

**VISVESVARAYA TECHNOLOGICAL UNIVERSITY
BELAGAVI-590 018, KARNATAKA.**



FINAL YEAR PROJECT REPORT

ON

“BRAIN TUMOR DETECTION USING DEEP NEURAL NETWORK”

Submitted in the partial fulfillment of the requirements for the award of Degree

B.E. in Computer Science & Engineering

PROJECT ASSOCIATES

MEGHANA G

4BD17CS063

NEHA U K

4BD17CS075

ROJA DEVINA

4BD17CS094

TEJASWINI

4BD17CS113

PROJECT GUIDE

Prof. Gangadharappa M.Tech.,
Asst. Professor,
Department of CS&E,
B.I.E.T., Davangere.



PROJET CO-ORDINATOR

Dr. Roopa G M Ph.D.,MISTE.

HEAD OF DEPARTMENT

Dr. Nirmala C R Ph.D.,MISTE.

**Department of Computer Science and Engineering
Bapuji Institute of Engineering & Technology
Davangere- 577004
2020-2021**

Bapuji Institute of Engineering and Technology

Davangere -577004



Department of Computer Science and Engineering

CERTIFICATE

This is to certify that **MEGHANA G, NEHA U K, ROJA DEVINA, TEJASWINI** bearing USNs **4BD17CS063, 4BD17CS075, 4BD17CS094, 4BD17CS113** respectively of Computer Science and Engineering department have satisfactorily submitted the Project Report entitled “**BRAIN TUMOR DETECTION USING DEEP NEURAL NETWORK**” in the partial fulfillment of the requirements for the award of Degree of Bachelor of Engineering (B.E.) in Computer Science & Engineering, under the VTU during the academic year 2020-2021.

Prof. Gangadharappa S M.Tech.,

Asst Professor

Guide

Prof. Roopa G M Ph.D.,

Project Co-ordinator

Dr. Nirmala C R Ph.D.,

Head of the Department

Dr. H B Aravind Ph.D.,

Principal

Date:

Place:

Bapuji Educational Association (Regd.)
Bapuji Institute of Engineering and Technology, Davangere – 577004
Department of Computer Science & Engineering

VISION

To be a center-of-excellence by imbibing state-of-the-art technology in the field of Computer Science and Engineering, thereby enabling students to excel professionally and be ethical.

MISSION

M1	Adapting best teaching and learning techniques that cultivates Questioning and Reasoning culture among the students.
M2	Creating collaborative learning environment that ignites the critical thinking in students and leading to the innovation.
M3	Establishing Industry Institute relationship to bridge the skill gap and make them industry ready and relevant.
M4	Mentoring students to be socially responsible by inculcating ethical and moral values.

PROGRAM EDUCATIONAL OBJECTIVES (PEOs)

PEO1	To apply skills acquired in the discipline of Computer Science and Engineering for solving societal and industrial problems with apt technology intervention.
PEO2	To continue their career in industry/academia or to pursue higher studies and research.
PEO3	To become successful entrepreneurs, innovators to design and develop software products and services that meets the societal, technical and business challenges.
PEO4	To work in the diversified environment by acquiring leadership qualities with effective communication skills accompanied by professional and ethical values.

PROGRAM SPECIFIC OUTCOMES (PSOs)

PSO1	Analyze and develop solutions for problems that are complex in nature by applying the knowledge acquired from the core subjects of this program.
PSO2	To develop Secure, Scalable, Resilient and distributed applications for industry and societal requirements.
PSO3	To learn and apply the concepts and construct of emerging technologies like Artificial Intelligence, Machine learning, Deep learning, Big Data Analytics, IoT, Cloud Computing, etc for any real time problems.

ACKNOWLEDGMENT

Salutations to our beloved and highly esteemed institute, “**BAPUJI INSTITUTE OF ENGINEERING AND TECHNOLOGY**” for having well-qualified staff and labs furnished with the necessary equipment.

We express our sincere thanks to our guide Prof. **Gangadharappa S** for giving us constant encouragement, support and valuable guidance throughout the project without whose stable guidance this project would not have been achieved.

We express wholehearted gratitude to our Project Co-ordinator **Dr. Roopa G M**. I wish to acknowledge her who made our task easy by providing with her valuable help and encouragement.

We express wholehearted gratitude to **Dr. Nirmala C R** who is our respectable H.O.D of Computer Science & Engineering Department. We wish to acknowledge her who made our task easy by providing with her valuable help and encouragement.

We also express our wholehearted gratitude to our respected Principal, **Dr. H B Aravind** for his moral support and encouragement.

We would like to extend our gratitude to all staff of **Department of Computer Science and Engineering** for the help and support rendered to us. We have benefited a lot from the feedback, suggestions given by them.

We would like to extend our gratitude to all our family members and friends especially for their advice and moral support.

MEGHANA G (4BD17CS063)

NEHA U K (4BD17CS075)

ROJA DEVINA (4BD17CS094)

TEJASWINI (4BD17CS113)

ABSTRACT

In recent decades, brain tumor detection has become one of the foremost difficult issues in medical science. Early detection of those tumors is highly required to supply treatment for patients. In this project we present a machine learning approach to detect whether an MRI image of a brain contains a tumor or not, if yes, which type of tumor. In the present environment, various classifying algorithms are available to classify the brain tumor images. According to observations made for detection of tumor, it is observed that the Decision Tree (DT) classifier gives 94.24% accuracy. Because of the lag in accuracy, we considered implementing Convolutional Neural Network (CNN) to classify and detect tumor presence in the brain through Magnetic Resonance Imaging (MRI) images. In addition to the accuracy criterion, we use the benchmarks of Sensitivity, Specificity, and Precision to evaluate CNN performance. This is a new method based on the combination of feature extraction techniques with the CNN for tumor detection from MRI images. CNN improves accuracy, which is important for diagnosing the tumor; the accuracy predicted by the model helps the doctors in diagnosing the tumor and treating the patient at earlier stages.

CONTENTS

TOPICS	PAGE NO.
Chapter 1: INTRODUCTION	01-08
1.1 Brain Anatomy	
1.2 Basic Classification of Brain Tumors	
1.3 Types of Brain Tumors	
1.4 Brain Tumor Classification according to Grade	
1.5 Dianosis of Brain Tumor	
1.6 Motivation	
Chapter 2: LITERATURE SURVEY	09-14
2.1 Literature Survey	
2.2 Summary of Literature Survey	
2.3 Problem Statement	
2.4 Proposed Solution	
2.5 Objectives	
Chapter 3: SOFTWARE REQUIREMENTS SPECIFICATION	15-16
3.1 Functional Requirements	
3.2 Non-Functional Requirements	
3.3 Hardware Requirements	
3.4 Software Requirements	
Chapter 4: SYSTEM DESIGN	17-21
4.1 Block Diagram	
4.2 Flow Diagram	
4.3 Methodology	

Chapter 5: IMPLEMENTATION

22-29

- 5.1 Installation Steps
 - 5.1.1 Steps to Install Anaconda Navigator
 - 5.1.2 To Install Jupyter Notebook
- 5.2 Dataset Description
- 5.3 Implementation Details
 - 5.3.1 Packages Used
- 5.4 Data Preparation
- 5.5 Image Preprocessing
- 5.6 Modules Implementation

Chapter 6: TESTING AND TRAINING

30-40

- 6.1 Overview
- 6.2 Training Process
 - 6.2.1 Transfer Learning
 - 6.2.2 Adding CNN Layers
 - 6.2.3 Summarizing and Compiling the Model
 - 6.2.4 Training the Model
 - 6.2.5 Epochs vs. Training and Validation Accuracy/Loss
- 6.3 Prediction
- 6.4 Evaluation
 - 6.4.1 Heatmap of Confusion Matrix
- 6.5 Testing Process
- 6.6 Failure Cases

Chapter 7: RESULTS AND DISCUSSIONS

41-43

- 7.1 Snapshots

CONCLUSION

FUTURE WORK

BIBLIOGRAPHY

LIST OF FIGURES

SL. NO.	FIGURE NO.	NAME	PAGE NO.
01	1.1	Brain Anatomy with various functions	1
02	1.2	Classification of Brain Tumor	3
03	1.3	Brain Tumor Location	6
04	1.4	MRI images of Brain Tumor	6
05	1.5	Proportion of death due to cancer	7
06	1.6	Comparision interms of survival rate	7
07	4.1	Steps to analyze the result	17
08	4.2	Flow Diagram represents the Steps	18
09	4.3	Block Diagram of Feature Extraction through digital Image Processing	19
10	6.1	Semantic Representation of CNN Architecture	31
11	6.2	Graph showing the relationship between Epochs vs. Training and Validation Accuracy/Loss	35
12	6.3	Heatmap of the Confusion Matrix	38
13	7.1	Login Page	40
14	7.2	SignUp Page	41
15	7.3	Home Page	41
16	7.4	Result Page	42
17	7.5	Patient Information Page	42
18	7.6	Report Page	43
19	7.7	Neurologist Reference Page	43

CHAPTER 1

INTRODUCTION

1.1 Brain Anatomy

Brain controls all the necessary functions of body. The most complex organ in human body is brain and it is part of the central nervous system (CNS). Skull covers brain and the brain is composed of three matters- -gray matter (GM), white matter (WM) and cerebrospinal fluid (CSF)ll. Cerebrospinal fluid (CSF) is a transparent liquid that encapsulates the brain and the spinal cord and provides many functions to CNS. It provides a barrier against shocks which consists of glucose, oxygen, and ions. It is distributed throughout the nervous tissue. CSF helps in removal of waste products away from nervous tissues. Figure 1.1 shows brain anatomy with various functions.

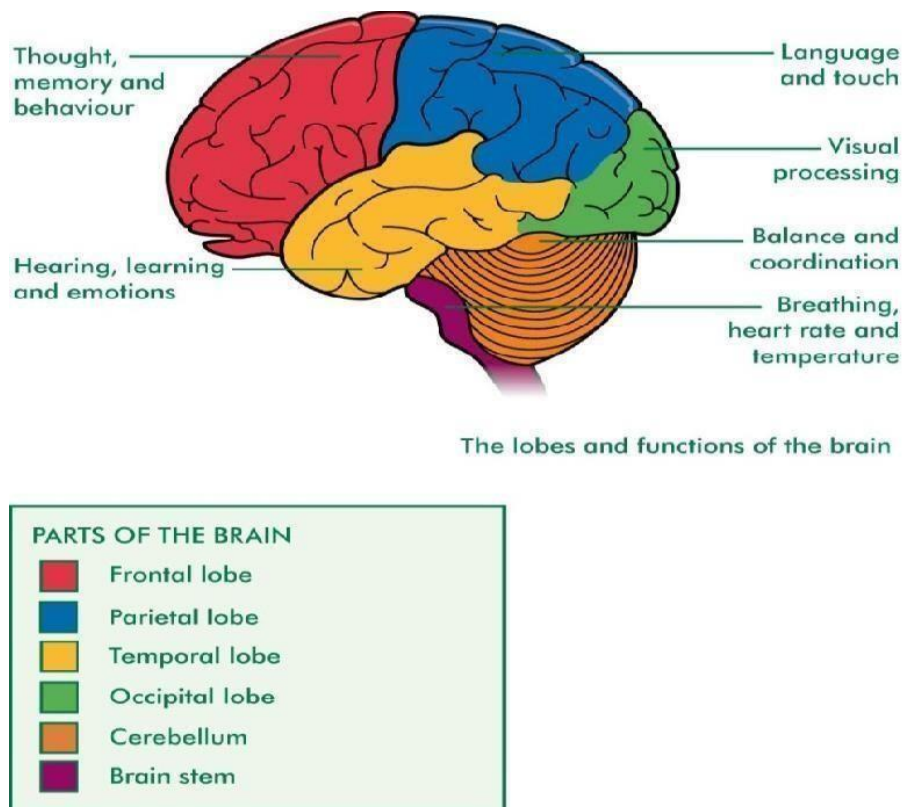


Fig 1.1 Brain anatomy with various Functions

1.2 Basic Classification of Brain tumors

Brain is the most complex organ of the body. The tumor is defined as uncontrolled growth of cells on any part of the body and respectively brain tumor is uncontrolled growth of brain cells .[1]

The body is made up of many types of cells. Each type of cell has special functions. Most cells in the body grow and then divide in an orderly way to form new cells as they are needed to keep the body healthy and working properly. When cells lose the ability to control their growth, they divide too often and without any order. The extra cells form a mass of tissue called a tumor. Tumors are benign or malignant.

- Benign brain tumors do not contain cancer cells. Usually these tumors can be removed, and they are not likely to recur. Benign brain tumors have clear borders. Although they do not invade nearby tissue, they can press on sensitive areas of the brain and cause symptoms.
- Malignant brain tumors contain cancer cells. They interfere with vital functions and are life threatening. Malignant brain tumors are likely to grow rapidly and crowd or invade the tissue around them. Like a plant, these tumors may put out "roots" that grow into healthy brain tissue. If a malignant tumor remains compact and does not have roots, it is said to be encapsulated. When an otherwise benign tumor is located in a vital area of the brain and interferes with vital functions, it may be considered malignant (even though it contains no cancer cells).

The Average life expectancy of HGG patient is 14 months. Brain tumors are scaled from grade I to IV and they are further classified in benign (class I II) and malignant (class III IV). Benign tumors are non-aggressive and mostly they don't move from their infected area. Whereas, malignant tumors are aggressive and more fatal than that of benign tumors. Benign tumors are treatable through chemotherapy and they can be reduced to a smaller size. Reducing it to an extent where it can be removed through operation. On the other hand, malignant tumors are non-operable, but they can be reduced through chemotherapy to some extent. It may increase the life expectancy of the patient up to 2 or 3 months, but it is not completely curable.

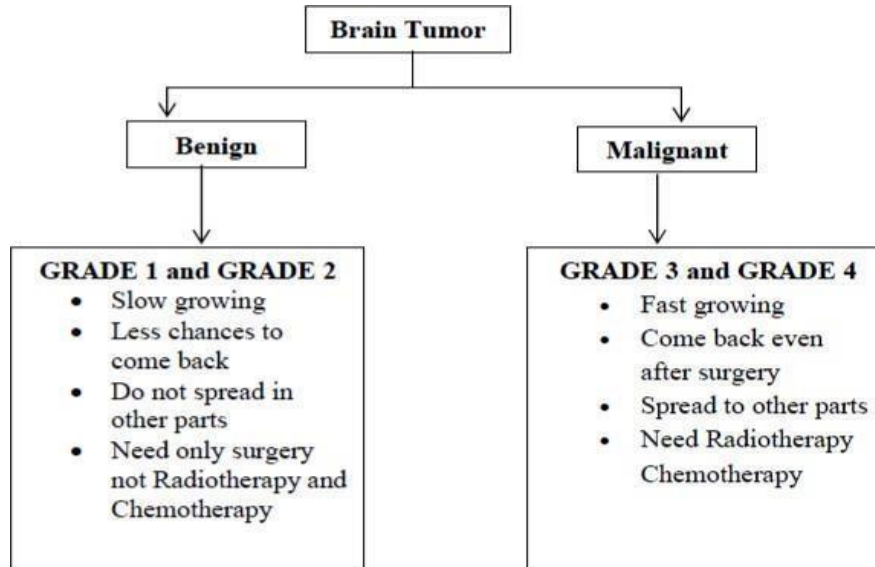


Fig 1.2 classification of Brain Tumor

1.3 Types of Brain tumors

According to World Health Organization (WHO), there are 120 types of brain tumor. WHO classification is based on cell's origin. Broadly, brain tumors are categorized in two types:

1. Primary Brain tumors
2. Secondary Brain tumors

Primary Brain Tumors are originated in the brain. Primary brain tumors are classified by the type of tissue in which they arise. Most common type of primary tumor is 'glioma' which occurs due to glial cells. Glial cells are supportive tissues of brain. There are several types of gliomas, such as Astrocytoma, Oligodendrogliomas, Ependymomas, Meningiomas, Schwannomas, Craniopharyngiomas, Germ cell tumors.

Secondary Brain Tumors are tumors caused from cancer that originates in another part of the body. These tumors are not the same as primary brain tumors. The spread of cancer within the body is called metastasis. Cancer that spreads to the brain is the same disease and has the same name as the original (primary) cancer.

Primary brain tumors include tumors that originate from the tissues of the brain or the brain's immediate surroundings. Primary tumors are categorized as benign or malignant. Metastatic

brain tumors include tumors that arise elsewhere in the body (such as the breast or lungs) and migrate to the brain, usually through the bloodstream. Metastatic tumors are considered cancer and are malignant. Brain tumor may or may not be symptomatic so they can be detected by symptoms exhibited by patients or can be identified by CT scan or MRI images .

According to World Health Organization, brain tumor was detected in more than 22000 patients in America in 2016. –National Brain Tumor Society estimates that every year 13000 patients die and 29000 patients suffer from primary brain tumors. World Health Organization reports states that there are 120 types of brain tumors which can be differentiated on the basis of size, shape, location and characteristics of brain tissue.

Some types of brain tumor such as Meningioma, Glioma, and Pituitary tumors are more common. Meningiomas are the most common type of tumors that originate in the thin membranes that surround the brain and spinal cord. The Glioma are a collection of tumors that grow within the substance of the brain and often mix with normal brain tissue. Pituitary tumors are abnormal growth of the brain cells. Pituitary tumors usually develop in the pituitary gland of the brain. These tumors can appear anywhere from the brain because of their intrinsic nature. Also, they do not have a uniform shape. They have different sizes, shapes, and contrasts. Gliomas and glioblastomas are the most common type of brain tumors among others. Gliomas are further differentiated in two LGG as low-grade gliomas and HGG as high-grade gliomas whereas glioblastomas are more severe, life threatening and more frequent in adults aging from 40 to 50.

1.4 Brain Tumor classification according to GRADE

In medical field, brain tumors are characterized by their GRADE. Tumor grade refers to a scale, ranging from I to IV, that reflects the potential aggressiveness of the tumor type. The GRADE of a tumor can be decided based on their behavior under microscopic observation:

- GRADE I: These tumors are known as benign. These tumors are slow growing, nonmalignant, and associated with long-term survival. This is the first stage of tumor.
- GRADE II: These tumors are known as malignant. GRADE I and GRADE II are known as low grade tumors. They can be malignant or nonmalignant.

- **GRADE III:** Their behavior is much different from normal cells and requires urgent treatment to cure. These tumors are malignant and often recur as higher-grade tumors.
- **GRADE IV:** These types of tumors show fastest growth rate and have worst abnormal behavior than other grades. These tumors reproduce rapidly and are very aggressive malignant tumors.

1.5 Diagnosis of Brain Tumor

To diagnose then brain tumor many methodologies can be used like CT(computerized tomography), MRI scan, BIOPSY, SPECT (Single Photon Emission Computed Tomography) scan.

Magnetic Resonance Imaging (MRI) is a medical imaging technique, which is extensively used for diagnosis and treatment of brain tumors in clinical practice. This method provides accurate images of the brain and is one of the most common and important methods for diagnosing and evaluating the patient's brain. In the field of Medical Detection Systems (MDS), MRI images provide better results than other imaging techniques such as Computed Tomography (CT), due to their higher contrast in soft tissue in humans.

The only best way is the detection of the tumor in the early stages and nipping the evil in the bud. The most efficient and precise tool used to identify brain tumors is Magnetic Resonance Imaging known as MRI. MRI has a different type of models including T1-weighted MRI, T1c, T2-weighted MRI, T2-Flair, etc. Most expert radiologist recommend MRI for detection a brain tumor as it is more effective and accurate regarding shape, size and location of brain tumor. MR Images detects an identifies the type of a brain tumor.

Benign tumors are basically confined to one specific area and they do not explicitly harm the structure of the brain. Converting MR Image into grey scale image where white part of the image describes the infected portion and grey scale portion shows the normal part of the brain. Detection of a benign tumor is relatively difficult from the detection of a malignant tumor. The reason behind it is the abnormality of the structure of the brain cannot be declared as benign tumor.

The structure of a normal brain can be different from the average structure without any reason. Identification and detecting the exact size, location and age of a brain tumor is solely dependent on skills and expertise of radiologist. Manual detection and classification of the brain tumor includes a lengthy manual procedure of radiologist with chances of human errors. By applying image processing with the help of machine learning algorithms.

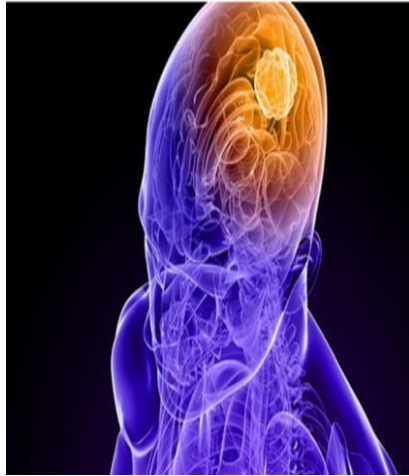


Fig.1.3 Brain Tumor Location

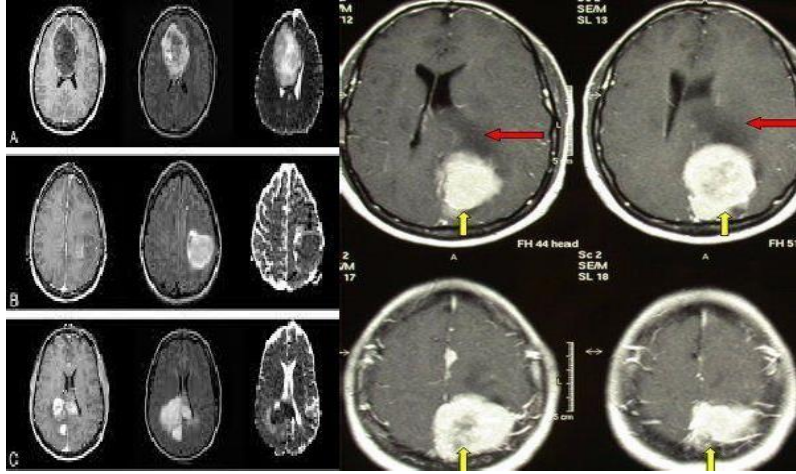
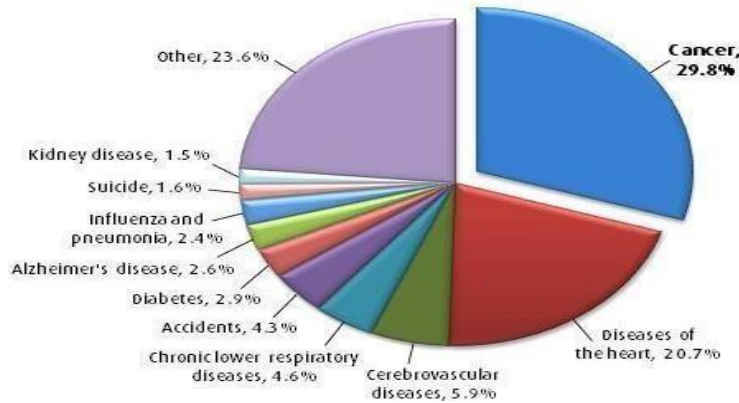


Fig 1.4 MRI images of Brain Tumor

Deep-learning-based techniques and methods are becoming popular in brain tumor segmentation studies, as their performance is superior in image analysis fields, such as object detection, image classification and semantic segmentation. These algorithmic models are classified into two categories: Generative and Discriminative. Generative models rely heavily on the basis of prior knowledge and use more of hand engineered components like Support Vector Machine (SVM) and other feature extraction techniques. Discriminative models have little prior knowledge and learn from the data given like Neural Networking, specifically Convolutional Neural Network (CNN) which is an efficient approach. Convolutional Neural Network is a cutting-edge method which is used for object and edge detection in images. In this project after minimal preprocessing on MR images. CNN is mostly used for brain tumor segmentation, classification, and prediction of survival time for patients. CNN is applied as an approach to train an effective system which can identify and classify a brain tumor. CNN is a layered structure which involves kernels or filters that work in a pipe line method to extract multiple complex features.

1.6 Motivation

Brain tumor represents an important group of diseases which has higher mortality rate (18.8%) and low survival rate (less than five years) than other common type of cancer



reported by New Philanthropy Capital (NPC). Figure 1.9 shows proportion of death due to cancer and other diseases reported by Canadian cancer society and Figure 1.9.1 shows comparison graph of various cancers in terms of survival rate.

Fig 1.5 Proportion of Death due to cancer

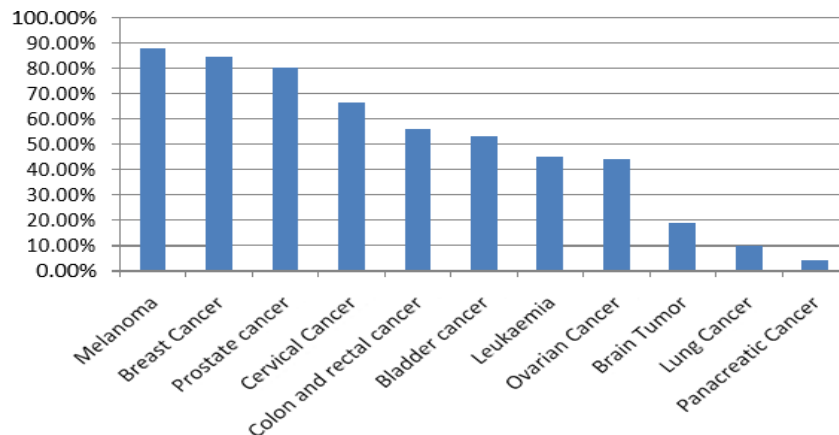


Fig1.6 Comparison in terms of survival rate

According to CBRTUS, it is estimated that most of the people are diagnosed for brain tumor per year around the world. It has become a serious issue in medical science to accurately detect brain tumor since it deals with human life. For this reason, Medical, Mathematical,

Computer science researchers have combined their knowledge and efforts to better diagnose the disease for effective treatment. Main motivation for this project work can be summarized as follows:

- Another motivation is to minimize human errors as it can possibly help physicians and radiologists with some segmenting software that will produce more accurate and fast results. Error minimization is necessary because it must deal with human lives.
- In these days, CAD (Computer Aided Diagnosis) software's are being used by radiologists for early detection so that treatment can start earlier. CAD helps medical expert in making final decision by providing second opinion.
- Automatic segmentation also encourages new application to be developed e.g. Content Based Image Retrieval (CBIR). In large medical databases, it is very easy to find out particular record for knowing medical history of patient and patient's family. Also, same type of tumor if treated by a doctor then previous record can help this time.

In this project, we are using convolution neural network, Google docs framework to detect and segment brain tumor types and boundaries. The MR images are combined with other image processing techniques to enhance accuracy of brain tumor segmentation.

CHAPTER 2

LITERATURE SURVEY

2.1 Literature Summary

In recent years, interest in designing tools for diagnosing brain tumors has been increasing. Mohammad Teshnelab et al [2] proposed a Convolutional Neural Network (CNN) has been used to detect a tumor through brain Magnetic Resonance Imaging (MRI) images. Images were first applied to the CNN. The accuracy of Softmax Fully Connected layer used to classify images obtained 98.67%. Also, the accuracy of the CNN is obtained with the Radial Basis Function (RBF) classifier 97.34% and the Decision Tree (DT) classifier, is 94.24%.

Halimesh Siar et al [3] proposed Alexnet model CNN is used to simultaneously diagnose MS(multiple sclerosis)and normal tumors, because the similarity between the tumor and MS. The CNN was able to accurately detect the network for simultaneous diagnosis of tumor and MS with accuracy of 96%.

F.samadi et al [4] recommended a deep learning-based supervised method is introduced to detect synthetic aperture radar (SAR) image changes. This method provided a dataset with an appropriate data volume and diversity for training the DBN using input images and the images obtained from applying the morphological operators on them. The detection performance of this method indicates the appropriately of deep learning based algorithms for solving the change detection problems.

Sindhia et al [5] proposed a Image process is employed within the medical tools for detection of tumor, solely MRI pictures aren't able to establish the tumorous region during this paper tend to a exploitation K-Means segmentation with pre-processing of image. That contains de-noising by Gaussian filter employed. It's expected that the experimental results of the projected system can offer higher lead to comparison to different existing systems.

Digvijay reddy et al [6] implemented a module in which MRI is taken as an input and tried to extract tumor cells from the input image. Pre-processing technique is used to remove noise from image. To this image, k-means clustering is applied and from this clustered image, skull was removed using morphological operations to identify tumor cells easily.

Md Rezwanul Islam et al [7] designed to detect brain tumor from MRI by integrated thresholding and morphological process with histogram based method and gives a thorough analysis. They have used BRATS database of magnetic resonance images. Their rate of successful detection of methodology is 86.84%.

Shanata Giraddi et al [8] proposed image processing techniques help in detecting the tumor images at an early stage. With the help of the scanned MRI images it is possible to detect the tumor and it's severity. The preprocessing which includes feature extraction followed by training the images on SVM classifier based on the extracted features and finally testing on the SVM classifier with various kernels with an accuracy of 90%.

Animesh Hazra et al [9] It needs to be detected at an early stage using MRI or CT scanned images when it is as small as possible because the tumor can possibly result to cancer. The proposed methodology consists of three stages i.e. pre-processing, edge detection and segmentation. Finally, the image is clustered using the k-means algorithm. Here they used MATLAB for the development of the project.

S.K.shil et al [10] proposed to detect and segment tumor from the MRI images designed an automated scheme. This scheme follows K means clustering for segmentation. To extract features and reduce the dimensions of features Discrete Wavelet Transform (DWT) followed by Principle Component Analysis (PCA) is respectively used. The reduced features were submitted to a Support Vector Machine (SVM) for classification. The performance evaluation of the proposed scheme is made by comparing the results with the ground truth of each processed image using Sensitivity, Specificity and Accuracy calculations.

M. Soltaninejad et al [11] an automated method is used to identify and categorize MRI images. This method is based on the Super Pixel Technique and the classification of each Super Pixel. Extremely randomized trees (ERT) classifier is compared with SVM to classify each super pixel into tumor and normal. This method has two datasets, which are 19 MRI FLAIR images and BRATS 2012 dataset. The results demonstrate the good performance of this method using ERT classifier.

K.Ramya et al [12] designed a watershed transformation technique is used with gradient magnitude with morphological open image and two important features is used as foreground and background to identify the tumor. First the Magnetic Resonance imaging (MRI) Scan of tumor is given as an input and it undergoes into watershed technique which is a topological boundary

dividing into two adjacent brain cells. With the gradient magnitude for segmentation technique the rate of inclination or declination of a tumor will be identified.

L. Sjlilagyi et al [13] implemented a multi-stage fuzzy c-means (FCM) framework for the automatic and accurate detection of brain tumors from multimodal 3D magnetic resonance image data. The proposed algorithm uses prior information at two points of the execution: (1) the clusters of voxels produced by FCM are classified as possibly tumorous and non-tumorous based on data extracted from train volumes; (2) the choice of FCM parameters is supported by train data as well.

Anupurba Nandi [14] The normal MR images are not that suitable for fine analysis, so segmentation is an important process required for efficiently analyzing the tumor images. Clustering is suitable for biomedical image segmentation as it uses unsupervised learning. This paper work uses K-Means clustering where the detected tumor shows some abnormality which is then rectified by the use of morphological operators along with basic image processing techniques to meet the goal of separating the tumor cells from the normal cells.

2.2 Summary of Literature Survey

From the literature survey it has been exposed that we can implement many computer aided techniques to determine the presence of lesion or not. There are various classification techniques for the detection of brain tumor have been reviewed. Calculation of tumor's area from MRI in fast, accurate and reproducible way is a tedious task. For distinguishing tumors from normal tissues by their image intensity, threshold-based or region growing techniques can be employed. However, the accuracy on brain tumor classification of the proposed automated methods is quite promising, but these approaches have not gained acceptance.

All methods are equally good for a particular type of image. Due to this, image classification remains a challenging problem in image processing. In fact, data classification algorithms typically employ two phases of processing — training and testing. At the first stage characteristic properties of image features are isolated and on the basis of this, a unique description of each classification category is created. And at subsequent testing stage, these feature space partitions are used to classify the images features to differentiate from each other. And in machine learning, image classification is used for both — supervised learning and unsupervised learning. Actually, Supervised and unsupervised classification is pixel-based classification process that creates square pixels and each pixel has a class. But object-based image classification groups pixels into representative shapes and sizes. Convolutional Neural Networks (CNN or ConvNet) are complex feed forward neural networks. CNNs are used for image classification and recognition because of its high accuracy. To address this problem, we propose a method of combining features from multiple layers in given CNN models. Moreover, already learned CNN models with training images are reused to extract features from multiple layers.

2.3 Problem Statement

Now a days we have seen most of the tumors are life threatening where brain tumor being one of them. As we know that brain tumor can be of any shape , size , location and intensity, therefore it is very difficult to detect tumor and diagnose it. The So automated identification of brain tumor from MRI images help alleviating the major issues and provide better results. Detection of brain tumor from the various symptoms of the patients has always been a major issue for the medical practitioner and pathologist for diagnosis and treatment planning. It is also fact that some tests may be time consuming and it gives workloads and difficulty for the pathologists to obtain the accuracy of the presence of the tumor.

“The manual identification of tumor from MRI images may vary from expert to expert depending on their expertise, due to lack of specific and accurate quantitative measures to classify the MRI images. So, we come up with an automated model to detect the brain tumor patients through MR images .”

2.4 Proposed Solution

Detection of Brain tumor at early stage is very difficult task for doctors to identify. MRI images are more prone to noise and other environmental interference. So it becomes difficult for doctors to identify tumor and their causes. So here we come up with the system, where system will detect brain tumor from images. So here we come up with the system, where system will detect brain tumor from images. User has to select the image, System will process the image by applying image processing steps. We applied cnn technique to detect tumor from brain image. In this method we applied image classification to detect tumor. Here we proposed image classification process and many image filtering techniques for accuracy. In this firstly we train the machine with some of the brain tumor images which predict that tumor is present or not. Based on the extracted features our model detects the tumor from MR images. Convolutional Neural Network (CNN) is used for learning how to segment and classify images. CNN extracts features directly from pixel images with minimal preprocessing.

It is a light deep neural network architecture designed for performing semantic preprocessing. The term –Convolutional in CNN denotes the mathematical function of convolution which is a special kind of linear operation wherein two functions are multiplied to produce a third function

which expresses how the shape of one function is modified by the other. In simple terms, two images which can be represented as matrices are multiplied to give an output that is used to extract features from the image.

2.5 Objectives

- To preprocesses the given MR image to enhance the image.
- To classify the affected tumor part for accurate examination of MR image.
- To detect the tumor and non tumor, CNN classifier algorithm is used.

CHAPTER 3

SOFTWARE REQUIREMENT SPECIFICATION

Requirement Specification

A Software Requirements Specification (SRS) is a description of a software system to be developed. It lays out function and non-functional requirements and may include set of use cases that describe user interactions that the software must provide.

The System Requirements Specification is a formal statement of the application's functional and operational requirements. It serves as a contract between the developer and the customer for whom the system is being developed.

3.1 FUNCTIONAL REQUIREMENTS

In software engineering, a functional requirement defines a function of a system and its component. A function is described as a set of inputs the behavior and the outputs. The functional requirements may be technical details, data manipulation and processing and other specific functionality that define what a system is supposed to accomplish. It includes the data and functional process requirements. These are statements of services the system should provide, how the system should react to inputs and how the system should behave situations. In some cases, the functional requirements may also explicitly state what the system should not do. The proposed system consists of the following functional requirements.

- The system should take the MR images of the patient as the input.
- The system must perform pre-processing of the MR images which includes removal irrelevant noise and background of undesired parts, smoothing regions of inner part and improvement in signal to noise ratio.
- The system should classify the MR image to detect the tumor.
- The system should extract the tumored feature with size, shape, composition and location of the tumor.

3.2 NON-FUNCTIONAL REQUIREMENTS

Non-functional requirements are requirements that are not directly concerned with the specific functions delivered by the system. Alternatively, they may define constraints on the system such as the capabilities of I/O devices and the data representations used in system interfaces. The basic nonfunctional requirements for any tumor segmentation technique are flexibility and scalability.

- Accuracy: The tumor classification must be precised with respect to the processing time, accuracy of tumor and identification of tumor location.
- Expeditious: The tumor classification must be a speedy process and must be efficient in detecting a tumorous or a non-tumorous image.

3.3 HARDWARE REQUIREMENTS

- PROCESSOR : Intel® Core™ i5-8250U CPU @ 1.60GHz 1.80 GHz
- RAM : 4 GB
- SYSTEM TYPE : 64-bit operating system

3.4 SOFTWARE REQUIREMENTS

- OPERATING SYSTEM : Windows 10
- INTERPRETER : PYTHON 3.6.1
- TOOLS : Anaconda (Python IDE)

CHAPTER 4

SYSTEM DESIGN

4.1 BLOCK DIAGRAM

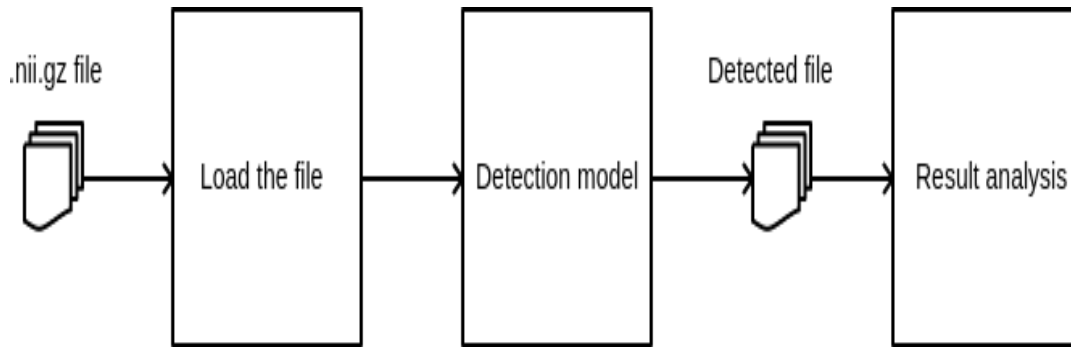


Fig 4.1 Steps to analyses the Result.

First .nii files are read & loaded to memory next the pre trained model will detect the tumor from the brain images which are loaded into memory and generates another .nii file with detected tumor markings, then using this generated file & 5 samples files are interpreted to give output result.

4.2 FLOW DIAGRAM

- Here we have created the application which will be helpful for both patient and doctor.
- Where they need use login page. If application is used client is a specialist they need to login first or they need to sign up it will to the signup page. In the event that client is a patient they needn't bother with any login or signup they can straightforwardly go to the Home page and can embed their MRI picture.
- This is the Home page here the client can browse and upload an MRI images. If user is an hospital related staff, first they need to login and upload an image or user is an patient they can directly upload image without login.
- After the uploaded MRI image is processed and sent to trained data and will give the desired output accordingly based on MRI image which we have uploaded.

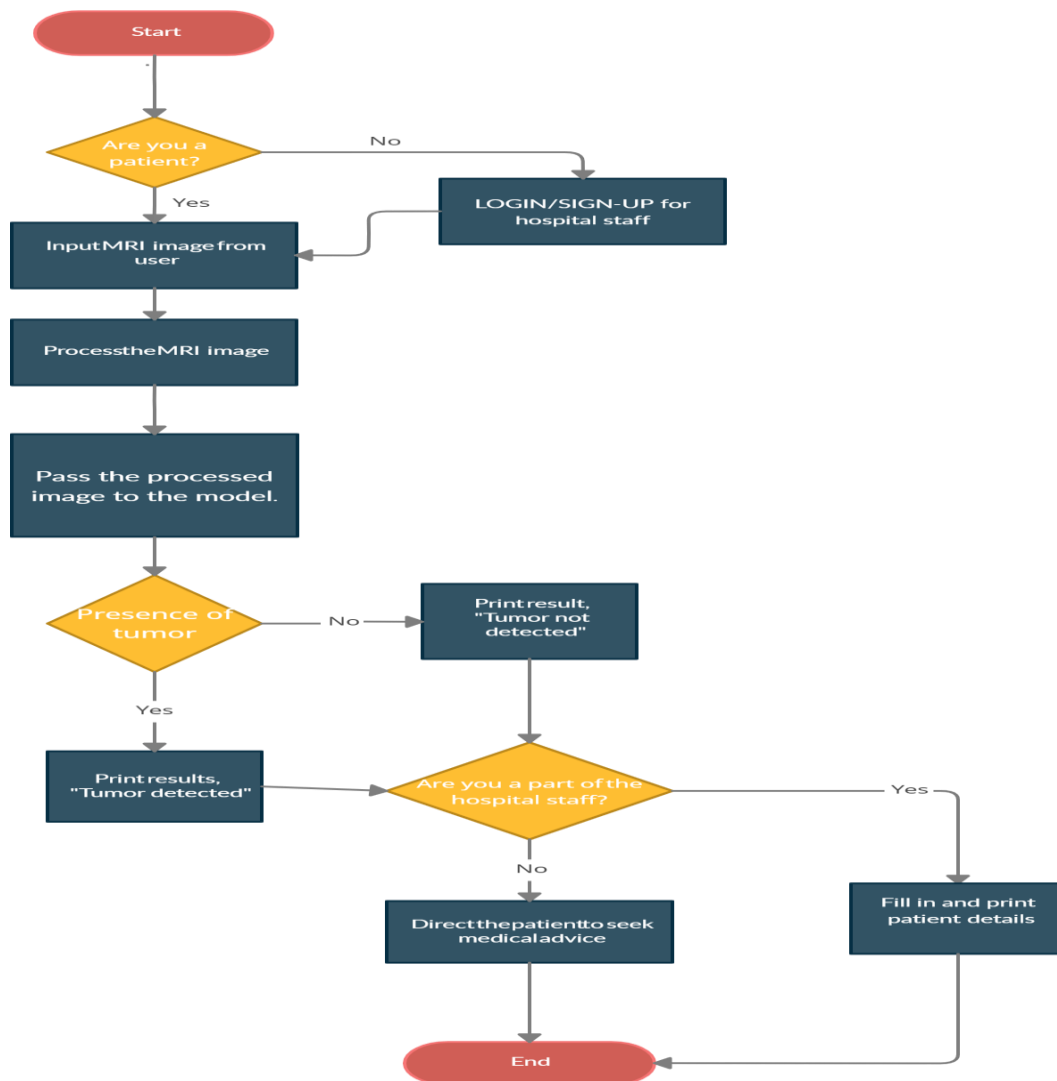


Fig 4.2 Flow Diagram Represents the Steps

- Then next page the specialist can fill the patient subtleties with tumor is distinguished or not and they can take print of patients report.
- Then they will provided with top neurologists in some top cities along with their Wikipedia information. Client can refer these information and consult them directly make appointment.

4.3 METHODOLOGY

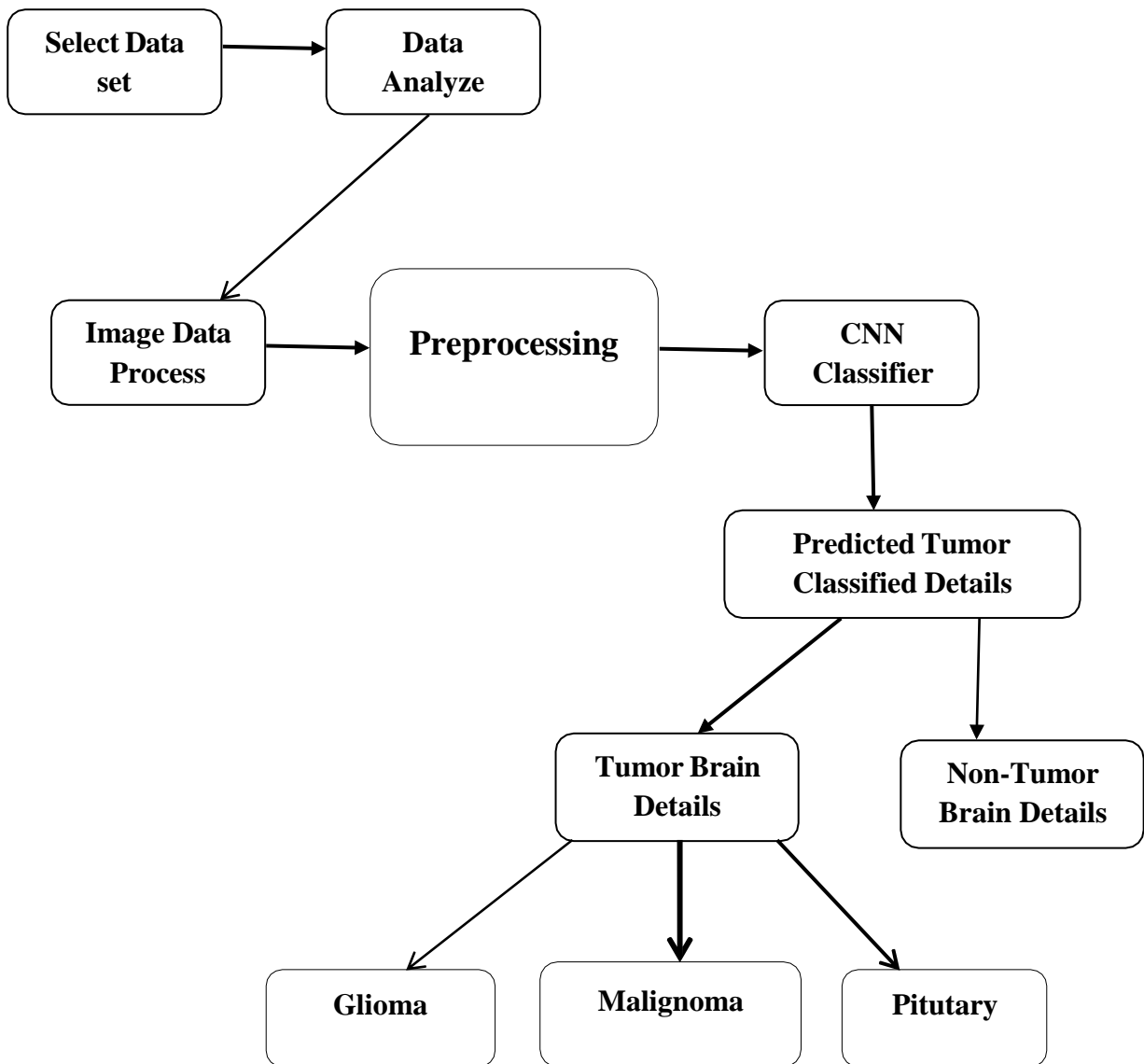


Fig 4.3 Block Diagram of Feature Extraction through Digital Image Processing

MR Images

Magnetic Resonance Imaging (MRI) is a non-invasive imaging technology that produces three dimensional detailed anatomical images. It is often used for disease detection, diagnosis, and treatment monitoring. MRI scanners are particularly well suited to image then on-bony parts or soft tissues of the body. MRI scans create diagnostic images without using harmful radiation. In the last few years the uses of Magnetic Resonance Imaging(MRI) scanners in medical field have grown.

Doctors may use MRI scans in diagnosing brain tumor and cancer. An MRI scan is an efficient way to look inside the human body without need to cut body. Most of the medical applications use 0.5to2.0 tesla. An electric current is passed through wired loops, which is used to create strong magnetic field. Magnets in other coils are used to send and receive radio waves. After excitation of molecules, energy is released. This is picked up by coil and then sent to a computer for final processing.

Preprocessing

Data pre-processing is an important step in the data mining process. The phrase "garbage in, garbage out" is particularly applicable to data mining and machine learning projects. Data- gathering methods are often loosely controlled, resulting in out-of-range values, impossible data combinations, missing values, etc. Analyzing data that has not been carefully screened for such problems can produce misleading results. Thus, the representation and quality of data is first and foremost before running an analysis. Often, data pre-processing is the most important phase of a machine learning project, especially in computational biology.

If there is much irrelevant and redundant information present or noisy and unreliable data, then knowledge discovery during the training phase is more difficult. Data preparation and filtering steps can take considerable amount of processing time. Data pre-processing includes cleaning, normalization, transformation, feature extraction and selection, etc. The product of data pre- processing is the final training set.

Data pre-processing may affect the way in which outcomes of the final data processing can be interpreted. This aspect should be carefully considered when interpretation of the results is a key point, such in the multivariate processing of chemical data.

Tasks of Data Pre-processing:

- I. Data cleansing
- II. Data editing
- III. Data reduction
- IV. Data wrangling

Classification

Classification or categorization is the process of classifying the objects or instances into a set of predefined classes. The use of machine learning approach makes a classifier system more dynamic. The goal of the ML approach is to build a concise model. This approach is to help to improve the efficiency of a classifier system. Every instance in a data set used by the machine learning and artificial intelligence algorithm is represented using the same set of features. These instances may have a known label, this is called the supervised machine learning algorithm. In contrast, if the labels are known, then its called the unsupervised. These two variations of the machine learning approaches are used for classification problems.

FLASK

Flask is an API of Python that allows to build up web-applications. It was developed by Armin Ronacher. Flask's framework is easy to learn because it has less base code to implement a simple web-Application. A Web-Application Framework or Web Framework is the collection of modules and libraries that helps the developer to write applications without writing the low-level codes such as protocols, thread management, etc. This web application can be some web pages, a blog, a wiki or go as big as a web-based calendar application or a commercial website.

Flask is based on WSGI (Web Server Gateway Interface) toolkit and Jinja2 template engine. Flask is part of the categories of the micro-framework. Micro-framework is normally framework with little to no dependencies to external libraries. This has pros and cons. Pros would be that the framework is light, there are little dependency to update and watch for security bugs, cons is that some time you will have to do more work by yourself or increase yourself the list of dependencies by adding plugins.

CHAPTER 5

IMPLEMENTATION

5.1 Installation Steps

5.1.1 Steps to Install Anaconda Navigator

1. Open an internet browser and Download Anaconda Installer.
2. Double click the installer to launch and click –Next.
3. Read the licensing terms and click –I Agree.
4. Select an install for –Just Me unless you’re installing for all users (which requires Windows Administrator privileges) and click –Next.
5. Select a destination folder to install Anaconda and click the –Next button
6. Choose whether to add Anaconda to your PATH environment variable. We recommend not adding Anaconda to the PATH environment variable, since this can interfere with other software. Instead, use Anaconda software by opening Anaconda Navigator or the Anaconda Prompt from the Start Menu
7. Choose whether to register Anaconda as your default Python. Unless you plan on installing and running multiple versions of Anaconda or multiple versions of Python, accept the default and leave this box checked.
8. Click the –Install button and click the –Next button
9. After a successful installation you will see the –Thanks for installing Anaconda dialog box:

5.1.2 To Install Jupyter Notebook

1. Go to Anaconda PROMPT
2. Type – conda install jupyter
3. Type y for yes when prompted. Once Jupyter is installed, type the command below into the Anaconda Prompt to open the Jupyter notebook file browser and start using Jupyter notebooks–
–jupyter notebook

5.1.3 To Install Spyder

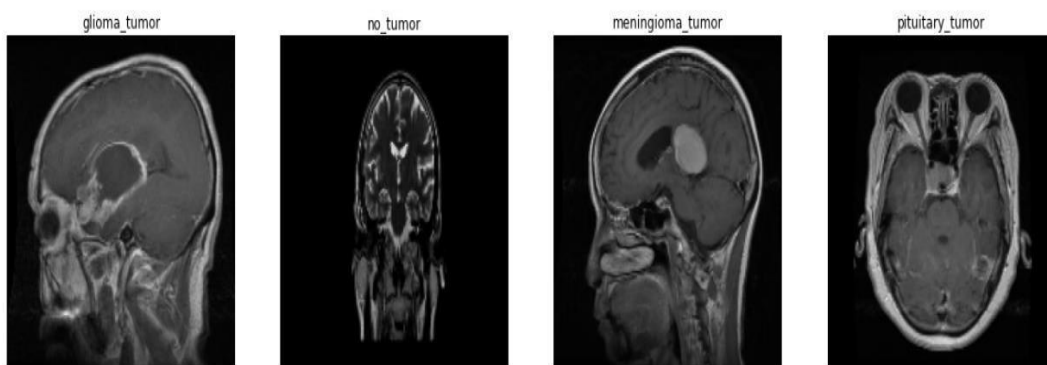
1. Go to Anaconda PROMPT
2. Type – conda install spyder
3. Type y for yes when prompted. Once Spyder is installed, type the command below into the Anaconda Prompt to open the Spyder and start using –Spyder

5.2 Dataset Description

The images employed for experimentation were obtained from the kaggle. The dataset is a targeted data collection containing MRI-multi-sequence images from few patients with glioma tumor of 800 images, 800 images of meningioma tumor ,800 images of pituitary tumor and 800 images which has no tumor . The total number of images in this dataset is 3500. Sample of our dataset as follows:

```
In [8]: 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
```

Sample Image From Each Label



Description: In the above code snippet labelling of the images are done.

5.3 Implementation Details

Implementation literally means to put into effect or carry out. The system implementation face of the software deals with the translation of the design specifications into the source code. The ultimate goal of the implementation is to write the source code and the internal documentation so that it can be verified easily. The code and the documentation should be written in a manner that uses testing, debugging and modification. A post-implementation review is an evaluation of the extent to which the system accomplishes stated objectives and actual project costs exceed initial estimates. It is usually a review of major problems that need converting and those that surface during the implementation stage.

5.3.1. Packages used

The packages that are used in this project are as follows:

- Numpy
- Matplotlib
- Tensorflow
- Keras
- Pandas
- Seaborn
- Sklearn

1. **Numpy** : It is a library for the Python programming language, adding support for large, multi-dimensional arrays and matrices, along with a large collection of high-level mathematical functions to operate on these arrays NumPy addresses the slowness problem partly by providing multidimensional arrays and functions and operators that operate efficiently on arrays, using these requires rewriting some code, mostly inner loops, using NumPy.
2. **Matplotlib**: Matplotlib is a plotting library for the Python programming language and its numerical mathematics extension NumPy. It provides an object-oriented API for embedding plots into applications using general-purpose GUI toolkits like Tkinter, wxPython, Qt, or GTK+. There is also a procedural "pylab" interface based on a state machine (like OpenGL), designed to closely resemble that of MATLAB, though its use is discouraged. SciPy makes use of Matplotlib.

3. **TensorFlow:** TensorFlow open-source software library for data flow and differentiable programming across a range of tasks. It is a symbolic math library, and is also used for machine learning applications such as neural networks. The applications for which TensorFlow is the foundation, are automated image captioning software, such as Deep Dream.
4. **Keras:** Keras is a deep learning API written in Python, running on top of the machine learning platform TensorFlow. It was developed with a focus on enabling fast experimentation. Being able to go from idea to result as fast as possible is key to doing good research.
5. **Pandas :** Pandas is one of the tools in Machine Learning which is used for data cleaning and analysis. It has features which are used for exploring, cleaning, transforming and visualizing from data. Pandas is an open-source python package built on top of Numpy developed by Wes McKinney.
6. **Seaborn :** Seaborn is a plotting library that offers a simpler interface, sensible defaults for plots needed for machine learning. Seaborn is a data visualization library built on top of matplotlib and closely integrated with pandas data structures in Python. Visualization is the central part of Seaborn which helps in exploration and understanding of data.
7. **Sklearn :** Scikit-learn (Sklearn) is the most useful and robust library for machine learning in Python. It provides a selection of efficient tools for machine learning and statistical modeling including classification, regression, clustering and dimensionality reduction via a consistent interface in Python.

5.4 Data Preparation

Data preparation (also referred to as –data preprocessing) is the process of transforming raw data so that data scientists and analysts can run it through machine learning algorithms to uncover insights or make predictions. Most machine learning algorithms require data to be formatted in a very specific way, so datasets generally require some amount of preparation before they can yield useful insights. Some datasets have values that are missing, invalid, or otherwise difficult for an algorithm to process. If data is missing, the algorithm can't use it. If data is invalid, the algorithm produces less accurate or even misleading outcomes. Some datasets are relatively clean but need to be shaped (e.g.,

aggregated or pivoted) and many datasets are just lacking useful business context (e.g., poorly defined ID values), hence the need for feature enrichment. Good data preparation produces clean and well-curated data which leads to more practical, accurate model outcomes.

```
In [5]: labels = ['glioma_tumor', 'no_tumor', 'meningioma_tumor', 'pituitary_tumor']
```

We start off by appending all the images from the directories into a Python list and then converting them into numpy arrays after resizing it.

```
In [7]: x_train = []
y_train = []
image_size = 150
for i in labels:
    folderPath = os.path.join('/', '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)

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

100%	826/826	[00:03<00:00, 265.97it/s]
100%	395/395	[00:01<00:00, 365.76it/s]
100%	529/529	[00:01<00:00, 304.75it/s]
100%	762/762	[00:02<00:00, 255.58it/s]

Description: The above code snippet shows preprocessing of the data, here we appended all the images from the local directories into python list and converting them into numpy arrays after resizing it.

5.5 Image Preprocessing

The goal of this step is to make your data ready for the ML model to make it easier to analyze and process computationally, as it is with images. Image processing could be simple tasks like image resizing. In order to feed a dataset of images to a convolutional network, they must all be the same size. Some of techniques involved are.

- **Convert colour images to grayscale to reduce computation complexity:**

In certain problems you'll find it useful to lose unnecessary information from your images to reduce space or computational complexity. For example, converting your coloured images to grayscale images. This is because in many objects, color isn't necessary to recognize and interpret an image. Grayscale can be good enough for recognizing certain objects. Because color images contain more information than black and white images, they can add unnecessary complexity and take up more space in memory.

- **Standardize images:** One important constraint that exists in some machine learning algorithms, such as CNN, is the need to resize the images in your dataset to a unified dimension. This implies that our images must be preprocessed and scaled to have identical widths and heights before fed to the learning algorithm.

- **Data augmentation:** Another common pre-processing technique involves augmenting the existing dataset with perturbed versions of the existing images. Scaling, rotations and other affine transformations are typical. This is done to enlarge your dataset and expose the neural network to a wide variety of variations of your images. This makes it more likely that your model recognizes objects when they appear in any form and shape.

```
In [31]: datagen = ImageDataGenerator(  
        rotation_range=30,  
        width_shift_range=0.1,  
        height_shift_range=0.1,  
        zoom_range=0.2,  
        horizontal_flip=True)  
  
        datagen.fit(X_train)  
        X_train.shape
```

```
Out[31]: (2780, 150, 150, 3)
```

Description: In the above code snippet ImageDataGenerator is applied to the model.

5.6 Modules Implementation

5.6.1. Interface design module

Flask: Flask is an API of Python that allows to build up web-applications. A Web-Application Framework or Web Framework is the collection of modules and libraries that helps the developer to write applications without writing the low-level codes such as protocols, thread management, etc.

```
1 from flask import Flask, render_template, request, redirect  
2 import sqlite3  
3 import numpy as np  
4  
5 app = Flask(__name__)  
6  
7 model = keras.models.load_model("assets/new/mymodel1.h5")  
8 app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER  
9  
10 if __name__ == '__main__':  
11  
12     app.run(debug=True, port=8000)  
13
```

```
42
43 @app.route("/mainPage", methods=["GET", "POST"])
44 def mainPage():
45
46     if request.method == 'POST':
47         file = request.files['mri']
48         if 'mri' not in request.files:
49             return render_template('btd.html', ses=ses, error="No File Found")
50         # if user does not select file, browser also
51         # submits an empty part without filename
52         if file.filename == '':
53             return render_template('btd.html', ses=ses, error="Filename Not Found")
54         if file:
55             filename = secure_filename(file.filename)
56             filepath = os.path.join(app.config['UPLOAD_FOLDER'], 'work.jpg')
57             file.save(filepath)
58             img = Image.open('./static/work.jpg')
59             dim = (150,150)
60             x = np.array(img.resize(dim))
61             x = x.reshape(1,150,150,3)
62             answ = model.predict_on_batch(x)
63             classification = np.where(answ == np.amax(answ))[1][0]
64             predicted_results = names[classification]+' Detected'
65             return render_template("result.html", filename=filename, predicted_results=predicted_results, ses=ses, error="")
66         return render_template("btd.html", ses=ses, error="")
```

Description: From the above code snippet we say that while testing the image it will be processed and sent to training model where based on that image will be predicted whether that MR image has brain tumor or non tumor and result will be obtained.

```
1 # create database using sqlite3
2 import sqlite3
3
4 db_local = 'patients.db'
5 conn = sqlite3.connect(db_local)
6 c = conn.cursor()
7
8 c.execute(""" CREATE TABLE doctors
9 (
10 dId int PRIMARY KEY, name varchar(20), contact varchar(10), description TEXT, website TEXT, city varc
11 )
12 """)
13
14 conn.commit()
15 conn.close()
```

Description: This is used and patient details to create a SQLite database for adding and saving data related to doctor

```
128 @app.route("/signup", methods=["GET", "POST"])
129 def signup():
130
131     if request.method == "POST":
132         conn = sqlite3.connect(db_local)
133         c = conn.cursor()
134         username = request.form['username']
135         password = request.form['password']
136         fullname = request.form['fullname']
137         emailid = request.form['emailid']
138         hname = request.form['hname']
139         position = request.form['position']
140         c.execute("SELECT * FROM logindb WHERE username = ?", [username])
141         if c.fetchone() is not None:
142             flash("The username is already taken")
143             return render_template('signup.html')
144         elif (checkpass(password) == True):
145             c.execute(
146                 "INSERT INTO logindb(username, password, fullname, emailid, hname, position) VALUES (?, ?, ?, ?, ?, ?)", (username, password,
147                 fullname, emailid, hname, position))
148             conn.commit()
149             conn.close()
150             return redirect(url_for('login'))
151         else:
152             return render_template('signup.html', ses=ses, error="The password is weak, please try again!")
153     conn.commit()
154     conn.close()
```

Description: The above snippet is used to create the signup page for doctor or hospital staff where they can register in this application and sign in and check for result using MRI images.

```
207 @ app.route("/doctors/<city>")
208 def doctors(city):
209     conn = sqlite3.connect(db_local)
210     c = conn.cursor()
211     city = str(city)
212     c.execute("select * from doctors where city=? ",
213              (city,))
214     data = c.fetchall()
215     conn.close()
216     return render_template('doctors.html', data=data)
217
218
```

Description: The above snippet is used, where the patient will be recommended with neurologist and information about that them will be provided, where patients can select the neurologist based on city.

CHAPTER 6

TRAINING AND TESTING

6.1 Overview

The training model is used to run the input data through the algorithm to correlate the processed output against the sample output. The result from this correlation is used to modify the model. This iterative process is called -model fitting. The accuracy of the training dataset or the validation dataset is critical for the precision of the model. There are several types of machine learning models, of which the most common ones are supervised and unsupervised learning. Whereas Testing is a critical element which assures quality and effectiveness of the proposed system in meeting its objectives. Testing is done at various stages in the System designing and implementation process with an objective of developing a transparent, flexible and secured system. Testing is an integral part of software development. Testing process, in a way certifies, whether the product, that is developed, complies with the standards, that it was designed to. Testing process involves building of test cases, against which, the product has to be tested.

6.2 Training Process

An algorithm takes a set of data known as -training data as input. The learning algorithm finds patterns in the input data and trains the model for expected results (target). The output of the training process is the machine learning model. After the pre-processing of the data, model building comes to picture. The Model is now trained by considering various attributes.

6.2.1 Transfer Learning

Deep convolutional neural network models may take days or even weeks to train on very large datasets. A way to short-cut this process is to re-use the model weights from pre-trained models that were developed for standard computer vision benchmark datasets, such as the ImageNet image recognition tasks. Top performing models can be downloaded and used directly, or integrated into a new model for your own computer vision problems.

Here we used the EfficientNetB0 model which will use the weights from the ImageNet dataset.

The include_top parameter is set to False so that the network doesn't include the top layer/ output layer from the pre-built model which allows us to add our own output layer depending upon our use case.

```
In [11]:  
effnet = EfficientNetB0(weights='imagenet',include_top=False,input_shape=(image_size,image_size,3))  
  
Downloading data from https://storage.googleapis.com/keras-applications/efficientnetb0_notop.h5  
16711680/16705208 [=====] - 0s 0us/step
```

6.2.2 Adding CNN Layers

The first layer of a Convolutional Neural Network is always a Convolutional Layer. Convolutional layers apply a convolution operation to the input, passing the result to the next layer. A convolution converts all the pixels in its receptive field into a single value. For example, if you would apply a convolution to an image, you will be decreasing the image size as well as bringing all the information in the field together into a single pixel. The final output of the convolutional layer is a vector. Based on the type of problem we need to solve and on the kind of features we are looking to learn, we can use different kinds of convolutions.

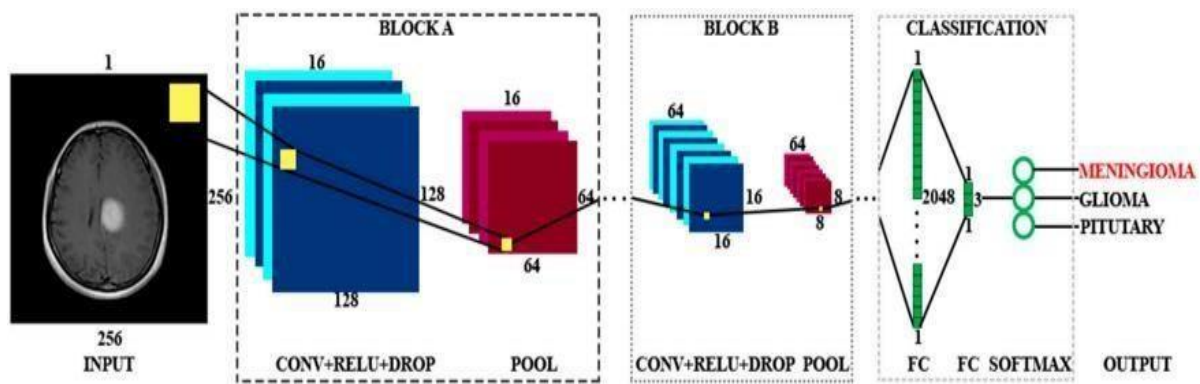


Fig 6.1 Schematic representation of convolutional neural network (CNN) architecture

```
In [12]:
model = effnet.output
model = tf.keras.layers.GlobalAveragePooling2D()(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)
```

- **GlobalAveragePooling2D** : This layer acts similar to the Max Pooling layer in CNNs, the only difference being is that it uses the Average values instead of the Max value while pooling. This really helps in decreasing the computational load on the machine while training.
- **Dropout** : This layer omits some of the neurons at each step from the layer making the neurons more independent from the neighbouring neurons. It helps in avoiding overfitting. Neurons to be omitted are selected at random. The rate parameter is the likelihood of a neuron activation being set to 0, thus dropping out the neuron
- **Dense** : This is the output layer which classifies the image into 1 of the 4 possible classes. It uses the softmax function which is a generalization of the sigmoid function.

6.2.3 Summarizing and Compiling the Model

Keras provides a way to summarize a model. The summary is textual and includes information about: The layers and their order in the model. The output shape of each layer. The number of parameters (weights) in each layer.

```
In [13]:
model.summary()
```

⌵ Hide output

```
Model: "model"
-----
Layer (type)                 Output Shape              Param #   Connected to
-----
input_1 (InputLayer)         [(None, 150, 150, 3)] 0
-----
rescaling (Rescaling)        (None, 150, 150, 3) 0      input_1[0][0]
-----
normalization (Normalization) (None, 150, 150, 3) 7      rescaling[0][0]
-----
stem_conv_pad (ZeroPadding2D) (None, 151, 151, 3) 0      normalization[0][0]
```

Before training the model we need to compile it and define the loss function, optimizers, and metrics for prediction. We compile the model using `. compile()` method. Optimizer, loss, and metrics are the necessary arguments.

```
In [14]: model.compile(loss='categorical_crossentropy', optimizer = 'Adam', metrics= ['accuracy'])
```

Callbacks : Callbacks can help you fix bugs more quickly, and can help you build better models. They can help you visualize how your model's training is going, and can even help prevent overfitting by implementing early stopping or customizing the learning rate on each iteration. By definition, "A callback is a set of functions to be applied at given stages of the training procedure. You can use callbacks to get a view on internal states and statistics of the model during training."

Here we used **TensorBoard**, **ModelCheckpoint** and **ReduceLROnPlateau** callback functions

- **TensorBoard Callback** is a visualization tool provided with TensorFlow. This callback logs events for TensorBoard, including: Metrics summary plots. Training graph visualization.
- **ModelCheckpoint Callback** class allows you to define where to checkpoint the model weights, how the file should be named and under what circumstances to make a checkpoint of the model. The API allows you to specify which metric to monitor, such as loss or accuracy on the training or validation dataset.
- **ReduceLROnPlateau Callback** is to reduce the learning rate when a metric has stopped improving. This callback monitors a quantity and if no improvement is seen for a patience number of epochs, the learning rate is reduced by factor value ($\text{new_lr} = \text{lr} * \text{factor}$).

```
In [15]: 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_de
lta = 0.001,
                                mode='auto', verbose=1)
```

6.2.4 Training the Model

A training model is a dataset that is used to train an ML algorithm. It consists of the sample output data and the corresponding sets of input data that have an influence on the output. The training model is used to run the input data through the algorithm to correlate the processed output against the sample output. The result from this correlation is used to modify the model.

This iterative process is called -model fitting|. The accuracy of the training dataset or the validation dataset is critical for the precision of the model.

Model training in machine language is the process of feeding an ML algorithm with data to help identify and learn good values for all attributes involved. There are several types of machine learning models, of which the most common ones are supervised and unsupervised learning.

```
In [20]: history = model.fit(X_train,y_train,validation_split=0.1, epochs =15, verbose=1, batch_size=32,
                             callbacks=[tensorboard,checkpoint,reduce_lr])

Epoch 00012: val_accuracy improved from 0.96018 to 0.96903, saving model to effnet.h5
Epoch 13/15
64/64 [=====] - 239s 4s/step - loss: 0.0024 - accuracy: 0.9995 - val_loss: 0.1162 - val_accuracy: 0.9690

Epoch 00013: val_accuracy did not improve from 0.96903
Epoch 14/15
64/64 [=====] - 238s 4s/step - loss: 0.0084 - accuracy: 0.9975 - val_loss: 0.1024 - val_accuracy: 0.9646

Epoch 00014: val_accuracy did not improve from 0.96903

Epoch 00014: ReduceLROnPlateau reducing learning rate to 2.70000040931627e-05.
Epoch 15/15
64/64 [=====] - 237s 4s/step - loss: 0.0034 - accuracy: 0.9990 - val_loss: 0.1011 - val_accuracy: 0.9690

Epoch 00015: val_accuracy did not improve from 0.96903
```

An **epoch** is a term used in machine learning and indicates the number of passes of the entire training dataset the algorithm has completed.

$$d * e = i * b$$

- d = dataset size
- e = number of epochs
- i = number of iterations
- b = batch size

6.2.5 Epochs vs. Training and Validation Accuracy/Loss

The below snippet plots the graph of the training loss vs. validation loss over the number of epochs. This will help the developer of the model to make informed decisions about the architectural choices that need to be made.

```
In [25]: filterwarnings('ignore')

epochs = [i for i in range(15)]
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()
```

Output :

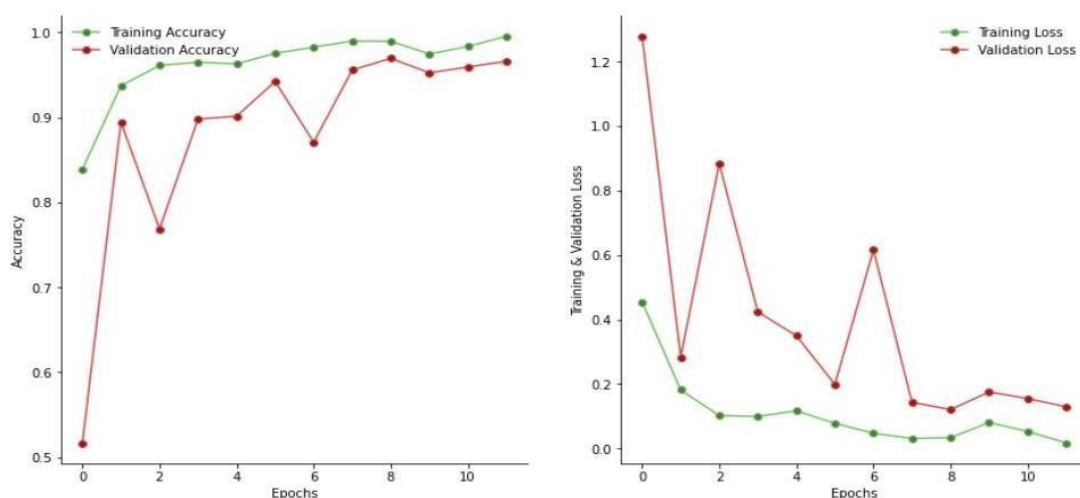


Fig 6.2 Graph showing the relationship between Epochs vs. Training and Validation Accuracy/Loss

Accuracy calculation :

$$\text{Accuracy} = (\text{TP} + \text{TN}) / (\text{TP} + \text{TN} + \text{FP} + \text{FN})$$

where: TP = True positive;

FP = False positive;

TN = True negative;

FN = False negative

6.3 Prediction

We've used the *argmax function* as each row from the prediction array contains four values for the respective labels. The **maximum** value which is in each row depicts the predicted output out of the 4 possible outcomes. So with *argmax*, I'm able to find out the index associated with the predicted outcome. The **argmax** function returns the index of the maximum value of a Numpy array. It's somewhat similar to the Numpy maximum function, but instead of returning the maximum value, it returns the index of the maximum value.

```
In [18]: pred = model.predict(X_test)
pred = np.argmax(pred,axis=1)
y_test_new = np.argmax(y_test,axis=1)
```

6.4 Evaluation

In this the label assigned as follows,

- 0 - Glioma Tumor
- 1 - No Tumor
- 2 - Meningioma Tumor
- 3 - Pituitary Tumor

Classification report is used to measure the quality of predictions from a classification algorithm. The report shows the main classification metrics precision, recall and f1-score on a per-class basis.

The metrics are calculated by using true and false positives, true and false negatives. The reported averages include macro average (averaging the unweighted mean per label), weighted average (averaging the support-weighted mean per label), and accuracy.

```
In [19]: print(classification_report(y_test_new, pred))
```

	precision	recall	f1-score	support
0	0.97	0.91	0.94	93
1	0.96	1.00	0.98	51
2	0.92	0.95	0.93	96
3	0.98	0.98	0.98	87
accuracy			0.95	327
macro avg	0.96	0.96	0.96	327
weighted avg	0.95	0.95	0.95	327

6.4.1 Heatmap of the Confusion Matrix

Heat map is used to draw the confusion matrix, and use the cmap parameter to set the heat map to the larger the value, the darker the color, which can help us clearly see the prediction effect of each category, and we can see which category is good or bad at a glance.

```
In [20]: 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()
```

Below diagram shows the heatmap of the Confusion Matrix classified as glioma_tumor, no_tumor, meningioma_tumor and pituitary_tumor.

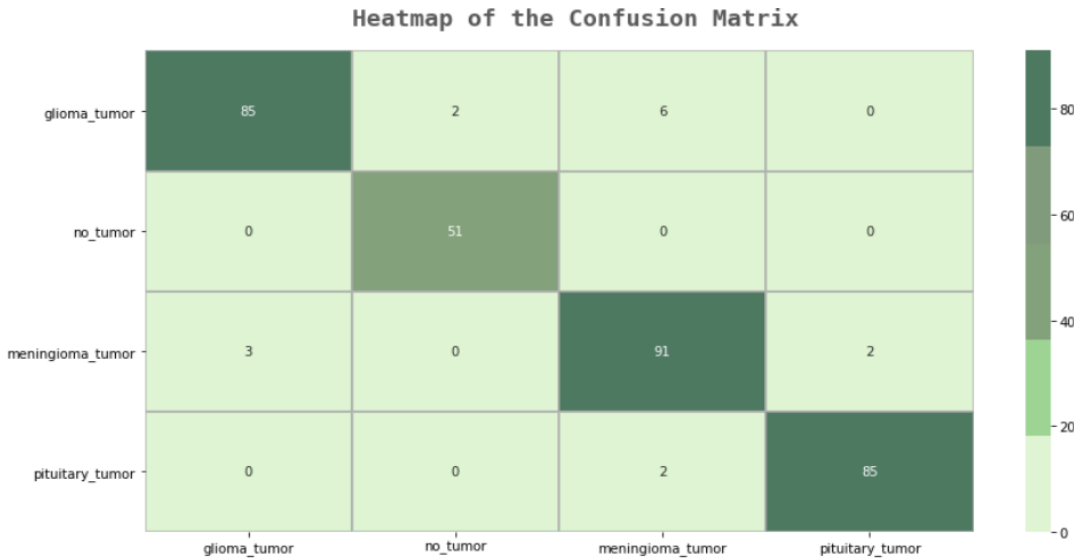


Fig 6.3 Heatmap of the Confusion Matrix

6.5 Testing process

Testing is the process of evaluating a system or its component(s) with the intent to find whether it satisfies the specified requirements or not. In simple words, testing is executing a system in order to identify any gaps, errors, or missing requirements in contrary to the actual requirements. In order to test a machine learning algorithm, tester defines three different datasets viz. Training dataset, validation dataset and a test dataset (a subset of training dataset). We've made the Widgets in which we can upload images from our local machine and predict whether the MRI scan has a Brain Tumour or not and to classify which Tumor it is

```
In [21]: 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}')
```

The above snippet is able to predict the type of tumor


```
In [22]: uploader = widgets.FileUpload()  
display(uploader)
```

Upload (1)

This is where we can upload the image by clicking on the **Upload** button

```
In [23]: 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

After uploading the image, you can click on the **Predict** button below to make predictions

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

Upload (1)

After uploading the image, you can click on the **Predict** button below to make predictions:

```
▶ 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 it is a Glioma Tumor

Here the Model predicts the present of tumor that is Glioma Tumor after uploading the image manually from the local drive

6.6 Failure Cases

Case I : Brain Tumor Segmentation from MRI

It is challenging to precisely model probabilistic distributions of brain tumors. In contrast, discriminative approaches extract features from images and associate the features with the tissue classes using discriminative classifiers. They often require a supervised learning set-up where images and voxel-wise class labels are needed for training.

Case II : Augmentation for Training and Testing

Considering the image acquisition process, one underlying anatomy can be observed with different conditions, such as various spatial transformations and intensity noise. Therefore, an acquired image can be seen as only one of many possible observations of the target. Directly applying CNNs to the single observed image may lead the result to be biased toward the specific transformation and noise in the given observation

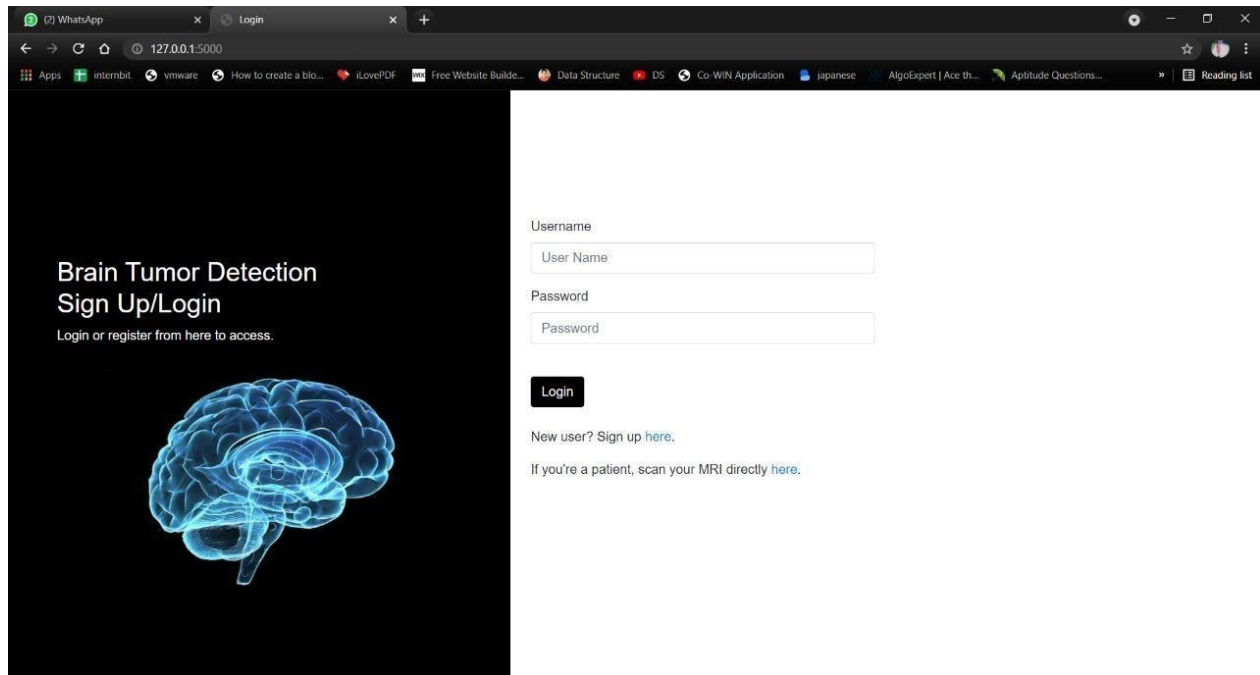
Case III : Uncertainty Estimation for CNNs

Uncertainty information can come from either the CNN models or the input images. For model-based (epistemic) uncertainty, exact Bayesian modeling is mathematically grounded but often computationally expensive and hard to implement.

CHAPTER 7

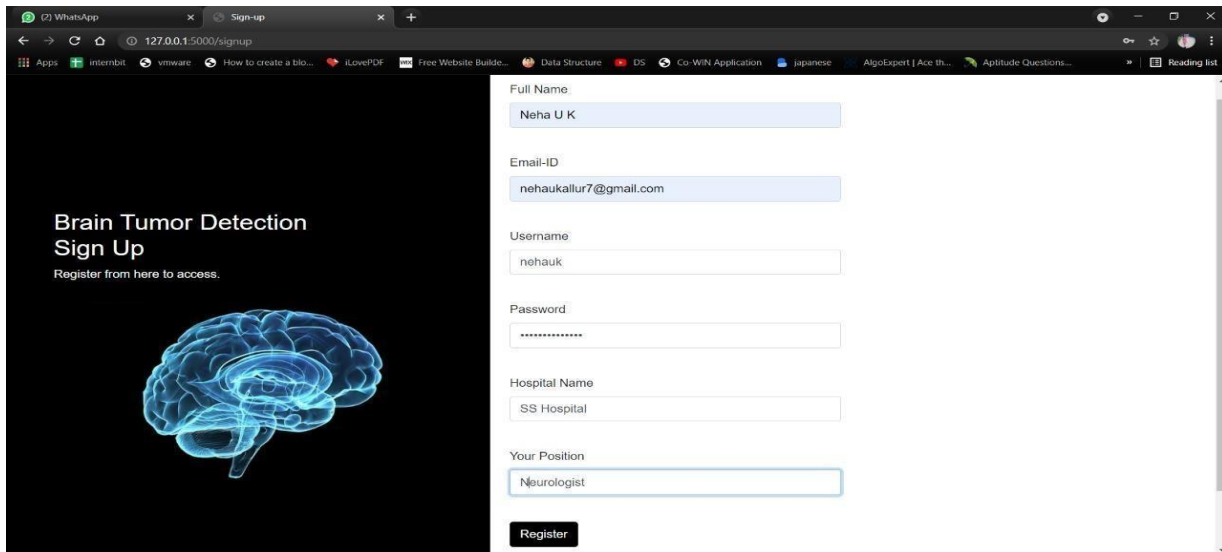
RESULTS & DISCUSSIONS

7.1 SNAPSHOTS



7.1 Login Page

Description: In this login page If client is a specialist they need to login first or they need to sign up it will to the signup page. In the event that client is a patient they needn't bother with any login or signup they can straightforwardly go to the Home page and can embed their MRI picture.



Brain Tumor Detection
Sign Up
Register from here to access.

Full Name
Neha U K

Email-ID
nehaukallur7@gmail.com

Username
nehauk

Password

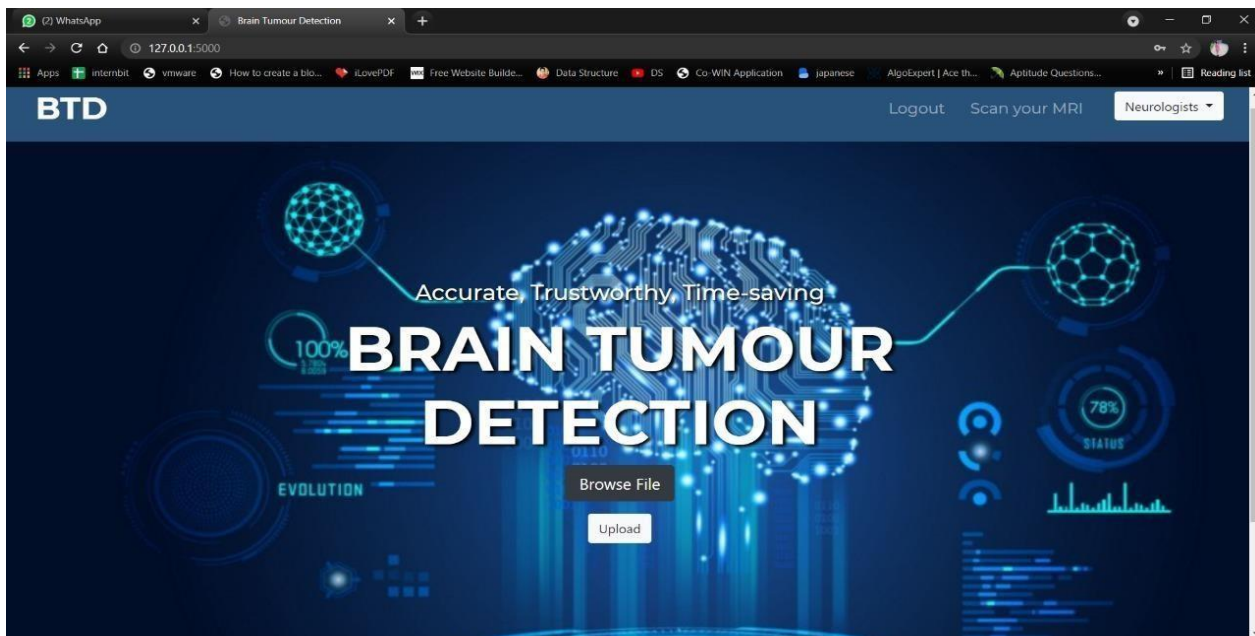
Hospital Name
SS Hospital

Your Position
Neurologist

Register

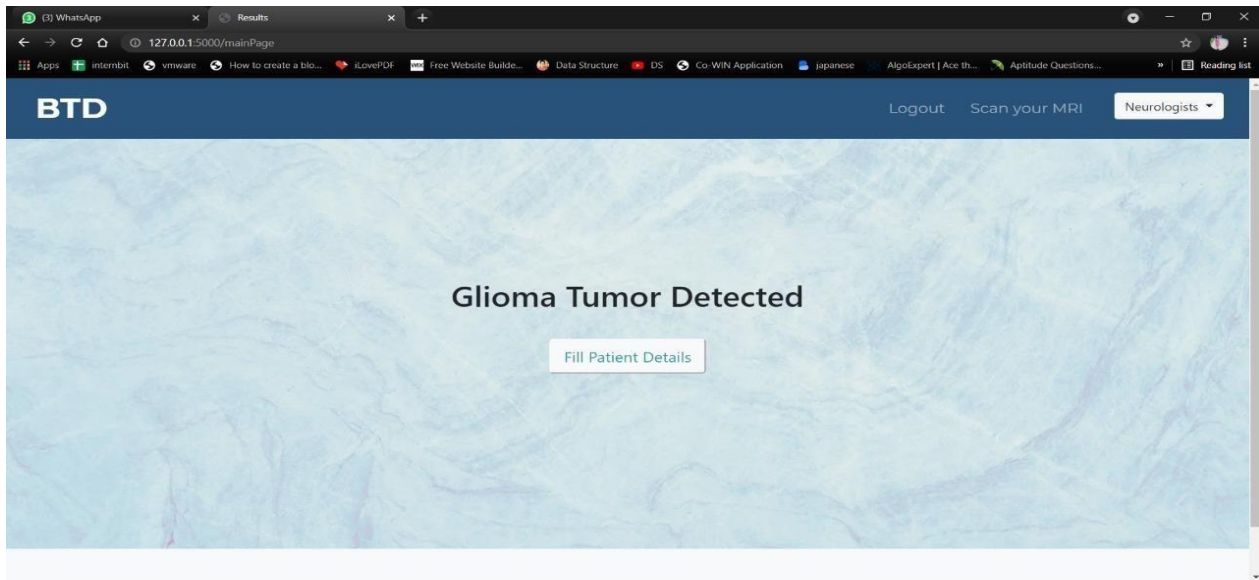
7.2 signup Page

Description: This is the signup page If the User Is an Hospital related staff they need to signup here.



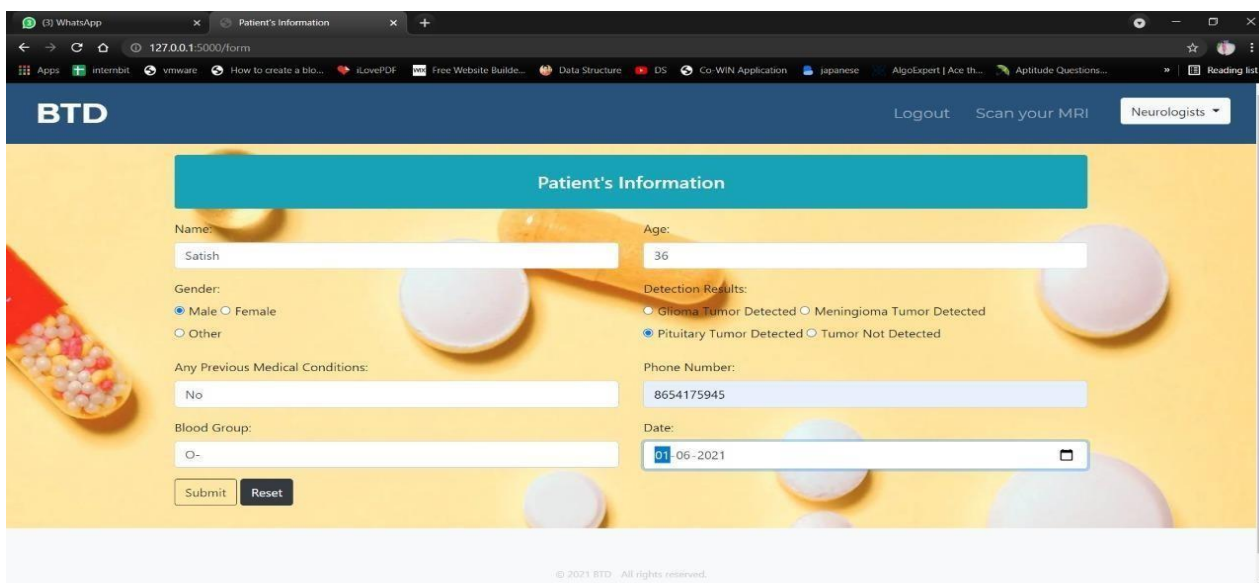
7.3 Home Page

Description: This is the Home page here the client can browse and upload an MRI images. If user is an hospital related staff, first they need to login and upload an image or user is an patient they can directly upload image without login. In this page we can also know about top neurologist in top cities.



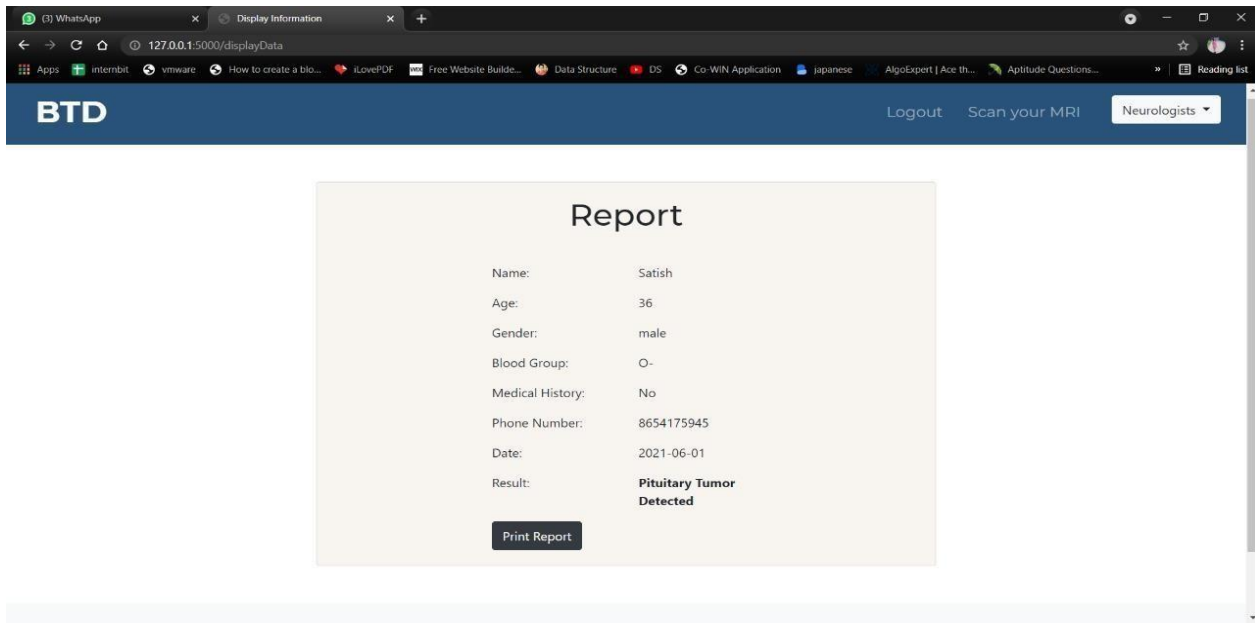
7.4 Result Page

Description: In this result page we can know whether the patient have Brain tumor or not. Glioma, Meningioma, Pituitary these are the 3 sorts of tumor this framework can identify and if tumor is absent it will result No tumor distinguished.



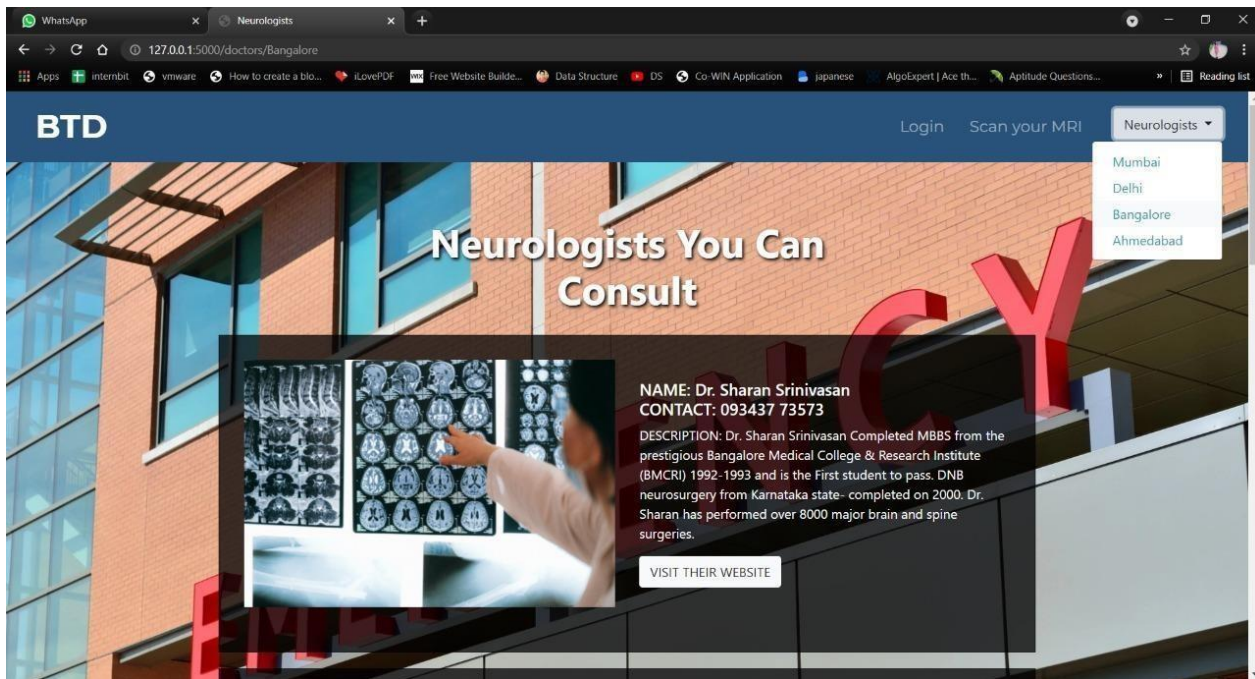
7.5 Patient Information Page

Description: In this page the specialist can fill the patient subtleties with tumor is distinguished or not.



7.6 Report Page

Description: In this Page we can take print of patients report.



7.7 Neurologist Reference Page

Description: In this page we can know about Top neurologists in some top cities along with their Wikipedia information. Client can refer these information and consult them.

CONCLUSION

In this paper, we developed a model based on CNN deep learning on brain tumor classification. The tumor examination is a difficult and precise efforts, certainty and accuracy are consistently considerable. As our proposed method describes to detect brain tumor from MR images. The system automates the manual process of tumor detection from MRI images and hence is economical in terms of time and human efforts. Here in the proposed model the Doctor and Patient can manually check for the tumor by uploading the MRI image by themselves. All the information are stored in the SQLite DataBase so that any data can be fetched easily if necessary. Detecting brain tumors is a complex and sensitive task, so preciseness and reliability will similarly show an important part of the chosen method. In future, the system can be improved by adapting more segmentation algorithm to suit the different medical image segmentation.

BIBLIOGRAPHY

- [1] <https://braintumor.org/brain-tumor-information/> from the journal of National brain tumor Society,2018.
- [2] Mohammad Teshanehalab et al, Brain Tumor Detection Using Deep Neural Network and Machine Learning Algorithm , 9th International Conference on Computer and Knowledge Engineering (ICCKE),2019.
- [3]Halimeh Siar, Mohammad Teshnehlal, Diagnosing and Classification Tumors and MS Simultaneous of Magnetic Resonance Images Using Convolution Neural Network, 7th Iranian Joint Congress on Fuzzy and Intelligent Systems (CFIS), 2019.
- [4]F. Samadi, G. Akbarizadeh, et al, Change Detection in SAR Images using Deep Belief Network: a New Training Approach based on Morphological Images, IET Image Processing, 2019.
- [5]Sindhia, Ramanitharan, Shankar Mahalingam, Soundarya Prithiesh, "Brain Tumor Detection Using MRI by Classification and Segmentation" SSRG International Journal of Medical Science 6.3 ,2019.
- [6]Digvijay reddy et al,Brain Tumor Detection Using Image Segmentation Techniques, International conference on communications and signal processing ,2018.
- [7]Md Rezwanul Islam et al, Detection and analysis of brain tumor from MRI by Integrated Thresholding and Morphological Process with Histogram based method,International Conference on Computer, Communication, Chemical, Material and Electronic Engineering, 2018.
- [8] Shanata Giraddi et al,Detection of Brain Tumor using Image Classification, International Conference on Current Trends in Computer, Electrical, Electronics and Communication, 2017.
- [9] Animesh Hazra et al,Brain tumor detection based on segmentation using MATLAB, International Conference on Energy, Communication, Data Analytics and Soft Computing, 2017.

- [10] S.K.Shil et al, An improved brain tumor detection and classification mechanism, International Conference on Information and Communication Technology Convergence, 2017.
- [11] M. Slotaninejad et al, Automated brain tumor detection and segmentation using superpixel based extremely randomized trees in FLAIR MRI, International journal of computer assisted radiology and surgery, 12(2), pp.183-203, 2017
- [12] K.Ramya et al, Brain tumor detection based on watershed transformation, International Conference on Communications and Signal Processing, 2016.
- [13] L. Szilagyi, et al, Automatic brain tumor segmentation in multispectral MRI volumes using a fuzzy c-means cascade algorithm, In 2015 12th international conference on fuzzy systems and knowledge discovery (FSKD), IEEE, pp. 285-291, 2015.
- [14] Anupurba Nandi, Detection of human brain tumour using MRI image segmentation and morphological operators 2015 IEEE International Conference on Computer Graphics, Vision and Information Security (CGVIS), 2015.