step 2: preprocessing of the dataset
Step 3: performing augmentation for the dataset.
Step 4: expected results
Step 5: Doing Classification using DNN models and transfer learning models.
Step 6: obtaining models result.

Step 1: Brain MRIs Dataset acquisition

Fig 5: snapshot of dataset in Kaggle

1-The dataset is a combination of the following three datasets [27] :
figshare
SARTAJ dataset
Br35H
1-This dataset consists of 7022 images of human brain MRI (Magnetic Resonance Imaging) images which are of JPEG type. A total of 22% images are reserved for the model testing and the rest approximately 78% are for model training, these images are further classified into four categories:
-Pituitary: An image for representation of Pituitary type tumor is shown below in fig
Here we have 1457 (.jpg) images of pituitary tumor for training.

Fig 6: pituitary tumor images

-meningioma: An image for representation of meningioma type tumor is shown below in fig
Here we have 1339 (.jpg) images of meningioma tumor for training.

Fig 7: meningioma tumor images

- no tumor: An image for representation of no tumor type is shown below in fig
Here we have 1595 (.jpg) images of no tumor for training.

Fig 8: No tumor images

2-SARTAJ dataset [28] has a problem that the glioma class images are not categorized correctly so we changed these images with glioma class images that are existing in this dataset ,This dataset is collected from Figshare dataset, All the images have been converted to **.jpg** for easier access

Fig 9: snapshot of dataset in Kaggle

- glioma: An image for representation of glioma type is shown below in fig 4
Here we have 1426 (.jpg) images of glioma for training.

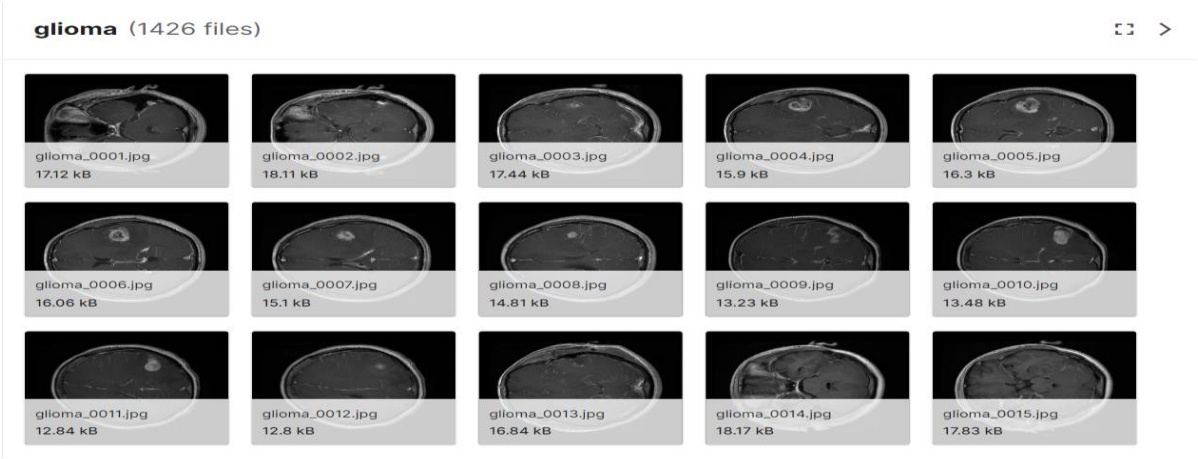


Fig 10: glioma tumor images

Table 24: Summary of Used Image Dataset

Class	Number of images
Glioma	1426
Meningioma	1645
Pituitary Tumor	1757
No tumor	2000
Total	**6828**

The percentages of tumors in the dataset (training folder) before augmentation or splitting: Glioma represents 20.61% of tumors, meningioma represents 24.21% of tumors, no tumor represents 28.84 % of total tumors and pituitary represents 26.33% of total tumors.

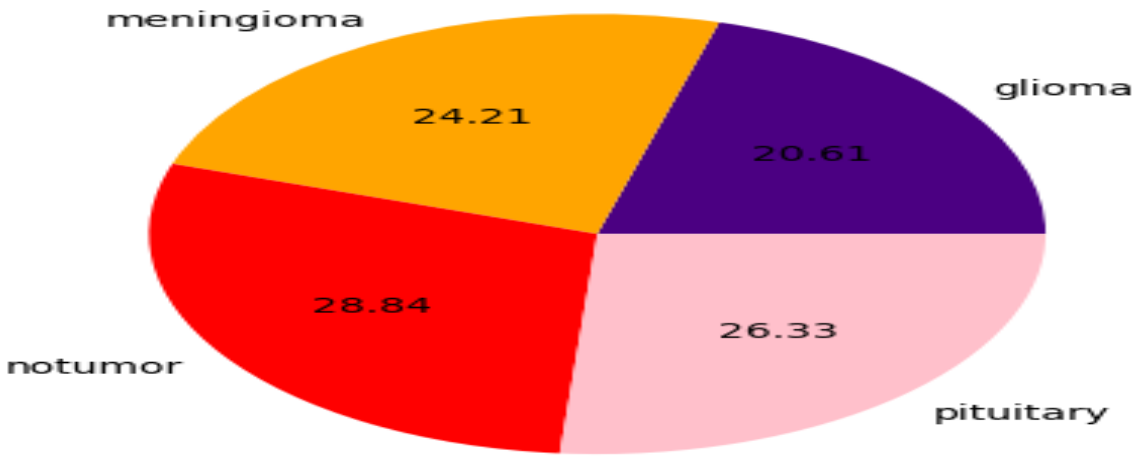


Fig 11: tumors percentages

Step 2: Preprocessing of the dataset

Our pre-processing pipeline involves several phases to enhance the quality of the image data. First, we rescaled the gray-level intensity values of the pixels in the range of 0-255 to reduce memory space and ensure consistency. Next, we apply Gaussian blur filter for noise removal, which has been found to be more effective than Median filter in preserving the outline of the brain while removing noise. Additionally, we utilize Binary Thresholding to segment the image and separate the tumor region from the rest of the brain. Finally, we apply morphological operations such as erosion and dilation to further refine the contours of the tumor region. These pre-processing steps are essential in preparing the image data for further analysis and subsequent tumor detection tasks.

Fig 12: preprocessing technique

Step 3: Performing augmentation for the dataset.

data augmentation was done using the ImageDataGenerator class in Python. Data augmentation is a technique used in machine learning and deep learning to artificially expand the size of the training dataset by applying various transformations to the original images. This helps in improving the model's ability to generalize and learn patterns from diverse data.
For augmentation we used:

1. rescale: This function rescales the pixel values of the images by dividing them by 255, which normalizes the pixel values to the range of 0 to 1.

2. featurewise_center and samplewise_center: These functions control whether to center the images' pixel values based on the meaning of the entire dataset (featurewise) or the meaning of each individual sample (samplewise). In this case, both are set to False, which means the images' pixel values will not be centered.

3. featurewise_std_normalization and samplewise_std_normalization: These functions control whether to normalize the pixel values of the images based on the standard deviation of the entire dataset (featurewise) or the standard deviation of each individual sample (samplewise). In this case, both are set to False, which means the pixel values will not be normalized based on standard deviation.

4. zca_whitening: This function controls whether to apply Zero-phase Component Analysis (ZCA) whitening to the images. ZCA whitening is a preprocessing step that can be used to decorate the pixel values of the images. In this case, it is set to False, which means ZCA whitening will not be applied.

5. shear_range: This function controls the range of shear (in radians) to be applied to the images. Shear is a transformation that changes the shape of the image by shifting one part of the image in a direction perpendicular to the lines in the image. In this case, the shear range is set to 0.2, which means the images will be randomly sheared within the range of -0.2 to 0.2 radians.

6. rotation_range: This function controls the range of rotation (in degrees) to be applied to the images. Rotation is a transformation that rotates the image around its center. In this case, the rotation range is set to 40 degrees, which means the images will be randomly rotated within the range of -40 to 40 degrees.

7. zoom_range: This function controls the range of zooming to be applied to the images. Zooming is a transformation that changes the scale of the image. In this case, the zoom range is set to 0.2, which means the images will be randomly zoomed within the range of 0.8 to 1.2.

8. width_shift_range and height_shift_range: These functions control the range of horizontal and vertical shifting to be applied to the images, respectively. Shifting is a transformation that moves the image in a horizontal or vertical direction. In this case, both the width and height shift ranges are set to 0.2, which means the images will be randomly shifted horizontally and vertically within the range of -0.2 to 0.2 times the image width and height, respectively.

9. horizontal_flip and vertical_flip: These functions control whether to randomly flip the images horizontally (horizontal_flip) or vertically (vertical_flip). In this case, horizontal flipping is set to True, which means the images will be randomly flipped horizontally.

10. fill_mode: This function controls how the pixels that are created by the above transformations are filled. In this case, it is set to 'nearest', which means the pixels will be filled with the nearest pixel value.

The total number of images after doing augmentation is 38612 images.

Data set	Training	Validation	Testing
Before augmentation	4424	1107	1297
After augmentation	25426	6356	1297

After Adding cleaned non-augmented data to the augmented data	29852	7463	1297

Table 25: dataset images

Step 4: Expected Results

The goal of this project is to develop a deep learning model that can accurately classify brain tumor images into different categories. In this chapter, we will discuss the expected results from the model.

Accuracy: We expect the model to achieve an accuracy rate of 98% to 99%. This level of accuracy is based on the performance of other similar models in the literature and is achievable given the size and quality of our dataset.

Precision and Recall: Apart from accuracy, precision and recall are also important evaluation metrics in a classification problem. We expect the model to have high precision and recall rates for each tumor type, indicating that it can accurately distinguish between different types of brain tumors.

Robustness: The model's robustness is another important factor that we will evaluate. We expect the model to perform well on unseen data and to be able to generalize to new brain tumor images.

In conclusion, we expect the deep learning model to achieve high accuracy, precision, and recall rates, and to be robust and generalizable. These results will demonstrate the potential of deep learning techniques for brain tumor classification and could lead to better diagnosis and treatment of brain tumors in the future.

Step 5 and 6 Doing Classification using DNN models, and our transfer learning models with results.
transfer learning models:

There are many Models for image classification with weights on ImageNet are Xception, VGG16, VGG19, ResnNet, ResNet2, ResNet 50, Inception v2, Inception v3, MobileNet, MobileNet v2, DenseNet, AlexNet, GoogleNet, NasNet etc.
For the implementation of Transfer Learning in our project, we have chosen CNN, VGG16, VGG19, xception, Inception v3, as our samples.
For every model we will present a table that shows results before and after augmentation and the results obtained from testing on unseen data

Our proposed work:

First model CNN:

By using sequential method, we added five convolutional layers, five maxpooling layers, five drop out layers to prevent overfitting, one flattens layer and three dense layers. The input layer to the CNN is our target image size (224X224X3) is a 3-chanalled RGB image. We adjusted each drop out layer at a rate of 0.25 and this means 25% of the nodes are dropped out randomly from the neural network so it improves the model as it prevents overfitting, Total no of parameters used are 2,655,044. A snapshot image of the model consisting of its layers has shown in fig.

```
Model: "sequential"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
conv2d (Conv2D)              (None, 220, 220, 32)      2432
_________________________________________________________________
max_pooling2d (MaxPooling2D) (None, 73, 73, 32)        0
_________________________________________________________________
dropout (Dropout)            (None, 73, 73, 32)        0
_________________________________________________________________
conv2d_1 (Conv2D)            (None, 69, 69, 64)        51264
_________________________________________________________________
max_pooling2d_1 (MaxPooling2 (None, 23, 23, 64)        0
_________________________________________________________________
dropout_1 (Dropout)          (None, 23, 23, 64)        0
_________________________________________________________________
conv2d_2 (Conv2D)            (None, 21, 21, 128)       73856
_________________________________________________________________
max_pooling2d_2 (MaxPooling2 (None, 10, 10, 128)       0
_________________________________________________________________
dropout_2 (Dropout)          (None, 10, 10, 128)       0
_________________________________________________________________
conv2d_3 (Conv2D)            (None, 8, 8, 256)         295168
_________________________________________________________________
max_pooling2d_3 (MaxPooling2 (None, 4, 4, 256)         0
_________________________________________________________________
dropout_3 (Dropout)          (None, 4, 4, 256)         0
_________________________________________________________________
conv2d_4 (Conv2D)            (None, 2, 2, 512)         1180160
_________________________________________________________________
max_pooling2d_4 (MaxPooling2 (None, 1, 1, 512)         0
_________________________________________________________________
dropout_4 (Dropout)          (None, 1, 1, 512)         0
_________________________________________________________________
flatten (Flatten)            (None, 512)               0
_________________________________________________________________
dense (Dense)                (None, 1024)              525312
_________________________________________________________________
dense_1 (Dense)              (None, 512)               524800
_________________________________________________________________
dense_2 (Dense)              (None, 4)                 2052
=================================================================
Total params: 2,655,044
Trainable params: 2,655,044
Non-trainable params: 0
_________________________________________________________________
```

Fig 13: model architecture

Before augmentation:

Model	Epochs	Preprocessing	Patience(reduceonplateau)	Patience (Early Stopping)	optimizer	Learning Rate (LR)	Batch size	AVG train Accuracy	AVG validation Accuracy	AVG test Accuracy
CNN	100	No	4	4	Adam	0.0001	32	94.7%	90.4%	90.2%
CNN	50	No	4	4	Adam	0.001	64	93.6%	89.2%	89%
CNN	150	No	4	4	Adam	0.001	32	87.8%	84.5%	83%

Table 26: CNN model results before augmentation

Results before augmentation and after preprocessing:

model	Epochs	Preprocessing	Patience (reduceonplateau)	Patience (Early Stopping)	optimizer	Learning Rate (LR)	Batch size	AVG train Accuracy	AVG validation Accuracy	AVG test Accuracy
CNN	100	Yes	4	4	Adam	0.0001	16	97.4%	90.9%	93%
CNN	50	Yes	4	4	Adam	0.001	64	86.5%	81.4%	84%
CNN	150	Yes	4	4	Adam	0.0001	32	80.8%	77.1%	79%

Table 27: CNN model results before augmentation and after preprocessing

Results after augmentation and after preprocessing:

model	Epochs	Preprocessing	Patience (reduceonplateau)	Patience (Early Stopping)	optimizer	Learning Rate (LR)	Batch size	AVG train Accuracy	AVG validation Accuracy	AVG test Accuracy
CNN	200	Yes	4	4	Adam	0.0001	32	99.4%	95.2%	98%
CNN	100	Yes	4	4	Adam	0.0001	32	98.8%	95.1%	97%
CNN	150	Yes	4	4	Adam	0.0001	32	98%	94.2%	97%
CNN	150	Yes	2	2	Adam	0.0001	32	96.3%	93%	95%
CNN	70	Yes	4	4	Adam	0.0001	32	95%	90.6%	94%

Table 28: CNN model results after augmentation

EVALUATION OF THE PREDICTIVE MODEL PERFORMANCE

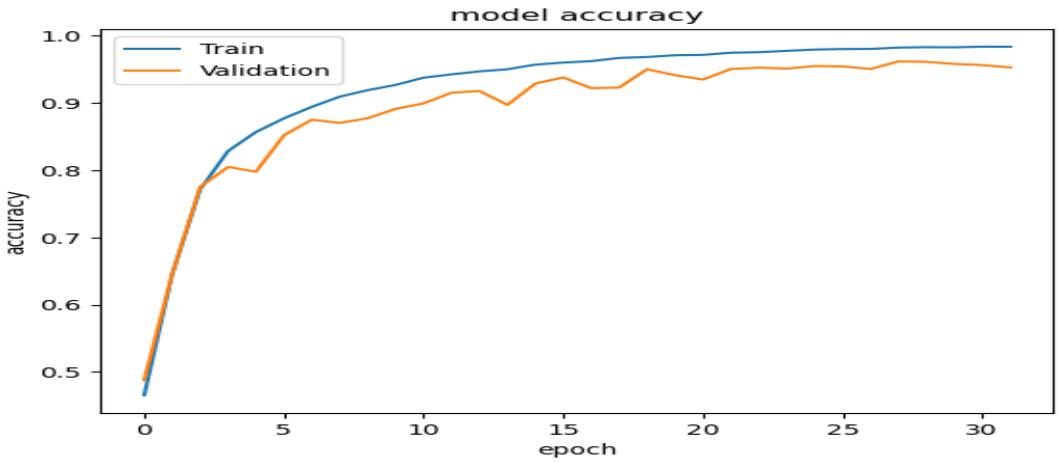


Fig 14: Training and Validation accuracy for the best model

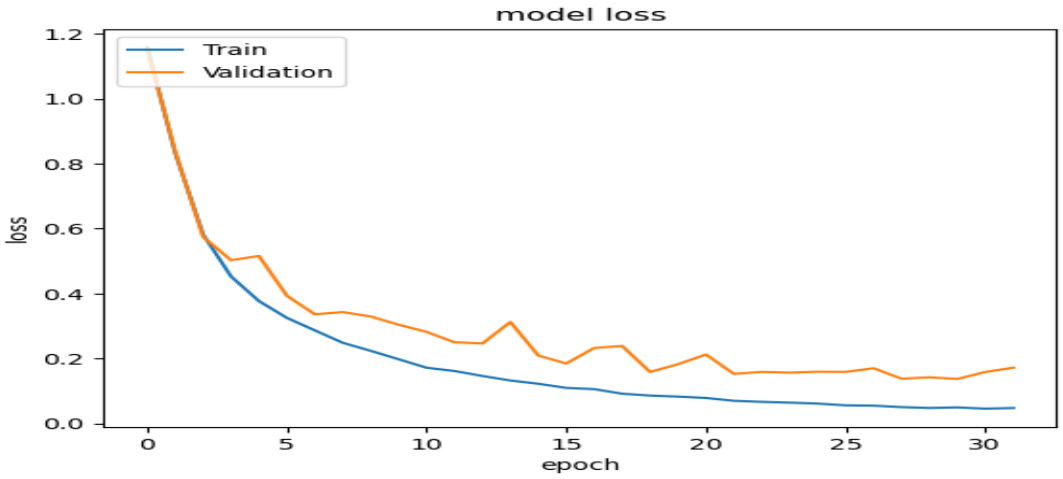


Fig 15: Training and Validation loss for the best model

	precision	recall	f1-score	support
glioma	0.98	0.97	0.97	286
meningioma	0.96	0.98	0.97	306
notumor	1.00	0.99	0.99	405
pituitary	0.98	0.99	0.98	300
accuracy			0.98	1297
macro avg	0.98	0.98	0.98	1297
weighted avg	0.98	0.98	0.98	1297

Fig 16: classification report for the best model

Second model xception model:

We have designed our model based on Convolution Neural Network. we have used Conv2D, Maxpool2D, flatten, we retrained some of the xception functional layers again, we used 5 dense layers with different number of neurons using relu activation function but in the last layer we use softmax activation function, and 9 dropout layers for our network. The image size is taken as 224*224.The kernel size has been reduced to 3*3 using dimension reduction feature. Total no of parameters used are 46,791,900. A snapshot image of the model consisting of its layers has shown in fig.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 xception (Functional)       (None, 7, 7, 2048)        20861480
 flatten (Flatten)           (None, 100352)            0
 dropout (Dropout)           (None, 100352)            0
 dense (Dense)               (None, 256)               25690368
 dropout_1 (Dropout)         (None, 256)               0
 dense_1 (Dense)             (None, 224)               57568
 dropout_2 (Dropout)         (None, 224)               0
 dense_2 (Dense)             (None, 220)               49500
 dropout_3 (Dropout)         (None, 220)               0
 dense_3 (Dense)             (None, 180)               39780
 dropout_4 (Dropout)         (None, 180)               0
 dense_4 (Dense)             (None, 160)               28960
 dropout_5 (Dropout)         (None, 160)               0
 dense_5 (Dense)             (None, 148)               23828
 dropout_6 (Dropout)         (None, 148)               0
 dense_6 (Dense)             (None, 148)               22052
 dropout_7 (Dropout)         (None, 148)               0
 dense_7 (Dense)             (None, 120)               17880
 dropout_8 (Dropout)         (None, 120)               0
 dense_8 (Dense)             (None, 4)                 484
=================================================================
Total params: 46,791,900
Trainable params: 46,356,428
Non-trainable params: 435,472
_________________________________________________________________
```

Fig 17: model architecture

Results before augmentation:

model	Epochs	prepro cessing	Patience(reduceon plateau)	Patience (Early Stopping)	optimizer	LR	Batch size	AVG train Accuracy	AVG validation Accuracy	AVG test Accuracy
Xception	50	NO	6	12	Adam	0.0001	12	99%	97%	85.5%
Xception	50	NO	6	10	Adam	0.0001	12	98.9%	96.7%	96%

Table 29: Xception model results before augmentation

Results before augmentation and after preprocessing:

model	Epochs	prepro cessing	Patience(reduceon plateau)	Patience (Early Stopping)	optimizer	LR	Batch size	AVG train Accuracy	AVG validation Accuracy	AVG test Accuracy
Xception	50	Yes	6	12	Adam	0.0001	12	99.6%	99.7%	97.5%
Xception	50	Yes	6	10	Adam	0.0001	12	99.7%	99%	97.9%

Table 30: Xception model results before augmentation and after preprocessing

Results after augmentation and after preprocessing:

model	Epochs	preproc essing	Patience(reduceon plateau)	Patience (Early Stopping)	optimizer	LR	Batch size	AVG train Accuracy	AVG validation Accuracy	AVG test Accuracy
Xception	50	Yes	6	12	Adam	0.0001	12	99.7%	99%	99.5%
xception	40	yes	6	12	Adam	0.0001	12	99.8%	99.2%	98.9%

Table 31: Xception model results after augmentation

EVALUATION OF THE PREDICTIVE MODEL PERFORMANCE

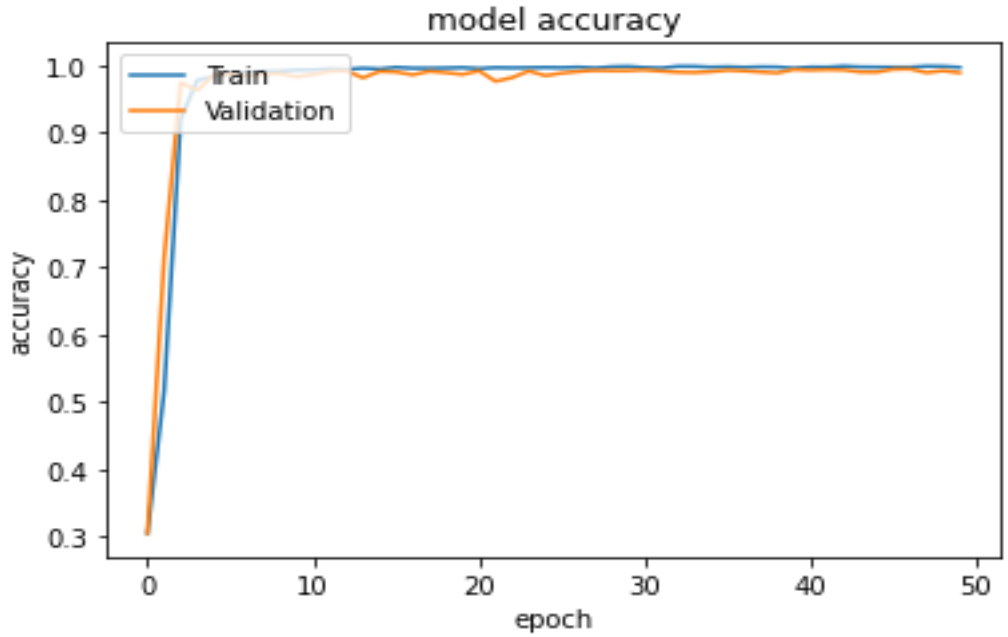


Fig 18: Training and Validation accuracy for the best model

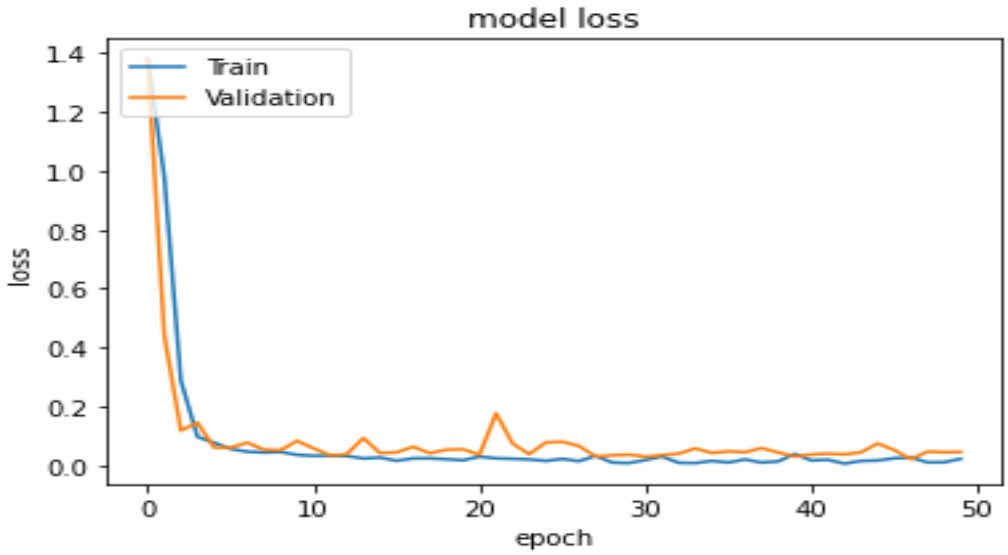


Fig 19: Training and Validation loss of the best model

	precision	recall	f1-score	support
glioma	1.00	0.98	0.99	286
meningioma	0.99	1.00	0.99	306
notumor	1.00	1.00	1.00	405
pituitary	1.00	1.00	1.00	300
accuracy			1.00	1297
macro avg	1.00	1.00	1.00	1297
weighted avg	1.00	1.00	1.00	1297

Fig 20: classification report for the best model

Third model inceptionv3 model:

We have designed our model based on Convolution Neural Network. we have used Conv2D, Maxpool2D, flatten, we retrain some of the inceptionv3 functional layers again, we used 8 dense layers with different number of neurons using relu activation function but in the last layer using softmax function and 9 dropout layers for our network. The image size is taken as 224*224.The kernel size has been reduced to 3*3 using dimension reduction feature. Total no of parameters used are 35,150,292. A snapshot image of the model consisting of its layers has shown in fig.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 inception_v3 (Functional)   (None, 5, 5, 2048)        21802784
 flatten (Flatten)           (None, 51200)             0
 dropout (Dropout)           (None, 51200)             0
 dense (Dense)               (None, 256)               13107456
 dropout_1 (Dropout)         (None, 256)               0
 dense_1 (Dense)             (None, 224)               57568
 dropout_2 (Dropout)         (None, 224)               0
 dense_2 (Dense)             (None, 220)               49500
 dropout_3 (Dropout)         (None, 220)               0
 dense_3 (Dense)             (None, 180)               39780
 dropout_4 (Dropout)         (None, 180)               0
 dense_4 (Dense)             (None, 160)               28960
 dropout_5 (Dropout)         (None, 160)               0
 dense_5 (Dense)             (None, 148)               23828
 dropout_6 (Dropout)         (None, 148)               0
 dense_6 (Dense)             (None, 148)               22052
 dropout_7 (Dropout)         (None, 148)               0
 dense_7 (Dense)             (None, 120)               17880
 dropout_8 (Dropout)         (None, 120)               0
 dense_8 (Dense)             (None, 4)                 484
=================================================================
Total params: 35,150,292
Trainable params: 35,115,860
Non-trainable params: 34,432
_________________________________________________________________
```

Fig 21: model architecture

Results before augmentation:

model	Epochs	preprocessing	Patience(reduceonplateau)	Patience (Early Stopping)	optimizer	LR	Batch size	AVG train Accuracy	AVG validation Accuracy	AVG test Accuracy
Inceptionv3	50	NO	6	12	Adam	0.0001	20	96.8%	94.5%	93.9%
Inceptionv3	50	NO	6	10	Adam	0.0001	20	99.8%	98.2%	97.9%

Table 32: InceptionV3 model results before augmentation

Results before augmentation but with preprocessing:

model	Epochs	preprocessing	Patience(reduceonplateau)	Patience (Early Stopping)	optimizer	LR	Batch size	AVG train Accuracy	AVG validation Accuracy	AVG test Accuracy
Inceptionv3	50	YES	6	12	Adam	0.0001	20	99%	98.8%	97.1%
Inceptionv3	50	YES	6	10	Adam	0.0001	20	99.8%	99.3%	98.5%

Table 33: InceptionV3 model results before augmentation and after preprocessing

Results after augmentation and after preprocessing:

model	Epochs	preprocessing	Patience(reduceonplateau)	Patience (Early Stopping)	optimizer	LR	Batch size	AVG train Accuracy	AVG validation Accuracy	AVG test Accuracy
Inceptionv3	50	YES	6	12	Adam	0.0001	20	99.9%	99.1%	99.2%

Inceptionv3	50	YES	6	10	Adam	0.0001	20	99.8%	99%	98.3%

Table 34: InceptionV3 model results after augmentation

EVALUATION OF THE PREDICTIVE MODEL PERFORMANCE

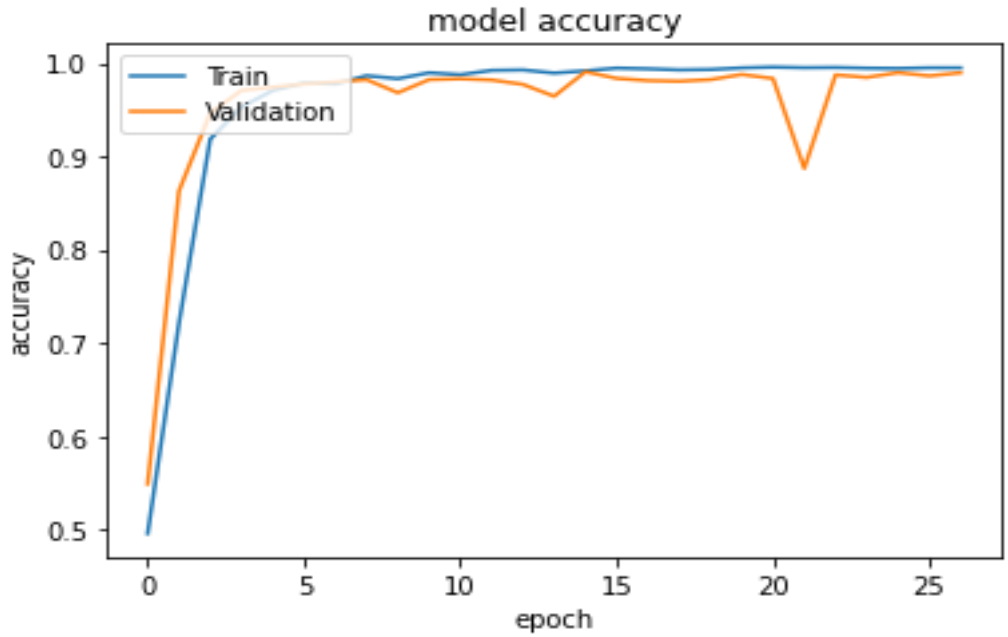


Fig 22: Training and Validation accuracy for the best model

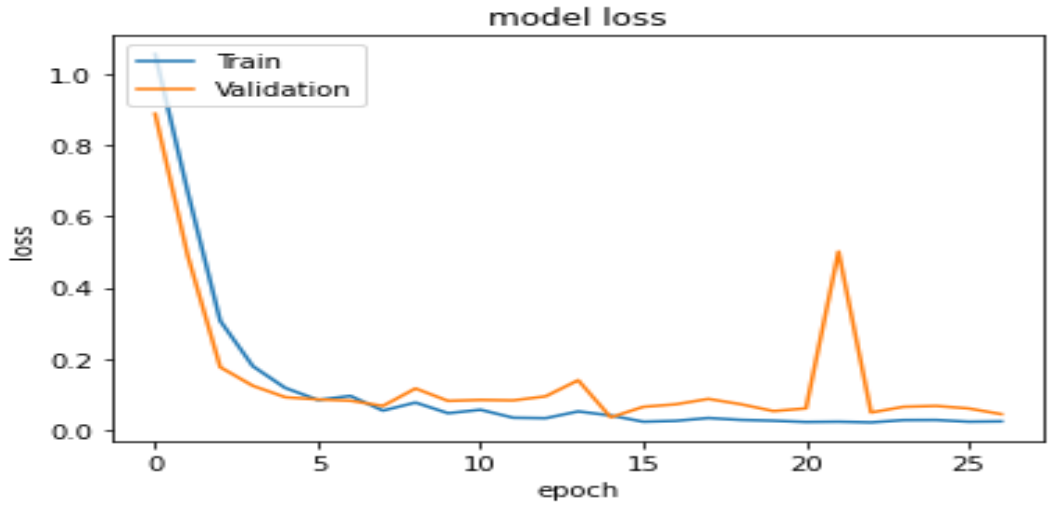


Fig 23: Training and Validation loss for the best model

	precision	recall	f1-score	support
glioma	0.99	0.99	0.99	286
meningioma	0.99	0.98	0.99	306
notumor	1.00	1.00	1.00	405
pituitary	0.99	0.99	0.99	300
accuracy			0.99	1297
macro avg	0.99	0.99	0.99	1297
weighted avg	0.99	0.99	0.99	1297

Fig 24: classification report for the best model

Fourth model VGG16 model:

We have designed our model based on Convolution Neural Network. We have used Conv2D, Maxpool2D, flatten, we retrained some of the vgg16 functional layers again, we used 4 dense layers with different number of neurons using relu function but in the last layer using softmax function and 5 dropout layers for our network. The image size is taken as 224*224.The kernel size has been reduced to 3*3 using dimension reduction feature. Total no of parameters used are 21,228,884. A snapshot image of the model consisting of its layers has shown in fig.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 vgg16 (Functional)          (None, 7, 7, 512)         14714688

 flatten (Flatten)           (None, 25088)             0

 dropout (Dropout)           (None, 25088)             0

 dense (Dense)               (None, 256)               6422784

 dropout_1 (Dropout)         (None, 256)               0

 dense_1 (Dense)             (None, 180)               46260

 dropout_2 (Dropout)         (None, 180)               0

 dense_2 (Dense)             (None, 148)               26788

 dropout_3 (Dropout)         (None, 148)               0

 dense_3 (Dense)             (None, 120)               17880

 dropout_4 (Dropout)         (None, 120)               0

 dense_4 (Dense)             (None, 4)                 484

=================================================================
Total params: 21,228,884
Trainable params: 13,593,620
Non-trainable params: 7,635,264
_________________________________________________________________
```

Fig 25: model architecture

Results before augmentation:

model	Epochs	Preprocessing	Patience(reduceonplateau)	Patience (Early Stopping)	optimizer	LR	Batch size	AVG train Accuracy	AVG validation Accuracy	AVG test Accuracy
VGG16	25	NO (using original data)	4	5	Adam	0.0001	20	99.8%	96.5%	97%
VGG16	25	NO	4	4	Adam	0.0001	20	99.4%	95.8%	95.5%
VGG16	25	NO	4	3	Adam	0.0001	20	98.8%	96%	97.7%

Table 35: VGG16 model results before augmentation

Results before augmentation and after preprocessing:

model	Epochs	Preprocessing	Patience(reduceonplateau)	Patience (Early Stopping)	optimizer	LR	Batch size	AVG train Accuracy	AVG validation Accuracy	AVG test Accuracy
VGG16	25	yes	4	5	Adam	0.0001	20	99.9%	99.5%	97.8%
VGG16	25	yes	4	4	Adam	0.0001	20	99.9%	99.4%	97.5%
VGG16	25	yes	4	3	Adam	0.0001	20	98.7%	97.6%	96.3%
VGG16	40	yes	4	10	Adam	0.0001	20	99.9%	99.1%	97.1%

Table 36: VGG16 model results before augmentation and after preprocessing

Results after augmentation and after preprocessing:

model	Epochs	Preprocessing	Patience(reduceonplateau)	Patience (Early Stopping)	optimizer	LR	Batch size	AVG train Accuracy	AVG validation Accuracy	AVG test Accuracy
VGG16	35	YES	4	5	Adam	0.0001	20	99.8%	98.4%	97.4%
VGG16	40	YES	4	10	Adam	0.0001	20	99.9%	98.3%	98.1%

Table 37: VGG16 model results after augmentation

We found that in order to increase accuracy and generalization of the model we can take best saved model from being trained on (original data) to be retrained again on augmentation data and there are the results [29]:

model	Epochs	Preprocessing	Patience(reduceonplateau)	Patience (Early Stopping)	optimizer	LR	Batch size	AVG train Accuracy	AVG validation Accuracy	AVG test Accuracy
VGG16	50	YES	4	10	Adam	0.0001	32	99.8%	98%	98.2%
VGG16	50	YES	4	4	Adam	0.0001	32	99.9%	97.5%	98.5%

Table 38: VGG16 model results after augmentation and retraining on saved model.

EVALUATION OF THE PREDICTIVE Model PERFORMANCE

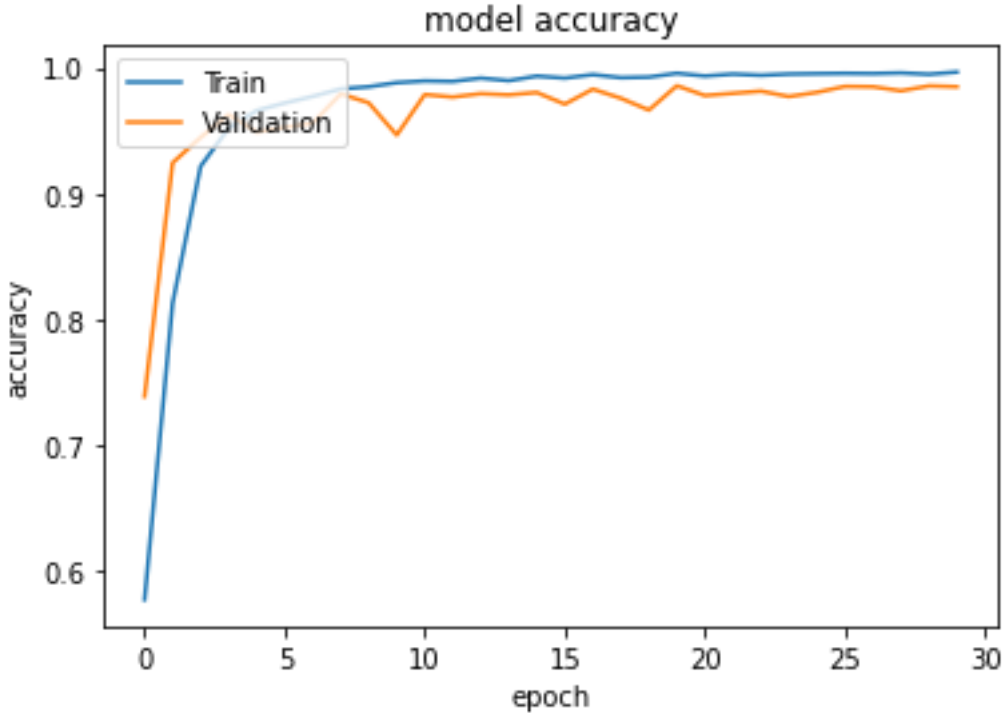


Fig 26: Training and Validation accuracy for the best model on original data

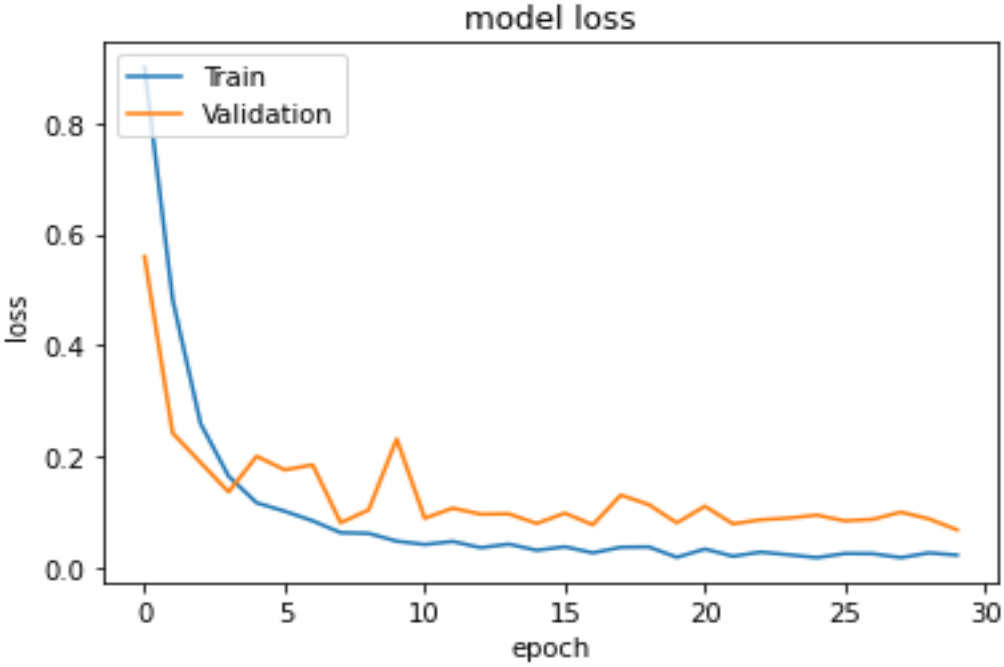


Fig 27: Training and Validation loss for the best model

	precision	recall	f1-score	support
glioma	0.99	0.95	0.97	286
meningioma	0.98	0.97	0.98	306
notumor	0.99	1.00	1.00	405
pituitary	0.96	0.99	0.98	300
accuracy			0.98	1297
macro avg	0.98	0.98	0.98	1297
weighted avg	0.98	0.98	0.98	1297

Fig 28: classification report for the best model

Fifth model VGG19 model:

We have designed our model based on Convolution Neural Network. We have used Conv2D, Maxpool2D, flatten, 4 dense layers with different number of neurons using relu function but in the last layer using softmax for classification and 5 dropout layers for our network. The image size is taken as 224*224.The kernel size has been reduced to 3*3 using dimension reduction feature. Total no of parameters used are 26,538,580. A snapshot image of the model consisting of its layers has shown in fig.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 vgg19 (Functional)          (None, 7, 7, 512)         20024384

 flatten (Flatten)           (None, 25088)             0

 dropout (Dropout)           (None, 25088)             0

 dense (Dense)               (None, 256)               6422784

 dropout_1 (Dropout)         (None, 256)               0

 dense_1 (Dense)             (None, 180)               46260

 dropout_2 (Dropout)         (None, 180)               0

 dense_2 (Dense)             (None, 148)               26788

 dropout_3 (Dropout)         (None, 148)               0

 dense_3 (Dense)             (None, 120)               17880

 dropout_4 (Dropout)         (None, 120)               0

 dense_4 (Dense)             (None, 4)                 484

=================================================================
Total params: 26,538,580
Trainable params: 13,593,620
Non-trainable params: 12,944,960
_________________________________________________________________
```

Fig 29: model architecture

Results before augmentation:

model	Epochs	Preprocessing	Patience(reduceonplateau)	Patience (Early Stopping)	optimizer	LR	Batch size	AVG train Accuracy	AVG validation Accuracy	AVG test Accuracy
VGG19	25	NO (using original data)	4	5	Adam	0.0001	20	99.2%	95.2%	96.3%
VGG19	25	NO	4	4	Adam	0.0001	20	99.1%	95.0%	94.8%

Table 39: VGG19 model results before augmentation

Results before augmentation and after preprocessing:

model	Epochs	Preprocessing	Patience(reduceonplateau)	Patience (Early Stopping)	optimizer	LR	Batch size	AVG train Accuracy	AVG validation Accuracy	AVG test Accuracy
VGG19	25	yes	4	5	Adam	0.0001	20	96.8%	95.9%	92.7%
VGG19	25	yes	4	4	Adam	0.0001	20	99.1%	97.1%	95%
VGG19	25	yes	4	3	Adam	0.0001	20	97%	94.9%	92.4%

Table 40: VGG19 model results before augmentation and after preprocessing

Results after augmentation:

model	Epochs	Preprocessing	Patience(reduceonplateau)	Patience (Early Stopping)	optimizer	LR	Batch size	AVG train Accuracy	AVG validation Accuracy	AVG test Accuracy
VGG19	35	YES	4	5	Adam	0.0001	20	98.9%	96.4%	95.6%
VGG19	40	YES	4	4	Adam	0.0001	20	99.3%	96.8%	97.8%

Table 41: VGG19 model results after augmentation

EVALUATION OF THE PREDICTIVE MODEL PERFORMANCE

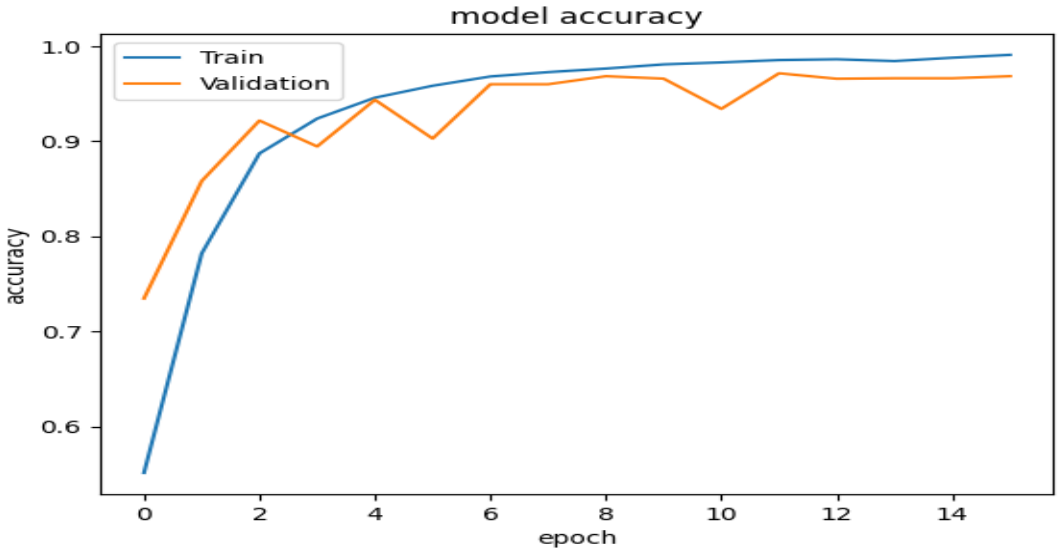


Fig 30: Training and Validation accuracy for the best model

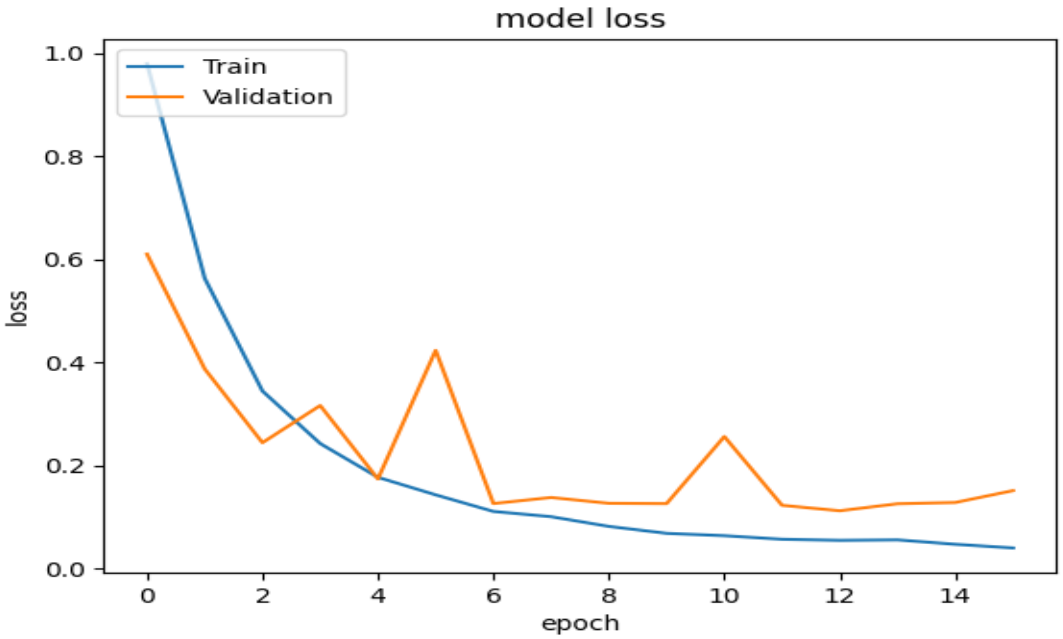


Fig 31: Training and Validation loss for the best model

	precision	recall	f1-score	support
glioma	0.99	0.95	0.97	286
meningioma	0.96	0.98	0.97	306
notumor	0.99	0.99	0.99	405
pituitary	0.98	0.99	0.98	300
accuracy			0.98	1297
macro avg	0.98	0.98	0.98	1297
weighted avg	0.98	0.98	0.98	1297

Fig 32: classification report for the best model

Sixth model DenseNet121:

We have designed our model based on Convolution Neural Network. We have used Conv2D, Maxpool2D, flatten, we retrained some of the DenseNet121 functional layers again, we used 2 dense layers with different number of neurons using relu activation function but in the last layer we use softmax activation function. The image size is taken as 224*224. Total no of parameters used are 7,300,932. A snapshot image of the model consisting of its layers has shown in fig.

```
Model: "sequential_2"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 densenet121 (Functional)    (None, 1024)              7037504

 flatten_2 (Flatten)         (None, 1024)              0

 dense_4 (Dense)             (None, 256)               262400

 dense_5 (Dense)             (None, 4)                 1028

=================================================================
Total params: 7,300,932
Trainable params: 7,217,284
Non-trainable params: 83,648
_________________________________________________________________
```

Fig 33: model architecture

Before augmentation:

Model	Epochs	Preprocessing	Patience(reduceonplateau)	Patience (Early Stopping)	optimizer	Learning Rate (LR)	Batch size	AVG train Accuracy	AVG validation Accuracy	AVG test Accuracy
DenseNet121	10	NO	4	4	Adam	0.0001	32	99.9%	96.6%	98%
DenseNet121	7	NO	4	4	Adadelta	0.01	32	99.7%	93.5%	94%
DenseNet121	12	NO	2	2	RMSprop	0.01	32	79.5%	78.1%	81%

Table 42: DenseNet121 model results before augmentation

Before augmentation and after preprocessing:

Model	Epochs	Preprocessing	Patience(reduceonplateau)	Patience (Early Stopping)	optimizer	Learning Rate (LR)	Batch size	AVG train Accuracy	AVG validation Accuracy	AVG test Accuracy
DenseNet121	10	Yes	4	4	Adam	0.0001	32	100.0%	96.5%	97.3%
DenseNet121	7	Yes	4	4	Adadelta	0.01	32	99.4%	90.9%	92%
DenseNet121	12	Yes	2	2	RMSprop	0.01	16	54.6%	54.2%	53%

Table 43: DenseNet121 model results before augmentation and after preprocessing

After augmentation and after preprocessing:

Model	Epochs	Preprocessing	Patience (reduceonplateau)	Patience (Early Stopping)	optimizer	Learning	Batch size	AVG train	AVG validatio	AVG test

						Rate (LR)		Accuracy	n Accuracy	Accuracy
DenseNet 121	5	Yes	2	2	Adadelta	0.1	32	100.0%	99.6%	99.3%
DenseNet 121	9	Yes	4	4	Adam	0.0001	32	99.32%	98.82%	98%
DenseNet 121	10	Yes	4	4	Adadelta	0.0001	32	57.43%	59.16%	59%

Table 44: DenseNet121 model results after augmentation

EVALUATION OF THE PREDICTIVE MODEL PERFORMANCE

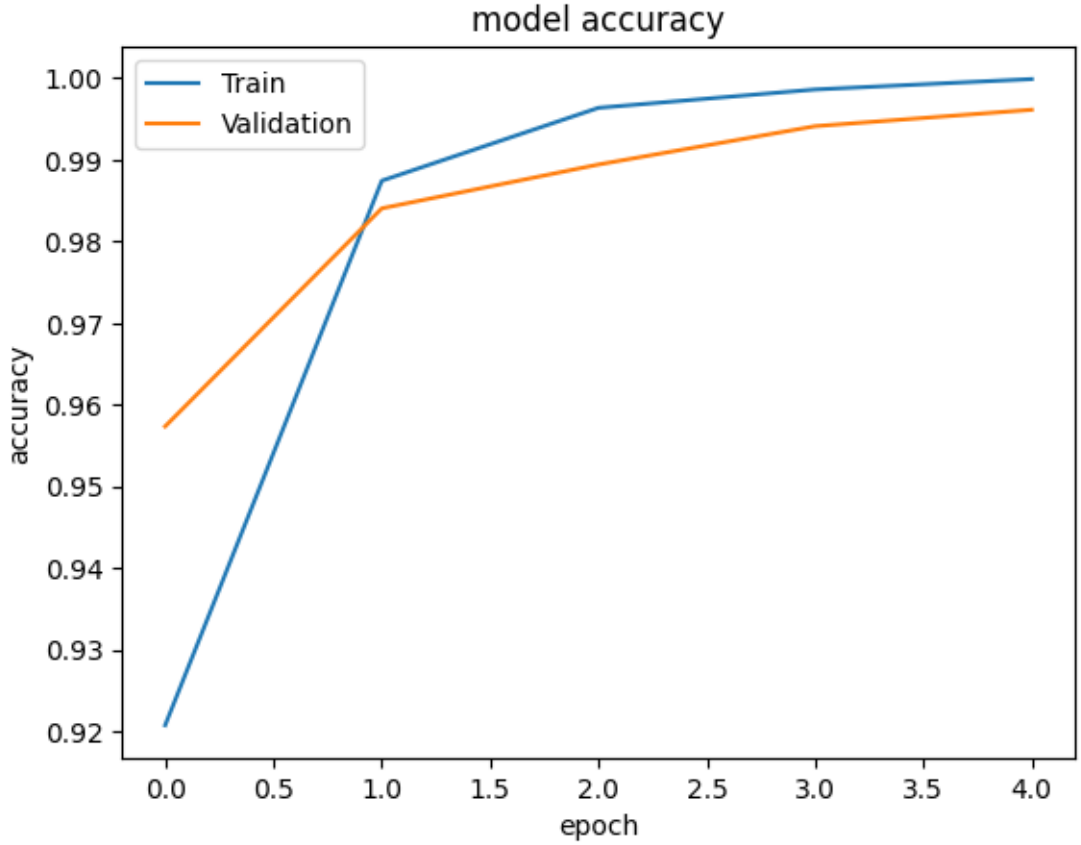


Fig 34: Training and Validation accuracy for the best model

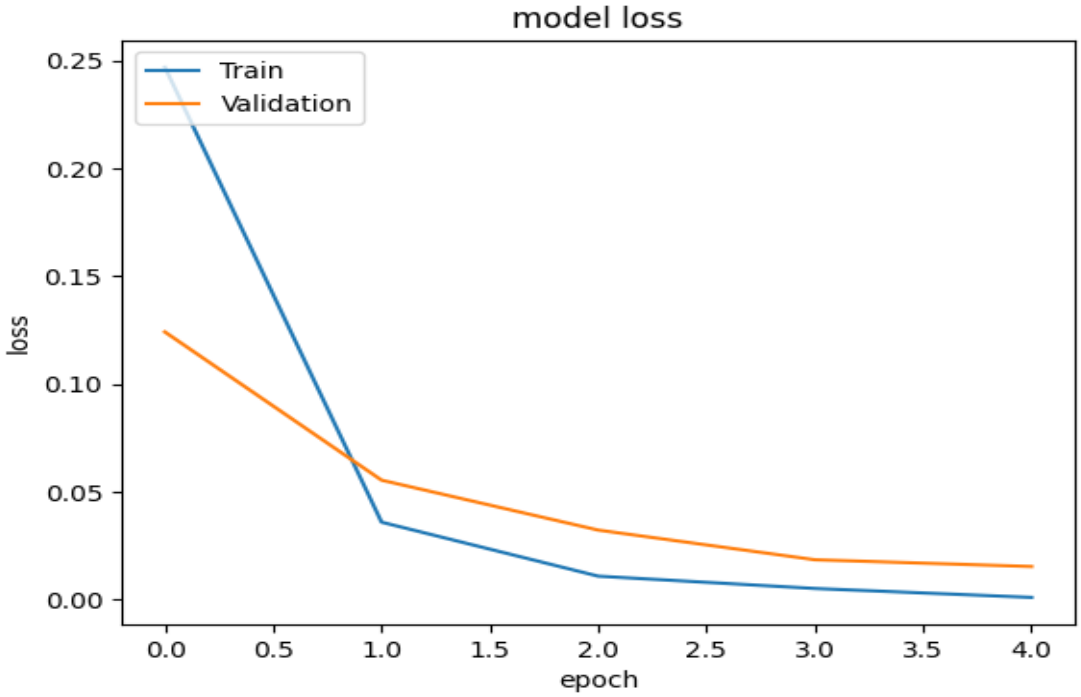


Fig 35: Training and Validation loss for the best model

	precision	recall	f1-score	support
glioma	1.00	0.97	0.99	286
meningioma	0.97	1.00	0.99	306
notumor	1.00	1.00	1.00	405
pituitary	1.00	0.99	0.99	300
accuracy			0.99	1297
macro avg	0.99	0.99	0.99	1297
weighted avg	0.99	0.99	0.99	1297

Fig 36: classification report for the best model

Seventh model resnet50:

We have designed our model based on Convolution Neural Network. We have used Conv2D, Maxpool2D, flatten, we retrained some of the ResNet50 functional layers again, we used 2 dense layers with different number of neurons using relu activation function but in the last layer we use softmax activation function. The image size is taken as 224*224.Total number of parameters used are 24,113,284. A snapshot image of the model consisting of its layer has shown in fig.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 resnet50 (Functional)       (None, 2048)              23587712

 flatten (Flatten)           (None, 2048)              0

 dense (Dense)               (None, 256)               524544

 dense_1 (Dense)             (None, 4)                 1028

=================================================================
Total params: 24,113,284
Trainable params: 24,060,164
Non-trainable params: 53,120
_________________________________________________________________
```

Fig 37: model architecture

Before augmentation:

Model	Epochs	Preprocessing	Patience(reduceon plateau)	Patience (Early Stopping)	optimizer	Learning Rate (LR)	Batch size	AVG train Accuracy	AVG validation Accuracy	AVG test Accuracy
ResNet50	10	NO	4	4	Adam	0.0001	32	98.0%	95.7%	96.2%
ResNet50	7	NO	4	4	Adadelta	0.01	32	93.9%	87.8%	86%
ResNet50	12	NO	2	2	RMSprop	0.01	32	24.3%	25.8%	25%

Table 38: ResNet50 model results before augmentation

Before augmentation and after preprocessing:

Model	Epochs	Preprocessing	Patience(reduceon plateau)	Patience (Early Stopping)	optimizer	Learning Rate (LR)	Batch size	AVG train Accuracy	AVG validation Accuracy	AVG test Accuracy
ResNet50	5	Yes	2	2	Adadelta	0.1	32	99.9%	99.1%	99.0%
ResNet50	6	Yes	4	4	Adadelta	0.01	32	53.7%	51.5%	54.0%
ResNet50	12	Yes	2	2	RMSprop	0.01	16	43.8%	43.5%	43.7%

Table 46: ResNet50 model results before augmentation and after preprocessing

After augmentation and after preprocessing:

Model	Epochs	Preprocessing	Patience(reduce onplateau)	Patience (Early Stopping)	optimizer	Learning Rate (LR)	Batch size	AVG train Accuracy	AVG validation Accuracy	AVG test Accuracy
ResNet50	4	YES	2	2	Adadelta	0.1	16	99.4%	98.4%	98.7%
ResNet50	3	YES	2	2	Adadelta	0.1	16	99.2%	98.2%	98.8%
ResNet50	3	YES	4	4	Adam	0.1	64	47.4%	47.0%	43.6%

Table 47: ResNet50 model results after augmentation

EVALUATION OF THE PREDICTIVE MODEL PERFORMANCE

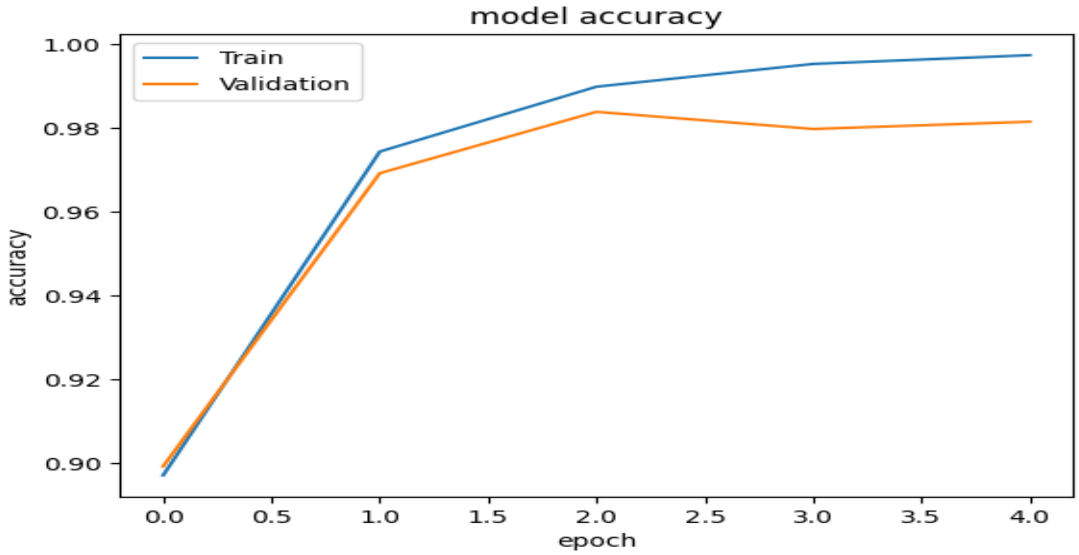


Fig 38: Training and Validation accuracy for the best model

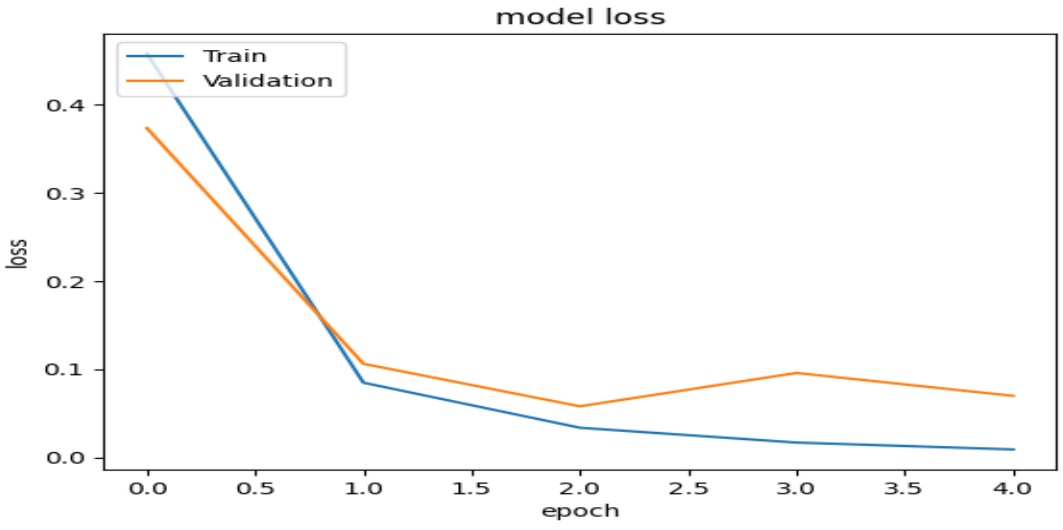


Fig 39: Training and Validation loss for the best model

	precision	recall	f1-score	support
glioma	1.00	0.95	0.97	286
meningioma	0.96	0.99	0.97	306
notumor	1.00	1.00	1.00	405
pituitary	0.98	1.00	0.99	300
accuracy			0.98	1297
macro avg	0.98	0.98	0.98	1297
weighted avg	0.98	0.98	0.98	1297

Fig 40: classification report for the best model

Eighth model ResNet152V2 model:

We have designed our model based on Convolution Neural Network. We have used Conv2D, Maxpool2D, flatten, we retrained some of the ResNet152V2 functional layers again, we used 2 dense layers with different number of neurons using relu activation function but in the last layer we use softmax activation function. The image size is taken as 224*224.Total number of parameters used are 58,857,220. A snapshot image of the model consisting of its layer has shown in fig.

```
Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 resnet152v2 (Functional)    (None, 2048)              58331648

 flatten (Flatten)           (None, 2048)              0

 dense (Dense)               (None, 256)               524544

 dense_1 (Dense)             (None, 4)                 1028

=================================================================
Total params: 58,857,220
Trainable params: 58,713,476
Non-trainable params: 143,744
```

Fig 41: model architecture

Before augmentation

Model	Epochs	Preprocessing	Patience(reduceonplateau)	Patience (Early Stopping)	optimizer	Learning Rate (LR)	Batch size	AVG train Accuracy	AVG validation Accuracy	AVG test Accuracy
ResNet152 V2	10	NO	4	4	Adam	0.0001	32	98.0%	95.7%	96.2%
ResNet152 V2	7	NO	4	4	Adadelta	0.01	32	93.9%	87.8%	86%
ResNet152 V2	12	NO	2	2	RMSprop	0.01	32	24.3%	25.85	25%

Table 48: ResNet152V2 model results before augmentation

Before augmentation and after preprocessing

Model	Epochs	Preprocessing	Patience(reduceonplateau)	Patience (Early Stopping)	optimizer	Learning Rate (LR)	Batch size	AVG train Accuracy	AVG validation Accuracy	AVG test Accuracy
ResNet152 V2	10	Yes	4	4	Adam	0.0001	32	99.9%	96.2%	96.8%
ResNet152 V2	6	Yes	4	4	Adadelta	0.01	32	83.6%	76%	76.7%
ResNet152 V2	12	Yes	2	2	RMSprop	0.01	16	44.6%	44.7%	42.8%

Table 49: ResNet152V2 model results before augmentation and after preprocessing

After augmentation and after preprocessing

Model	Epochs	Preprocessing	Patience(reduceonplateau)	Patience (Early Stopping)	optimizer	Learning Rate (LR)	Batch size	AVG train Accuracy	AVG validation Accuracy	AVG test Accuracy
ResNet152 V2	5	Yes	2	2	Adadelta	0.1	32	99.9%	99.1%	99.0%
ResNet152 V2	9	Yes	4	4	Adam	0.0001	32	99.02%	98.05%	98%
ResNet152 V2	8	Yes	4	4	Adam	0.0001	32	98.94%	98.09%	98.9%
ResNet152 V2	6	Yes	4	4	Adadelta	0.0001	32	98.61%	98.28%	98.8%

Table 50: ResNet152V2 model results after augmentation

EVALUATION OF THE PREDICTIVE MODEL PERFORMANCE

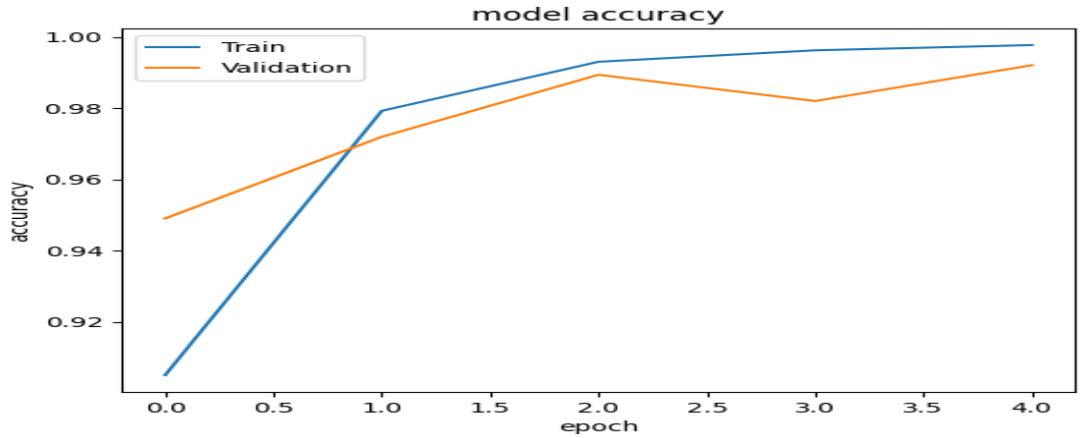


Fig 42: Training and Validation accuracy for the best model

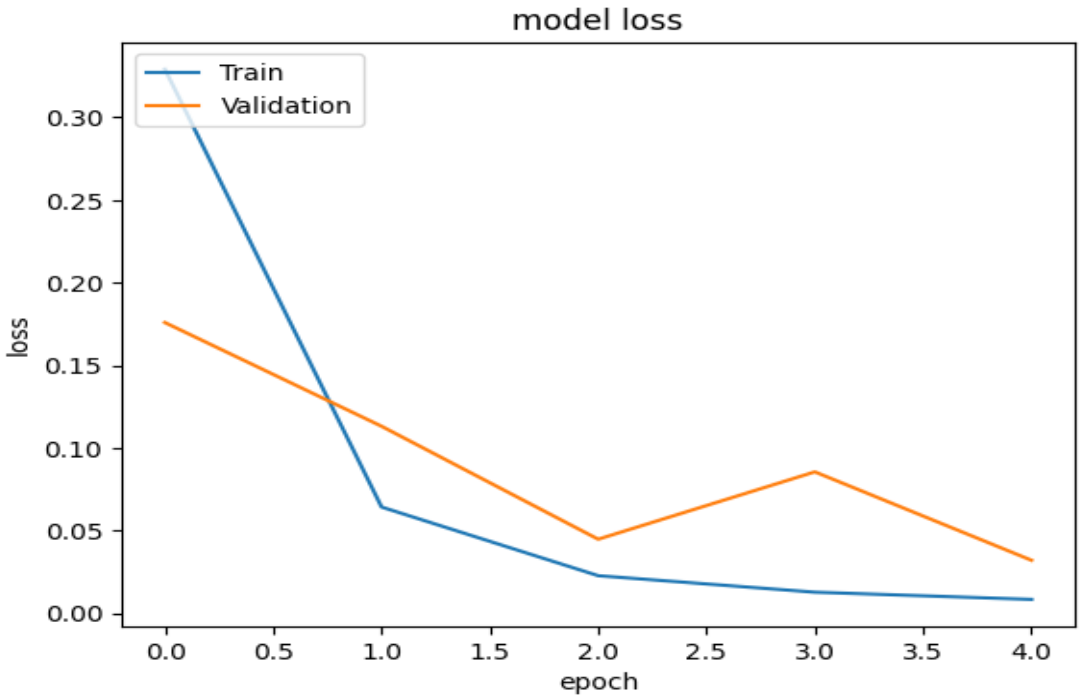


Fig 43: Training and Validation loss for the best model

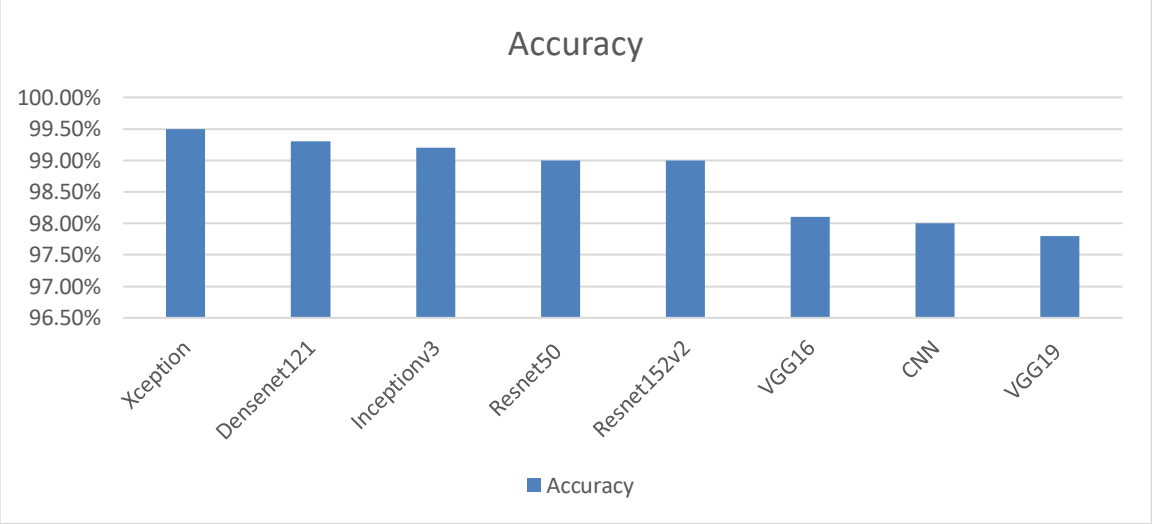
	precision	recall	f1-score	support
glioma	0.99	0.98	0.99	286
meningioma	0.98	0.99	0.98	306
notumor	1.00	1.00	1.00	405
pituitary	0.99	0.99	0.99	300
accuracy			0.99	1297
macro avg	0.99	0.99	0.99	1297
weighted avg	0.99	0.99	0.99	1297

Fig 44: classification report for the best model

Classification Results

Model	Epochs	Preprocessing	Patience (reduceonplateau)	Patience (Early Stopping)	optimizer	Learning Rate (LR)	Batch size	AVG train Accuracy	AVG validation Accuracy	AVG test Accuracy
Xception	50	YES	6	12	Adam	0.0001	12	99.7%	99%	99.5%
DenseNet121	5	YES	2	2	Adadelta	0.1	32	100.0%	99.6%	99.3%
Inceptionv3	50	YES	6	12	Adam	0.0001	20	99.9%	99.1%	99.2%
ResNet50	5	Yes	2	2	Adadelta	0.1	32	99.9%	99.1%	99.0%
ResNet152V2	5	Yes	2	2	Adadelta	0.1	32	99.9%	99.1%	99.0%
VGG16	40	YES	4	10	Adam	0.0001	20	99.9%	98.3%	98.1%
CNN	200	YES	4	4	Adam	0.0001	32	99.4%	95.2%	98%
VGG19	40	YES	4	4	Adam	0.0001	20	99.3%	96.8%	97.8%

Table 51: Classification models results.



5.2 Brain tumor segmentation

(FLOWCHART FOR DESIGN AND DEVELOPMENT OF PROPOSED PROJECT)

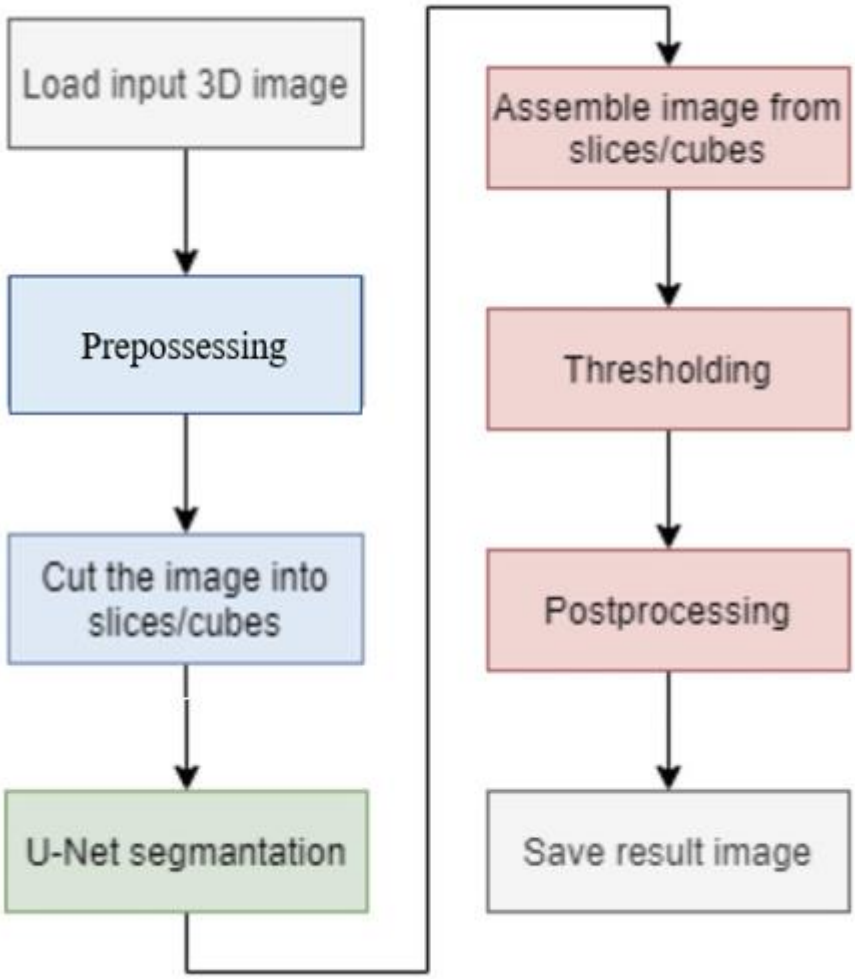


Fig 45: segmentation workflow

5.2.1 Software Requirements:

1. Python [version 3.10.9] - Python is chosen as the programming language for this project due to several reasons:

2. Python code is compact and readable, making it easier to understand and maintain compared to MATLAB.

3. Python offers superior data structures compared to MATLAB, making it more versatile for handling complex data.

4. Python is an open-source language with a large community, providing access to a wide range of graphic packages and data sets.

5. Keras (with TensorFlow backend version 2.12.0) - Keras is a popular neural network API that is built on top of TensorFlow, CNTk, Theano, and other deep learning frameworks.

6. Python packages like NumPy, Matplotlib, Pandas - These packages are used for mathematical computation and plotting graphs in the project.

7. Kaggle - Kaggle was used to obtain the online dataset for the project.

8. GitHub and chatGPT - These platforms were used for reference in case of programming syntax errors and for finding solutions to coding issues.

9. OpenCV (Open-Source Computer Vision) - OpenCV is a popular library of programming functions used for real-time computer vision, including image processing and operations related to images such as reading and writing images, modifying image quality, noise removal using Gaussian Blur, etc. It supports multiple programming languages including C++, Java, C, and Python, and is commonly used in applications like CamScanner, Instagram, and GitHub.

10. Mayavi - Mayavi is a Python package used for scientific data visualization in 3D. It is built on top of the Visualization Toolkit (VTK) and integrates with other scientific Python packages such as NumPy, SciPy, and Matplotlib. Mayavi provides a Pythonic interface for creating and manipulating 3D visualizations and supports a wide range of data formats and visualization techniques. It also provides interactivity and animation capabilities and is widely used in scientific research and engineering applications.

5.2.2 Hardware Requirements:

1-Operating system: windows 11 pro
2-Minimum CPU or processor speed. 2.50 GHz
3-Minimum GPU or video memory.
4-Minimum system memory (RAM) 8 GB
5-Minimum free storage space. 140 GB

6-Audio hardware (sound card, speakers, etc) intel SST Realtek(R) Audio

5.2.3 Methodology

Our proposed methodology is based on the Unet.

The proposed methodology for segmenting brain tumors is as follows:

Step 1: brain tumor segmentation Dataset acquisition

Step 2: preprocessing of the dataset

Step 3: expected results

Step 4: segmentation using Unet model.

Step 5: obtaining models result.

Step 1: Brain tumor segmentation dataset acquisition

Fig 46: snapshot of dataset in Kaggle

All BraTS multimodal [30] scans are available as NIfTI files (. nii.gz)

Four channels of information (4 different volumes of the same region)

- native (T1)
- post-contrast T1-weighted (T1ce)
- T2-weighted (T2)
- T2 Fluid Attenuated Inversion Recovery (FLAIR) volumes

sub-regions considered for evaluation are: 1) the "enhancing tumor" (ET), 2) the "tumor core" (TC), and 3) the "whole tumor" (WT) [see figure below]. The ET is described by areas that show hyper-intensity in T1Gd when compared to T1, but also when compared to “healthy” white matter in T1Gd. The TC describes the bulk of the tumor, which is what is typically resected. The TC entails the ET, as well as the necrotic (fluid-filled) and the non-enhancing (solid) parts of the tumor. The appearance of the necrotic (NCR) and the non-enhancing (NET) tumor core is typically hypo-intense in T1-Gd when compared to T1. The WT describes the complete extent of the disease, as it entails TC and the peritumoral edema (ED), which is typically depicted by hyper-intense signal in FLAIR.

The provided segmentation labels have values of

- For everything else (label 0)
- Necrotic and non-enhancing tumor core (NCR/NET — label 1)
- Peritumoral edema (ED — label 2)
- whole tumor (WT — no pixels in all volumes contain label 3)
- GD-enhancing tumor (ET — label 4)

Step 2: Preprocessing of the dataset

1- fixing missing files and fixing it.
2- Crop unnecessary pixels, cropping x, y, and z to 128x128x128.
3- Reassign mask values 4 to 3.

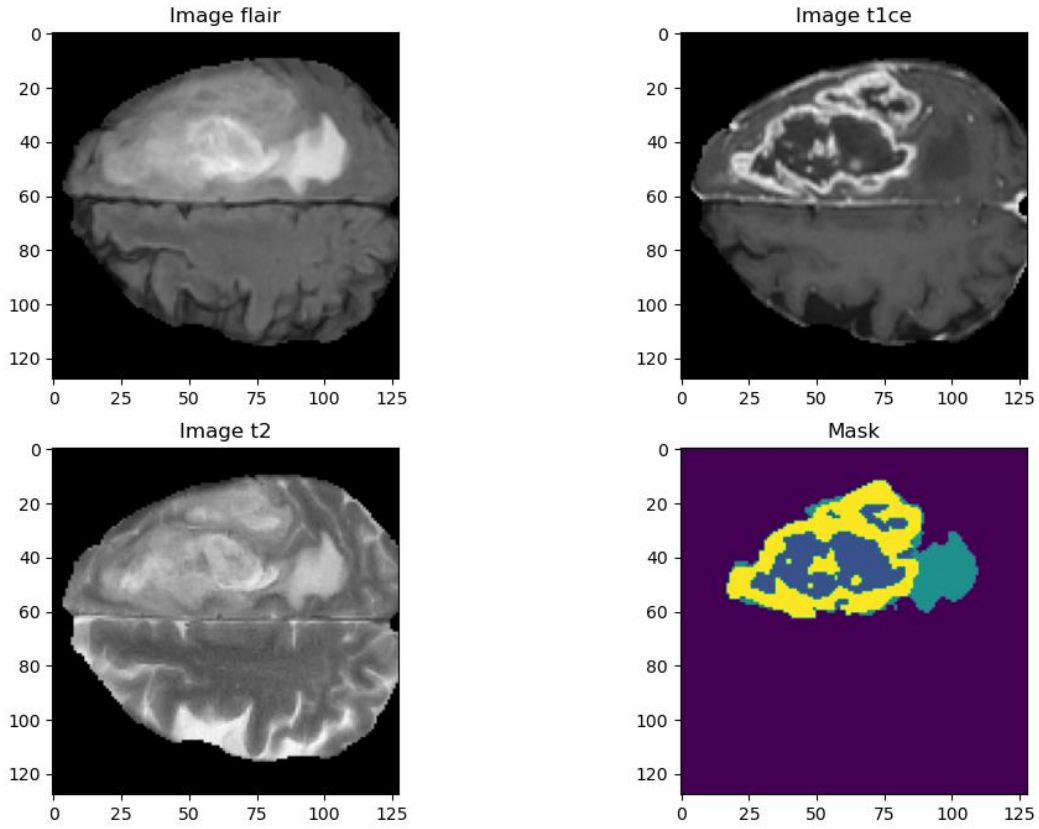


Fig 47: preprocessing technique

Step 3: Expected results.

The expected result of segmenting the BraTS2020 dataset with this U-Net architecture would be a pixel-wise segmentation map of the four classes: background, necrotic and non-enhancing tumor, edema, and enhancing tumor. The model would take a 256x256x3 input image and output a 256x256x4 tensor, where each pixel value corresponds to the probability of that pixel belonging to one of the four classes.

The model architecture consists of a contracting path and an expanding path, where the contracting path applies a series of convolutional and pooling layers to extract features from the input image, and the expanding path applies a series of convolutional and upsampling layers to generate a pixel-wise segmentation map. The model uses ReLU activations, max pooling, and dropout regularization to prevent overfitting, and softmax activation in the last layer to generate class probabilities.

The segmentation performance of the model can be evaluated using metrics such as dice coefficient or intersection over union (IoU) on a test set of BraTS2020 images and their ground truth. for accurate segmentation of brain tumors, the model should achieve high values for the following metrics:

Accuracy: Since the dataset is imbalanced, accuracy is not always a reliable metric for evaluating segmentation performance on the BraTS2020 dataset. However, a model that predicts all pixels as belonging to the majority class (background) would achieve an accuracy of around 0.85. Therefore, a good performance on this metric would be a higher accuracy than this baseline value, but it should not be the primary metric for evaluating the model's performance.

Intersection over Union (IoU): A high IoU value indicates that the predicted segmentation is like the ground truth segmentation for a given class. For example, a model with an IoU of 0.8 or higher for the enhancing tumor class would be considered to have good performance on that class. However, since the dataset is imbalanced, a model that performs well on the majority class (background) could achieve a high overall IoU value. Therefore, it is important to also consider other metrics such as the dice coefficient and the Volumetric Similarity metric to evaluate the model's performance on the different classes and overall.

Dice coefficient: The dice coefficient measures the overlap between the predicted segmentation and the ground truth segmentation. A high dice coefficient value (close to 1) indicates high overlap and accuracy of the segmentation. For example, for a U-Net model trained on the BraTS2020 dataset, a dice coefficient value of 0.6 or higher would be considered a good performance.

Precision: The precision metric measures the proportion of true positive predictions among all positive predictions. A high precision value indicates that the model is accurately predicting the positive examples. a precision value of 0.85 or higher would be considered a good performance.

Sensitivity: The sensitivity metric measures the proportion of true positive predictions among all positive examples in the ground truth. A high sensitivity value indicates that the model is accurately detecting positive examples. a sensitivity value of 0.85 or higher would be considered a good performance.

Specificity: The specificity metric measures the proportion of true negative predictions among all negative examples in the ground truth. A high specificity value indicates that the model is accurately detecting negative examples. a specificity value of 0.85 or higher would be considered a good performance.

Step 4: Segmentation using Unet model.

This model is an implementation of the U-Net architecture for brain tumor segmentation.

introduction about Unet:

Convolutional Networks for Biomedical Image Segmentation

The u-net is convolutional network architecture for fast and precise segmentation of images. Up to now it has outperformed the prior best method (a sliding-window convolutional network) on the ISBI challenge for segmentation of neuronal structures in electron microscopic stacks. It has won the Grand Challenge for Computer-Automated Detection of Caries in Bitewing Radiography at ISBI 2015, and it has won the Cell Tracking Challenge at ISBI 2015 on the two most challenging transmitted light microscopy categories (Phase contrast and DIC microscopy) by a large margin (See also our annoucement).

Here are some details about the layers:

- The input layer is an input image with a shape of (None, 128, 128, 4), where "None" represents a variable batch size, and the shape (128, 128) represents the height and width of the input image in pixels. The "4" in the final dimension represents the number of input channels, which correspond to different MRI modalities.
- The first layer is a convolutional layer with 32 filters and a kernel size of 3x3. This layer takes the input image as input and applies a set of filters to extract features from the input.
- The second layer is another convolutional layer with 32 filters and a kernel size of 3x3. This layer applies another set of filters to the output of the previous layer, further extracting features from the input.
- The third layer is a max pooling layer with a pool size of 2x2. This layer reduces the spatial resolution of the feature maps by a factor of 2, while increasing the number of channels to 32.
- The model then repeats this pattern of convolutional and max pooling layers several times, gradually reducing the spatial resolution of the feature maps while increasing the number of channels.
- After several layers of down-sampling, the model begins to up-sample the feature maps, gradually increasing the spatial resolution while decreasing the number of channels.
- The first up-sampling layer is an up-sampling layer with a size of 2x2. This layer doubles the size of the feature maps in each dimension.
- The model then concatenates the output of the up-sampling layer with the output of the corresponding convolutional layer from the down-sampling path. This is done using the concatenate layer.

- The concatenated feature maps are then passed through two convolutional layers with 256 filters each and a kernel size of 3x3.
- The model repeats this pattern of up-sampling and concatenation several times, gradually increasing the spatial resolution of the feature maps while decreasing the number of channels.
- The final layer is a convolutional layer with 4 filters and a kernel size of 1x1. This layer produces the output of the model, which is a segmentation mask with 4 channels, corresponding to the different tumor classes.

The model has a total of 7,760,484 parameters, all of which are trainable. During training, these parameters are updated using a loss function that compares the predicted segmentation mask to the ground truth segmentation mask. The goal is to minimize the difference between the predicted and ground truth masks, while avoiding overfitting to the training data. A snapshot image of the model consisting of its layers has shown in fig.

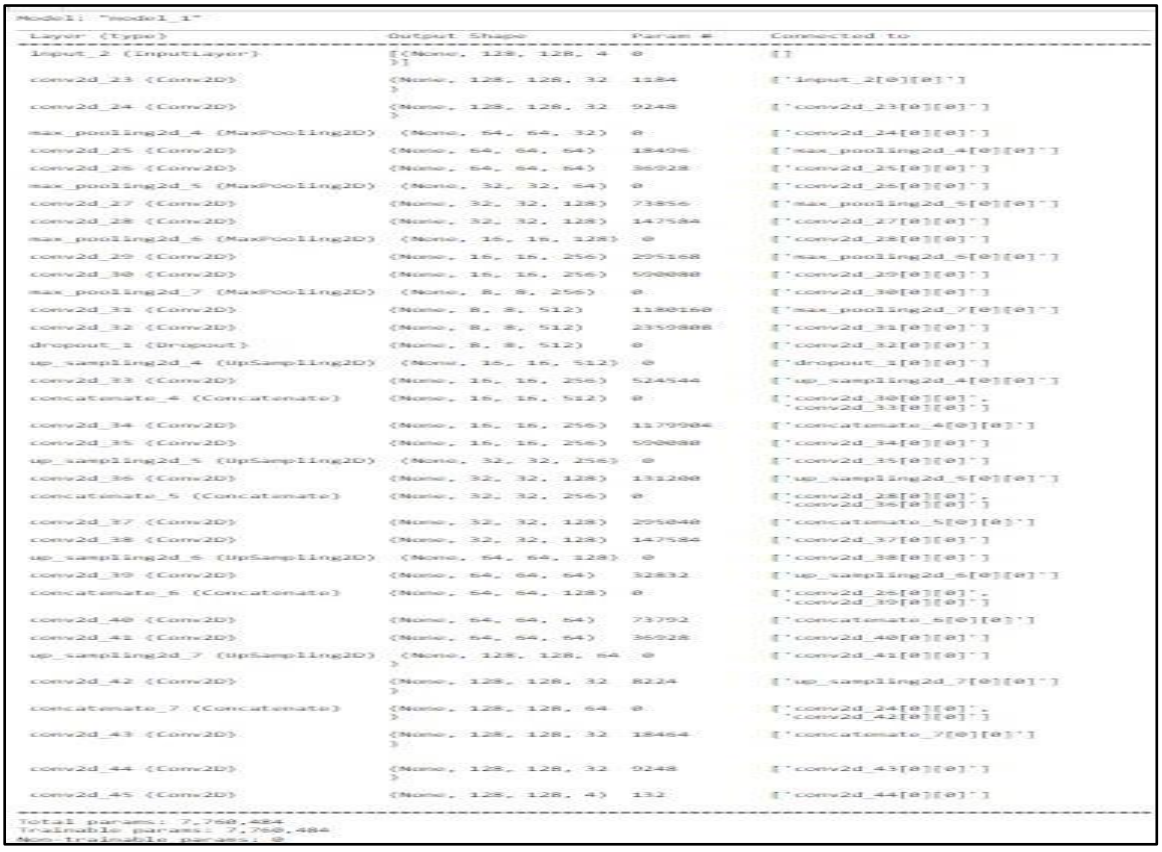


Fig 48 : model architecture

Step 5: Our proposed work results

model	Epochs	Patience (reduceonplateau)	Patience (Early Stopping)	optimizer	Learning Rate (LR)	Batch size	train Accuracy	validation Accuracy	test Accuracy	Train IoU	validation IoU	test IoU
Unet	37	2	2	Adam	0.001	8	0.99464	0.99395	0.99432	0.802342	0.798671	0.800713
Unet	35	2	2	Adam	0.001	8	0.994501	0.992621	0.9944	0.835553	0.831702	0.8003

Table 52: segmentation model results

EVALUATION OF THE PREDICTIVE MODEL PERFORMANCE

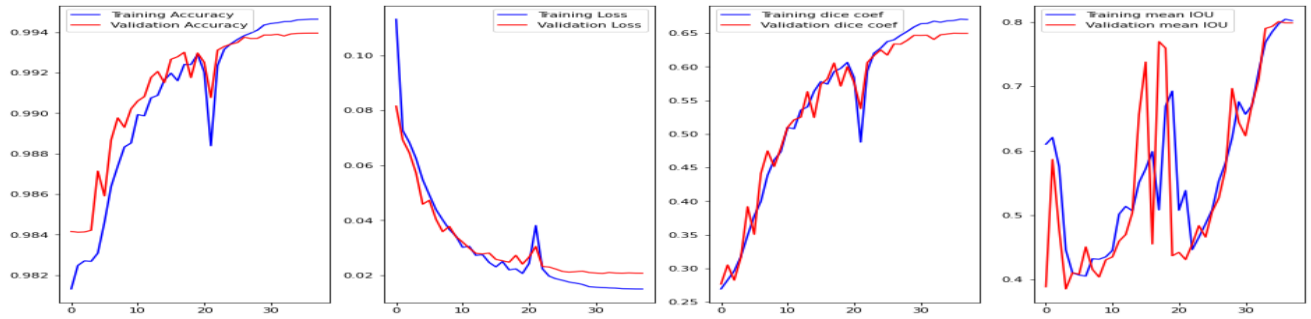


Fig 49: Training and Validation accuracy and loss for the best model

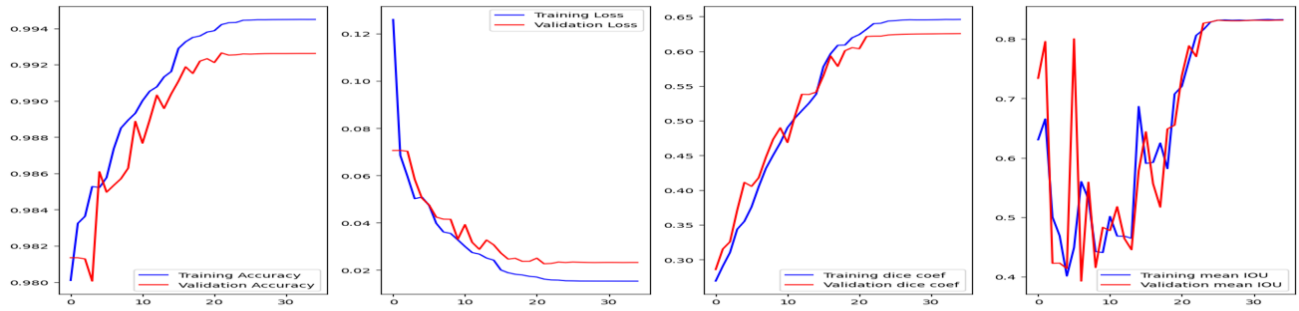


Fig 50: Training and Validation accuracy and loss for the best model after retraining

5.3 Our online medical system

5.3.1 implementation methodology

This documentation will provide an overview of the website's features and functionality, as well as instructions on how to use it.

1- Getting Started To use our website, you will need a web browser that supports HTML, CSS, and JavaScript. Simply navigate to our website.

2- Uploading MRI Images To classify/segment a brain tumor, you will need to upload an MRI image of the patient's brain. You can do this by clicking the "Upload Image" button on the classification/segmentation page and selecting the appropriate image file from your computer. Our online medical system will process the image and display the

results.

3- Tumor Classification/Segmentation Our online medical system uses Flask and deep learning algorithms to classify/segment brain tumors into different categories, such as glioma or meningioma etc. The results of the classification are displayed after the MRI image has been processed. You can view the classification results.

4- The application includes various pages, such as an information page about tumors with tumor distribution and survival data, a treatment and FAQ page, a positive story page about us page, and a memory game for patients. All the pages are built using HTML, CSS, and JavaScript, without any database integration.

5- Server-Side: Since the application does not require a database, server-side development will only involve the Flask framework. Flask will be used to host the application and ensure that the pages are delivered to users accurately and quickly.

6- Deployment: The completed application will be deployed on a web server such as Heroku. This will allow the application to be accessible to users from anywhere with an internet connection.

7- User Interface We have used HTML, CSS, Bootstrap and Charts.js to design the user interface of the website, making it easy to navigate and use. The online medical system is responsive and mobile-friendly, allowing users to access it from any device.

Chapter six: conclusion and future work

6.1 Conclusion

In this project, we developed deep learning models like (CNN, vgg16, vgg19, etc.) for the classification of brain tumors using magnetic resonance images. And used a model like (U-NET) for the segmentation task. The models achieved high accuracy in classification (99.5%) and in segmentation (IoU 80%) and demonstrated the potential for accurate and efficient diagnosis and segmentation of brain tumors.

Through the utilization of a large and diverse dataset, we successfully trained the models to accurately classify different types of brain tumors. Additionally, we incorporated advanced deep learning architectures, enabling us to achieve remarkable accuracy with a relatively small number of training epochs.

The outcomes of this project carry significant implications for the diagnosis and treatment of brain tumors. By providing accurate and efficient classification and segmentation of brain tumors, our models hold the potential to improve patient outcomes and reduce healthcare costs. Moreover, by integrating these models into clinical decision support systems, we can offer valuable assistance to clinicians in diagnosing and treating brain tumors.

While the current project has fulfilled its objectives, there remain several promising directions for future work that can further enhance our findings and expand the project's scope. These include increasing the size and diversity of the dataset to improve generalization, exploring alternative deep learning architectures specifically tailored for segmentation tasks, and investigating the interpretability of the models to gain insights into their decision-making processes.

6.2 Future work

1-Increasing the size and diversity of the dataset: While the current project used a large and diverse dataset, it is possible to further expand the dataset to include more cases or different types of data (such as genomic data or clinical data) that could help improve the accuracy of both the classification and segmentation models.

2-Exploring different deep learning architectures: While the current project used a state-of-the-art deep learning architecture for classification, it is essential to explore and develop specialized architectures for segmentation tasks. Investigating and adapting architectures like U-Net, Mask R-CNN, or DeepLab can potentially enhance the accuracy of the segmentation model.

3-Investigating the interpretability of the models: While the current project achieved high accuracy in both classification and segmentation of brain tumor images, it is important to delve into the interpretability of the models. Understanding how the models make decisions and identifying potential biases or errors can provide valuable insights for refining the algorithms and improving overall performance.

4-Incorporating multi-modal data fusion: In future work, consider integrating multiple imaging

modalities (such as MRI, CT, PET) to leverage their complementary information. Developing fusion techniques that combine the strengths of each modality can potentially enhance the accuracy and robustness of both classification and segmentation tasks.

5-Incorporating clinical decision support systems: Future work could focus on integrating both the classification and segmentation models into clinical decision support systems. These systems can assist clinicians in accurately diagnosing and treating brain tumors by providing automated, accurate, and real-time predictions, as well as generating informative visualizations of tumor segmentations.

References

[1] Karlin Kelley, "what is artificial intelligence: types, history and future", March 9, 2023.

[2] Kiprono Elijah koech, "the basics of neural networks (neural network series) – part 1 ", May 3, 2022.

[3] Nagesh Singh Chauhan, "introduction to convolutional neural networks", June 3, 2022

[4] Muktha Sai Ajay, "introduction to transfer learning", Aug 4, 2020

[5] mayo foundation for medical education and research

[6] J. Seetha1, et al, "Brain Tumor Classification Using Convolutional Neural Networks " Biomedical and pharmacology journal

**[7] lizhi sun," Towards Reinforced Brain Tumor Segmentation on MRI Images Based on Temperature Changes on Pathologic Area
", 03 March 2019, international journal of biomedical imaging.**

[8] Haq, A.u., Li, J.P., Khan, S. *et al.* DACBT: deep learning approach for classification of brain tumors using MRI data in IoT healthcare environment. *Sci Rep* 12, 15331 (2022).

[9] Md. Saikat Islam Khan, Anichur Rahman et al. "accurate brain tumor detection using deep convolutional neural networks" , national library of medicine, 2022 Aug 27

**[10] Ullah, N.; Khan, J.A.; Khan, M.S.; Khan, W.; Hassan, I.; Obayya, M.; Negm, N.; Salama, A.S. An Effective Approach to Detect and Identify Brain Tumors Using Transfer Learning. Appl. Sci. 2022, 12, 5645.
https://doi.org/10.3390/ app12115645**

[11]- Sartaj, B.; Ankita, K.; Prajakta, B.; Sameer, D.; Swati, K. Brain Tumor Classification (MRI). Kaggle 2020. [CrossRef]

[12]- Abdul Hannan Khan, Sagheer Abbas, Muhammad Adnan Khan, Umer Farooq, Wasim Ahmad Khan, Shahan Yamin Siddiqui, Aiesha Ahmad, "Intelligent Model for Brain Tumor Identification Using Deep Learning", *Applied Computational Intelligence and Soft Computing*, vol. 2022, Article ID 8104054, 10 pages, 2022.

[13] Hossam H. Sultan, Nancy M. Salem, Walid Al-Atabany. "Multi-classification of brain tumor images using deep neural network," IEEE Access, 2019

[14] J. Cheng, et al., "Enhanced performance of brain tumor classification via tumor region augmentation and partition," PloS one, vol. 10, no. 12, Oct. 2015.

[15] J. S. Paul, A. J. Plassard, B. A. Landman, and D. Fabb, "Deep learning for brain tumor classification,"' Proc. SPIE, Med. Imag., Biomed. Appl. Mol., Struct., Funct. Imag., vol. 10137, Mar. 2017.

[16] P. Afshar, et al., "Capsule networks for brain tumor classification based on MRI images and course tumor boundaries," 2018. [Online].

[17] Amin Kabir Anaraki, Moosa Ayati, Foad Kazemi, "Magnetic resonance imaging-based brain tumor grades classification and grading via convolutional neural networks and genetic algorithms," Biocybernetics and Biomedical Engineering, vol. 39, no. 1, pp. 63-74, 2019

[18] Z. N. K Swati, et al., "Brain tumor classification for MR images using transfer learning and fine-tuning," Computerized Medical Imaging and Graphics, vol. 75, pp. 34-46, 2019.

[19] M. Talo, et al., "Convolutional neural networks for multi-class brain disease detection using MRI images," Computerized Medical Imaging and Graphics, vol. 78, Dec. 2019.

[20] Abhishekh sawner, Vandana roy, sarang Kapoor et al., "4-way classification of brain tumor using shallow convolution neural network", November 23, 2021

[21] Osman ozkaraca, okan ihsan bagriacik, et al.," multiple brain tumor classification with dense CNN architecture using brain MRI images", MDPI open access journal, 28 January 2023.

[22] K. Alhajeri et al., "Brain Tumor Segmentation Using a Hybrid Deep Convolutional Neural Network", International Journal of Intelligent Computing and Cybernetics in 2021

[23] Y. Wu et al.," A Multimodal Deep Learning Approach with U-Net for Brain Tumor Segmentation", Frontiers in Neuroscience in 2021

[24] Naeem Ullah, et al, "Overview of the proposed method for brain tumor classification."

[25] Mohamed A. Naser, et al, "segmentation model flowchart"

[26] ResearchGate," Proposed block diagram of brain tumor classification system"

[27] Brain Tumor MRI Dataset, Kaggle

[28] Brain Cancer - C3 , Kaggle

[29] Jason Brownlee, "how to update neural network models with more data", March 2021.

[30] Brain Tumor Segmentation (BraTS2020), Kaggle