A Major Project Report

On

BRAIN TUMOR CLASSIFICATION USING CNN OF BRAIN MRI IMAGES

Submitted in partial fulfillment of the

Requirements for the award of degree of

Bachelor of Technology

In

Computer Science and Engineering

by

BOOSA GOWTHAM GOUD,
(18H65A0508)

JULAKANTI SRI TEJASWINI,
(17H61A0585)

BEERELLY DHEERAJ MAHANIDHI
(17H61A0567)

Under the Guidance of

Mrs. P. RAJESHWARI
Assistant Professor, CSE Department

Department of Computer Science and Engineering

ANURAG GROUP OF INSTITUTIONS

(Formerly CVSR College of Engineering)

(An Autonomous Institution, Approved by AICTE and NBA Accredited)
Venkatapur (V), Ghatkesar (M), Medchal(D)., T.S-500088
(2017-2021)

Anurag Group of Institutions

An Autonomous Institution

(Formerly CVSR College of Engineering)

Venkatapur (V), Ghatkesar (M), Ranga Reddy (Dist.)

Ph: 08499953666, 08499963666. www.anurag.edu.in

Department of Computer Science and Engineering

CERTIFICATE

This is to certify that the project entitled **"BRAIN TUMOR CLASSIFICATION USING CNN OF BRAIN MRI IMAGES"** being submitted by **B. Gowtham** bearing the Hall Ticket number **18H65A0508** and **J. Tejaswini** bearing the Hall Ticket number **17H61A0585** and **B. Dheeraj Mahanidhi** bearing the Hall Ticket number **17H61A0567** in partial fulfillment of the requirements for the award of the degree of the **Bachelor of Technology** in **Computer Science and Engineering** to **Anurag Group of Institutions (Formerly CVSR College of Engineering)** is a record of bonafide work carried out by them under my guidance and supervision from April 2021 to July 2021.

The results presented in this project have been verified and found to be satisfactory. The results embodied in this project report have not been submitted to any other University for the award of any other degree or diploma.

Internal Guide **External Examiner**
Mrs. P. RAJESHWARI
Assistant Professor

Dr.G.Vishnu Murthy
Professor and Head
Dept. of CSE

ACKNOWLEDGEMENT

It is our privilege and pleasure to express profound sense of respect, gratitude, and indebtedness to our guide **Mrs. P. Rajeshwari, Assistant Professor, CSE**, Dept. of Computer Science and Engineering, Anurag Group of Institutions (Formerly CVSR College of Engineering), for her indefatigable inspiration, guidance, cogent discussion, constructive criticisms and encouragement throughout this dissertation work.

We express our sincere gratitude to **Dr. G. Vishnu Murthy, Professor & Head**, Department of Computer Science and Engineering, Anurag Group of Institutions (Formerly CVSR College of Engineering), for his suggestions, motivations and co-operation for the successful completion of the work.

We extend our sincere thanks to **Dr. V. Vijaya Kumar**, **Dean,** Research and Development, Anurag Group of Institutions, for his encouragement and constant help.

We extend our sincere thanks to **Dr. K.S. Rao, Director**, Anurag Group of Institutions for his encouragement.

B. Gowtham
18H65A0508

J. Tejaswini
17H61A0585

B. Dheeraj Mahanidhi
17H61A0567

DECLARATION

We hereby declare that the project work entitled "**BRAIN TUMOR CLASSIFICATION USING CNN OF BRAIN MRI IMAGES**" submitted to the **Anurag Group of Institutions(Formerly CVSR College of Engineering)** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology (B.Tech)** in Computer Science and Engineering is a record of an original work done by us under the guidance of **Mrs. P. Rajeshwari, Assistant Professor, CSE Department,** and this project work have not been submitted to any other university for the award of any other degree or diploma.

B. Gowtham (18H65A0508),

J. Tejaswini (17H61A0585),

B. Dheeraj (17H61A0567)

Date: -

ABSTRACT

Brain tumor is a severe cancer disease caused by uncontrollable and abnormal partitioning of cells and it requires early and accurate detection methods. The brain tumors, are the most common and aggressive disease, leading to a very short life expectancy in their highest grade. Thus, treatment planning is a key stage to improve the quality of life of patients. MRI images are used to diagnose tumor in the brain. However, the huge amount of data generated by MRI scan thwarts manual classification of tumor vs non-tumor in a particular time. Recent study in the field of deep learning has helped the health industry in Medical Imaging for Medical Diagnostic of many diseases. For Visual learning and Image Recognition tasks CNN is the most prevalent and commonly used machine learning algorithm. Similarly, we introduce the convolutional neural network (CNN) approach along with Data Augmentation and Image Processing to categorize brain MRI scan images into cancerous and non-cancerous, that is presence of tumor or not. CNN is the most effective model in classifying images. Here the model is trained to specifically identify tiny aberrations from MRI images and predict the images into cancerous and non-cancerous.

CONTENTS

S NO	PAGE NO

1.INTRODUCTION

1.1 Motivation

Brain tumor is one of the most rigorous diseases in the medical science. An effective and efficient analysis is always a key concern for the radiologist in the premature phase of tumor growth. Histological grading, based on a stereotactic biopsy test, is the gold standard and the convention for detecting the grade of a brain tumor. The biopsy procedure requires the neurosurgeon to drill a small hole into the skull from which the tissue is collected. There are many risk factors involving the biopsy test, including bleeding from the tumor and brain causing infection, seizures, severe migraine, stroke, coma and even death. But the main concern with the stereotactic biopsy is that it is not 100% accurate which may result in a serious diagnostic error followed by a wrong clinical management of the disease.

Tumor biopsy being challenging for brain tumor patients, non-invasive imaging techniques like Magnetic Resonance Imaging (MRI) have been extensively employed in diagnosing brain tumors. Therefore, development of systems for the detection and prediction of the grade of tumors based on MRI data has become necessary. But at first sight of the imaging modality like in Magnetic Resonance Imaging (MRI), the proper visualization of the tumor cells and its differentiation with its nearby soft tissues is somewhat difficult task which may be due to the presence of low illumination in imaging modalities or its large presence of data or several complexity and variance of tumors-like unstructured shape, viable size, and unpredictable locations of the tumor.

1.2 Problem Definition

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 presence of a tumor by combining several procedures to provide a foolproof method of tumor detection in MRI brain images. The methods utilized are filtering, contrast adjustment, negation of an image, image subtraction, erosion, dilation, threshold, and outlining of the tumor. The focus of this project is MRI brain images tumor extraction and its representation in simpler form such that it is understandable by everyone.

1.3 Objective of the project

Here we have developed a model which helps in recognizing the brain tumor and classify it with the help of MRI scan images by training the machine using cnn model. This classification and recognition should be done with great accuracy and good performance. It can be implemented with less complexity. And the main thing is we have to reduce human error in diagnosis.

2. LITERATURE SURVEY

In, the Fuzzy C-Means (FCM) segmentation is applied to separate the tumor and non-tumor region of brain. Also wavelet feature are extracted by using multilevel Discrete Wavelet Transform (DWT). Finally, Deep Neural Network (DNN) is incorporated for brain tumor classification with high accuracy. This technique is compared with KNN, Linear Discriminant Analysis (LDA) and Sequential Minimal Optimization (SMO) classification methods. An accuracy rate of 96.97% in the analysis of DNN based brain tumor classification, But the complexity is very high and performance is very poor. In, a novel bio-physiomechanical tumor growth modeling is presented to analyze the step by steps tumor growth of patients. It will be applied for gliomas and solid tumor with individual margins to seizure the significant tumor mass effect. The discrete and continuous methods are combined to make a tumor growth modeling. The proposed scheme provides the likelihood to tacitly segment tumor-bearing brain images based on atlas-based registration. This technique is mainly used for brain tissue segmentation. But the computation time is high. In, new multi-fractal (MultiFD) feature extraction and improved AdaBoost classification schemes are used to detect and segment the brain tumor. The texture of brain tumor tissue is extracted by using MultiFD feature extraction scheme. The improved AdaBoost classification methods are used to find the given brain tissue is tumor or non-tumor tissue. Complexity is high.In,4 local independent projection-based classification (LIPC) method is used to classify the voxel of the brain. Also path feature is extracted in this method. Hence no need to perform explicit regularization in LIPC. The accuracy is low. In, the survey of brain tumor segmentation is presented. Discuss about Various segmentation methods such as Region based segmentation, threshold based segmentation, fuzzy C Means segmentation, Atlas based segmentation, Margo Random Field (MRF) segmentation, deformable model, geometric deformable model, The accuracy, robustness, validity are analyzed for all the methods. In,8 hybrid feature selection with ensemble classification is applied for brain tumor diagnosis process. The GANNIGMAC, decision Tree, Bagging C based wrapper approach is used to obtain the decision rules. Also simplify the decision rules by using hybrid feature selection, which contains the combination of (GANNIGMAC + MRMR C+ Bagging C + Decision Tree). In, the fuzzy based control theory is used for brain tumor segmentation and classification method. The Fuzzy Interference System (FIS) is a one special technique, which is mainly used for brain segmentation. Supervised classification is used to create a membership function of fuzzy controller. The performance is high and accuracy is low. In,10 the adaptive histogram equalization is used to improve the contrast of the image. Then Fuzzy CMeans (FCM) based segmentation is performed to separate the tumor from the whole brain image. After that Gabor feature are extracted to filter the abnormal cells of brain. Finally, the fuzzy with K Nearest Neighbor (KNN) classification is applied to find the abnormality of brain MRI image. The complexity is high. But the accuracy is low. In this work, a novel automatic brain tumor classification is performed by convolutions neural network.

Artificial intelligence and deep learning are primarily used in image processing techniques to segment, identify, and classify MRI Images and are also used to classify and detect brain tumors. So many works have already been done on the classification and segmentation of brain MRI images. Some of the international journals we reviewed on the detection and classification of brain tumor using deep learning are Sheikh Basheera et al., [4] proposed a method for classifying brain tumors where the tumor is initially segmented from an MRI image and segmented portion is then extracted through a pre-trained convolutional neural network using stochastic gradient descent. Muhammad Sajjad et al. [5] suggested classification of multi-grade tumors by applying data augmentation technique to Mri images and then tuning it using a pre-trained VGG-19 CNN Model. Carlo, Ricciardi et al., [6] presented an approach for classifying pituitary adenomas tumor MRIs by using multinomial logistic regression and k-nearest neighbor algorithms. The approach achieved an accuracy of 83% on multinomial logistic regression and 92% on a k-nearest neighbor with an AUC curve of 98.4%. Khwaldeh, saed et al., [7] presented a framework for classification of brain MRI images into healthy and unhealthy, and a grading system for categorizing unhealthy brain images into low and high grades, by modifying the Alex-Net CNN model which revealed 91% accuracy. Nyoman Abiniwanda et al., [8] trained a convolutional neural network to classify three specific brain tumors classes, namely Meningioma, Glioma, and Pituitary, which achieved 98.51% training accuracy and 84.19% validation accuracy. Sunanda Das et al., [9] also trained a CNN model with an image processing technique to identify various brain tumor types and achieved 94.39% accuracy with an average precision of 93.33%. Romeo, Valeria et al., [10] presented a radiomic machine learning approach to predict tumor grades and nodal status from CT scans of primary tumor lesions and got the highest accuracy of 92.9% by Naive Bayes and k-nearest neighbor. Muhammed Talo et al., [11] used the ResNet34 pre-trained CNN model a transfer learning approach along with Data Augmentation to classify normal and abnormal brain MRI images and got 100% accuracy. Arshia Rehman et al., [12] used three different pre-trained CNN models (VGG16, AlexNet, and GoogleNet) to classify the brain tumors into pituitary, glioma, and meningioma. During this Transfer learning approach, VGG16 acquires the highest accuracy that is 98.67%. Ahmet Çinar et al., [13] modified the pre-trained ResNet50 CNN model by removing its last 5 layers and adding 8 new layers instead and comparing its accuracy with other pre-trained models such as GoogleNet, AlexNet, ResNet50. The updated ResNet50 model showed effective results by achieving 97.2% accuracy.

A Convolutional Neural Network (Conv Net/CNN) is a Deep Learning algorithm which can take in an input image, assign importance (learnable weights and biases) to various aspects/objects in the image and be able to differentiate one from the other. The pre-processing required in a Conv Net is much lower as compared to other classification algorithms. While in primitive methods filters are hand-engineered, with enough training, Conv Nets have the ability to learn these filters/characteristics. The architecture of a Conv Net is analogous to that of the connectivity pattern of Neurons in the Human Brain and was inspired by the organization of the Visual Cortex. Individual neurons respond to stimuli only in a restricted region of the visual field known as the Receptive Field. A collection of such fields overlap to cover the entire visual area.An image is nothing but a matrix of pixel values, right? So why not just flatten the image (e.g. 3x3 image matrix into a 9x1 vector) and feed it to a Multi-Level Perceptron for classification purposes? In cases of extremely basic binary images, the method might show an average precision score while performing prediction of classes but would have little to no accuracy when it comes to complex images having pixel dependencies throughout.

A ConvNet is able to successfully capture the Spatial and Temporal dependencies in an image through the application of relevant filters. The architecture performs a better fitting to the image dataset due to the reduction in the number of parameters involved and reusability of weights. In other words, the network can be trained to understand the sophistication of the image better. The objective of the Convolution

Operation is to extract the high-level features such as edges, from the input image. Conv Nets need not be limited to only one Convolutional Layer. Conventionally, the first Conv Layer is responsible for capturing the Low-Level features such as edges, color, gradient orientation, etc. With added layers, the architecture adapts to the High-Level features as well, giving us a network which has the wholesome understanding of images in the dataset, similar to how we would. There are two types of results to the operation — one in which the convolved feature is reduced in dimensionality as compared to the input, and the other in which the dimensionality is either increased or remains the same. This is done by applying Valid Padding in case of the former, or Same Padding in the case of the latter.

When we augment the 5x5x1 image into a 6x6x1 image and then apply the 3x3x1 kernel over it, we find that the convolved matrix turns out to be of dimensions 5x5x1. Hence the name — Same Padding**.** On the other hand, if we perform the same operation without padding, we are presented with a matrix which has dimensions of the Kernel (3x3x1) itself — Valid Padding. The following repository houses many such GIFs which would help you get a better understanding of how Padding and Stride Length work together to achieve results relevant to our needs. Similar to the Convolutional Layer, the Pooling layer is responsible for reducing the spatial size of the Convolved Feature. This is to decrease the computational power required to process the data through dimensionality reduction. Furthermore, it is useful for extracting dominant features which are rotational and positional invariant, thus maintaining the process of effectively training of the model.

There are two types of Pooling: Max Pooling and Average Pooling. Max Pooling returns the maximum value from the portion of the image covered by the Kernel. On the other hand, Average Pooling returns the average of all the values from the portion of the image covered by the Kernel. Max Pooling also performs as a Noise Suppressant. It discards the noisy activations altogether and also performs de-noising along with dimensionality reduction. On the other hand, Average Pooling simply performs dimensionality reduction as a noise suppressing mechanism. Hence, we can say that Max Pooling performs a lot better than Average Pooling. The Convolutional Layer and the Pooling Layer, together form the i-th layer of a Convolutional Neural Network.

Depending on the complexities in the images, the number of such layers may be increased for capturing low-levels details even further, but at the cost of more computational power.

After going through the above process, we have successfully enabled the model to understand the features. Moving on, we are going to flatten the final output and feed it to a regular Neural Network for classification purposes. Adding a Fully-Connected layer is a (usually) cheap way of learning non-linear combinations of the high-level features as represented by the output of the convolutional layer. The Fully-Connected layer is learning a possibly non-linear function in that space. Now that we have converted our input image into a suitable form for our Multi-Level Perceptron, we shall flatten the image into a column vector. The flattened output is fed to a feed-forward neural network and backpropagation applied to every iteration of training. Over a series of epochs, the model is able to distinguish between dominating and certain low-level features in images and classify it.

3. ANALYSIS

3.1 Existing System

The existing system uses a manual approach by a person physically that is person should check whether a person is having brain tumor or not. The person in charge must check each and every person's report to classify whether they are having a tumor. As this approach is very difficult and it takes a lot of time to check each and every person's report manually.

Drawbacks

- Most detection and diagnosis methods depend on decision of neuro specialists and radiologist for image evaluation.
- It is very difficult and time taking process.
- Additional people must work.
- This may also lead to human errors.

3.2 Proposed System

The proposed system focuses on how to identify the person on image with or without tumor with the help of deep learning algorithm by using the CNN, OpenCV, Tensor flow, Kera's library.

Advantages

- A CNN uses convolution kernels to convolve with the original images or feature maps to extract higher-level features, thus resulting in a very powerful tool for Computer Vision tasks.
- Saves time and effort.
- Gives more accuracy.

3.3 Software Requirement Specification

Software Requirement Specification (SRS) is the starting point of the software developing activity. As system grew more complex it became evident that the goal of entire system cannot be easily comprehended. The SRS is the means of translating the ideas of the minds of clients (the input) into a formal document (the output of the requirement of the phase). The SRS phase consists of two basic activities.

1. **The Problem/Requirement Analysis**

   The process is order and more nebulous of the two, deals with understand the problem, the goal and constraints.

2. **Requirement Specification**

   Here, the focus is on specifying what has been found giving analysis such as representation, specification languages and tools, and checking the specifications are addressed during this activity.

   - The requirement phase terminates with the production of the validate SRS document.
   - Producing the SRS document is the basic goal of this phase.

3.3.1 Purpose

The purpose of the Software Requirement Specification is to reduce the communication gap between the clients and the developers. SRS is the medium through which the client and the user needs are accurately specified. A good SRS should satisfy all the parties involved in the system.

3.3.2 Scope

This Document is the only one that describes the requirements of the file system. It is meant for the use by the developers and will also be the basis for validating the final delivered system. Any changes made to the requirements in the future will have to go through a formal change approval process. The developer is responsible for asking the clarifications, where necessary and will not make any alterations without the permission of the client.

3.3.3 Overall Description

Hardware Requirements

System:	Pentium IV 2.4 GHz/ above
Hard Disk:	Minimum of 50 GB
Monitor:	15 VGA Color
Ram:	2 GB or Higher

Software Requirements

Operating System:	Windows Family
Documentation:	MS Office
Coding language:	Python
Frameworks:	Keras, Opencv, Tenser Flow
IDE:	Anaconda (Jupiter NoteBook)
API:	Flask (Web framework)

4. DESIGN

4.1 UML Diagrams

UML (Unified Modeling Language) is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems. UML was created by the Object Management Group (OMG) and UML 1.0 specification draft was proposed to the OMG in January 1997. It was initially started to capture the behavior of complex software and non-software system and now it has become an OMG standard. This tutorial gives a complete understanding on UML.

UML is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems.UML was created by the Object Management Group (OMG) and UML 1.0 specification draft was proposed to the OMG in January 1997.

- UML stands for Unified Modelling Language.
- UML is different from the other common programming languages such as C++, Java, COBOL, etc.
- UML is a pictorial language used to make software blueprints.
- UML can be described as a general-purpose visual modelling language to visualize, specify, construct, and document software system.
- Although UML is generally used to model software systems, it is not limited within this boundary. It is also used to model non-software systems as well. For example, the process flow in a manufacturing unit, etc.

UML is not a programming language but tools can be used to generate code in various languages using UML diagrams. UML has a direct relation with object oriented analysis and design. After some standardization, UML has become an OMG standard. The users can be developers, testers, business people, analysts, and many more. Hence, before designing a system, the architecture is made with different perspectives in mind. The most important part is to visualize the system from the perspective of different viewers. The better we understand the better we can build the system.

UML plays an important role in defining different perspectives of a system. These perspectives are –

- Design
- Implementation
- Process
- Deployment

GOALS

- Provide users a ready to use, expressive visual modelling Language so that they can develop and exchange meaningful models.
- Provide extendibility and specialization mechanisms to extend the core concepts.
- Be independent of particular programming languages and development process.
- Provide a formal basis for understanding the modelling language.
- Encourage the growth of OO tools market.
- Support higher level development concepts such as collaborations, frameworks, patterns, and components.
- Integrate best practices.

4.1.1 Use Case Diagram

In UML, there are five diagrams available to model the dynamic nature and use case diagram is one of them. Now as we have to discuss that the use case diagram is dynamic in nature, there should be some internal or external factors for making the interaction. These internal and external agents are known as actors. Use case diagrams consists of actors, use cases and their relationships. The diagram is used to model the system/subsystem of an application. A single use case diagram captures a particular functionality of a system. Hence to model the entire system, a number of use case diagrams are used.

The purpose of use case diagram is to capture the dynamic aspect of a system. However, this definition is too generic to describe the purpose, as other four diagrams (activity, sequence, collaboration, and Statechart) also have the same purpose. We will look into some specific purpose, which will distinguish it from other four diagrams.Use case diagrams are used to gather the requirements of a system including internal and external influences. These requirements are mostly design requirements. Hence, when a system is analyzed to gather its functionalities, use cases are prepared and actors are identified. When the initial task is complete, use case diagrams are modelled to present the outside view. Use case diagrams specify the events of a system and their flows. But use case diagram never describes how they are implemented. Use case diagram can be imagined as a black box where only the input, output, and the function of the black box is known.

These diagrams are used at a very high level of design. This high level design is refined again and again to get a complete and practical picture of the system. A well-structured use case also describes the pre-condition, post condition, and exceptions. These extra elements are used to make test cases when performing the testing.

Purpose of Use Case Diagrams

- Used to gather the requirements of a system.
- Used to get an outside view of a system.
- Identify the external and internal factors influencing the system.

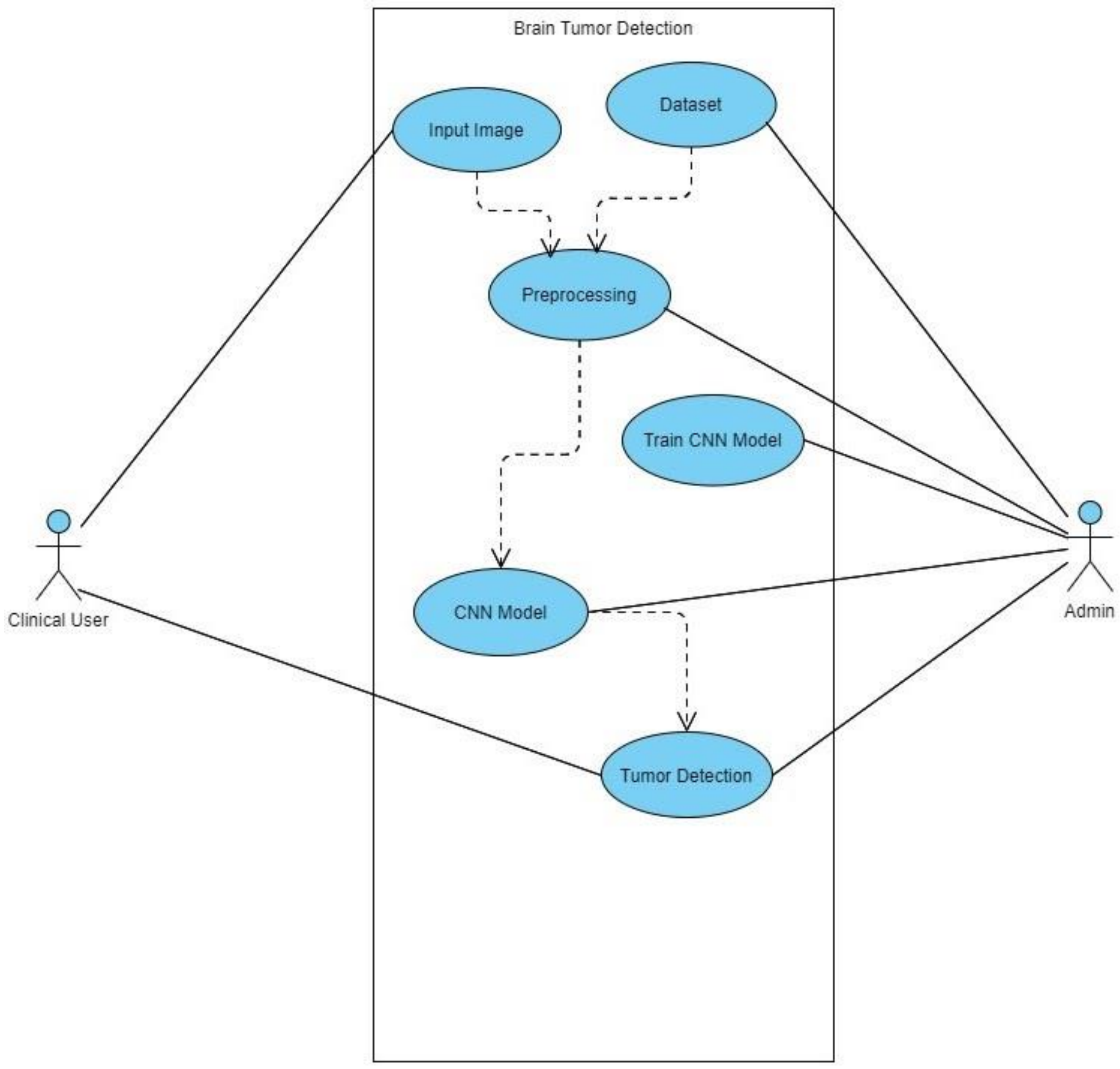


Fig 4.1.1.1 Use Case Diagram

4.1.2 Class Diagram

Class diagram is a static diagram. It represents the static view of an application. Class diagram is not only used for visualizing, describing, and documenting different aspects of a system but also for constructing executable code of the software application. Class diagram describes the attributes and operations of a class and also the constraints imposed on the system. The class diagrams are widely used in the modelling of object oriented systems because they are the only UML diagrams, which can be mapped directly with object-oriented languages. Class diagram shows a collection of classes, interfaces, associations, collaborations, and constraints. It is also known as a structural diagram. The purpose of class diagram is to model the static view of an application. Class diagrams are the only diagrams which can be directly mapped with object-oriented languages and thus widely used at the time of construction.

UML diagrams like activity diagram, sequence diagram can only give the sequence flow of the application, however class diagram is a bit different. It is the most popular UML diagram in the coder community. Class diagrams have a lot of properties to consider while drawing but here the diagram will be considered from a top level view. Class diagram is basically a graphical representation of the static view of the system and represents different aspects of the application. A collection of class diagrams represents the whole system.

Purpose of Class Diagrams

The purpose of the class diagram can be summarized as −

- Analysis and design of the static view of an application.
- Describe responsibilities of a system.
- Base for component and deployment diagrams.
- Forward and reverse engineering

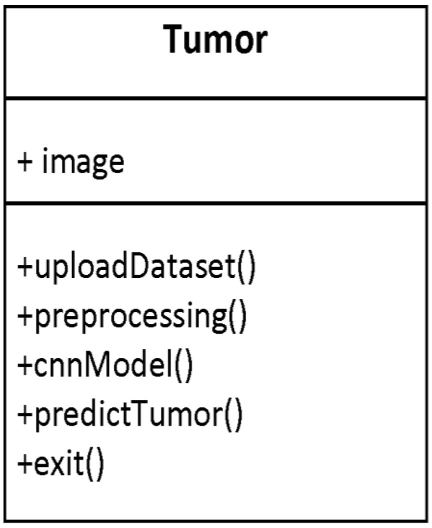


Fig 4.1.2.1 Class Diagram

4.1.3 Sequence Diagram

The sequence diagram captures the time sequence of the message flow from one object to another. A sequence diagram simply depicts interaction between objects in a sequential order i.e. the order in which these interactions take place. We can also use the terms event diagrams or event scenarios to refer to a sequence diagram. Sequence diagrams describe how and in what order the objects in a system function. These diagrams are widely used by businessmen and software developers to document and understand requirements for new and existing systems.

Actors

An actor in a UML diagram represents a type of role where it interacts with the system and its objects. It is important to note here that an actor is always outside the scope of the system we aim to model using the UML diagram. We use actors to depict various roles including human users and other external subjects. We represent an actor in a UML diagram using a stick person notation. We can have multiple actors in a sequence diagram.

Lifelines

A lifeline is a named element which depicts an individual participant in a sequence diagram. So basically, each instance in a sequence diagram is represented by a lifeline. Lifeline elements are located at the top in a sequence diagram. The standard in UML for naming a lifeline follows the following format – Instance Name : Class Name

Messages

Communication between objects is depicted using messages. The messages appear in a sequential order on the lifeline. We represent messages using arrows. Lifelines and messages form the core of a sequence diagram.

Purpose of Sequence Diagrams

- To model the flow of control by time sequence.
- To model the flow of control by structural organizations.
- For forward engineering.

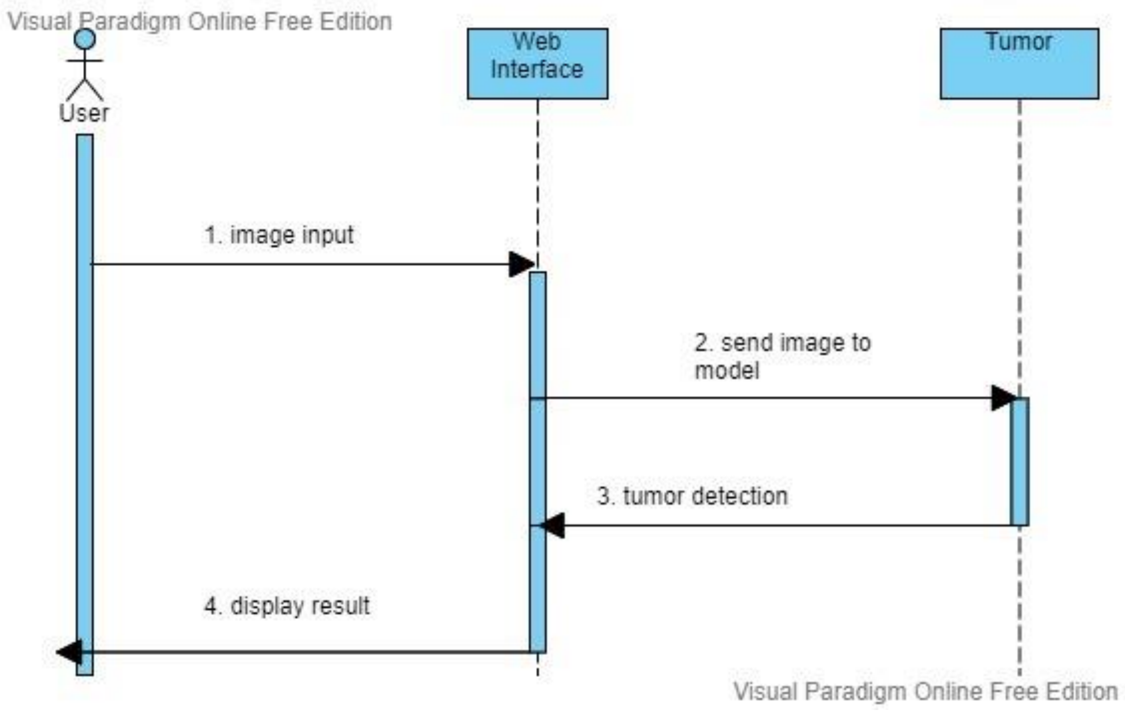


Fig 4.1.3.1 Sequence Diagram

4.1.4 Activity Diagram

Activity diagram is another important diagram in UML to describe the dynamic aspects of the system.Activity diagram is basically a flowchart to represent the flow from one activity to another activity. The activity can be described as an operation of the system.The control flow is drawn from one operation to another. This flow can be sequential, branched, or concurrent. Activity diagrams deal with all type of flow control by using different elements such as fork, join, etcThe basic purposes of activity diagrams is similar to other four diagrams. It captures the dynamic behavior of the system. Other four diagrams are used to show the message flow from one object to another but activity diagram is used to show message flow from one activity to another.

Activity is a particular operation of the system. Activity diagrams are not only used for visualizing the dynamic nature of a system, but they are also used to construct the executable system by using forward and reverse engineering techniques. The only missing thing in the activity diagram is the message part.It does not show any message flow from one activity to another. Activity diagram is sometimes considered as the flowchart. Although the diagrams look like a flowchart, they are not. It shows different flows such as parallel, branched, concurrent, and single.

Purpose of Activity Diagrams

The purpose of an activity diagram can be described as −

- Draw the activity flow of a system.
- Describe the sequence from one activity to another.
- Describe the parallel, branched and concurrent flow of the system.

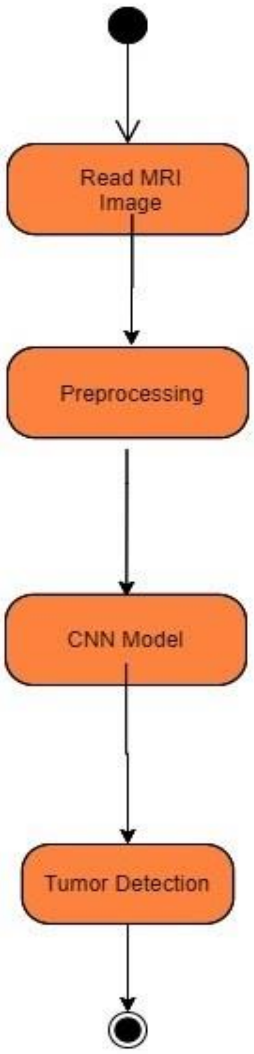


Fig 4.1.4.1 Activity Diagram

5.IMPLEMENTATION

5.1 Modules

A module is a collection of source files and build settings that allow you to divide your project into discrete units of functionality. Your project can have one or many modules and one module may use another module as a dependency. Each module can be independently built, tested, and debugged A module in project-open is a high-level description of a functional area, consisting of a group of processes describing the functionality of the module and a group of packages implementing the functionality.

We have Four modules in our project namely

1. User Module
2. Pre-processing Module
3. Training Module
4. Web Detection Module

5.2 Module Description

1. User Module

In User module, the user gives the input image to the system and the system preprocess the given image and produces an output.

2. Preprocessing Module

In this module, the images are preprocessed to extract the useful features from the given image and it also does various functions like image adjustment, pixel changing .

3. Training Module

Once the model is created, it has to be trained. We have predefined dataset which contains the images with tumor and without tumor. Using this dataset the model is trained.

4. Web Detection Module

Through flask we have converted the project source code into a website, where clinical user can upload images, and then result will be displayed. After the image has been uploaded in the website, the output will be shown tumor or no tumor.

5.3 Introduction To Technologies Used

5.3.1 Python

Python is an interpreted, object-oriented, high-level programming language with dynamic semantics. Its high-level built in data structures, combined with dynamic typing and dynamic binding, make it very attractive for Rapid Application Development, as well as for use as a scripting or glue language to connect existing components together. Python's simple, easy to learn syntax emphasizes readability and therefore reduces the cost of program maintenance. Python supports modules and packages, which encourages program modularity and code reuse. The Python interpreter and the extensive standard library are available in source or binary form without charge for all major platforms, and can be freely distributed.

Often, programmers fall in love with Python because of the increased productivity it provides. Since there is no compilation step, the edit-test-debug cycle is incredibly fast. Debugging Python programs is easy: a bug or bad input will never cause a segmentation fault. Instead, when the interpreter discovers an error, it raises an exception. When the program doesn't catch the exception, the interpreter prints a stack trace. A source level debugger allows inspection of local and global variables, evaluation of arbitrary expressions, setting breakpoints, stepping through the code a line at a time, and so on. The debugger is written in Python itself, testifying to Python's introspective power. On the other hand, often the quickest way to debug a program is to add a few print statements to the source: the fast edit-test-debug cycle makes this simple approach very effective.

Python is often compared to other interpreted languages such as Java, JavaScript, Perl, Tcl, or Smalltalk. Comparisons to C++, Common Lisp and Scheme can also be enlightening. In this section I will briefly compare Python to each of these languages. These comparisons concentrate on language issues only. In practice, the choice of a programming language is often dictated by other real-world constraints such as cost, availability, training, and prior investment, or even emotional attachment. Since these aspects are highly variable, it seems a waste of time to consider them much for this comparison.

Python programs are generally expected to run slower than Java programs, but they also take much less time to develop. Python programs are typically 3-5 times shorter than equivalent Java programs. This difference can be attributed to Python's built-in high-level data types and its dynamic typing. For example, a Python programmer wastes no time declaring the types of arguments or variables, and Python's powerful polymorphic list and dictionary types, for which rich syntactic support is built straight into the language, find a use in almost every Python program. Because of the run-time typing, Python's run time must work harder than Java's. For example, when evaluating the expression a+b, it must first inspect the objects a and b to find out their type, which is not known at compile time. It then invokes the appropriate addition operation, which may be an overloaded user-defined method. Java, on the other hand, can perform an efficient integer or floating point addition, but requires variable declarations for a and b, and does not allow overloading of the + operator for instances of user-defined classes.

For these reasons, Python is much better suited as a "glue" language, while Java is better characterized as a low-level implementation language. In fact, the two together make an excellent combination. Components can be developed in Java and combined to form applications in Python; Python can also be used to prototype components until their design can be "hardened" in a Java implementation. To support this type of development, a Python implementation written in Java is under development, which allows calling Python code from Java and vice versa. In this implementation, Python source code is translated to Java bytecode (with help from a run-time library to support Python's dynamic semantics).

5.3.2 Anaconda

Anaconda is a distribution of packages built for data science. It comes with conda, a package, and environment manager. We usually used conda to create environments for isolating our projects that use different versions of Python and/or different version of packages. We also use it to install, uninstall, and update packages in our project environments. When you download Anaconda first time it comes with conda, Python, and over 150 scientific packages and their dependencies. Anaconda is a fairly large download (~500 MB) because it comes with the most common data science packages in Python, for people who are conservative about disk space, there is also Miniconda, a smaller distribution that includes only conda and Python. You can still install any of the available packages with conda, that comes by default with the standard version. Conda is a program we will be using exclusively from the command line

5.3.3 Frameworks

5.3.3.1 Tenser Flow

TensorFlow is a free and open-source software library for dataflow 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. Itis used for both research and production at Google, TensorFlow is Google Brain's second-generation system. Version 1.0.0 was released on February 11, While the reference implementation runs on single devices, TensorFlow can run on multiple CPUs and GPUs (with optional CUDA and SYCL extensions for general-purpose computing on graphics processing units). Tensor Flow is available on 64-bit Linux, macOS, Windows, and mobile computing platforms including Android and iOS. Its flexible architecture allows for the easy deployment of computation across a variety of platforms (CPUs, GPUs, TPUs), and from desktops to clusters of servers to mobile and edge devices. The name Tensor Flow derives from the operations that such neural networks perform on multidimensional data arrays, which are referred to as tensors. During the Google I/O Conference in June 2016, Jeff Dean stated that 1,500 repositories on GitHub mentioned TensorFlow, of which only 5 were from Google.Unlike other numerical libraries intended for use in Deep Learning like Theano, TensorFlow was designed for use both in research and development and in production systems, not least RankBrain in Google search and the fun DeepDream project.It can run on single CPU.

5.3.3.2 Keras

Keras is an API designed for human beings, not machines. Keras follows best practices for reducing cognitive load: it offers consistent & simple APIs, it minimizes the number of user actions required for common use cases, and it provides clear & actionable error messages. It also has extensive documentation and developer guides. Keras contains numerous implementations of commonly used neuralnetwork building blocks such as layers, objectives, activation functions, optimizers, and a host of tools to make working with image and text data easier to simplify the coding necessary for writing deep neural network code. The code is hosted on GitHub, and community support forums include the GitHub issues page, and a Slack channel. Keras is a minimalist Python library for deep learning that can run on top of Theano or Tensor Flow. It was developed to make implementing deep learning models as fast and easy as possible for research and development.It runs onPython 2.7 or 3.5 and can seamlessly execute on GPUs and CPUs given the underlying frameworks. It is released under the permissive MIT license. Keras was developed and maintained by François Chollet, a Google engineer using four guiding principles: · Modularity: A model can be understood as a sequence or a graph alone. All the concerns of a deep learning model are discrete components that can be combined in arbitrary ways. · Minimalism: The library provides just enough to achieve an outcome, no frills and maximizing readability. · Extensibility: New components are intentionally easy to add and use within the framework, intended for researchers to trial and explore new ideas. · Python: No separate model files with custom file formats. Everything is native Python. Keras is designed for minimalism and modularity allowing you to very quickly define deep learning models and run them on top of a Theano or TensorFlow backend.

5.3.3.3 Opencv

OpenCV (Open Source Computer Vision Library) is an open source computer vision and machine learning software library. OpenCV was built to provide a common infrastructure for computer vision applications and to accelerate the use of machine perception in the commercial products. Being a BSD-licensed product, OpenCV makes it easy for businesses to utilize and modify the code. The library has more than 2500 optimized algorithms,which includes a comprehensive set of both classic and state-ofthe-art computer vision and machine learning algorithms. These algorithms can be used to detect and recognize faces, identify objects, classify human actions in videos, track camera movements, track moving objects, extract 3D models of objects, produce 3D point clouds from stereo cameras, stitch images together to produce a high resolution image of an entire scene, find similar images from an image database, remove red eyes from images taken using flash, follow eye movements, recognize scenery and establish markers to overlay it with augmented reality, etc.

OpenCV has more than 47 thousand people of user community and estimated number of downloads exceeding 18 million. The library isused extensively in companies, research groups and by governmental bodies.Along withwell-established companies like Google, Yahoo, Microsoft,Intel,IBM, Sony,Honda,Toyota that employ the library, there are many startups such as Applied Minds, Video Surf, and Zeitera, that make extensive use of OpenCV. OpenCV’s deployed uses span the range from stitching street view images together, detecting intrusions in surveillance video in Israel, monitoring mine equipment in China, helping robots navigate and pick up objects atWillow Garage, detection of swimming pool drowning accidents in Europe, running interactive art in Spain and New York, checking runways for debris in Turkey, inspecting labels on products in factories around the world on to rapid face detection in Japan. It has C++, Python, Java and MATLAB interfaces and supports Windows, Linux, Android and Mac OS. OpenCV leans mostly towards real-time vision applications and takes advantage of MMX and SSE instructions when available. A full-featured CUDA and OpenCL interfaces are being actively developed right now. There are over 500 algorithms and about 10 times as many functions that compose or support those algorithms. OpenCV is written natively in C++ and has a template interface that works seamlessly with STL containers.

5.3.4 Flask:

Flask is a micro web framework written in Python. It is classified as a microframework because it does not require particular tools or libraries.[2] It has no database abstraction layer, form validation, or any other components where pre-existing third-party libraries provide common functions. However, Flask supports extensions that can add application features as if they were implemented in Flask itself. Extensions exist for object-relational mappers, form validation, upload handling, various open authentication technologies and several common frameworks related tools.

WSGI

The Web Server Gateway Interface (Web Server Gateway Interface, WSGI) has been used as a standard for Python web application development. WSGI is the specification of a common interface between web servers and web applications.

Werkzeug

Werkzeug is a WSGI toolkit that implements requests, response objects, and utility functions. This enables a web frame to be built on it. The Flask framework uses Werkzeg as one of its bases.

jinja2

jinja2 is a popular template engine for Python.A web template system combines a template with a specific data source to render a dynamic web page.

Microframework

Flask is often referred to as a micro framework. It is designed to keep the core of the application simple and scalable.

Instead of an abstraction layer for database support, Flask supports extensions to add such capabilities to the application.

5.4 Sample code

5.4.1 Data Preprocessing code

```
#DATA AUGUMENTATION

datagen = ImageDataGenerator(rotation_range=10,
                             width_shift_range=0.1,
                             height_shift_range=0.1,
                             shear_range=0.1,
                             brightness_range=(0.3, 1.0),
                             horizontal_flip=True,
                             vertical_flip=True,
                             fill_mode='nearest')
paths = []
```

```
for r, d, f in os.walk(r'D:\Study\B.Tech\Semesters\SEM 8\Major
Project\Code\dataset\no'):
    for file in f:
        if '.jpg' in file:
            paths.append(os.path.join(r, file))
for path in paths:
    img = load_img(path)
    x = img_to_array(img)
    x = x.reshape((1,) + x.shape)
    i = 0
    for batch in datagen.flow(x,
batch_size=1,save_to_dir=r'D:\Study\B.Tech\Semesters\SEM 8\Major
Project\Code\dataset\no_after_aug', save_prefix='n', save_format='jpg'):
        i += 1
        if i > 20:
            break
paths = []
for r, d, f in os.walk(r"D:\Study\B.Tech\Semesters\SEM 8\Major
Project\Code\dataset\yes"):
    for file in f:
        if '.jpg' in file:
            paths.append(os.path.join(r, file))
for path in paths:
    img = load_img(path)
    x = img_to_array(img)
    x = x.reshape((1,) + x.shape)
    i = 0
    for batch in datagen.flow(x,
batch_size=1,save_to_dir=r'D:\Study\B.Tech\Semesters\SEM 8\Major
Project\Code\dataset\yes_after_aug', save_prefix='y', save_format='jpg'):
        i += 1
        if i > 12:
            break
#IMAGE CONTURING

def crop_brain_contour(image, plot=False):
# Convert the image to grayscale, and blur it slightly
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    gray = cv2.GaussianBlur(gray, (5, 5), 0)
# Threshold the image, then perform a series of erosions +
# dilations to remove any small regions of noise
    thresh = cv2.threshold(gray, 45, 255, cv2. THRESH_BINARY)[1]
    thresh = cv2.erode(thresh, None, iterations=2)
    thresh = cv2.dilate(thresh, None, iterations=2)
    # Find contours in thresholded image, then grab the Largest one
    cnts = cv2.findContours(thresh.copy(), cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
    cnts = imutils.grab_contours(cnts)
    c = max(cnts, key=cv2.contourArea)
    extLeft = tuple(c[c[:, :, 0].argmin()][0])
    extRight = tuple (c[c[:, :, 0].argmax()][0])
    extTop = tuple(c[c[:, :, 1].argmin()][0])
    extBot = tuple(c[c[:, :, 1].argmax()][0])
# crop new image out of the original image using the four extreme points (left, right,
top, bottom,
    new_image =  image[extTop[1]: extBot[1], extLeft[0]: extRight[0]]
    if plot:
        plt.figure()
        plt.subplot(1, 2, 1)
        plt.imshow(image)
        plt.tick_params (axis='both', which='both',top=False, bottom=False, left=False,
right=False, labelbottom=False, labeltop=False, labelleft=False, labelright=False)
        plt.title('Original Image')
        plt.subplot(1, 2, 2)
        plt.imshow(new_image)
        plt.tick_params (axis='both', which='both',
        top=False, bottom=False, left=False, right=False, labelbottom=False,
labeltop=False, labelleft=False, labelright=False)
        plt.title('Cropped Image')
        plt.show()
    return new_image
paths=[]
for r, d, f in os.walk(r'D:\Study\B.Tech\Semesters\SEM 8\Major
Project\Code\dataset\no_after_aug'):
    for file in f:
        if '.jpg' in file:
            paths.append(os.path.join(r, file))
i=0
for path in paths:
    i=i+1
    ex_img= cv2.imread(path)
    ex_new_img= crop_brain_contour(ex_img, True)
    cv2.imwrite(os.path.join(r'D:\Study\B.Tech\Semesters\SEM 8\Major
Project\Code\dataset\no_after_crop' , str(i+1)+'.jpg'), ex_new_img)
```

```
paths=[]
for r, d, f in os.walk(r'D:\Study\B.Tech\Semesters\SEM 8\Major
Project\Code\dataset\yes_after_aug'):
    for file in f:
        if '.jpg' in file:
            paths.append(os.path.join(r, file))
i=0
for path in paths:
    i=i+1
    ex_img= cv2.imread(path)
    ex_new_img= crop_brain_contour(ex_img, True)
    cv2.imwrite(os.path.join(r'D:\Study\B.Tech\Semesters\SEM 8\Major
Project\Code\dataset\yes_after_crop' , str(i+1)+'.jpg'), ex_new_img)
```

5.4.2 CNN Model Code

```
model = Sequential()
model.add(Conv2D(32, kernel_size=(2, 2), input_shape=(128, 128,3), padding =
'Same'))
model.add(Conv2D(32, kernel_size=(2, 2),  activation ='relu', padding = 'Same'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2, 2)))
model.add(Dropout(0.25))
model.add(Conv2D(64, kernel_size = (2,2), activation ='relu', padding = 'Same'))
model.add(Conv2D(64, kernel_size = (2,2), activation ='relu', padding = 'Same'))
model.add(BatchNormalization())
model.add(MaxPooling2D(pool_size=(2,2), strides=(2,2)))
model.add(Dropout(0.25))
model.add(Flatten())
model.add(Dense(512, activation='relu'))
model.add(Dropout(0.5))
model.add(Dense(2, activation='softmax'))
model.compile(loss = "categorical_crossentropy",
optimizer='Adamax',metrics=['accuracy'])
print(model.summary())
history = model.fit(x_train, y_train, epochs = 30, batch_size = 40, verbose =
1,validation_data = (x_test, y_test))
```

5.4.3 Tumor Detection Code (Flask)

```
import numpy as np
from keras.models import load_model
from keras.preprocessing.image import load_img
import pandas
import cv2
from PIL import Image, ImageOps
from flask import Flask, request, jsonify, render_template
app = Flask(__name__)
model = load_model('D:\Study\B.Tech\Semesters\SEM 8\Major
Project\Code\mymodel.h5')
@app.route('/')
def home():
    return render_template('index.html')
@app.route('/predict',methods=['POST'])
def predict():
    img = Image.open(request.files['img'])
    img = img.resize((128,128))
    print(img)
    img = img.convert("RGB")
    x = np.array(img)
    print(x.shape,end="\n")
    x = x.reshape((1,) + x.shape)
        res = model.predict_on_batch(x)
    classification = np.where(res == np.amax(res))[1][0]
    return render_template('index.html',
prediction_text='{}'.format(names(classification)))
def names(number):
    if number==0:
        return 'Yes, Its a Tumor'
    else:
        return 'No, Its not a tumor'
@app.route('/predict_api',methods=['POST'])
def predict_api():
    '''
    For direct API calls trought request
    '''
    data = request.get_json(force=True)
    prediction = model.predict([np.array(list(data.values()))])
    output = prediction[0]
    return jsonify(output)
if __name__ == "__main__":
    app.run(debug=True)
```

5.4.4 HTML Code (Flask)

```
<!doctype html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, user-scalable=no, initial-
scale=1.0, maximum-scale=1.0, minimum-scale=1.0">
    <meta http-equiv="X-UA-Compatible" content="ie=edge">
    <title>Brain Tumor Detection</title>
```

```
   <link type="text/css" href="{{ url_for('static', filename='css/bootstrap.min.css') }}"
rel="stylesheet" />
   <link rel="stylesheet"
href="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/css/bootstrap.min.css"
integrity="sha384-
Vkoo8x4CGsO3+Hhxv8T/Q5PaXtkKtu6ug5TOeNV6gBiFeWPGFN9MuhOf23Q9Ifjh"
crossorigin="anonymous">
<script>
var loadFile = function(event) {
        var image = document.getElementById('output');
        image.src = URL.createObjectURL(event.target.files[0]);
};
</script>
</head>
<body class="bg-light">
   <nav class="navbar navbar-expand-lg navbar-dark bg-primary">
      <!-- <a class="navbar-brand" href="#">Cancer Predictor and Detector</a> -->
      <a class="navbar-brand" href="#">Brain Tumor Detection</a>
      <button class="navbar-toggler" type="button" data-toggle="collapse" data-
target="#navbarSupportedContent" aria-controls="navbarSupportedContent" aria-
expanded="false" aria-label="Toggle navigation">
         <span class="navbar-toggler-icon"></span>
      </button>
      <div class="collapse navbar-collapse" id="navbarSupportedContent">
         <ul class="navbar-nav ml-auto">
            <li class="nav-item active">
               <a class="nav-link" href="./">Anurag University <span class="sr-
only">(current)</span></a>
            </li>
         </ul>
      </div>
   </nav>
   <div class="container mt-auto" style="padding-top: 3%;">
<form action="{{ url_for('predict')}}"method="post" enctype="multipart/form-data">
          <div class="row mt-5">
            <div class="col-md-2"></div>
            <div class="col-md-8 justify-content-center center">
               <h3 class="display-5 text-center">Upload MRI image of Brain: </h3>
               <input type="file" id="img" name="img" accept="image/*"
onchange="loadFile(event)" class="file-control col-md-6 d-flex justify-content-center"
style="display: block; margin-left: auto; margin-right: auto; width: 40%;" required>
               <div id="msg"></div>
<p><img id="output" width="200" /></p>
               <div class="mt-5">
                  <button type="submit" class="btn btn-primary btn-lg btn-block col-md-8
mr-auto ml-auto" id="upload-button">Check Tumor Status</button>
               </div>
      </form>
```

```
        </div>
        <div class="col-md-2"></div>
        </div>
<br>
<br>
<center><h1>{{ prediction_text }}</h1></center>
   </div>
   <script src="https://code.jquery.com/jquery-3.4.1.slim.min.js" integrity="sha384-
J6qa4849blE2+poT4WnyKhv5vZF5SrPo0iEjwBvKU7imGFAV0wwj1yYfoRSJoZ+n"
crossorigin="anonymous"></script>
   <script src="https://cdn.jsdelivr.net/npm/popper.js@1.16.0/dist/umd/popper.min.js"
integrity="sha384-
Q6E9RHvbIyZFJoft+2mJbHaEWldlvI9IOYy5n3zV9zzTtmI3UksdQRVvoxMfooAo"
crossorigin="anonymous"></script>
   <script src="https://stackpath.bootstrapcdn.com/bootstrap/4.4.1/js/bootstrap.min.js"
integrity="sha384-
wfSDF2E50Y2D1uUdj0O3uMBJnjuUD4Ih7YwaYd1iqfktj0Uod8GCExl3Og8ifwB6"
crossorigin="anonymous"></script>
</body>
</html>
```

6.TEST CASES

A **test case** is nothing, but a series of step executed on a product, using a predefined set of input data, expected to produce a pre-defined set of outputs, in a given environment. It describes “how” to implement those **test cases**. **Test case** specifications are useful as it enlists the specification details of the items. A test case is a document, which has a set of test data, preconditions, expected results and post conditions, developed for a particular test scenario in order to verify compliance against a specific requirement.

Test Case acts as the starting point for the test execution, and after applying a set of input values, the application has a definitive outcome and leaves the system at some end point or also known as execution postcondition.The test case is defined as a group of conditions under which a tester determines whether a software application is working as per the customer's requirements or not. Test case designing includes preconditions, case name, input conditions, and expected result. A test case is a first level action and derived from test scenarios

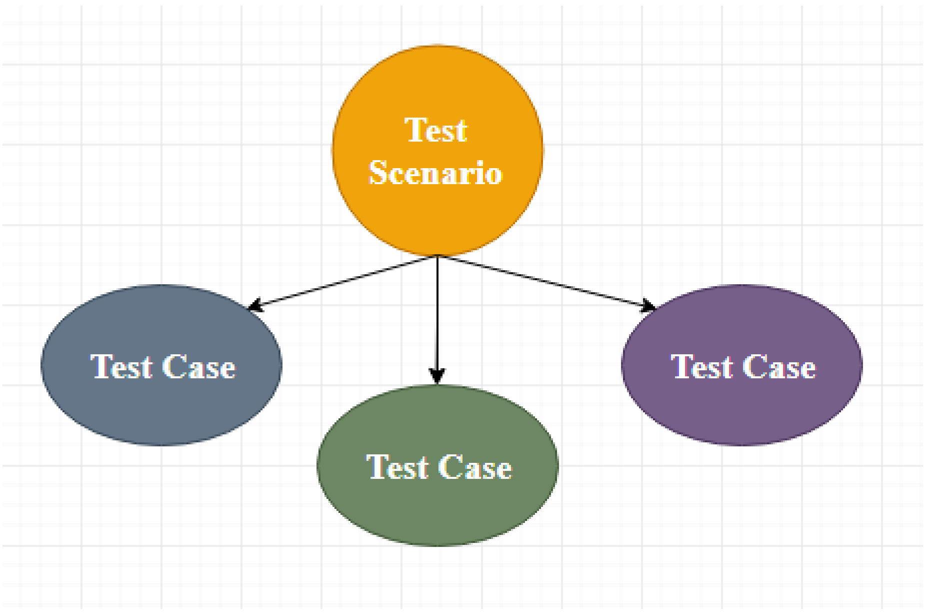


Fig 6.0 Test Cases

Test Cases

Test Case ID	Test Scenario	Test Steps	Expected Results	Actual Results	Status
1	Check website is loading	Open chrome and enter the URL	Website is loaded with User Interface	As expected	Pass
2	Check image is uploaded or not	Click on choose file and select an image	The selected image is displayed on the website	As expected	Pass
3	Check tumorous image is being detected	After uploading the MRI image which contains tumor click on Check Tumor Status	A result will be displayed as Yes, it is a Tumor	As expected	Pass
4	Check non tumorous image is being detected	After uploading the MRI image which do not contain tumor click on Check Tumor Status	A result will be displayed as No, it is not a Tumor	As expected	Pass

7.SCREENSHOTS

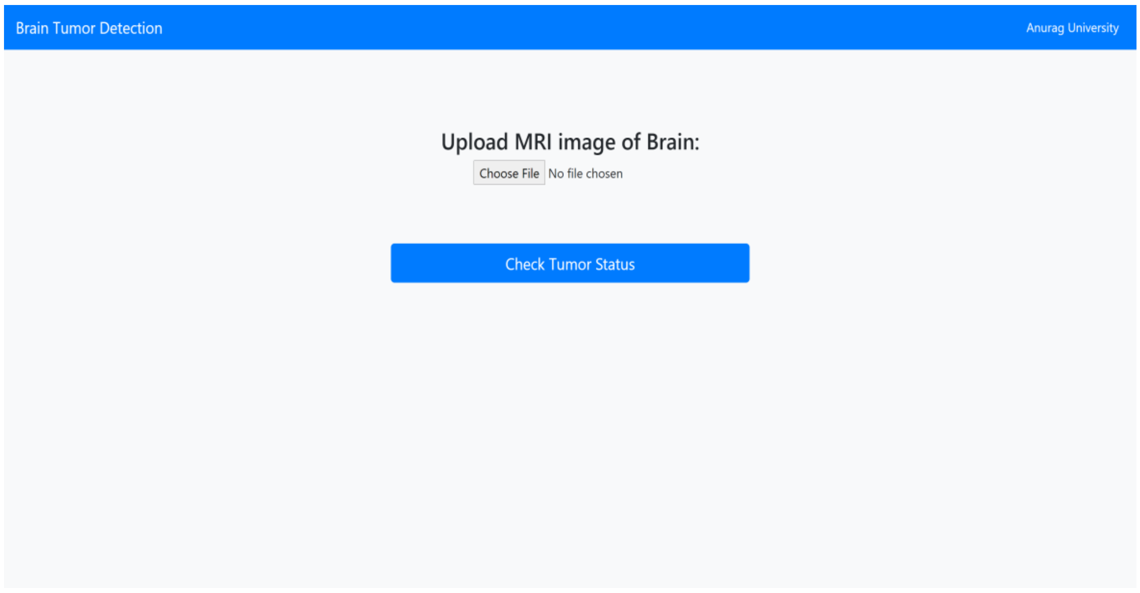


Fig 7.1 Web Interface

Web Interface which runs on the server, where clinical user can use to classify the MRI images.

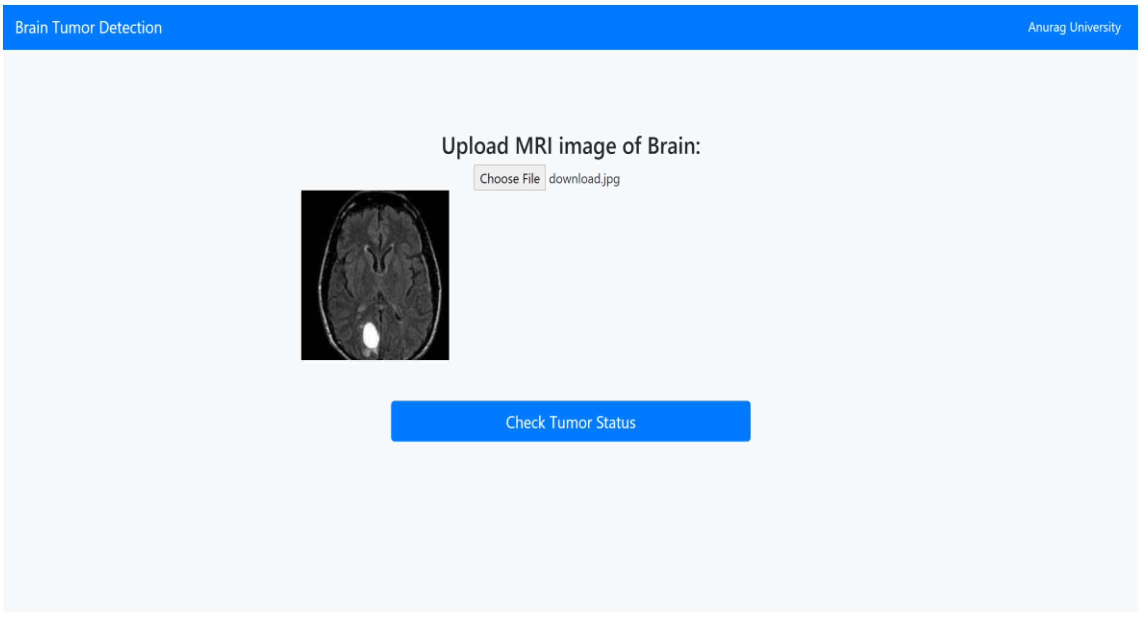
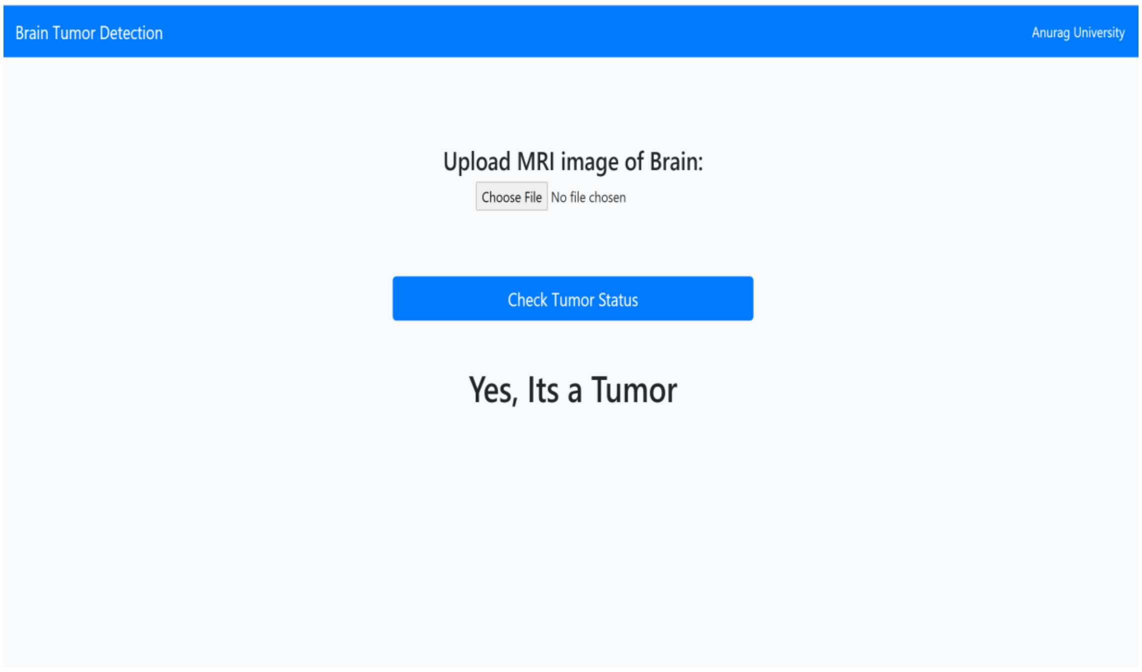


Fig 7.2 Uploaded Tumor Image

The user can upload the MRI image containing tumor using the choose file button on the webpage and click upload.



Yes, Its a Tumor

Fig 7.3 Tumor Detected

The user then can click on Check Tumor Status to get whether the MRI contains tumor or not, it is displayed on the screen.

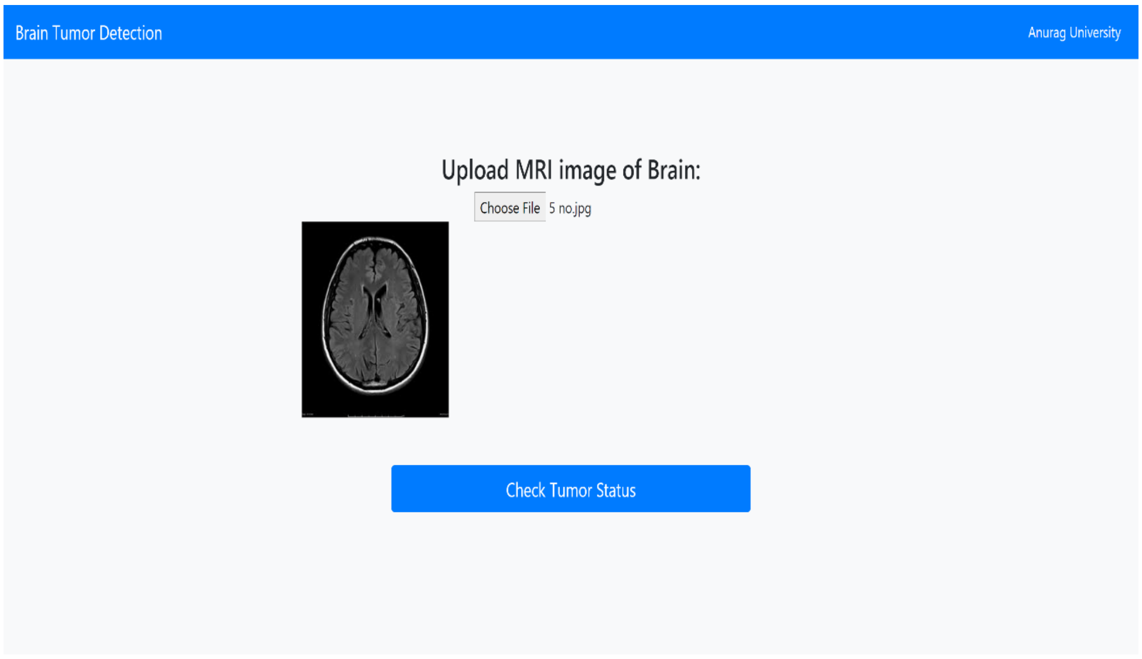
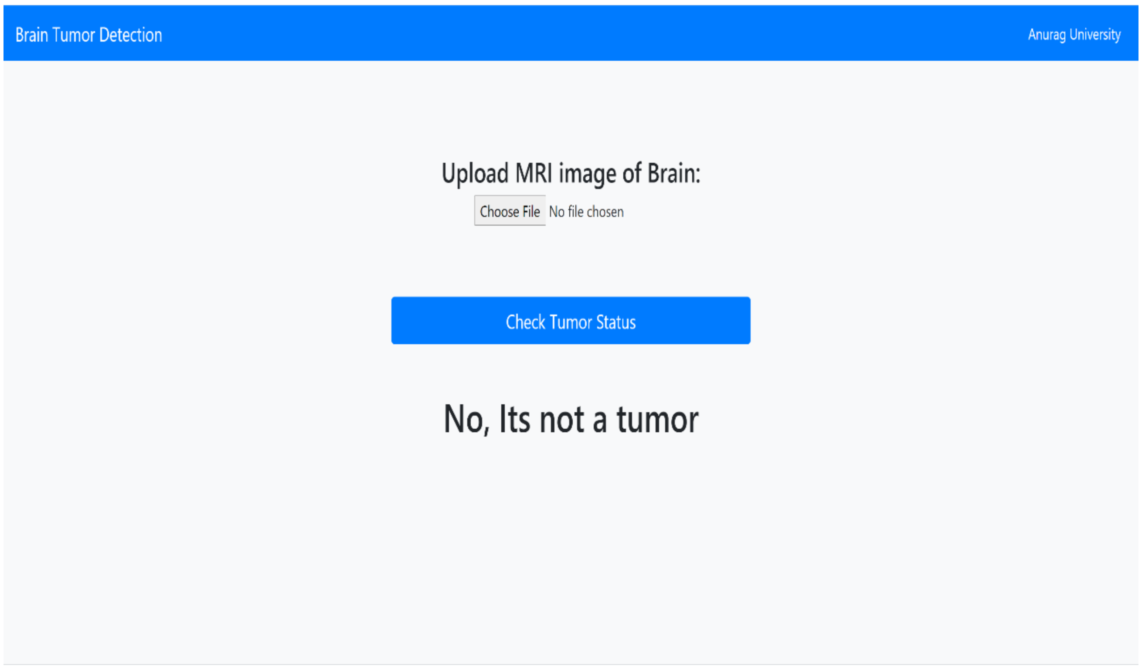


Fig 7.4 Uploaded Non-Tumor Image

The user can upload the MRI image not containing tumor using the choose file button on the webpage and click upload.



No, Its not a tumor

Fig 7.5 No Tumor Detected

The user then can click on Check Tumor Status to get whether the MRI contains tumor or not, it is displayed on the screen.

8.CONCLUSION

We are able to create and train a model which give good results and accuracy in classifying the presence of tumor. Our system can play an effective role in the early diagnosis of dangerous disease in other clinical domains related to medical imaging. using the image edge detection technique, we find the region of interest in MRI images and cropped them then, we used the data augmentation technique for increasing the size of our training data. We provide an efficient methodology for brain tumor classification by proposing a simple CNN network.

9.FUTURE ENHANCEMENT

Build an app-based user interface in hospitals which allows doctors to easily determine the impact of tumor and suggest treatment accordingly Since performance and complexity of ConvNets depend on the input data representation we can try to predict the location as well as stage of the tumor from Volume based 3D images. By creating three dimensional (3D) anatomical models from individual patients, training, planning and computer guidance during surgery is improved. Ultimately, we would like to use very large and deep convolutional nets on video sequences where the temporal structure provides very helpful information that is missing or far less obvious in static images. Unsupervised transfer learning may attract more and more attention in the future.

BIBLIOGRAPHY

[1]. Kasban, Hany & El-bendary, Mohsen & Salama, Dina. (2015). “A Comparative Study of Medical Imaging Techniques”. International Journal of Information Science and Intelligent System. 4. 37-58.J. Clerk Maxwell, A Treatise on Electricity and Magnetism, 3rd ed., vol. 2. Oxford: Clarendon, 1892, pp.68–73.

[2]. D. Surya Prabha and J. Satheesh Kumar, “Performance Evaluation of Image Segmentation using Objective Methods”, Indian Journal of Science and Technology, Vol 9(8), February 2016.

[3]. Brain Tumor: Statistics, Cancer.Net Editorial Board, 11/2017 (Accessed on 17th January 2019)

[4]. Kavitha Angamuthu Rajasekaran and Chellamuthu Chinna Gounder, Advanced Brain Tumour Segmentation from MRI Images, 2018.

[5]. General Information About Adult Brain Tumors". NCI. 14 April 2014. Archived from the original on 5 July 2014. Retrieved 8 June 2014. (Accessed on 11th January 2019)

[6]. M. Young, The Technical Writer’s Handbook. Mill Valley, CA: University Science, 1989.

[7]. B. Devkota, Abeer Alsadoon, P.W.C. Prasad, A. K. Singh, A. Elchouemi, “Image Segmentation for Early Stage Brain Tumor Detection using Mathematical Morphological Reconstruction,” 6th International Conference on Smart Computing and Communications, ICSCC 2017, 7-8 December 2017, Kurukshetra, India.

[8]. Song, Yantao & Ji, Zexuan & Sun, Quansen & Yuhui, Zheng. (2016). “A Novel Brain Tumor Segmentation from Multi-Modality MRI via A Level-Set-Based Model”. Journal of Signal Processing Systems. 87. 10.1007/s11265-016-1188-4.

[9]. Ehab F. Badran, Esraa Galal Mahmoud, Nadder Hamdy, “An Algorithm for Detecting Brain Tumors in MRI Images”, 7th International Conference on Cloud Computing, Data Science & Engineering - Confluence, 2017.

[10]. Pei L, Reza SMS, Li W, Davatzikos C, Iftekharuddin KM. “Improved brain tumor segmentation by utilizing tumor growth model in longitudinal brain MRI”. Proc SPIE Int Soc Opt Eng. 2017.

[11]. Dina Aboul Dahab, Samy S. A. Ghoniemy, Gamal M. Selim, “Automated Brain Tumor Detection and Identification using Image Processing and Probabilistic Neural Network Techniques”, IJIPVC, Vol. 1, No. 2, pp. 1-8, 2012.

[12]. Mohd Fauzi Othman, Mohd Ariffanan and Mohd Basri, “Probabilistic Neural Network for Brain Tumor Classification”, 2nd International Conference on Intelligent Systems, Modelling and Simulation, 2011.

[13]. A. Rajendran, R. Dhanasekaran, “Fuzzy Clustering and Deformable Model for Tumor Segmentation on MRI Brain Image: A Combined Approach,” International Conference on Communication Technology and System Design 2011.

[14]. Sobhaninia, Zahra & Rezaei, Safiyeh & Noroozi, Alireza & Ahmadi, Mehdi & Zarrabi, Hamidreza & Karimi, Nader & Emami, Ali & Samavi, Shadrokh. (2018). “Brain Tumor Segmentation Using Deep Learning by Type Specific Sorting of Images”.

[15]. Gupta, Gaurav and Vinay Singh. “Brain Tumor segmentation and classification using Fcm and support vector machine.” (2017).

[16]. Anam Mustaqeem, Ali Javed, Tehseen Fatima, “An Efficient Brain Tumor Detection Algorithm Using Watershed & Thresholding Based Segmentation”, I.J. Image, Graphics and Signal Processing, 2012, 10, 34-39.

[17]. Seetha, J & Selvakumar Raja, S. (2018). “Brain Tumor Classification Using Convolutional Neural Networks. Biomedical and Pharmacology Journal”. 11. 1457-1461. 10.13005/bpj/1511.

[18]. Mariam Saii, Zaid Kraitem, “Automatic Brain tumor detection in MRI using image processing techniques”, Biomedical Statistics and Informatics, Vol. 2, No. 2, pp. 73-76, 2017.