D.Y.PATIL COLLEGE OF ENGINEERING AND TECHNOLOGY

KASABA BAWADA, KOLHAPUR

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

A

Project Report

" Classification of Brain Lesion and Grade Using MRI Texture and Shape in Machine Learning Scheme "

Submitted by

Sr no.	Name	Roll no
1	Jeet Aryan	167
2	Shivam Rajesh Mandlik	168
3	Vivek Vishwanath Patil	169

Under the guidance of

Prof. Miss. A. R. Chougale

Group no. : 36 **Class : B.E** **Div : B**

Year: 2021-2022 Sem-VIII

D. Y. PATIL COLLEGE OF ENGINEERING AND TECHNOLOGY,

KASABA BAWADA, KOLHAPUR

DEPARTMENT OF COMPUTER SCIENCE AND ENGINEERING

CERTIFICATE

This is to certify that the project group consisting of following members have satisfactorily completed the project-I work entitled "**Classification of Brain Lesion and Grade Using MRI Texture and Shape in Machine Learning Scheme**" at BE (CSE) Semester – VII as prescribed in the syllabus of Shivaji University for the academic year 2021-2022.

Sr no.	Name	Roll no
1	Jeet Aryan	167
2	Shivam Rajesh Mandlik	168
3	Vivek Vishwanath Patil	169

Prof. A.R. Chougale	Prof. M. A. Pardesi	Prof. G. A. Patil
Project Guide	**Project Coordinator**	**HOD**

External Examiner

Prof. S. D. Chede
Principal

Date : _____

Place : Kolhapur

ACKNOWLEDGEMENT

The satisfaction and euphoria that accompany the successful completion of any task would be incomplete without the mention of the people who made it possible. Success is the epitome of hard work, perseverance and most of all those guidance and encouragement crowned our efforts and success.

We owe a debt of thanks to our guide Prof. A.R. Chougale, who stood as a backbone to our project, having worked meticulously all through with special vigilance, zeal and criticism. This contribution to the project is unbounded and mere words are not enough to express our deepest sense of gratitude.

We thank Prof. G. A. Patil, Head of the Department and Prof. M. A. Pardesi Project Coordinator for their constant encouragement throughout the year. Their advice and co-operation in the completion of project is really unforgettable. We are really indebted to them.

We are also grateful to all teaching and non- teaching staff of our department who helped directly or indirectly for successful completion of

Sr no.	Name	Sign
1	Jeet Aryan	_____
2	Shivam Rajesh Mandlik	_____
3	Vivek Vishwanath Patil	_____

INDEX

1. Signed Copy of Synopsis

Group Number: G36

Roll No.	Name	Sign
167	Jeet Aryan	
168	Shivam Rajesh Mandlik	
169	Vivek Vishwanath Patil	

Date: / /

Place: Kolhapur

Prof. Miss A.R.Chougale	Prof. M.A.Pardesi	Prof.Dr. G.A.Patil
(Guide)	(Project Co-ordinator)	(HOD)

2. Introduction :

2.1. Description :

A brain tumor is collection of the abnormal cells in the brain. A tumor may lead to cancer, which is a major leading cause of death. The cancer rate is growing at an alarming rate in the world, nearly 11% of people are died every year due to the cancer. The abnormal growth of cells and division of that cells in the brain may lead to brain tumor, and the further growth of brain tumors may leads to brain cancer. So, detection of the tumor is very important in their earlier stages. Great knowledge and experience on radiology are require for accurate tumor detection in medical imaging processing. The brain tumor is depending upon the combination of factors like the type of tumor, its position, its size, Grade of tumor and its state of growth.

This project deals with such a system, which uses computer and mobile based procedures to detect the tumor parts and classify the type and grade of tumor using Convolution Neural Network (CNN) Algorithm for MRI images of different patients. Different types of image processing techniques like image segmentation technique, image enhancement technique and feature extraction technique which are used for the brain tumor detection in the MRI images of the cancer- affected patients.

Based on our machine, we will predict whether there is any brain tumor or not. The resultant outcomes will be examined through various performance examined metrics that include accuracy, sensitivity, and specificity, etc. Detecting the Brain tumor using various Image Processing techniques that are involves the four stages which are Image Pre-Processing, Image segmentation, Feature Extraction, and Classification. Image processing and neural network techniques are used for improve the performance of detecting and classifying brain tumor in MRI images.

2.2. Need of Work & Problem Statement :

2.2.1. Need of the Work :

A brain tumor is defined as abnormal growth of cells within the brain or the central spinal canal. Some tumors can be cancerous so they need to be detected and cured in time. The exact cause of brain tumors is not clear and neither is exact set of symptoms defined, thus, so many peoples may be suffering from cancer without realizing the danger. Primary brain tumors can be either malignant (contain cancer cells) or benign (do not contain cancer cells).

Determining the right type of brain lesion manually requires an expert who has a good knowledge in brain diseases. Instead of going to doctor in physical mode we can go through online mode and save our time.

Determining and Detection is time consuming and tedious process for a lot of images on manually. Human errors are possible and consequently false detection may cause a wrong procedure and treatment.

2.2.2. Problem Statement :

Problem Statement for our Project :

To design and implement the system which will predict the Brain Abnormalities using MRI images and Detection and Classify Grade through Machine Learning Techniques.

2.3. Objectives / Goals of Project :

The aim of our project is tumor identification in brain MR images. The main reason for detection of brain tumors is to provide aid to clinical diagnosis.

The aim is to provide an algorithm that guarantees the detectuing presence of a tumor by combining several procedures to provide a convenient method of tumor detection in MR brain images. The methods utilized are filtering, erosion, dilation, threshold, and outlining of the tumor such as edge detection.

The focus of this project is MRI brain images tumor extraction and its representation in simpler form such that it is understandable by everyone. The objective of this work is to bring some useful information in simpler form in front of the users, especially for the medical staff treating the patient.

The main aim of this work is to define an algorithm that will result in extracted image of the tumor from the MRI brain images. The resultant image will be able to provide information like size, dimension and position of the tumor, and its boundary provides us with information related to the tumor that can prove useful for various cases, which will provide a better base to decide the procedure.

A computer-assisted classification method combining conventional MRI and perfusion MRI is developed and used for differential diagnosis.

The proposed scheme consists of several steps including region-of interest, definition, feature extraction, feature selection, and classification. The extracted features include lesion shape and intensity characteristics, as well as rotation invariant texture features, etc.

3. Requirement Analysis & Specification :

3.1. Information Gathering :

We referred journals as our main research data source. We initially searched using key words such as, “Detection and Classification of Brain Lesions” and “Detection of Brain Tumor”; this approach returned many empirical studies related to tumor detection in related systems. We then filtered the items using terms such as, “Brain Tumor Types” and, “Tumor Grades” studies and also defined our search boundaries as research conducted. When it comes to “ Brain cancer Detection,”

we hypothesized that every System or methods carried out in the medical management would not be in trusted form i.e Erroers can be Happened from the humans, Doctors also. We found some of the algorithms that are effective in finding the Tumor from MRI images and can predict using the balanced dataset which is used to train the machine first. So for getting the dataset to train the module or machine we referred to the “Kaggle” site that provides various datasets that are helpful for the projects which uses Machin Learning algorithms to train the machine accordingly.

3.2. Project Literature Survey & Feasibility Study :

In Medical diagnosis, robustness as well as accuracy of the prediction algorithms are very important, because the result is crucial and important for the treatment of patients. There are many classification and clustering algorithms that are used for prediction. The goal of clustering is the medical image is to simplify i.e the representation of an image into a meaningful image and make it easier to analyze. Several Clustering and Classification algorithms are aimed at enhancing the prediction accuracy of diagnosis process in detecting abnormalities.

In the literature survey we provide a brief summary of the different methods that have been proposed for clustering over the period of 2002 to 2018.We have been though 25 papers each of which has a unique approach towards segmentation in some parameter or the other. The summaries of each of the papers are provided below.

Sr. no.	Title	Litrature Review
1	"A Novel Based Approach for Extraction Of Brain Tumor In MRI Images Using Soft Computing Techniques," International Journal Of Advanced Research In Computer And Communication Engineering, Vol. 2, Issue 4, April 2013.	A.Sivaramakrishna et al. (2013) [1] projected an efficient and innovative discovery of the brain tumor vicinity from an image that turned into finished using the Fuzzy Capproach grouping algorithm and histogram equalization.
2	B.Sathya and R.Manavalan, Image Segmentation by Clustering Methods: Performance Analysis, International Journal of Computer Applications (0975 – 8887) Volume 29– No.11, September 2011.	Sathya et al. (2011) [3], provided a different clustering algorithm such as K-means, Improvised K-means, C-means, and improvised C-means algorithms. Their paper presented an experimental analysis for massive datasets consisting of unique photographs. They analyzed the discovered consequences using numerous parametric tests.
3	K. Sudharani, T. C. Sarma and K. Satya Rasad, "Intelligent Brain Tumor lesion classification and identification from MRI images usinga K-NN technique," 2015 International Conference on Control, Instrumentation, Communication andComputational Technologies (ICCICCT), Kumaracoil, 2015, pp. 777-780. DOI: 10.1109/ICCICCT.2015.7475384	K. Sudharani et al. [5] presented a K- nearest neighbor algorithm to the MR images to identify and confine the hysterically full-fledged part within the abnormal tissues. The proposed work is a sluggish methodology but produces exquisite effects. The accuracy relies upon the sample training phase.
4	Survey on Brain Tumor Detection Techniques Using Magnetic Resonance Images.	The brain tumor is an abnormal growth of cells inside the skull which causes damage to the other cells necessary for functioning human brain. Brain tumor detection is a challenging task due to the complex structure of the human brain. MRI images generated from MRI scanners using strong magnetic fields and radio waves to form images of the body which helps for medical diagnosis. This paper gives an overview of the various techniques

		used to detect the tumor in the human brain usingMRI
5	Detection of Tumor in MRI ImagesUsing Artificial Neural Networks	Automatic defects detection in MR images is very important in many diagnostic and therapeutic applications. This work has introduced one automatic brain tumor detection method to increase the accuracy and yield and decrease the diagnosis time. The goal is classifying the tissues into two classes of normal and abnormal. MR images that have been used here are MR images from normal and abnormal brain tissues. This method uses from neural network to do this classification. The purpose of this project is to classify the brain tissues to normal and abnormal classes automatically, which saves the radiologist time, increases accuracy and yield of diagnosis.
6	A Neural Network-based Method for Brain Abnormality Detection in MR Images Using Gabor Wavelets.	Nowadays, automatic defects detection in MR images is very important in many diagnostic and therapeutic applications. This paper introduces a Novel automatic brain tumor detection method that uses T1, T2_weighted and PD, MR images to determine any abnormality in brain tissues. Here, ithas been tried to give a clear description from brain tissues using Gabor wavelets, energy, entropy, contrast and some other statistic features such as mean, median, variance, correlation, values of maximum and minimum intensity. It is used from a feature selection method to reduce the feature space too. this method uses from neural network to do this classification. The purpose of this project is to classify the brain tissues to normaland abnormal classes automatically, which saves the radiologist time, increases accuracy .

3.2.1 Economical Feasibility :

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research & development of the system is limited. The expenditures must be justified. Thus the developed system as well within the budget and this was achieved because most of the technologies used are freely available.

3.2.2. Technical Feasibility :

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

3.2.3. Social Feasibility :

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

3.3 Project Life Cycle Model :

This provides the architecture of the system that would be developed by our hands. It consists of six steps where the execution starts from taking an input image from the data set followed by the image pre-processing, image enhancement, Image segmentation using binary thresholding and the brain tumor classification using Convolutional Neural Network. Finally, the output is observed after all the above mentioned steps are completed.

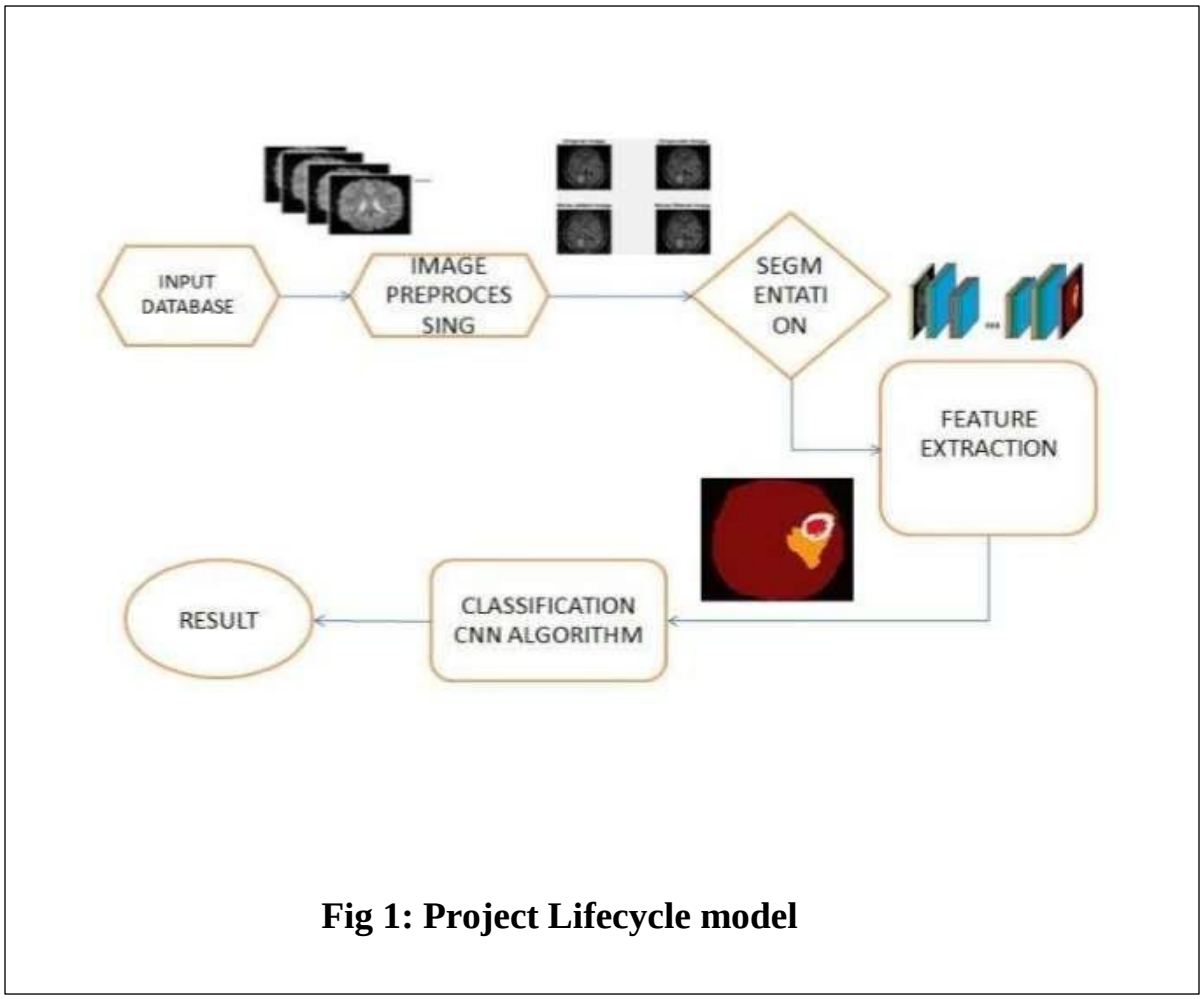


Fig 1: Project Lifecycle model

4. Software Requirement Specification (SRS) :

4.1. Introduction :

This document lays a project plan for the development of the "**Classification of Brain Lesion and Grade Using MRI Texture and Shape in Machine Learning Scheme** " The plan will include a summary of the system functionality, the scope of the project from the perspective of the Detection of Lesions Online, Types of Lesions, the process by which the project will be developed and the metrics and measurements that will be recorded while the project.

4.2. Overview :

The focus of this project is MR brain images tumor extraction and its representation in simpler form such that it is understandable by everyone. The objective of this work is to bring some useful information in simpler form in front of the users, especially for the medical staff treating the patient. The aim of this work is to define an algorithm that will result in extracted image of the tumor from the MR brain image. The resultant image will be able to provide information like size, dimension and position of the tumor, and its boundary provides us with information related to the tumor that can prove useful for various cases, which will provide a better base for the staff to decide the curing procedure. Finally, we detect whether the given MR brain image has tumor or not using Convolution Neural Network.

4.3. Scope:

Our aim is to develop an automated system for enhancement, segmentation and classification of brain tumors. The system can be used by neurosurgeons and healthcare specialists. The system incorporates image processing, pattern analysis, and computer vision techniques and is expected to improve the sensitivity, specificity, and efficiency of brain tumor screening.

The primary goal of medical imaging projects is to extract meaningful and accurate information from these images with the least error possible. The proper combination and parameterization of the phases enables the development of adjunct tools that can help on the early diagnosis or the monitoring of the tumor identification and locations.

4.4. Performance measures:

The proposed algorithm has been assessed through various performance evaluation metrics that include True Positive, True Negative the former one that designates how many times does the proposed algorithm is able to correctly recognize the damaged region as damaged region and the later one designates how many times does the proposed algorithm correctly identified non-damaged region as non-damaged region. And the False Positive (FN) and False Negative (FN) the former one designates how many times does the proposed algorithm fails to recognize the damaged region correctly, and the later represents how many times does the proposed algorithm fails to identify the non-tumors region as non-tumors regions. Basing on values of TP, TN, FP, and FN, the values of Accuracy, Specificity and sensitivity are calculated of the proposed algorithm.

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

$$Specificity = \frac{TN}{TN + FP}$$

$$Sensitivity = \frac{TN}{TP+FN}$$

4.4. Functionality :

- Firstly, the input MRI Image will be take as a input to our system.
- These MRI image will contain some noise or extra details which will find out and reduce that noise in image preprocessing stage. The RGB MRI image is converted to gray scale image and the median filter is applied to brain MR images for noise removal. The noise has to be removed for further processing as high accuracy is required.
- The edges are detected from filtered image using canny edge detection. The detected image of the edges are required for segmentation of the image.

- Further, Watershed segmentation is performed for finding the location of the tumor in the brain image
- The segmented brain MRI image is used and texture features are extracted from the segmented image which shows the texture property of the image.
- These features are extracted using Gray Level Co-occurrence Matrix (GLCM) as it is robust method with high performance.
- Energy, Contrast, homogeneity, corealation, These features are used to differentiate between normal and abnormal brain.

4.5. Platform :

The system will be launched in the Android Application form so that the user can access it any time through any Android System as per the user preference and won‟t need much specs or requirement for the software.

4.6. Goals and Scopes :

To capture MRI images to analyse the presence of tumor using machine learning techniques.

To perform segmentation process on MRI images to separate the tumor from normal brain tissues.

To extract features from captured images to find the tumor.

To detect the presence of tumor.

4.7. Scheduling and Estimates :

Sr. no.	Activity	Completion
1	Information Search About Project	October
2	Finalization of Project Topics	November
3	Literature Survey	December
4	Planning	December
5	Implementation (Initial Step)	December
6	Implementation of Module	December
7	Implementation of Module	January
8	Configuration	February
9	Deployment	March
10	Project Report	April

4.8. Technical Specification

4.8.1. Software Requirements

Windows: Python 3.6.2 or above,
PIP and NumPy 1.13.1, OpenCV (Open source computer vision), Keras, Tensor Flow,
Jupyter Notebook Software: Android Studio.

4.8.2. Hardware Requirements:

Processor: Intel core i5 or above. 64-bit, quad-core, 2.5 GHz minimum per core
Ram: 4 GB or more
Hard disk: 10 GB of available space or more.
Display: Dual XGA (1024 x 768) or higher resolution monitors
Operating system: Windows 10,11

5. Design :

5.1. Modular Structure :

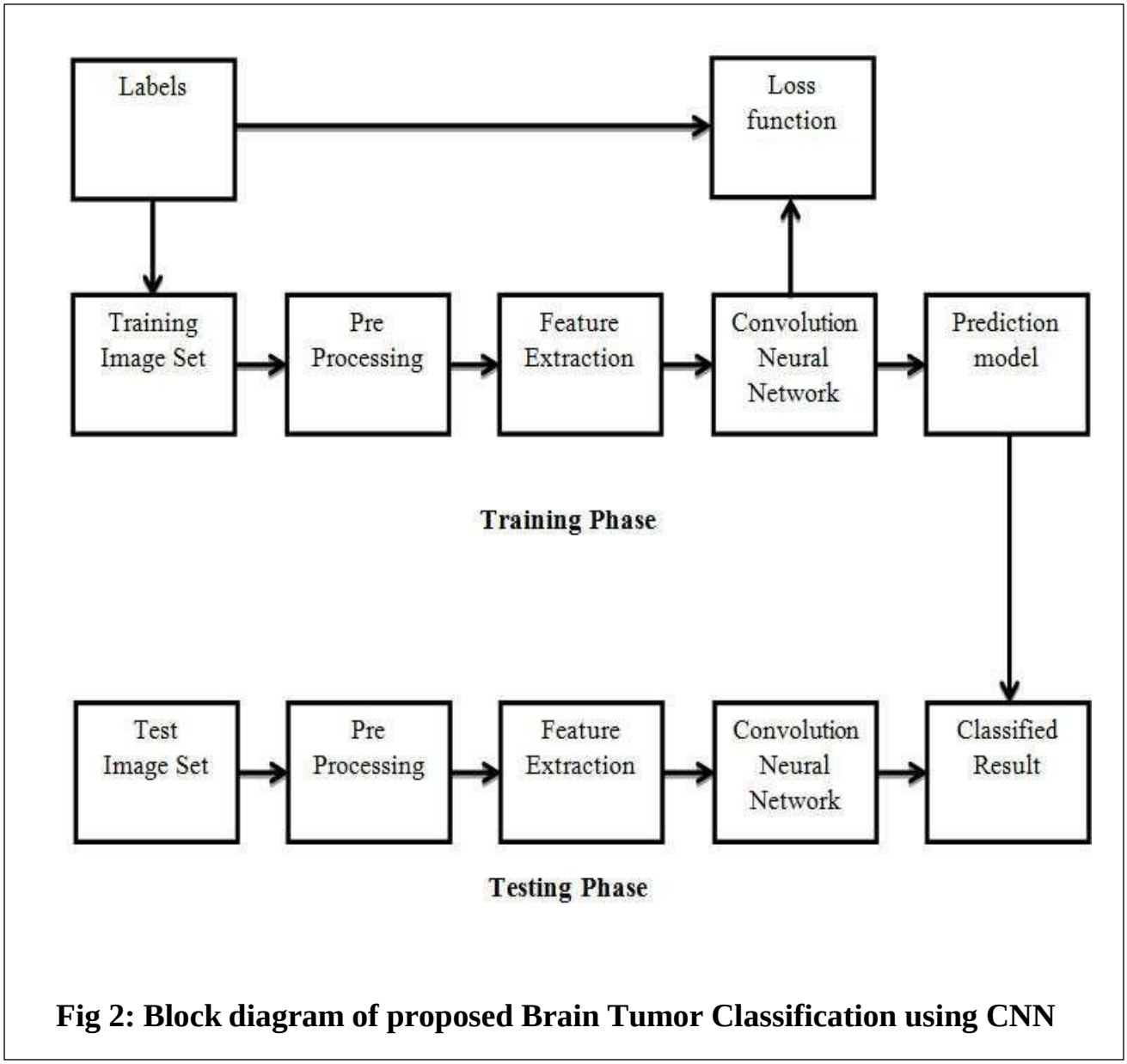


Fig 2: Block diagram of proposed Brain Tumor Classification using CNN

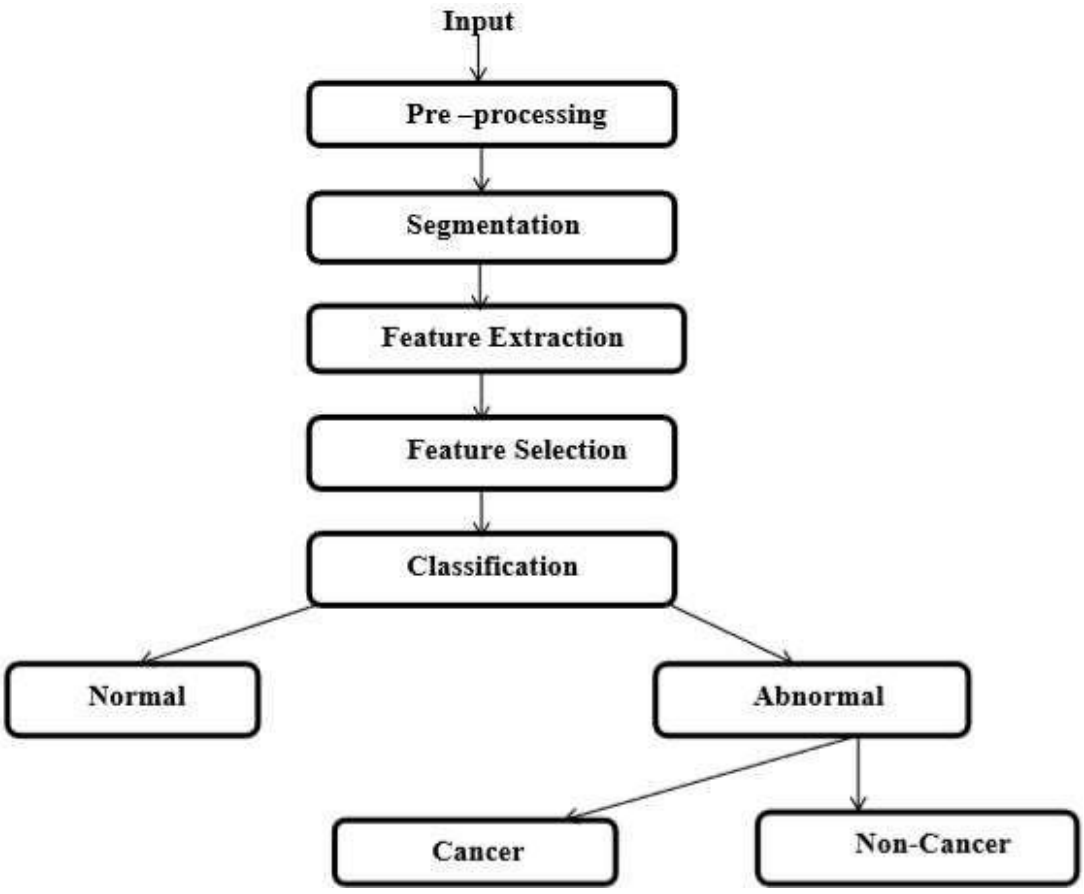


Fig 3: Classification Architecture

Module 1:- MRI Image Acquisition

Module 2:- Pre-processing

Module 3:- Segmentation

Module 4:- Feature extraction

Module 5:- Classification

Module 6:- Prediction

MODULE 1: IMAGE PREPROCESSING AND IMAGE ENHANCEMENT

We take Dataset of Brain MRI image from the Kaggle. The MRI dataset consists of a MRI images, including normal, benign, and malignant and No Tumor. These MRI images are taken as input to the primary step. The pre-processing is an essential and initial step in improving the quality of the brain MRI Image. The critical steps in pre-processing are the reduction of impulsive noises and image resizing. In the initial phase, we convert the brain MRI image into its corresponding gray-scale image. The removal of unwanted noise is done using the technique called adaptive bilateral filtering technique to remove the distorted noises that are present in the brain images. This improves the diagnosis and also increase the classification accuracy rate.

IMAGE ACQUISITION FROM DATASET:

In image processing, image acquisition is done by retrieving an image from dataset for processing. It is the first step in the workflow because, without an image there is no any processing is possible. The image which is acquired is completely unprocessed. Here we process the image using the file path from the local device, or giving the Drive path.

FILTERS: In image processing, filters are mainly used to suppress the high frequencies in the image.

1. **Median filter:** It is a non-linear filtering technique used to remove noise from the images. It is performed by sorting all the pixel values from the window into numerical order and then replacing the pixel being considered with the median pixel value. This filter removes the speckle noise and salt and pepper noise through „ON‟ and „OFF‟ of pixels by white and dark spots.

2. **Bilateral filter:** It is also a non-linear, noise-reducing smoothing filter for images. It replaces the intensity of each pixel with a weighted average of intensity values from nearby pixels. This weight is based on the Gaussian distribution. Bilateral filtering smooth images while conserving edges utilizing a nonlinear grouping of neighbouring image pixels. This filtering technique is simple, local, and concise. It syndicates a grey level grounded on their likeness and the symmetrical nearness and chooses near vales to farther values in both range and domain.

IMAGE ENHANCEMENT:

Image enhancement is a technique used to improve the images quality by applying various computer-aided softwares. This Image Enhancement technique includes both objective and subjective enhancements. This technique includes on points and local operations. The local operations depend on the district input pixel values. Image enhancement has two types: Spatial and Transform domain techniques. The Spatial techniques work directly on the pixel level, while the transform technique works on the spatial technique.

Edge detection is a segmentation technique that uses the border recognition of objects or regions. This technique identifies the discontinuity of the objects. Edge Detection technique is used mainly in image study and to recognize the parts of the image where a huge variation in intensity arises.

1. SOBEL FILTER:

The Sobel filter is used for Edge detection technique. The filter works by calculating the gradient of image intensity at each pixel within that image. This filter is widely used in image analysis to help locate the edges in images. Sobel filter is used for segmentation purpose. This technique can be dependent on the central difference which tends toward the central pixels on average. This technique can be expressed as 3 × 3 matric to the first derivative of the Gaussian kernel. It combines smoothing and differentiation. For Sobel edge detection the gradient of the image is calculated for each pixel position in the image.

MODULE 2: IMAGE SEGMENTATION USING BINARY THRESHOLD

Image segmentation is a technique of segregating the image into various parts. The main aim of the segregation technique is to make the images easy to analyze and interpret with preserving the quality of image. This technique is also used to trace the objects borders within the images i.e. Edge detect. This technique labels the pixels according to their intensity and characteristics. Those parts represent the entire original image and acquire its characteristics such as intensity and similarity.

The image segmentation technique is used to create contours of the body for clinical purposes. Segmentation is used in machine perception, malignant disease analysis, tissue volumes, anatomical and functional analyses, virtual reality visualization, and anomaly analysis, and object definition and detection. Segmentation methods has ability to detect or identify the abnormal portion from the image which is useful for analyzing the size, volume, location, texture and shape of the extracted image.

THRESHOLDING:

Thresholding is the simplest method of image segmentation. It is a non-linear operation that converts a grey-scale image into a binary image where the two levels are assigned to pixels that are below or above the specified threshold value. In Open CV, we use cv2.threshold() function:

cv2.threshold(src, thresh, maxval, type[dst])

This function applies fixed-level thresholding to a single-channel array. The function is typically used to get a bi-level (binary) image out of a grayscale image for removing a noise, that is, filtering out pixels with too small or too large values. “maxval” is the set threshold value which compares with input values, when the input is greater than the set threshold value it gives output is set maxval value and it is shown with white color in gray images. when the input pixel intensity values are less than the set threshold, its output is black color.

Morphological Transformations :

Morphological transformations technique removes noise from a brain to make sure that all brain is converted to white, to manage from removing boundary (skull) without effect on other parts of a brain. That is mainly done by the morphology transformation. Morphological transformations are some simple operations on binary images. It needs two inputs, one is our original image, and the second one is called structuring element or the kernel.

Morphological operators :

1) **Erosion:**

   Erosion erodes away the boundaries of foreground object by making a kernel slides through the image (as 2D convolution). If only all the pixels under the kernel is 1, a pixel in the original image (either 1 or 0) will be considered 1, otherwise, it made equal to zero. It is useful for removing small white cause the simply white region decreases in an image.

2) **Dilation:**

   Dilation is just the opposite of erosion. Dilation increases the white region in the image.

Edge Detector:

We detecting tumor by using Canny Edge Detector which is developed by John F in 1986. Canny Edge detector technique aims to satisfy three main criteria i.e. low error rate, good localization, and minimal response.

Steps of Canny Edge Detector:

1. **Noise Reduction:**

First step is to remove the noise in the image with a 5x5 Gaussian filter.

.**2. Finding Intensity Gradient of the image:**

A) Filtering with Sobel kernel in both horizontal (Gx) and vertical direction (Gy).

B) Find the gradient strength and direction.

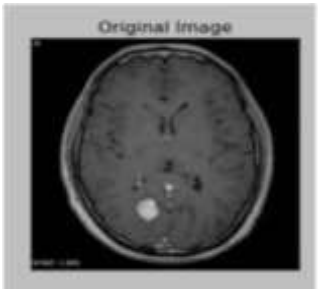
Steps of Image Pre-Processing and Segmentation

First: Convert to gray style:

At first, we should remove any colors in the image to convert it to the gray scale by CvtColor function. Converting an image from 3d to 2d makes the processing faster.

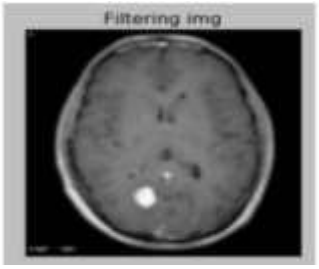
cv.CvtColor (Image_Input, image_output, code)

```
gray_image = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
```



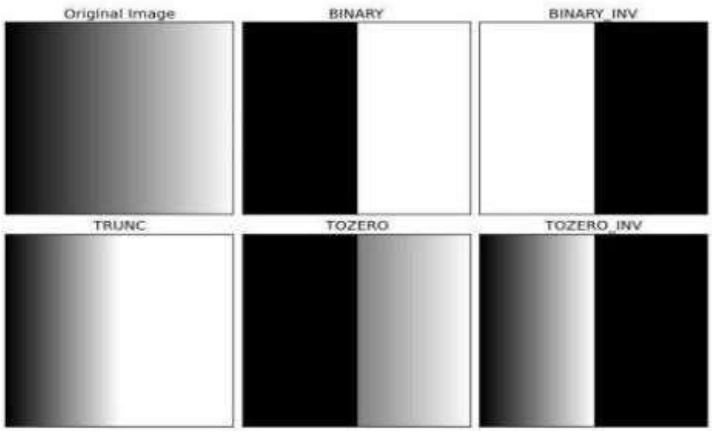
Second: Filters:

```
#Filtering
median = cv2.medianBlur(img,5)
Gaussian = cv2.GaussianBlur(median,(5,5),0)
```

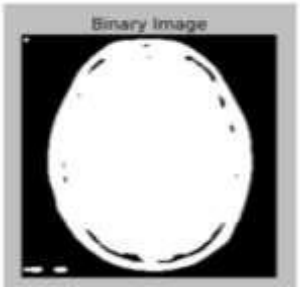


Third: Binarization using thresholding techniques

Img = cv2.imread('gradient.png',0) ret,
thresh1 = cv2.threshold(img,127,255,cv2.THRESH_BINARY)
ret,thresh2 = cv2.threshold(img,127,255,cv2.THRESH_BINARY_INV)
ret,thresh3 =cv2.threshold(img,127,255,cv2.THRESH_TRUNC)
ret,thresh4 = cv2.threshold(img,127,255,cv2.THRESH_TOZERO)
ret,thresh5 = cv2.threshold(img,127,255,cv2.THRESH_TOZERO_INV)




```
#binarization
ret, thresh1 = cv2.threshold(img,25,255,cv2.THRESH_BINARY)
```



Fourth: Morphological transformations

1. Erosion:

```
img = cv2.imread('j.png',0)
Kernel = np.ones((5,5),np.uint8)
Erosion = cv2.erode(img, Kernel, iterations = 1)
```

2. Dilation:

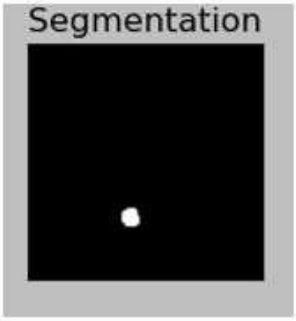
```
Dilation = cv2.dilate(Img, kernel, iterations = 1)
```

```
#removing skull
erosion = cv2.erode(closing1,kernel,iterations = 6)
NO_skull = img * (-erosion);
```

```
#convert to white
kernel = np.ones((7,7),np.uint8)
closing1 = cv2.morphologyEx(thresh1, cv2.MORPH_CLOSE, kernel)
```

Fifth : Threshold again

```
#Segmentation
ret, thresh2 = cv2.threshold(img,120,255,cv2.THRESH_BINARY)
```

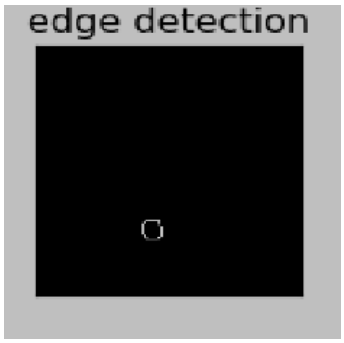


Sixth : Edge Detector

Steps of Canny Edge Detector:

1. Noise Reduction: First step is to remove the noise in the image with a 5x5 Gaussian filter.
2. Finding Intensity Gradient of the image.
   A) Filtering with Sobel kernel in both horizontal (Gx) and vertical direction (Gy):
   B) Find the gradient strength and direction

```
#edge detection
edges = cv2.Canny(img,100,200)
```



MODULE 3: FEATURE EXTRACTION AND SELECTION

Features are the main properties that quantify significant characteristics of the object. After getting an image when ending segmentation, start features steps on an original image. Applying features selection then feature reduction using PCA (Principal component analysis) is used.

Feature extraction :

Extraction stage is an important step to identifying the brain tumor where is exactly located and helps in predicting next stage. There are three kinds of features can be selected and extracted.

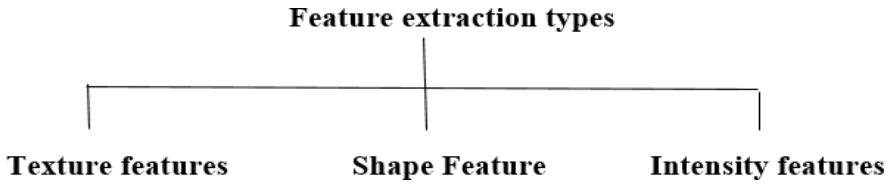


Fig 4: Feature Extraction Types

Texture features: based on surface properties. Such as homogeneity, energy…etc.

Shape features: based on the boundary of an image. Such as: area, Centroid , Compactness .

Intensity: based on region properties. Such as: median, intensity …etc.

Texture features

It is characterized by the spatial distribution of gray levels in a neighborhood. Since texture shows its characteristics both by pixel co-ordinates and pixel values. Image texture depends on the scale or resolution at which it‟s displayed.

Texture based on gray level co-occurrence matrix GLCM

Concerned with properties of two pixels at specific relative positions, the matrix describe how many times a pair of pixels appears in an image, the first pixel is referenced the other is a neighbor, the goal is to assign an unknown sample image to one of a set of known texture classes.

Calculate GLCM matrix

1. Create framework matrix, if the original image has G Gray levels, then the dimension of the co-occurrence matrix is GxG.
2. Describe spatial relation between reference and neighbor pixels using two parameters (d, Θ), which d is the distance between the pair of pixels and Θ is the angle between them.
3. Count the occurrence and fill in framework matrix by checking all pixel pairs with distance d and direction Θ assign value i to pixel 1 and value j to pixel 2.

Contrast

Measure the quantity of local changes in an image.

It returns the measure of intensity contrast between the pixel and its neighbor.

Rang = [0 (size (GLCM, 1)-1) ^2], If it is small, texture becomes strong.

Correlation

How correlated pixel is to its neighbor.it is the measure of gray tone linear changes in the image.

Rang = [-1 1].

-1 indicating prefect negative correlation.

1 indicating prefect positive correlation .

Homogeneity (inverse different moment)

It scaled the local changes of image texture.it returns value that measures the closeness of the distribution of elements in the GLCM.

Rang = [0 1].

It became large if local texture only has minimal changes.

Energy

It measures the degree of pixel pair repetitions. Only similarity gray level pixel is present.

Rang = [0 1].

Dissimilarity

Measure of it allows comparison between segmentations created by different algorithms.

Texture based on histogram

Concerned with properties of one pixel.it is important to calculate statistical value such as: mean, skew ….Etc.

Mean

It is the average value, so it tells us something about the general brightness.

-a bright image has a high mean.

-a dark image has a low mean.

Skew

Measure the asymmetry. The image should have low skew (one peak at each side of the mean).

Standard deviation (SD)

Known as the square root of variance describes contrast. High contrast should have high standard deviation

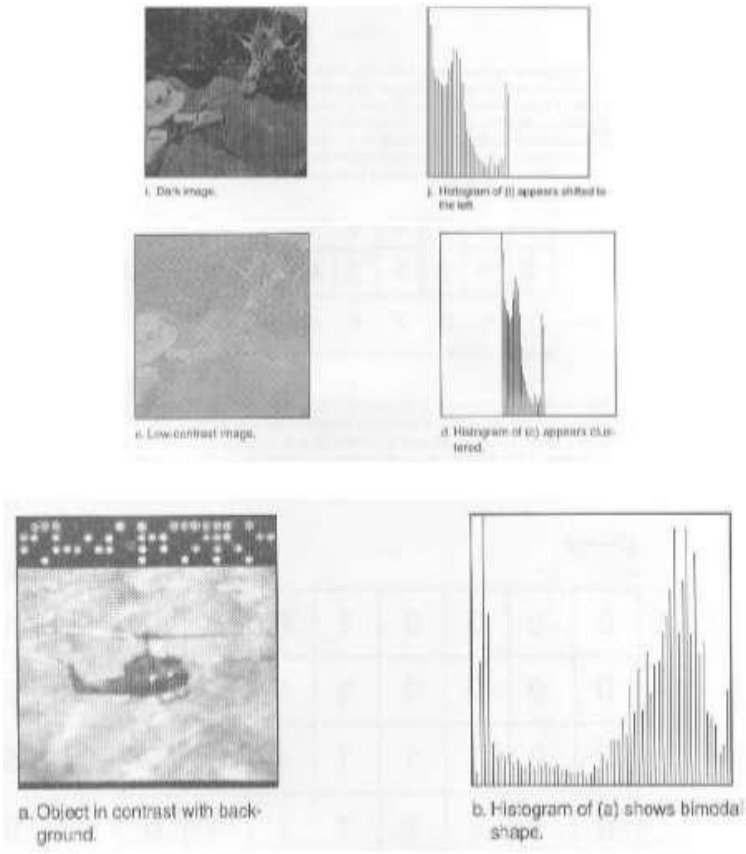


Fig 5 : Different histogram for different contrast image

Shape features

Features describe the boundary of the image like area and centroid, help to extract specific information about the shape of the region.

To calculate shape features we should know image moments, An Image moment is a number calculated using a certain formula, asset of moments computed from the digital image represented characteristics of image shape and give information about the geometric feature of the image.

Area

Calculate a number of pixels in the region. For Calculating area, we need to calculate its zeroes moment In a binary image, there are two possibilities 0 or 1, so in white pixel add 1 to moment to get all area.

Centroid

Cendroid are the coordinates for the region center of mass.

$$centroid = \left(\frac{\mu_{1,0}}{\mu_{0,0}}, \frac{\mu_{0,1}}{\mu_{0,0}}\right)$$

Compactness

Calculated from dividing the area by perimeter square, where perimeter is the length of the boundary of the object, when region more likely to be a circle it goes to maximum compactness

Eccentricity

The ratio between the major axis and the minor axis in the region detected, when the ratio is 1, the region becomes a circle.

Major axis (A): longest horizontal axis in region.

Minor axis (B): longest vertical axis in region.

Eccentricity = length of A / length of B

Feature Reduction

In machine learning and statistics, dimensionality reduction or dimension reduction is the process of reducing the number of random variables under consideration, via obtaining a set of principal variables. It can be divided into feature selection and feature extraction.

Feature extraction transforms the data in the high-dimensional space to a space of fewer dimensions. The data transformation may be linear, as in principal component analysis (PCA), but many nonlinear dimensionality reduction techniques also exist. For multidimensional data, tensor representation can be used in dimensionality reduction through multilinker subspace learning.

Methods of feature reduction

Principal Component Analysis (PCA):

Principal Component Analysis (PCA) is a statistical procedure that orthogonally transforms the original n coordinates of a data set into a new set of n coordinates called principal components.

Random Forests / Ensemble Trees:

Decision Tree Ensembles, also referred to as random forests, are useful for feature selection in addition to being effective classifiers. One approach to dimensionality reduction is to generate a large and carefully constructed set of trees against a target attribute and then use each attribute‟s usage statistics to find the most informative subset of features.

1 -Texture Feature:

Mean, Standard deviation, contrast, correlation, dissimilarity, energy, homogeneity, kurtosis, skew

2 -Shape Feature:

Area, convex area, Eccentricity, Solidity, Euler number, Extent, Orientation.

A - First we use Low Variance Filter to remove features that have low variance data samples

B - Then we normalize the rest of features to make feature values Convergent as possible using Standard deviation and Mean

MODULE 5:- CLASSIFICATION

- Classification of MR brain image either as normal or abnormal.
- In MLP, each node is a neuron with a nonlinear activation function.
- With the help of above modules we can detect weather the Tumor is present or not.
- This process helps in identifying the size, shape and position of the tumor.

6. Implementation & Coding :

6.1. Important Algorithms :

6.1.1. Support vector machine (SVM):

It is an algorithm of supervised machine learning, is used for both classification and regression.

It distributes data items as points in n dimensional space where n is number of features, the data points is called the support vectors .Then the hyper plane is detected to separate points into two classes to made classification very well.

The hyper plane is line used for splitting input data distribute in n dimensional space.

6.1.2. K nearest-neighbors algorithm

K nearest neighbors is a supervised machine learning algorithm, it is calculating as the minimum distance between the unknown instance and the training data samples. The k is number of neighbors and must be odd and positive. It stores dataset so it no learning required.

The equation is :

$$d(x, x') = \sqrt{(x_1 - x'_1)^2 + (x_2 - x'_2)^2 + \ldots + (x_n - x'_n)^2}$$

The knn classifier: it is a non-parametric, instance based and lazy learning.

1) Instanced based: is the training instance to use in predication.

2) Lazy learning: not required learning.

3) Non parametric: not has assumption about of functional form of problem solved.

6.1.3. Linear regression

Linear regression is very simple in representation. The representation is a linear equation between set of input values (x) and the predicted output for that set (y). As such, both input values (x) and output values (y) are numeric. This linear equation assigns one scale factor to each input value, called a coefficient and represented by the capital Greek letter Beta (B)

One additional Machine Learning coefficient is also added, giving the line an additional degree of freedom (e.g. moving up and down on a two-dimensional plot) and is often called the intercept or the bias coefficient. For example, in a simple regression problem (a single x and a single y), the form of the model is:

$$y = B0 + B1*x$$

Four techniques to prepare a linear regression model:

- **Simple Linear Regression**

When we have a single input, we can use statistics to estimate the coefficients. This requires that you calculate statistical properties from the data such as means, standard deviations, correlations and covariance. All of the data must be available to traverse and calculate statistics.

- **Ordinary Least Squares**

When we have more than one input we can use Ordinary Least Squares to estimate the values of the coefficients.

- **Gradient Descent**

When there are one or more inputs you can use a process of optimizing the values of the coefficients by iteratively minimizing the error of the model on your training data.
It is useful when you have a very large dataset either in the number of rows or the number of columns that may not fit into memory.

- **Regularization**

These seek to both minimize the sum of the squared error of the model on the training data (using ordinary least squares) but also to reduce the complexity of them.

6.1.4. K-Means:

K-means clustering is an unsupervised learning, which is used when you have unlabeled data (data without defined categories or groups). The goal of this algorithm is to find groups in this data, with the number of groups represented by (K).

The algorithm works iteratively to assign each data point to one of K groups based on the features that are provided. Data points are clustered based on feature similarity. The results of the K-means clustering algorithm are:

1. The centroids of the K clusters, which can be used to label new data
2. Labels for the training data (each data point is assigned to a single cluster)

6.2. Methods of Evaluating a Classifier used in this Project :

6.2.1 Classification accuracy:

Is the percentage of Correct Predictions, it computed using the true test class labels and predicted class labels

```
('True: ', array([1, 1, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1, 0, 1, 1, 1]))
('pred: ', array([1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 1, 0, 1, 1]))
```

This how to calculate accuracy in our project:

```
# Calculate accuracy
from sklearn import metrics
print(metrics.accuracy_score(y_test , y_pred_class))
```

```
0.764705882353
```

6.2.2. Confusion Matrix:

Table that describes the performance of a classification model.

n=192	Predicted: 0	Predicted: 1
Actual: 0	118	12
Actual: 1	47	15

Every observation in the testing set is represented in exactly one box

It's a 2x2 matrix because there are 2 response classes

The format shown here is not universal

Basic terminology

1. True Positives (TP): we correctly predicted that they do have cancer .

2. True Negatives (TN): we correctly predicted that they don't have cancer.

3. False Positives (FP): we incorrectly predicted that they do have cancer (a "Type I error")

4. False Negatives (FN): we incorrectly predicted that they don't have cancer (a "Type II error")

```
]: # important : true values first then predicted valuse
confusion = metrics.confusion_matrix(y_test, y_pred_class)
print(confusion)
```

```
[[ 0  3]
 [ 1 13]]
```

```
('TP :', 13)
('TN', 0)
('FP', 3)
('FN', 1)
```

6.2.3. Sensitivity:

when the actual values are positive, how often the prediction is correct or how 'sensitive' is the classifier to detecting positive instances. Also known as 'true positive rate ' or recall:

```
print( TP / float(TP + FN))
print( metrics.recall_score(y_test, y_pred_class) )
```

```
0.928571428571
0.928571428571
```

6.2.4. Specificity:

when the actual value is negative, how often is the prediction correct or how 'specific' or 'selective' is the classifier in prediction positive instances:

```
print( TN / float(TN + FP ))
```

6.2.5.Precision:

when a positive value is predicted, how often the prediction is correct or how "precise" is the classifier when predicting positive instances:

```
print( TP / float(TP + FP))
print( metrics.precision_score(y_test, y_pred_class))
```

```
0.8125
0.8125
```

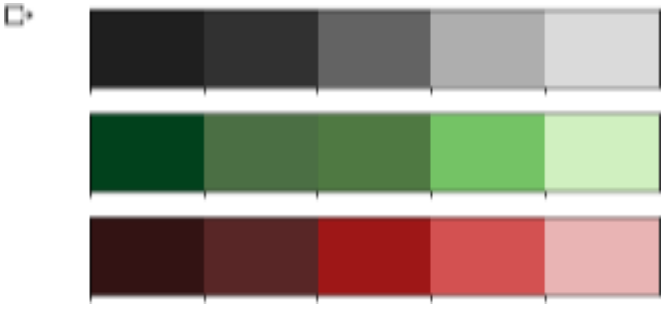
6.3. Coding and Implementation :

6.3.1. Importing the Dataset:

```
import matplotlib.pyplot as pltimport numpy as np
import pandas as pd import seaborn as snsimport cv2
import tensorflow as tf
from tensorflow.keras.preprocessing.image import ImageDataGeneratorfrom
tqdm import tqdm
import os
from sklearn.utils import shuffle
from sklearn.model_selection import train_test_split from
tensorflow.keras.applications import EfficientNetB0from
tensorflow.keras.callbacks import EarlyStopping,
ReduceLROnPlateau, TensorBoard, ModelCheckpoint
from sklearn.metrics import classification_report,confusion_matriximport
ipywidgets as widgets
import io
from PIL import Image
from IPython.display import display,clear_outputfrom warnings import
filterwarnings
for dirname, _, filenames in os.walk('/kaggle/input'):for filename in
filenames:
print(os.path.join(dirname, filename))
```

6.3.2. color_palette(), which can be used for coloring the plot. Using the palette we can generate the point with different colors.

```
colors_dark = ["#1F1F1F", "#313131", '#636363', '#AEAEAE', '#DADADA']
colors_red = ["#331313", "#582626", '#9E1717', '#D35151', '#E9B4B4']
colors_green = ['#01411C','#4B6F44','#4F7942','#74C365','#D0F0C0']
sns.palplot(colors_dark) sns.palplot(colors_green)sns.palplot(colors_red)
```



6.3.3. Give labels

```
labels = ['glioma_tumor','no_tumor','meningioma_tumor','pituitary_tumor']
```

6.3.4. Mount google Drive

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

6.3.5. we will read the images and store it in a separate list

```
X_train = [] y_train = [] image_size = 150for i in labels:
    folderPath = os.path.join('/content/drive/MyDrive/ Brain-Tumor-
Classification-DataSet-master','Training',i)
for j in tqdm(os.listdir(folderPath)):
img = cv2.imread(os.path.join(folderPath,j)) img =
cv2.resize(img,(image_size, image_size))X_train.append(img)
y_train.append(i)

for i in labels:
folderPath = os.path.join('/content/drive/MyDrive
/Brain-Tumor-Classification-DataSet-master','Testing',i)for j in
tqdm(os.listdir(folderPath)):
img = cv2.imread(os.path.join(folderPath,j)) img =
cv2.resize(img,(image_size,image_size))X_train.append(img)
y_train.append(i)

X_train = np.array(X_train)y_train = np.array(y_train
```

```
100%|██████████| 826/826 [00:14<00:00, 56.13it/s]
100%|██████████| 395/395 [00:05<00:00, 68.54it/s]
100%|██████████| 822/822 [00:14<00:00, 57.67it/s]
100%|██████████| 827/827 [00:13<00:00, 60.21it/s]
100%|██████████| 100/100 [00:01<00:00, 63.27it/s]
100%|██████████| 105/105 [00:01<00:00, 95.84it/s]
100%|██████████| 115/115 [00:13<00:00,  8.72it/s]
100%|██████████| 74/74 [00:20<00:00,  3.64it/s]
```

6.3.6. Take Sample Images from Datasets

```
k=0
fig, ax = plt.subplots(1,4,figsize=(20,20))
fig.text(s='Sample Image From Each
Label',size=18,fontweight='bold',
fontname='monospace',color=colors_dark[1],y=0.62,x=0.4,alpha=0.8)
for i in labels:
j=0
```

```
while True :
if y_train[j]==i: ax[k].imshow(X_train[j]) ax[k].set_title(y_train[j])
ax[k].axis('off')
k+=1
break
j+=1
```


```
X_train, y_train = shuffle(X_train,y_train, random_state=101)
X_train.shape
```

Output

```
(3264, 150, 150, 3)
```

6.3.7. Split arrays / matrices into random train and test subsets

```
X_train,X_test,y_train,y_test = train_test_split (X_train,y_train,
test_size=0.1,random_state=101)
```

```
y_train_new = [] for i in y_train:
y_train_new.append(labels.index(i))y_train = y_train_new
y_train = tf.keras.utils.to_categorical(y_train)y_test_new = []
for i in y_test: y_test_new.append(labels.index(i))
y_test = y_test_new
y_test = tf.keras.utils.to_categorical(y_test)
```

6.3.8. Split arrays / matrices into random train and test subsets

```
X_train,X_test,y_train,y_test = train_test_split (X_train,y_train,
test_size=0.1,random_state=101)


y_train_new = [] for i in y_train:
    y_train_new.append(labels.index(i))y_train = y_train_new
y_train = tf.keras.utils.to_categorical(y_train)y_test_new = []
for i in y_test: y_test_new.append(labels.index(i))
y_test = y_test_new
y_test = tf.keras.utils.to_categorical(y_test)
```

6.3.9. use EfficientNetB0 for classifying 1000 classes of images from imagenet.

```
effnet = EfficientNetB0(weights='imagenet',include_top=False,
input_shape=(image_size,image_size,3))
```

output

```
Downloading data from https://storage.googleapis.com/keras-
applications/efficientnetb0_notop.h5
16711680/16705208 [==============================] - 0s 0us/step
16719872/16705208 [==============================] - 0s 0us/step
```

6.3.10. To create our own classification layers stack on top of the EfficientNet convolutional base model. We adapt GlobalMaxPooling2D to convert 4D the (batch_size, rows, cols, channels) tensor into 2D tensor with shape (batch_size, channels). GlobalMaxPooling2D results in a much smaller number of features compared to the Flatten layer, which effectively reduces the number of parameters.

```
model = effnet.output
model = tf.keras.layers.GlobalMaxPooling2D()(model)model =
tf.keras.layers.Dropout(rate=0.5)(model)
model = tf.keras.layers.Dense(4,activation='softmax')(model)
model = tf.keras.models.Model(inputs=effnet.input, outputs = model)
model.summary()
```

6.3.11. A metric is a function that is used to judge the performance of our model

```
model.compile(loss='categorical_crossentropy',optimizer = 'Adam',metrics=
['accuracy'])


tensorboard = TensorBoard(log_dir = 'logs')
checkpoint = ModelCheckpoint("effnet.h5",monitor="val_accuracy",
save_best_only=True, mode="auto",verbose=1)
reduce_lr = ReduceLROnPlateau(monitor = 'val_accuracy',factor = 0.3,
patience = 2, min_delta = 0.001,
mode='auto',verbose=1)
```

6.3.12. Fitting the model

```
history = model.fit(X_train,y_train,validation_split=0.1,epochs =12,
verbose=1, batch_size=32, callbacks=[tensorboard,checkpoint,reduce_lr])
```

6.3.13. Training, Loss vs epoch

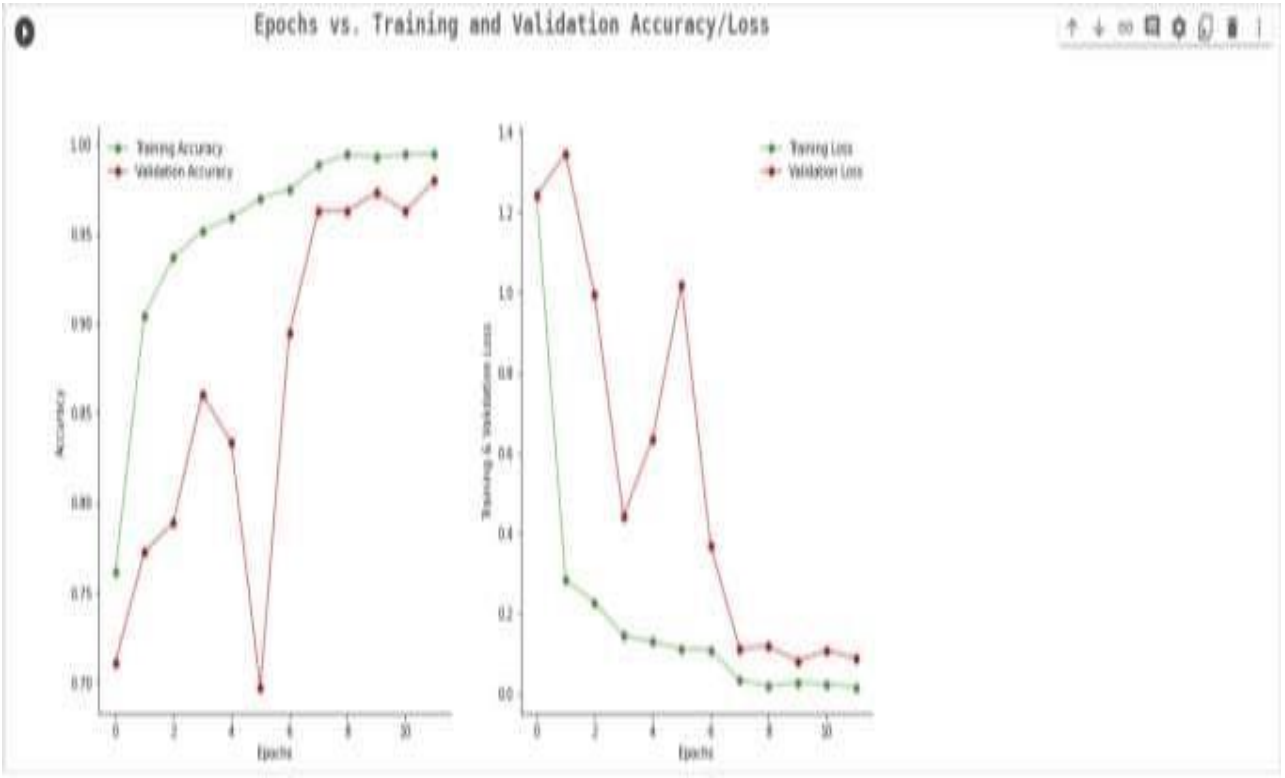
```
filterwarnings('ignore') epochs = [i for i in range(12)]
fig, ax = plt.subplots(1,2,figsize=(14,7))train_acc =
history.history['accuracy'] train_loss = history.history['loss'] val_acc =
history.history['val_accuracy'] val_loss = history.history['val_loss']

fig.text(s='Epochs vs. Training and Validation Accuracy/Loss',size=18,
fontweight='bold',fontname='monospace',color=colors_dark[1],
y=1,x=0.28,alpha=0.8)
sns.despine()ax[0].plot(epochs,            train_acc,            marker='o',
markerfacecolor=colors_green[2],color=colors_green[3],  label   =   'Training
Accuracy')
ax[0].plot(epochs, val_acc, marker='o',
markerfacecolor=colors_red[2],color=colors_red[3],label = 'Validation
Accuracy') ax[0].legend(frameon=False) ax[0].set_xlabel('Epochs')
ax[0].set_ylabel('Accuracy')

sns.despine()
ax[1].plot(epochs, train_loss, marker='o',markerfacecolor=colors_green[2],
color=colors_green[3],label ='Training Loss')
ax[1].plot(epochs, val_loss, marker='o',
```

```
markerfacecolor=colors_red[2],color=colors_red[3],label = 'Validation Loss')
ax[1].legend(frameon=False) ax[1].set_xlabel('Epochs')
ax[1].set_ylabel('Training & Validation Loss')

fig.show()
```




```
pred = model.predict(X_test) pred = np.argmax(pred,axis=1)
y_test_new = np.argmax(y_test,axis=1)
```

6.3.14. Classification report.

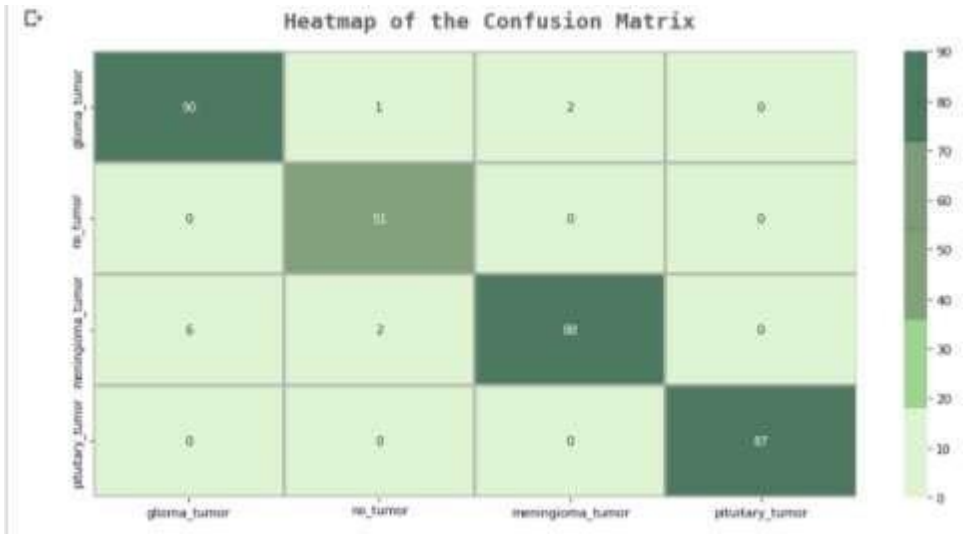
```
print(classification_report(y_test_new,pred))
```

	precision	recall	f1-score	support
0	0.94	0.97	0.95	93
1	0.94	1.00	0.97	51
2	0.98	0.92	0.95	96
3	1.00	1.00	1.00	87
accuracy			0.97	327
macro avg	0.96	0.97	0.97	327
weighted avg	0.97	0.97	0.97	327

6.3.15. Confusion Matrix

```
fig,ax=plt.subplots(1,1,figsize=(14,7))
sns.heatmap(confusion_matrix(y_test_new,pred),
ax=ax,xticklabels=labels,yticklabels=labels,annot=True,
cmap=colors_green[::-1],alpha=0.7,linewidths=2,linecolor=colors_dark[3])
fig.text(s='Heatmap of the Confusion Matrix',size=18,fontweight='bold',
fontname='monospace',color=colors_dark[1],y=0.92,x=0.28,alpha=0.8)

plt.show()
```



6.3.16. OUTPUT

```
def img_pred(upload):
for name, file_info in uploader.value.items(): img =
Image.open(io.BytesIO(file_info['content']))
opencvImage = cv2.cvtColor(np.array(img), cv2.COLOR_RGB2BGR)img =
cv2.resize(opencvImage,(150,150))
img = img.reshape(1,150,150,3)
```

```
p = model.predict(img)
p = np.argmax(p,axis=1)[0]if p==0: p='Glioma Tumor' elif p==1:
    print('The model predicts that there is no tumor')elif p==2:
p='Meningioma Tumor'
else:
    p='Pituitary Tumor'if p!=1:
print(f'The Model predicts that it is a {p}')

uploader = widgets.FileUpload()display(uploader)
```

```
[ ] uploader = widgets.FileUpload()
    display(uploader)
```

Upload (3)

```
button = widgets.Button(description='Predict')out = widgets.Output()
def on_button_clicked(_):
with out: clear_output()try:
    img_pred(uploader)except:
      print('No Image Uploaded/Invalid Image File')
button.on_click(on_button_clicked) widgets.VBox ([button,out])
```

```
button = widgets.Button(description='Predict')
out = widgets.Output()
def on_button_clicked(_):
    with out:
        clear_output()
        try:
            img_pred(uploader)

        except:
            print('No Image Uploaded/Invalid Image File')
button.on_click(on_button_clicked)
widgets.VBox([button,out])
```

Predict

The model predicts that there is no tumor

6.4. Output – Application Preview

6.5. About The Application

It mainly involves 4 steps:-

1. **Training and saving Tensorflow Model:-** Firstly we need to train a model using Keras framework and save the model in .H5 or.PB format to load it any time we require.

2. **Creating a TensorFlow Lite Converter:-**The TensorFlow Lite converter is a tool available as a Python API that converts trained TensorFlow models into the TensorFlow Lite format. It can also introduce optimizations.

3. **Converting TensorFlow Lite Converter into .tflite format:-** Now we need to convert this object to tflite format which will be further used in the android application.

4. **Deploying .tflite into Android Studio and run the Inference:-** Now we will use Tensorflow Interpreter API in an android studio to run the .tflite model with data to produce output

7. Testing :

7.1. Testing Mechanism :

7.1.1 UNIT TESTING :

The procedure level testing is made first. By giving improper inputs, the errors occurred are noted and eliminated.

7.1.2 VALIDATION TESTING :

In the project, validation testing is made in various forms. i.e MRI image is valiated image or not .

Sr.No	Action	Inputs	Expected Outputs	Actual Output	Test Browser	Test Result	Test Comments
1.	Launch Application	MRI image	Normal or Abnormal	Normal	Google	Pass	Successful
2.	Input Image	MRI Image	Meningioma Tumor, or Pituitary Tumor, or Gliomatumor	Menin-gioma tumor	Google	Pass	Successful
3.	Input image	MRI image	Meningioma Tumor, or Pituitary Tumor, or Gliomatumor	Pituitary tumor	Google	Pass	Successful
4.	Input image	MRI image	Meningioma Tumor, or PituitaryTumor, or Glioma tumor	Glioma tumor	Google	Pass	Successful
5.	Input text file	Text file	Meningioma Tumor, or PituitaryTumor, or Glioma tumor	Select image	Google	fail	Select only .jpg fies

8. Conclusion & Further Work :

8.1. Conclusion:

We proposed a computerized method for the segmentation and identification of a brain tumor using the Convolution Neural Network. The input MR images are read from the local device using the file path and converted into grayscale images. These images are pre-processed usingan adaptive bilateral filtering technique for the elimination of noises that are present inside theoriginal image. The binary thresholding is applied to the denoised image, and Convolution Neural Network segmentation is applied, which helps in figuring out the tumor region in the MR images. The proposed model had obtained an accuracy of 84% and yields promising results without any errors and much less computational time.

9. Conclusion & Further Work :

9.1. Conclusion:

We proposed a computerized method for the segmentation and identification of a brain tumor using the Convolution Neural Network. The input MR images are read from the local device using the file path and converted into grayscale images. These images are pre-processed usingan adaptive bilateral filtering technique for the elimination of noises that are present inside theoriginal image. The binary thresholding is applied to the denoised image, and Convolution Neural Network segmentation is applied, which helps in figuring out the tumor region in the MR images. The proposed model had obtained an accuracy of 84% and yields promising results without any errors and much less computational time.

9. References :

1. "Classification of Brain Lesion Type and Grade Using MRI Texture and Shape in a Machine Learning Scheme," Magnetic Resonance in Medicine, vol. 62, no. 6, pp. 1609-1618, Dec 2009.
2. Deep Learning Approach for Brain Tumor Classification and Segmentation Using a Multiscale Convolutional Neural Network. Healthcare 2021, 9, 153.
3. Brain tumor segmentation using convolutional neural networks in MRI images. IEEE T Med Imaging 2016; 35: 1240-1251 13.
4. An Optimized Technique for Brain Tumor Classification and Detection with Radiation Dosage Calculation in MR Image, Microprocessors and Microsystems (2019), PII: S0141-9331(19)30323-0
5. MRI simulation-based evaluation of image-processing and classification methods. IEEE T Med Imaging 1999; 18: 1085-1097.
6. Classification of CT brain images based on deep learning networks. Comput Meth Prog Bio 2017; 138: 49-56.
7. Automatic brain tumor segmentation and extraction in MR images. In: Conference on Advances in Signal Processing (CASP); 9-11 June 2016; Pune, India. pp. 104-107.
8. "A quantitative study of shape descriptors from glioblastoma multiforme phenotypes for predicting survival outcome," Br J Radiol, vol. 89, no. 1068, p. 20160575
9. Brain tumors detection and segmentation in MR images: Gabor wavelet vs. statistical features. Computers & Electrical Engineering 2015; 45: 286-301.
10. "Brain Lesion MRI Segmentation and Classification Using Ensemble Classifier", International Journal of Recent Technology and Engineering (IJRTE) ISSN: 22773878, Volume- 8, Issue-1S4, 2019.
11. MRI-Based Brain Lesion Classification Using Ensemble of Deep Features and Machine Learning Classifiers. Sensors 2021, 21, 2222. https://doi.org/10.3390/s21062222
12. Brain tumor segmentation using convolutional neural networks in MRI images. IEEE T Med Imaging 2016; 35: 1240-1251
13. Deep Learning Approach for Brain Tumor Classification and Segmentation Using a Multiscale
14. Convolutional Neural Network. Healthcare 2021, 9, 153.