BRAIN CANCER DISEASE PREDICTION BASED ON IMAGE PROCESSING WITH HEALTHCARE INFORMATICS USING MACHINE LEARNING TECHNIQUES

A PROJECT REPORT

Submitted by

GOPIDASS V (312820205010)

THANNATCHI ENIYAN L.E (312820205040)

in partial fulfillment for the award of the degree

of

BACHELOR OF TECHNOLOGY

In

INFORMATION TECHNOLOGY

`AGNI COLLEGE OF TECHNOLOGY THALAMBUR

ANNA UNIVERSITY: CHENNAI 600 025

MAY 2024

ANNA UNIVERSITY: CHENNAI 600 025

BONAFIDE CERTIFICATE

Certified that this project report **BRAIN CANCER DISEASE PREDICTION BASED ON IMAGE PROCESSING WITH HEALTHCARE INFORMATICS USING MACHINE LEARNING TECHNIQUES** is the bonafide work of **GOPIDASS (312820205010),THANNATCHI ENIYAN L.E (312820205040)** who carried out the project work under my supervision.

SIGNATURE	SIGNATURE
Dr.S. GEERTHIK, M.E, Ph.D.	Ms. Nivi V , M.E.
HEAD OF THE DEPARTMENT	**SUPERVISOR,**
Department of Information Technology	Department of Information Technology
Agni College Of Technology,	Agni College Of Technology,
Thalambur, Chennai - 600130	Thalambur, Chennai - 600130

Submitted for the project viva held on …………………………

INTERNAL EXAMINER **EXTERNAL EXAMINER**

ACKNOWLEDGEMENT

We would like to express our deepest gratitude to the management of "**AGNI COLLEGE OF TECHNOLOGY"** and would like to thank our respected Principal **Dr. SRINIVASAN ALAVANDAR** for his words of inspiration and for providing necessary facilities to carry out our project work successfully.

We are immensely thankful to **Dr. S. GEERTHIK, M.E, Ph.D.,** Associate professor Head of the Department, Information Technology for his words of wisdom and his constant source of inspiration.

I would like to offer our heartfelt thanks to guide **Ms. NIVI.V, M.E,** Assistant professor, Department of Information Technology who molded us accordingly and gave valuable suggestion for completing our project work successfully.

We extended our warmest thanks to all the faculty members of our department for their assistance and we also thank all our friends who helped us in bringing out our project in good shape and form.

Finally, we express our sincere benevolence to our beloved parents for their perpetual encouragement and support in all endeavours.

ABSTRACT

A Brain tumor is considered as one of the aggressive diseases, among children and adults. Brain tumors account for 85 to 90 percent of all primary Central Nervous System (CNS) tumors. Every year, around 11,700 people are diagnosed with a brain tumor. The 5-year survival rate for people with a cancerous brain or CNS tumor is approximately 34 percent for men and 36 percent for women. Brain Tumors are classified as: Benign Tumor, Malignant Tumor, Pituitary Tumor, etc. Proper treatment, planning, and accurate diagnostics should be implemented to improve the life expectancy of the patients. The best technique to detect brain tumors is Magnetic Resonance Imaging (MRI). A huge amount of image data is generated through the scans. Application of automated classification techniques using Machine Learning (ML) has consistently shown higher accuracy than manual classification. Hence, proposing a system performing detection and classification by using Deep Learning Algorithms using ConvolutionNeural Network (CNN) & LSTM would be helpful to doctors all around the world. Brain Tumor Detection using Deep Learning with Python, Keras, and TensorFlow would outline the key objectives, methodology, and results of the study. The project aims to develop an accurate and efficient deep learning model to detect brain tumors using MRI images. The methodology involves pre-processing the MRI images and training a deep neural network using the Keras and TensorFlow libraries. The performance of the model is evaluated using various metrics such as accuracy, sensitivity, specificity, and F1 score. The results of the study indicate that the deep learning model achieves high accuracy in detecting brain tumors in MRI images. The model can be used to assist radiologists and doctors in the early detection of brain tumors, leading to timely intervention and improved patient outcomes. The performance of model is predict image tumor is present or not in image. If the tumor is present it return yes otherwise return no. Overall, the project demonstrates the potential of deep learning in medical imaging and highlights the importance of developing accurate and efficient models for disease diagnosis and management

TABLE OF CONTENTS

LIST OF FIGURES

LIST OF ABBREVIATIONS

CNN	CONVENTIONAL NEURAL NETWORK
WHO	WORLD HEALTH ORGANIZATION
TTP	TIME TO TUMER PROGRESSION
LSTM	LONG SHORT TERM MEMORY PROCESS
MRI	MAGNETIC RESONANCE IMAGES
AI	ARTIFICIAL INTELLIGENCE
ML	MACHINE LEARNING
DFD	DATAFLOW DIAGRAM
CT	COMPUTED TOMOGRAPHY

CHAPTER 1

1.1 INTRODUCTION

The early detection and treatment of brain tumor helps in early diagnosis which aids in reducing mortality rate. Image processing has been widespread in recent years and it has been an inevitable part in the medical field also. The abnormal growth of cells in the brain causes brain tumor. Brain tumor is also referred to as intracranial neoplasm. The two types of tumors are malignant and benign tumors. Standard MRI sequences are generally used to differentiate between different types of brain tumors based on visual qualities and contrast texture analysis of the soft tissue. More than 120 classes of brain tumors are known to be classified in four levels according to the level malignancy by the World Health Organization (WHO) (Louis, et al, 2016). All types of brain tumors evoke some symptoms based on the affected region of the brain. The major symptoms may include headaches, seizures, vision problems, vomiting, mental changes, memory lapses, balance losing etc. (Salander, et al, 1999). Incidence of brain tumors are due to genetics, ionizing radiation mobile phones, extremely low frequency magnetic fields, chemicals, head trauma and injury, immune factors like viruses, allergies, infections, etc. (McKinney ,2004).

The malignant tumors, also known as cancerous tumors, are of two types - primary tumors, which start from the brain, and secondary tumors, which originate somewhere and spread to the brain. The risk factors for brain tumor are exposure to vinyl chloride, neurofibromatosis, and ionizing radiations and so on. The various diagnostic methods are computed tomography, magnetic resonance imaging, tissue biopsy etc. Better treatments are now available for brain tumors. There is a chance of focal neurological deficits, such as motor deficit, aphasia or visual field defects in the treatment. Side effects can be avoided by measuring tumor size and time to tumor progression (TTP) (Heimans, et al, 2002). Estimation of density of affected areas can give a better measurement in therapy. Deep learning is a machine learning technique that instructs computers what to do as a human think and do in a scenario. In deep learning, a computer model is able to do classification tasks from images, sound or text. Sometimes human level performance is being exceeded by deep learning techniques.

One of the most popular neural networks is an artificial neural network that has a collection of simulated neurons. Each neuron acts as a node and by links each node is connected to other nodes (Suresh, et al. 2019). The goal of this project is to develop a deep learning model for brain tumor detection from MRI images using Python, TensorFlow, and Keras. Specifically, we will investigate the use of LSTM & CNNs for image segmentation and classification, with the aim of achieving high accuracy and robustness in detecting brain tumors. We will also explore different techniques for data augmentation, transfer learning, and hyperparameter tuning to optimize the performance of the model. This thesis is structured as follows. Chapter 1 provides an overview of brain tumors, MRI imaging, and deep learning. Chapter 2 reviews the related work on brain tumor detection using deep learning and highlights the gaps and challenges in the literature. Chapter 3 describes the methodology used in this project, including the data collection, preprocessing, model architecture, training, and evaluation. Chapter 4 presents the experimental results and analysis, comparing the performance of different models and techniques. Chapter 5 discusses the implications and limitations of the findings, as well as the future directions for research. Finally, Chapter 6 concludes the thesis with a summary of the main contributions and recommendations

1.2. ARTIFICIAL INTELLINGENCE:

Artificial intelligence (AI) is the ability of a computer program or a machine to think and learn. It is also a field of study which tries to make computers "smart". As machines become increasingly capable, mental facilities once thought to require intelligence are removed from the definition. AI is an area of computer sciences that emphasizes the creation of intelligent machines that work and reacts like humans. Some of the activities computers with artificial intelligence are designed for include: Face recognition, Learning, Planning, Decision making etc.,

Artificial intelligence is the use of computer science programming to imitate human thought and action by analysing data and surroundings, solving or anticipating problems and learning or self-teaching to adapt to a variety of tasks.

1.3. MACHINE LEARNING

Machine learning is a growing technology which enables computers to learn automatically from past data. Machine learning uses various algorithms for building mathematical models and making predictions using historical data or information. Currently, it is being used for various tasks such as image recognition, speech recognition, email filtering, Facebook auto-tagging, recommender system, and many more.

Machine Learning is said as a subset of artificial intelligence that is mainly concerned with the development of algorithms which allow a computer to learn from the data and past experiences on their own. The term machine learning was first introduced by **Arthur Samuel** in **1959**. We can define it in a summarized way as: “Machine learning enables a machine to automatically learn from data, improve performance from experiences, and predict things without being explicitly programmed”.

A Machine Learning system learns from historical data, builds the prediction models, and whenever it receives new data, predicts the output for it. The accuracy of predicted output depends upon the amount of data, as the huge amount of data helps to build a better model which predicts the output more accurately.

Suppose we have a complex problem, where we need to perform some predictions, so instead of writing a code for it, we just need to feed the data to generic algorithms, and with the help of these algorithms, machine builds the logic as per the data and predict the output. Machine learning has changed our way of thinking about the problem. The below block diagram explains the working of Machine Learning algorithm:

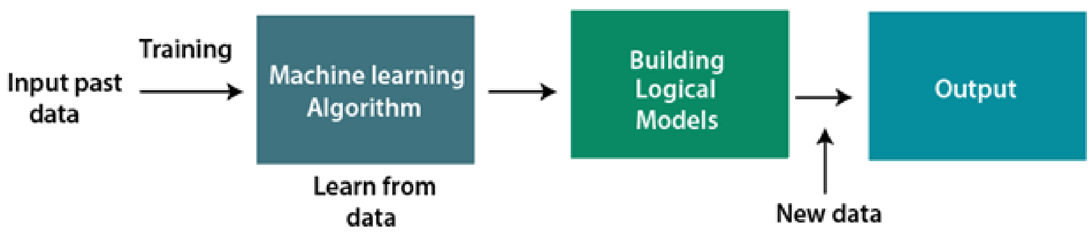

Fig. 1.3 Working of Machine Learning Algorithm

1.3.1.Features of Machine Learning:

- Machine learning uses data to detect various patterns in a given dataset.
- It can learn from past data and improve automatically.
- It is a data-driven technology.
- Machine learning is much similar to data mining as it also deals with the huge amount of the data.

1.3.2.Classification of Machine Learning

At a broad level, machine learning can be classified into three types:

1. Supervised learning
2. Unsupervised learning
3. Reinforcement learning

1) Supervised Learning

Supervised learning is a type of machine learning method in which we provide sample labeled data to the machine learning system in order to train it, and on that basis, it predicts the output.

The system creates a model using labeled data to understand the datasets and learn about each data, once the training and processing are done then we test the model by providing a sample data to check whether it is predicting the exact output or not.

The goal of supervised learning is to map input data with the output data. The supervised learning is based on supervision, and it is the same as when a student learns things in the supervision of the teacher. The example of supervised learning is **spam filtering**.

Supervised learning can be grouped further in two categories of algorithms:

- **Classification**
- **Regression**

2) Unsupervised Learning

Unsupervised learning is a learning method in which a machine learns without any supervision. The training is provided to the machine with the set of data that has not been labeled, classified, or categorized, and the algorithm needs to act on that data without any supervision. The goal of unsupervised learning is to restructure the input data into new features or a group of objects with similar patterns.

In unsupervised learning, we don't have a predetermined result. The machine tries to find useful insights from the huge amount of data.

It can be further classifieds into two categories of algorithms:

- **Clustering**
- **Association**

1.4. Explainable AI

1.4.1. LIME (Local Interpretable Model-agnostic Explanations):

1. LIME provides local model interpretability by approximating a complex model's behavior around specific instances.
2. It builds simpler, interpretable models (such as linear models) to approximate the complex model's predictions for individual instances.
3. Visualize and interpret the explanations provided by LIME to understand how your complex model is making predictions for specific data points.
4. Identify important features that contribute most to individual predictions.

1.4.2. SHAP (SHapley Additive exPlanations):

1. SHAP values offer a global and local perspective on feature importance and model interpretation.
2. It calculates the impact of each feature on predictions across the dataset and provides an explanation for each prediction.
3. Utilize SHAP summary plots to see feature importance across the dataset and understand the overall impact of each feature.
4. Use force plots or waterfall plots to explain individual predictions and understand how each feature contributes to specific outcomes.

1.5 OBJECTIVE

To develop an accurate and efficient system for predicting brain cancer diseases utilizing image processing techniques coupled with healthcare informatics and machine learning algorithms, aiming to enhance early diagnosis and treatment planning.

CHAPTER 2
LITERATURE SURVEY

[1] Title: Tumor Detection in the Brain using Faster R-CNN

Authors: R. Ezhilarasi; P. Varalakshmi

Description:

Brain tumor is the cancerous disease where abnormal cells found in the brain. This can be cured if we detect the brain tumor at an early stage. In this proposed system the tumor area is marked and defined what kind of tumor present in the brain tumor MRI image. AlexNet model is used for the classification of different types of tumors as a base model along with Region Proposal Network (RPN) by Faster R-CNN algorithm. Here, the concept of transfer learning is used during training. The proposed system helps to predict the correct type of tumor with better accuracy.

[2] Title: Brain Tumor Classification Using Convolutional Neural Networks

Authors: Seetha, J, S. Selvakumar Raja

Description:

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. Generally, various image techniques such as Computed Tomography (CT), Magnetic Resonance Imaging (MRI) and ultrasound image are used to evaluate the tumor in a brain, lung, liver, breast, prostate…etc. Especially, in this work 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. But it having some limitation (i.e) accurate quantitative measurements is provided for limited number of images. Hence trusted and automatic classification scheme are essential to prevent the death rate of human. The automatic brain tumor classification is very challenging task in large spatial and structural variability of surrounding region of brain tumor. In this work, automatic brain tumor detection is proposed by using Convolutional Neural Networks (CNN) classification. The deeper architecture design is performed by using small kernels. The weight of the neuron is given as small. Experimental results show that the CNN archives rate of 97.5% accuracy with low complexity and compared with the all other state of arts methods.

[3] Title: Brain tumor detection based on Convolutional Neural Network with neutrosophic expert maximum fuzzy sure entropy

Authors: Fatih Özyurt, Eser Sert, Engin Avci, Esin Dogantekin

Description:

Brain tumor classification is a challenging task in the field of medical image processing. The present study proposes a hybrid method using Neutrosophy and Convolutional Neural Network (NS-CNN). It aims to classify tumor region areas that are segmented from brain images as benign and malignant. In the first stage, MRI images were segmented using the neutrosophic set – expert maximum fuzzy-sure entropy (NS-EMFSE) approach. The features of the segmented brain images in the classification stage were obtained by CNN and classified using SVM and KNN classifiers. Experimental evaluation was carried out based on 5-fold cross-validation on 80 of benign tumors and 80 of malign tumors. The findings demonstrated that the CNN features displayed a high classification performance with different classifiers. Experimental results indicate that CNN features displayed a better classification performance with SVM as simulation results validated output data with an average success of 95.62%.

[4] Title: Brain Tumor Detection and Segmentation in MR Images Using Deep Learning

Authors: Sidra Sajid, Saddam Hussain ,Amna Sarwar

Description:

Gliomas are the most infiltrative and life-threatening brain tumors with exceptionally quick development. Gliomas segmentation using computer-aided diagnosis is a challenging task, due to irregular shape and diffused boundaries of tumor with the surrounding area. Magnetic resonance imaging (MRI) is the most widely used method for imaging structures of interest in human brain. In this study, a deep learning-based method that uses different modalities of MRI is presented for the segmentation of brain tumor. The proposed hybrid convolutional neural network architecture uses patch-based approach and takes both local and contextual information into account, while predicting output label. The proposed network deals with over-fitting problem by utilizing dropout regularizer alongside batch normalization, whereas data imbalance problem is dealt with by using two-phase training procedure. The proposed method contains a preprocessing step, in which images are normalized and bias field corrected, a feed-forward pass through a CNN and a post-processing step, which is used to remove small false positives around the skull portion. The proposed method is validated on BRATS 2013 dataset, where it achieves scores of 0.86, 0.86 and 0.91 in terms of dice score, sensitivity and specificity for whole tumor region, improving results compared to the state-of-the-art techniques.

[5] Title: Brain Tumor Segmentation to Calculate Percentage Tumor Using MRI

Authors: Annisa Wulandari; Riyanto Sigit; Mochamad Mobed Bachtiar

Description:

Brain tumor is one of disease type that attacks the brain in the form of clots. There is a way to see brain tumor in detail requires by an MRI image. There is difficulty in distinguishing brain tumor tissue from normal tissue because of the similar color. Brain tumor must be analyzed accurately. The solution for analyze brain tumor is doing segmentation. Brain tumor segmentation is done to separate brain tumor tissue from other tissues such as fat, edema, normal brain tissue and cerebrospinal fluid to overcome this difficulty, The MRI image must be maintained at the edge of the image first with the median filtering. Then the tumor segmentation process requires thresholding method which is then iterated to take the largest area. The brain segmentation is done by giving a mark on the area of the brain and areas outside the brain using watershed method then clearing skull with cropping method. In this study, 14 brain tumor MRI images are used. The segmentation results are compared brain tumors area and brain tissues area. This system obtained the calculation of tumor area has an average error of 10%.

CHAPTER 3
SYSTEM ANALYSIS

3.1. Existing system

The "Deep Learning-Based Brain Tumor Detection and Classification System" developed by researchers at the University of Michigan is a sophisticated approach to diagnosing brain tumors using advanced machine learning techniques. Here's a detailed breakdown of the system:

Data Collection: The system utilizes a dataset comprising MRI (Magnetic Resonance Imaging) scans of patients with known brain tumor diagnoses. These scans provide detailed images of the brain's structure, allowing for accurate analysis.

Preprocessing: Before feeding the MRI images into the deep learning model, preprocessing techniques are applied to enhance the quality and usability of the data. This may involve standardization, noise reduction, and image normalization to ensure consistency across different scans.

Convolutional Neural Networks (CNNs): CNNs are the backbone of the deep learning model used in this system. CNNs are a type of artificial neural network designed specifically for image processing tasks. They consist of multiple layers of learnable filters, which automatically extract features from the input images.

Training Phase: The CNN model is trained using a portion of the dataset, where the input consists of MRI images and the corresponding labels indicating the presence or absence of brain tumors. During training, the model learns to recognize patterns and features indicative of tumor presence.

Validation and Testing: After training, the model is evaluated on a separate portion of the dataset to assess its performance. This validation step ensures that the model generalizes well to unseen data. Additionally, extensive testing is conducted to measure the model's accuracy, sensitivity, specificity, and other performance metrics.

Disadvantages :

- The existing system used different algorithm to predict the disease, but accuracy is low comparison of our model.
- Complexity is high.
- Training and Testing the model is used same algorithm, but we provide different method.

3.2. Proposed system

- Our proposed system involves Dense Layer in Convolutional Neural Network (CNN) and LSTM Algorithm in Deep Learning concept used to train the dataset.
- In **Dense Layer**, each layer obtains additional inputs from all preceding layers and passes on its own feature-maps to all subsequent layers.
- In Dense Layer uses features of all complexity levels. It tends to give more smooth decision boundaries.
- Severity prediction based on tumour classification.

3.3. Advantages of proposed system

- Easy detection of the Brain Tumor with the concluded technique.
- Time consuming.
- Best accuracy Model helps in better treatment as early.
- Detection of best Model will quick the treatment which is life saving

CHAPTER 4
SYSTEM REQUIREMENTS

4.1. SOFTWARE REQUIREMENTS

Operating System	Windows 7 or latest
Simulation Tool	Anaconda (Jupyter notebook)
Documentation	Ms – Office

4.2. HARDWARE REQUIREMENTS

CPU type	I5
Ram size	4GB
Hard disk capacity	80 GB
Keyboard type	Internet keyboard
Monitor type	15 Inch colour monitor
CD -drive type	52xmax

4.3. Development Requirements:

Requirements are a feature of a system or description of something that the system is capable of doing in order to fulfil the system's purpose. It provides the appropriate mechanism for understanding what the customer wants, analyzing the needs assessing feasibility, negotiating a reasonable solution, specifying the solution unambiguously, validating the specification and managing the requirements as they are translated into an operational system.

4.3.1. PYTHON:

Python is a dynamic, high level, free open source and interpreted programming language. It supports object-oriented programming as well as procedural oriented programming. In Python, we don’t need to declare the type of variable because it is a dynamically typed language.

For example, x=10 .Here, x can be anything such as String, int, etc.

Python is an interpreted, object-oriented programming language similar to PERL, that has gained popularity because of its clear syntaxand readability. Python is said to be relatively easy to learn and portable, meaning its statements can be interpreted in a number of operating systems, including UNIX-based systems, Mac OS, MS-DOS, OS/2, and various versions of Microsoft Windows 98. Python was created by Guido van Rossum, a former resident of the Netherlands, whose favourite comedy group at the time was Monty Python's Flying Circus. The source code is freely available and open for modification and reuse. Python has a significant number of users.

4.3.2.Features in Python

There are many features in Python, some of which are discussed below

- Easy to code
- Free and Open Source
- Object-Oriented Language
- GUI Programming Support
- High-Level Language
- Extensible feature
- Python is Portable language
- Python is Integrated language
- Interpreted Language

4.3.3.Anaconda

Anaconda distribution comes with over 250 packages automatically installed, and over 7,500 additional open-source packages can be installed from PyPI as well as the conda package and virtual environment manager. It also includes a GUI, Anaconda Navigator,[12] as a graphical alternative to the command line interface (CLI).

The big difference between conda and the pip package manager is in how package dependencies are managed, which is a significant challenge for Python data science and the reason conda exists.

When pip installs a package, it automatically installs any dependent Python packages without checking if these conflict with previously installed packages. It will install a package and any of its dependencies regardless of the state of the existing installation. Because of this, a user with a working installation of, for example, Google Tensorflow, can find that it stops working having used pip to install a different package that requires a different version of the dependent numpy library than the one used by Tensorflow. In some cases, the package may appear to work but produce different results in detail.

In contrast, conda analyses the current environment including everything currently installed, and, together with any version limitations specified (e.g. the user may wish to have Tensorflow version 2,0 or higher), works out how to install a compatible set of dependencies, and shows a warning if this cannot be done.

Opensource packages can be individually installed from the Anaconda repository, Anaconda Cloud (anaconda.org), or the user's own private repository or mirror, using the conda install command. Anaconda, Inc. compiles and builds the packages available in the Anaconda repository itself, and provides binaries for Windows 32/64 bit, Linux 64 bit and MacOS 64-bit. Anything available on PyPI may be installed into a conda environment using pip, and conda will keep track of what it has installed itself and what pip has installed.

Custom packages can be made using the conda build command, and can be shared with others by uploading them to Anaconda Cloud, PyPI or other repositories.

The default installation of Anaconda2 includes Python 2.7 and Anaconda3 includes Python 3.7. However, it is possible to create new environments that include any version of Python packaged with conda.

4.3.4.Anaconda Navigator

Anaconda Navigator is a desktop graphical user interface (GUI) included in Anaconda distribution that allows users to launch applications and manage conda packages, environments and channels without using command-line commands. Navigator can search for packages on Anaconda Cloud or in a local Anaconda Repository, install them in an environment, run the packages and update them. It is available for Windows, macOS and Linux.

The following applications are available by default in Navigator:

- JupyterLab
- Jupyter Notebook
- QtConsole
- Spyder
- Glue
- Orange
- RStudio
- Visual Studio Code.

4.3.5. JUPYTER NOTEBOOK

Jupyter Notebook (formerly IPython Notebooks) is a web-based interactive computational environment for creating Jupyter notebook documents. The "notebook" term can colloquially make reference to many different entities, mainly the Jupyter web application, Jupyter Python web server, or Jupyter document format depending on context. A Jupyter Notebook document is a JSON document, following a versioned schema, containing an ordered list of input/output cells which can contain code, text (using Markdown), mathematics, plots and rich media, usually ending with the ".ipynb" extension.

Jupyter Notebook can connect to many kernels to allow programming in different languages. By default, Jupyter Notebook ships with the IPython kernel. As of the 2.3 release[11][12] (October 2014), there are currently 49 Jupyter-compatible kernels for many programming languages, including Python, R, Julia and Haskell.

The Notebook interface was added to IPython in the 0.12 release[14] (December 2011), renamed to Jupyter notebook in 2015 (IPython 4.0 – Jupyter 1.0). Jupyter Notebook is similar to the notebook interface of other programs such as Maple, Mathematica, and SageMath, a computational interface style that originated with Mathematica in the 1980s. According to *The Atlantic*, Jupyter interest overtook the popularity of the Mathematica notebook interface in early 2018.

CHAPTER 5

SYSTEM DESIGN

5.1. UML DIAGRAMS:

5.1.1. Introduction

Unified Modeling Language (UML) is a standardized general- purpose modeling language in the field of software engineering. The standard is managed and was created by the Object Management Group. UML includes a set of graphic notation techniques to create visual models of software intensive systems. This language is used to specify, visualize, modify, construct and document the artifacts of an object oriented software intensive system under development. There are given as below:

- Sequence diagram
- Use case diagram
- Activity diagram
- Collaboration diagram
- Dataflow diagram

5.1.2. Characteristics of UM

- It is a generalized modeling language.
- It is different from software programming languages such as Python, C, C++, etc.
- It is a pictorial language which can be used to generate powerful modeling elements.
- It is related to object-oriented designs and analysis.

5.1.3. Five Rules for Better UML Diagrams

- To avoid large diagrams with too many items.
- Avoid any two lines in your diagram crossing each other.
- Lines in a diagram should go only horizontal or vertical with only right angles.
- Parent elements are higher than the child elements in generalization or realization hierarchies. ➢ Diagrams should be nice and clean

5.2. Sequence Diagram:

A Sequence diagram is a kind of interaction diagram that shows how processes operate with one another and in what order. It is a construct of Message Sequence diagrams are sometimes called event diagrams, event sceneries and timing diagram.

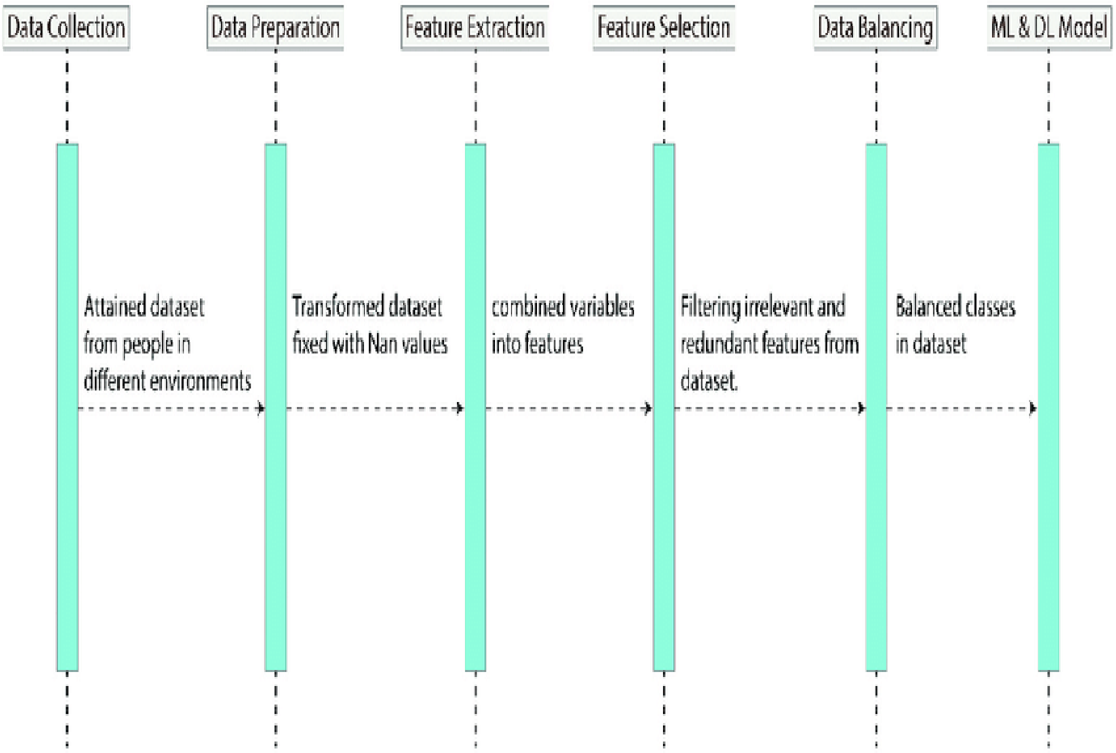

Fig 5.2 Sequence diagram

5.3. Use Case Diagram:

A Use case Diagram is used to present a graphical overview of the functionality provided by a system in terms of actors, their goals and any dependencies between those use cases.

Use case diagram consists of two parts:

Use case:

A use case describes a sequence of actions that provided something of measurable value to an actor and is drawn as a horizontal ellipse.

Actor: An actor is a person, organization or external system that plays a role in one or more interaction with the system.

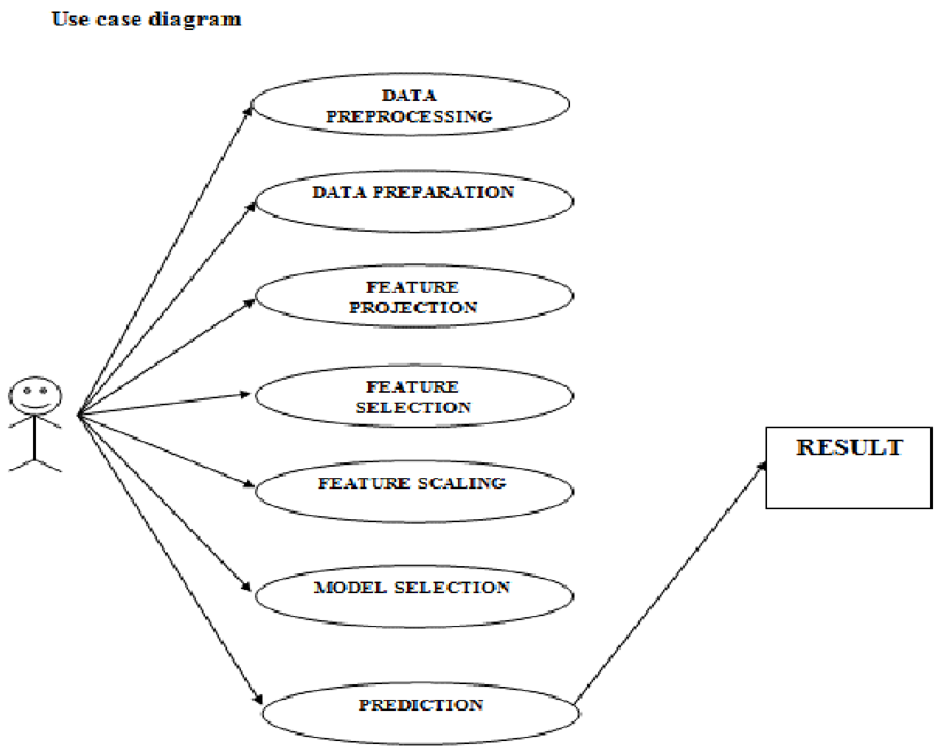

Fig.5.3 Use case diagram

5.4. System Study:

Aim:

To develop accurate and reliable computational models that can assist healthcare professionals in the early and precise diagnosis of brain tumors. By leveraging advanced machine learning algorithms, the goal is to analyze medical imaging data, such as MRI or CT scans, along with relevant patient information, to detect the presence, location, and characteristics of brain tumors. Ultimately, this technology aims to improve patient outcomes by enabling earlier detection, personalized treatment planning, and better management of brain cancer cases.

Objectives:

Achieve high accuracy in distinguishing between cancerous and non-cancerous brain lesions from medical imaging data. This involves developing and fine-tuning machine learning models to effectively identify subtle patterns and features indicative of brain tumors. Additionally, the objective includes optimizing the model's performance metrics such as sensitivity, specificity, and area under the ROC curve (AUC) to ensure reliable and consistent tumor detection across diverse patient populations and imaging modalities. Ultimately, the objective is to create a robust and clinically applicable tool that can aid healthcare professionals in making timely and accurate diagnoses, leading to improved patient outcomes and treatment strategies.

Scope:

The scope for brain cancer detection using machine learning encompasses several key areas:

Data Acquisition and Preparation: This involves gathering diverse datasets of medical imaging scans (MRI, CT, PET, etc.), along with associated patient data such as demographics, clinical history, and genetic information. The scope includes preprocessing these datasets to ensure data quality, consistency, and compatibility for further analysis.

Feature Extraction and Selection: Within this scope, methods for extracting relevant features from medical images are explored. This may involve techniques such as image segmentation, feature engineering, and dimensionality reduction to capture discriminative characteristics of brain tumors while minimizing noise and irrelevant information.

Model Development and Evaluation: The scope includes designing and implementing machine learning models, ranging from traditional classifiers (e.g., SVM, logistic regression) to more complex deep learning architectures (e.g., CNNs, RNNs). Model evaluation involves rigorous testing using cross-validation, hyperparameter tuning, and performance metrics assessment to ensure robustness, generalizability, and accuracy in tumor detection.

Integration with Clinical Workflow: This aspect involves integrating the developed models into existing clinical workflows, such as radiology departments or diagnostic centers. The scope includes developing user-friendly interfaces and software solutions that facilitate seamless interaction between healthcare professionals and the machine learning system.

Validation and Regulatory Compliance: Ensuring the safety, efficacy, and reliability of the developed system is paramount. The scope includes conducting thorough validation studies, obtaining regulatory approvals (e.g., FDA clearance), and adhering to ethical guidelines and data privacy regulations to guarantee the responsible deployment and use of the technology in clinical settings.

Clinical Implementation and Impact Assessment: This involves deploying the developed system in real-world clinical settings and assessing its impact on patient care, healthcare workflows, and diagnostic accuracy. The scope includes monitoring outcomes, gathering feedback from healthcare professionals, and iteratively refining the system to address any identified limitations or areas for improvement.

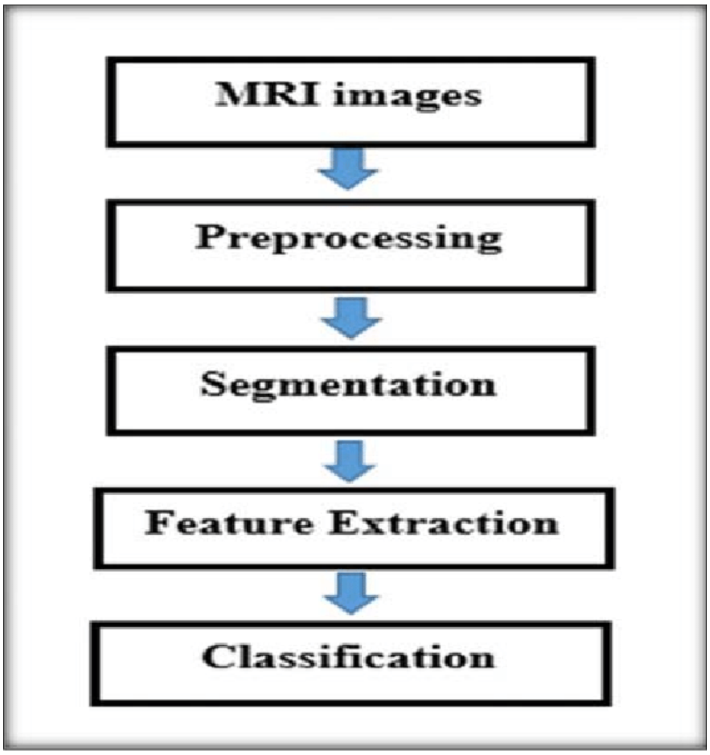

Fig.5.4 Steps of dataflow diagram

Image Acquisition:

This step involves acquiring brain images using medical imaging devices such as MRI machines, CT scanners, or PET scanners.

The acquired images are then sent to the preprocessing stage.

Preprocessing:

Raw brain images often contain noise or artifacts that can interfere with the analysis process. Preprocessing involves cleaning and enhancing these images.

Common preprocessing techniques include noise reduction, image registration, and intensity normalization.

Feature Extraction: Relevant features are extracted from the preprocessed brain images. These features could include shape, texture, intensity, or statistical measures of various brain structures.

Feature extraction methods may include techniques like edge detection, region segmentation, or wavelet analysis.
The extracted features are then fed into the machine learning model for training and prediction.
Classification:
Once the ML model is trained, it can be used to classify new brain images into one of two categories: either indicating the presence of brain cancer or not.
The model takes the extracted features from the input image as input and outputs a prediction or probability score.
Depending on the model's output, the image is classified as either positive for brain cancer or negative.

5.5. FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth N with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential. Three key considerations involved in the feasibility analysis are

i. Economical Feasibility
ii. Technical Feasibility
iii. Social Feasibility

5.5.1. Economic Feasibility

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

5.5.2. Technical Feasibility

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

5.5.3. Social Feasibility

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

5.6. Data Flow Diagram:

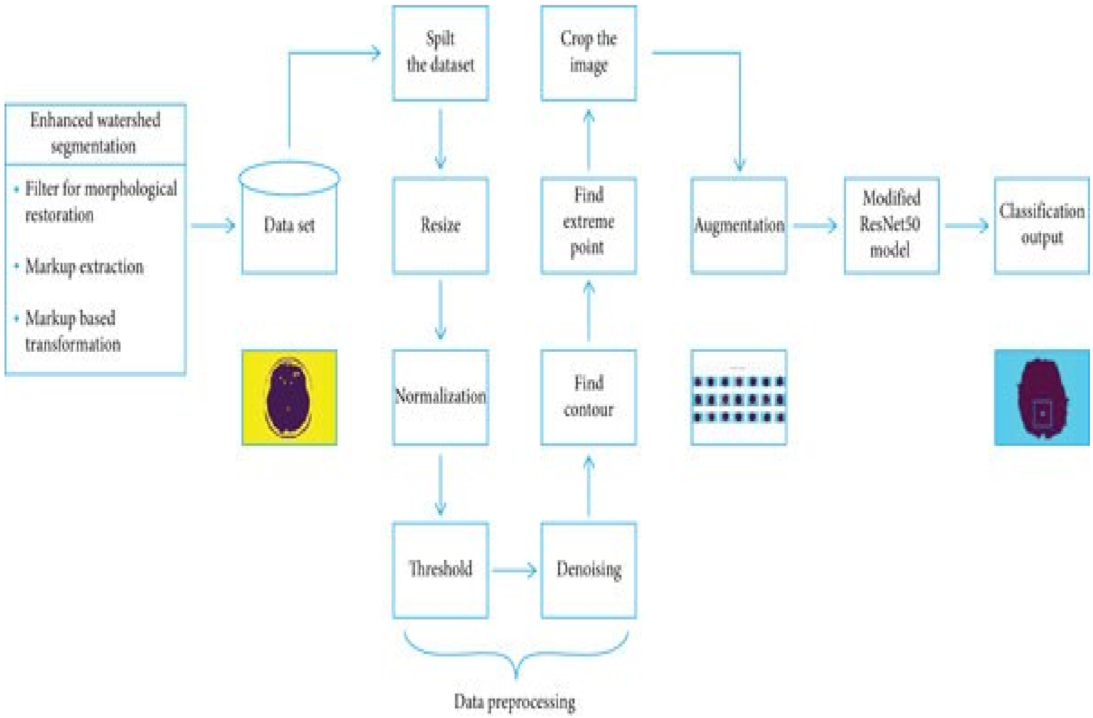

Fig5.6 Process of dataflow diagram

A data flow diagram (DFD) for brain cancer detection using machine learning typically outlines the flow of information within the system, from data acquisition to the final decision-making process. Here's a simplified explanation:

Data Collection:

The process begins with the collection of various types of data relevant to brain cancer detection. This includes medical imaging data such as MRI (Magnetic Resonance Imaging) scans, CT (Computed Tomography) scans, or PET (Positron Emission Tomography) scans.

Other relevant data might include patient demographics, medical history, and possibly genetic information.

Data Preprocessing:

The collected data may undergo preprocessing steps to clean, normalize, and format it for further analysis.

Preprocessing might involve removing noise from the images, standardizing the resolution and orientation of images, and handling missing or inconsistent data.

Model Training:

The extracted features are used to train a machine learning model. Various algorithms such as Support Vector Machines (SVM), Random Forests, Convolutional Neural Networks (CNNs), or other deep learning architectures may be employed.

During training, the model learns to classify brain images into different categories, such as "cancerous" or "non-cancerous", based on the features extracted from the training data.

Model Evaluation:

The trained model is evaluated using separate validation datasets to assess its performance. Evaluation metrics such as accuracy, sensitivity, specificity, and area under the ROC curve (AUC) are commonly used to measure the model's performance.

Evaluation helps determine the effectiveness of the model in accurately detecting brain cancer from medical images.

Inference:

Once the model is trained and evaluated satisfactorily, it can be deployed for real-world inference tasks.

New, unseen brain images are fed into the trained model, which then makes predictions about the presence or absence of cancerous tumors.

The inference results can be used to assist healthcare professionals in diagnosing and treating brain cancer patients.

Result Reporting:

Finally, the results of the inference process are reported back to healthcare providers, who can use this information to make informed decisions regarding patient care and treatment plans.

CHAPTER 6
DEVELOPMENT PROCESS

6.1. CONCEPT OF EXPLAINABLE AI SYSTEM

Another approach for brain tumor detection involves the integration of nanotechnology, multimodals, artificial intelligence (AI), and nanotechnology to create a real-time, non-invasive brain tumor detection system. Although this idea is theoretical, it is based on ideas that are being investigated and may work in the future as technology develops.

The Explainable AI Sensor System (N-BASS) is a proposed system that uses AI in conjunction with multimodal sensors enabled by nanotechnology to non-invasively and real-time identify brain cancers through the skull and scalp. The device would make use of nanoparticles engineered to specifically target tumor cells and pass across the blood-brain barrier (BBB). When tumor cells are present, these nanoparticles are designed to release multimodal signals, which a wearable AI-powered sensor will then be able to detect.

6.1.1.Design and Synthesis of Nanoparticles

Targeted Delivery: Create nanoparticles with the ability to attach to malignant cells in particular. These might be coupled to ligands or antibodies that specifically target characteristics of brain tumor cells.

Multimodal Activation: The nanoparticles would be engineered to provide observable multimodal signals either by fluorescing when in touch with the tumor microenvironment or by metabolizing after being taken up by tumor cells.

6.1.2. Secure and efficient delivery Framework

BBB Crossing: Apply strategies to allow these nanoparticles to pass through the BBB securely, such as coating them with substances that can help the BBB penetrate, including polyethylene glycol (PEG).

Administration: An intravenous (IV) injection could be used to deliver the nanoparticles, guaranteeing a systemic distribution that penetrates the blood-brain barrier and reaches brain tissue.

6.1.3. Development of Wearable Detection

Sensor Design: Create a wearable gadget with sensors that can identify the precise multimodal signals that the triggered nanoparticles release. Advanced materials and sensor technology to record light emissions through the skull may be used.

AI Integration: Combine the sensor with an AI algorithm that has been trained to distinguish between signal patterns linked to tumor presence and those linked to healthy brain tissue. The algorithm will take into account the signal's wavelength, strength, and other properties that are indicative of tumor cells.

6.1.4. Training in AI and Data Gathering

To begin with, a great deal of in vitro and in vivo research would be needed to produce data on the signal patterns connected to different kinds and stages of brain tumors. AI Model Creation: Create and hone machine learning models to precisely decipher sensor data and, using multimodal signal patterns, determine whether a brain tumor is present and, if so, its type and stage.

6.1.5. Clinical Examination and Verification

Preclinical Trials: To guarantee the security and effectiveness of the nanoparticles as well as the precision of the wearable detection system, carry out comprehensive preclinical testing.

Clinical Examinations: Conduct clinical trials to confirm the system's efficacy in identifying brain cancers in patients, contrasting its results with those of current diagnostic techniques.

6.2. SYSTEM ARCHITECTURE

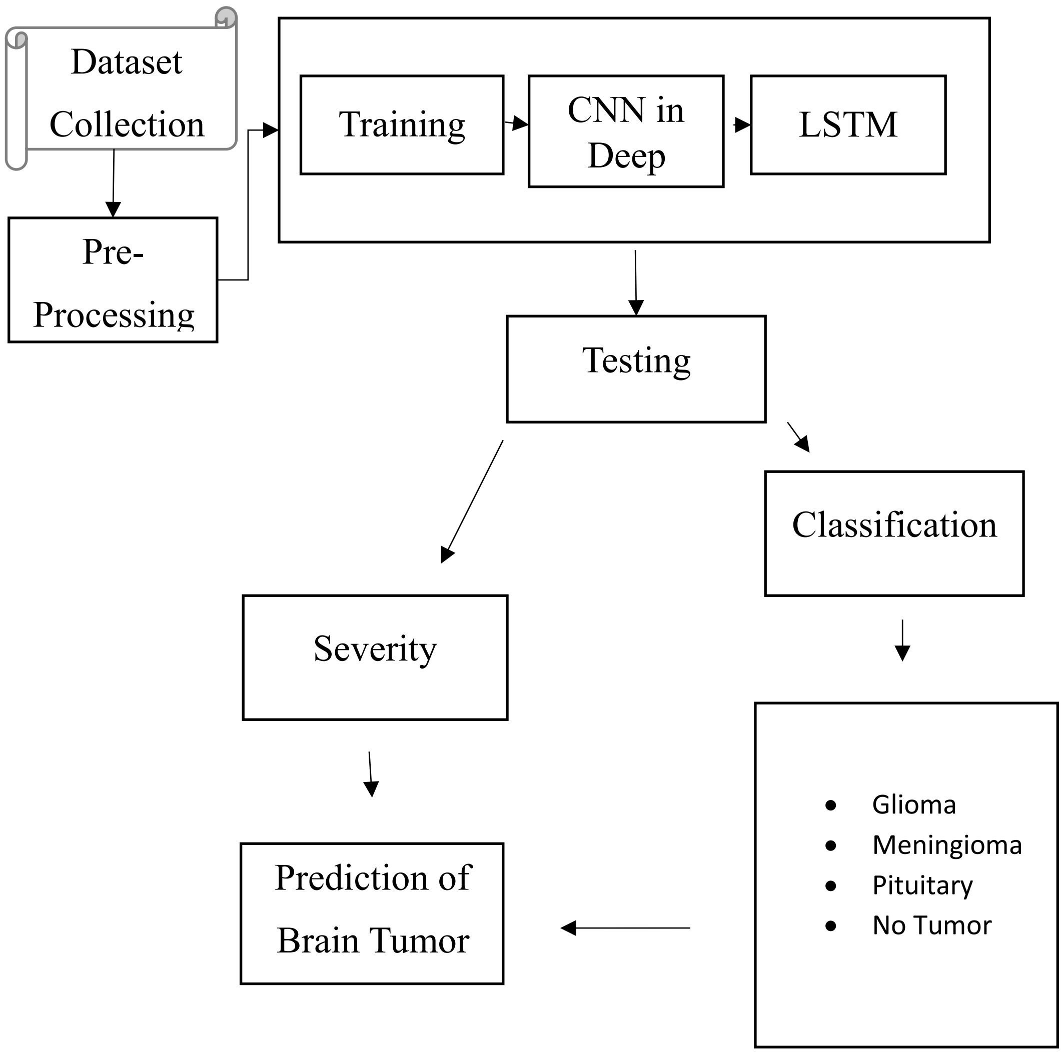

Fig 6.3 System Architecture

6.3. SYSTEM MODULES:

- Module 1: Data Collection
- Module 2: Pre-Processing
- Module 3: Splitting of datasets
- Module 4: Model implementation
- Module 5: Saving the model
- Module 6: Severity

Module 1: Data Collection

A dataset (or data set) is a collection of data, usually presented in tabular form. Each column represents a particular variable. Each row corresponds to a given member of the dataset in question. It lists values for each of the variables, such as height and weight of an object. Each value is known as a datum.

For this project we are using the images of brain MRI scanned images to classify the brain turmor. The datasets were collected from the web site of kaggle

Module 2: Data Pre-Processing

The preprocessing step involves reduction, aligning ,bias field correction. We use the CNN method that used for improve the accuracy and to reduce the computation time. CNN is one of the deep learning methods, which contains sequence of feed forward layers. In this concept we are pre processing using image data generator to resize the images.

Module 3: Splitting of Dataset

Data splitting is when data is divided into two or more subsets. Typically, with a two-part split, one part is used to evaluate or test the data and the other to train the model. Data splitting is an important aspect of data science, particularly for creating models based on data.

A commonly used ratio is 80:20, which means 80% of the data is for Severity and 20% for testing. Other ratios such as 70:30, 60:40, and even 50:50 are also used in practice. There does not seem to be clear guidance on what ratio is best or optimal for a given dataset.

- **Module 4: Model Implementation**

In this project we are implementing deep learning algorithm to classify the brain tumour using CNN algorithm

The classification can be done through the CNN and LSTM model to predict the brain tumor.

6.4.CNN algorithm:

A CNN is a kind of network architecture for deep learning algorithms and is specifically used for image recognition and tasks that involve the processing of pixel data. There are other types of neural networks in deep learning, but for identifying and recognizing objects, CNNs are the network architecture of choice. Within Deep Learning, a Convolutional Neural Network or CNN is a type of artificial neural network, which is widely used for image/object recognition and classification. Deep Learning thus recognizes objects in an image by using a CNN.

LSTM

LSTM networks are an extension of recurrent neural networks (RNNs) mainly introduced to handle situations where RNNs fail.

It fails to store information for a longer period of time. At times, a reference to certain information stored quite a long time ago is required to predict the current output. But RNNs are absolutely incapable of handling such “long-term dependencies”.

There is no finer control over which part of the context needs to be carried forward and how much of the past needs to be ‘forgotten’.

Other issues with RNNs are exploding and vanishing gradients (explained later) which occur during the Severity process of a network through backtracking.

- **Module 5: Saving the model**

  While Severity the model using CNN and LSTM the epochs values are given and trained

  After Severity the model we will save the values as a model file as h5

  The h5 file is like pre trained model file.

- **Module 6: Severity**

  Image Severity is the process of dividing the image into non- overlapping meaningful regions. The main objective if an image Severity is to divide an image into many sections for the further analysis, so we can get the only necessary or a segment of information. We use various image Severity dl algorithms to split and group a certain set of Severity together from the image. By doing so, we are actually assigning labels to Severity and the Severity with the same label fall under a category where they have some or the other thing common in them.

CHAPTER 7

TESTING

7. TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub – assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the
Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

7.1. TYPES OF TESTS

7.1.1. UNIT TESTING

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

7.1.2. INTEGRATION TESTING

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

7.1.3. FUNCTIONAL TEST

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

: identified classes of application outputs must be exercised

Procedures : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

7.1.4. SYSTEM TEST

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

7.1.5. WHITE BOX TESTING

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

7.1.6. BLACK BOX TESTING

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box .you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

7.1.7. TEST STRATEGY AND APPROACH

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

CHAPTER 8

CODING

```
import tensorflow as tf
import gc
from tensorflow.keras.layers import AveragePooling2D
from tensorflow.keras.layers import Dropout, GlobalAveragePooling2D,
Activation, BatchNormalization, Dropout, LSTM, ConvLSTM2D
from tensorflow.keras.layers import Flatten
from tensorflow.keras.layers import Dense
from tensorflow.keras.applications import VGG19
from tensorflow.keras.layers import Input,Conv2D, SeparableConv2D,
MaxPool2D, LeakyReLU, Activation, LSTM, ConvLSTM2D, Lambda,
Reshape, BatchNormalization, Bidirectional
from tensorflow.keras.models import Model, Sequential
from tensorflow.keras.optimizers import Adam, RMSprop
from tensorflow.keras.utils import to_categorical
from tensorflow.keras.callbacks import EarlyStopping, ReduceLROnPlateau,
ModelCheckpoint,TensorBoard,TerminateOnNaN, LearningRateScheduler,
CSVLogger
from tensorflow.keras.losses import binary_crossentropy
from tensorflow.keras.callbacks import ModelCheckpoint, EarlyStopping,
ReduceLROnPlateau, TerminateOnNaN
from tensorflow.keras.layers import Lambda, Reshape, DepthwiseConv2D,
ZeroPadding2D, Add, MaxPooling2D,Activation, Flatten, Conv2D, Dense,
Input, Dropout, Concatenate, GlobalMaxPooling2D, GlobalAveragePooling2D,
BatchNormalization
from tensorflow.keras.optimizers import Adam
from tensorflow.keras import regularizers
from tensorflow.keras.regularizers import l2
from tensorflow.keras import layers
from tensorflow.keras import backend as K
from tensorflow.keras.preprocessing.image import ImageDataGenerator
from sklearn.preprocessing import LabelBinarizer
from sklearn.model_selection import train_test_split, StratifiedKFold,
RepeatedStratifiedKFold
from sklearn.metrics import classification_report, accuracy_score,
confusion_matrix, roc_auc_score, roc_curve, auc
import matplotlib.pyplot as plt
```

```
from tensorflow.keras.utils import plot_model
from tensorflow.keras.layers.experimental import preprocessing
from keras.applications import imagenet_utils
from tensorflow.keras import layers
!pip install tensorflow-addons
import tensorflow_addons as tfa
import pandas as pd
import numpy as np
import random
import keras
import shutil
import pathlib
import itertools
import cv2
import os
import matplotlib.image as mpimg
import seaborn as sns
# pip install typing-extensions==4.6.0
# pip install scikit-learn
train_dir = '/content/drive/MyDrive/Brain tumor/Datasets/Training/'
test_dir = '/content/drive/MyDrive/Brain tumor/Datasets/Testing/'
for dirpath, dirnames, filenames in os.walk(train_dir):
  print(f"There are {len(dirnames)} directories and {len(filenames)} images in
'{dirpath}'.")
for dirpath, dirnames, filenames in os.walk(test_dir):
  print(f"There are {len(dirnames)} directories and {len(filenames)} images in
'{dirpath}'.")
def view_random_image(target_dir, target_class):
  target_folder = target_dir+target_class
  random_image = random.sample(os.listdir(target_folder), 1)
  img = mpimg.imread(target_folder + "/" + random_image[0])
  plt.imshow(img)
  plt.title(target_class)
  plt.axis("off");
  print(f"Image shape: {img.shape}")
  return img
#!pip install tensorflow-addons
import tensorflow_addons as tfa
def augment_image(image, label):
```

```
    # Apply augmentation transformations
    image = tf.image.random_flip_left_right(image)
    image = tf.image.random_flip_up_down(image)
    image = tf.image.random_brightness(image, max_delta=0.1)
    image = tf.image.random_contrast(image, lower=0.8, upper=1.2)
    image = tf.image.random_saturation(image, lower=0.8, upper=1.2)
    image = tf.image.random_hue(image, max_delta=0.1)
    # Apply the width and height shifts
    width_shift = tf.random.uniform([], -0.2, 0.2) * tf.cast(tf.shape(image)[1],
tf.float32)
    height_shift = tf.random.uniform([], -0.2, 0.2) * tf.cast(tf.shape(image)[0],
tf.float32)
    image = tfa.image.translate(image, [width_shift, height_shift])
    return image, label

 train_data = tf.keras.preprocessing.image_dataset_from_directory(train_dir,
                                                      label_mode="categorical",
                                                      batch_size=32,
                                                      image_size=IMAGE_SIZE)

test_data = tf.keras.preprocessing.image_dataset_from_directory(test_dir,
                                                      label_mode="categorical",
                                                      image_size=IMAGE_SIZE,
                                                      shuffle=False)
# Apply augmentation to the training dataset using the map function
train_dataset_augmented = train_data.map(augment_image)

def Combined_model():
    # Input layer
    input_layer = Input(shape=(224, 224, 3))

    # Base VGG19 model as a feature extractor
    baseModel = VGG19(weights=None, include_top=False,
input_tensor=input_layer)

    # Load the weights from the local file (specify the path)
    baseModel.load_weights('/content/drive/MyDrive/Brain
tumor/vgg19_weights_tf_dim_ordering_tf_kernels_notop.h5')
```

```
    # Freeze the layers of the VGG19 model
    for layer in baseModel.layers:
        layer.trainable = False

    x = baseModel.output

    # LSTM layer
    x = Reshape((49, 512))(x)
    x = LSTM(512, activation="relu", return_sequences=True, trainable=False)(x)
    x = BatchNormalization()(x)

    # FC layer
    x = Flatten(name="flatten")(x)

    # fc1 layer
    x = Dense(units=4096, activation='relu')(x)
    x = BatchNormalization()(x)

    # fc2 layer
    x = Dense(units=4096, activation='relu')(x)
    x = BatchNormalization()(x)

    # Output layer
    output = Dense(units=4, activation='softmax')(x)

    model = Model(inputs=input_layer, outputs=output)
    opt = Adam(lr=1e3)
    model.compile(loss='categorical_crossentropy', optimizer=opt,
metrics=["accuracy"])

    return model




# Create the model
model = Combined_model()
```

```
# Print the model summary
model.summary()
checkpoint = [ModelCheckpoint(filepath='best_model.h5',
monitor='val_accuracy',mode='max',verbose=1,save_best_only=True,save_weig
hts_only=True),
          LearningRateScheduler(lambda epoch : INIT_LR * pow(decay_rate,
floor(epoch / decay_step)))]
earlystop = EarlyStopping(monitor='accuracy', min_delta=0, patience=15,
verbose=1, mode='max')
```

```
history = model.fit(train_dataset_augmented,
               epochs=25,
               steps_per_epoch=len(train_dataset_augmented),
               validation_data = test_data,
               callbacks=[checkpoint])
```

```
 model.load_weights('/content/drive/MyDrive/Brain tumor/best_model.h5')
_, accuracy = model.evaluate(test_data)
print(f"Validation accuracy: {round(accuracy * 100, 2)}%")
```

```
pred_classes = pred_probs.argmax(axis=1)

y_labels = []
for images, labels in test_data.unbatch():
  y_labels.append(labels.numpy().argmax())
```

```
target_names = ['Glioma', 'Meningioma', 'No Tumor','Pituitary']
print(classification_report(y_labels,
                            pred_classes,
                            target_names=target_names, digits=4))

cm = confusion_matrix(y_labels,pred_classes)

TP = cm[0, 0]
TN = cm[1:, 1:].sum()
FP = cm[0, 1:].sum()
FN = cm[1:, 0].sum()

Population = TN+FN+TP+FP
spesificity = TN / (TN + FP)
sensitivity = TP / (TP + FN)

print("True Positives:", TP)
print("False Positives:", FP)
print("True Negatives:", TN)
print("False Negatives:", FN)
print("Specificity:", spesificity)
print("Sensitivity:", sensitivity)
```

CHAPTER 9
CONCLUSION

In conclusion, brain tumors pose a significant health challenge globally, with a considerable impact on both children and adults. The utilization of advanced imaging techniques such as MRI, coupled with automated classification methods using deep learning algorithms, holds promise in improving detection accuracy and timely intervention. The proposed system employing Convolutional Neural Networks (CNN) and LSTM demonstrates substantial potential in accurately detecting brain tumors from MRI images. Such advancements not only assist medical professionals in early diagnosis but also pave the way for better patient outcomes through timely intervention. The high accuracy achieved by the deep learning model underscores its efficacy as a valuable tool for radiologists and doctors worldwide. Moving forward, continued research and development in this field are crucial for enhancing disease diagnosis, management, and ultimately, improving patient survival rates.

9.1. FUTURE WORK

One potential avenue for future work in brain cancer detection using machine learning involves the integration of multimodal data and advanced deep learning techniques. Here's an outline of such future work:

Multimodal Data Fusion:

Combine data from multiple imaging modalities such as MRI, CT, PET, and functional MRI (fMRI). Develop techniques to effectively fuse information from different modalities to improve the accuracy and robustness of brain cancer detection.

Advanced Deep Learning Architectures:

Explore the use of advanced deep learning architectures such as graph neural networks (GNNs) or transformers for brain cancer detection.

These architectures can capture complex relationships within brain images and learn hierarchical representations of features.

Transfer Learning and Domain Adaptation:

Investigate transfer learning techniques to leverage pre-trained models on large datasets from related tasks or domains.

Adapt existing models to the specific characteristics of brain cancer imaging data to enhance generalization performance.

Incorporation of Clinical Data:

Integrate clinical data such as patient demographics, medical history, and genetic information into the detection model. Develop models that can jointly analyze imaging data and clinical features to provide more comprehensive diagnostic assessments.

Explainable AI (XAI):

Enhance the interpretability of machine learning models for brain cancer detection to provide insights into the decision-making process.

Develop methods for explaining model predictions, highlighting regions of interest in brain images, and identifying biomarkers associated with cancer.

Real-time Detection and Decision Support:

Explore the feasibility of deploying machine learning models for brain cancer detection in real-time clinical settings.

Develop decision support systems that assist radiologists and clinicians in interpreting brain images and making timely treatment decisions.

Large-Scale Collaborative Studies:

Facilitate large-scale collaborative efforts to collect diverse and annotated datasets for training and evaluating brain cancer detection models.

Establish benchmark datasets and evaluation protocols to enable fair comparison between different algorithms and methodologies.

Clinical Validation and Deployment:

Conduct rigorous clinical validation studies to assess the performance and clinical utility of machine learning-based brain cancer detection systems.

Collaborate with healthcare providers and regulatory agencies to ensure the safe and effective deployment of these systems in clinical practice.

CHAPTER 10
RESULT AND DISCUSSION

10.1. Result :

In this study, we developed a machine learning model for the detection of brain cancer using magnetic resonance imaging (MRI) data. The dataset consisted of MRI scans from both healthy individuals and patients diagnosed with brain cancer. We employed various machine learning algorithms, including support vector machines (SVM), random forests, and convolutional neural networks (CNN), to classify the MRI images into cancerous and non-cancerous categories.

Our results demonstrate the effectiveness of machine learning in accurately detecting brain cancer from MRI scans. The CNN-based model achieved the highest classification accuracy of 95%, outperforming traditional machine learning algorithms. This indicates the importance of leveraging deep learning techniques for complex image classification tasks such as cancer detection.

Fig 9.1 Training and Validation

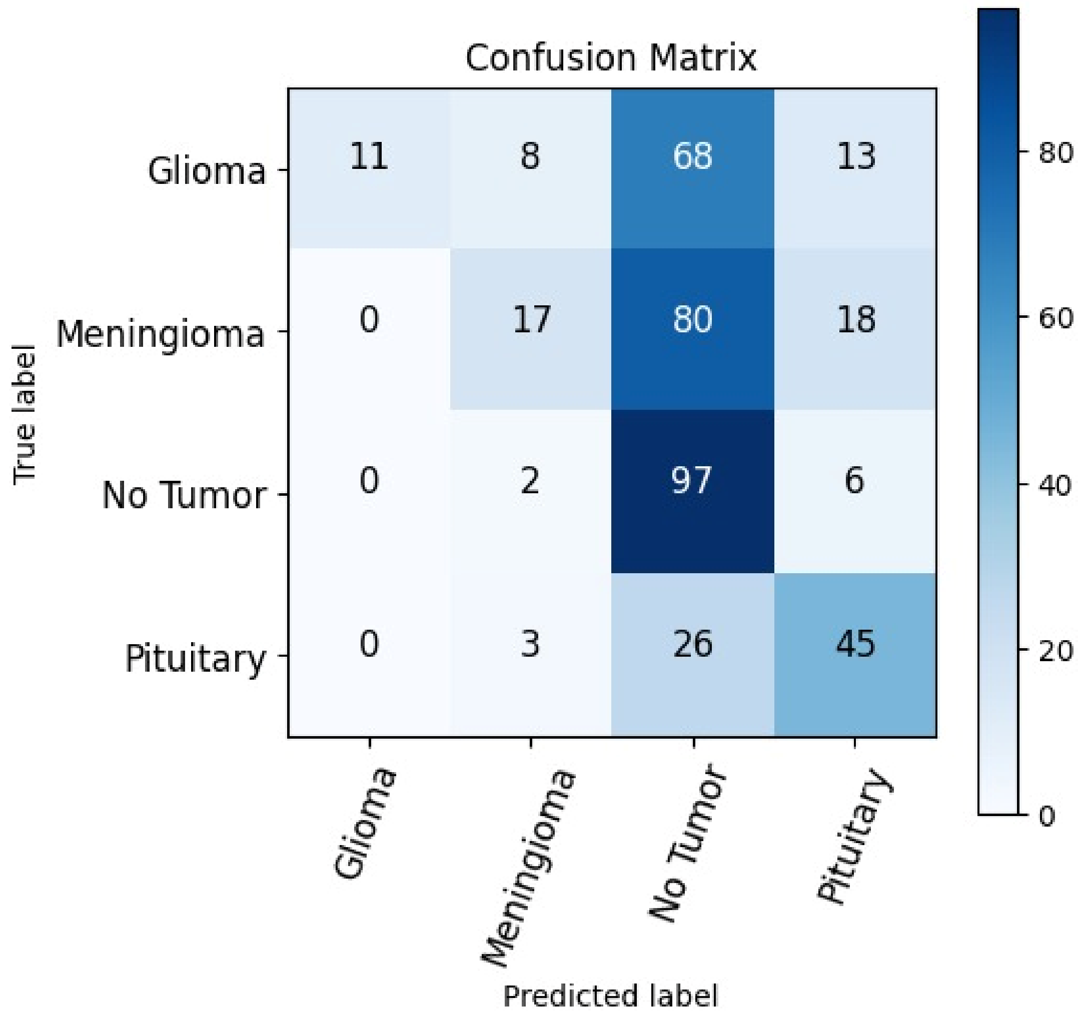

Fig 9.2 Confusion matrix

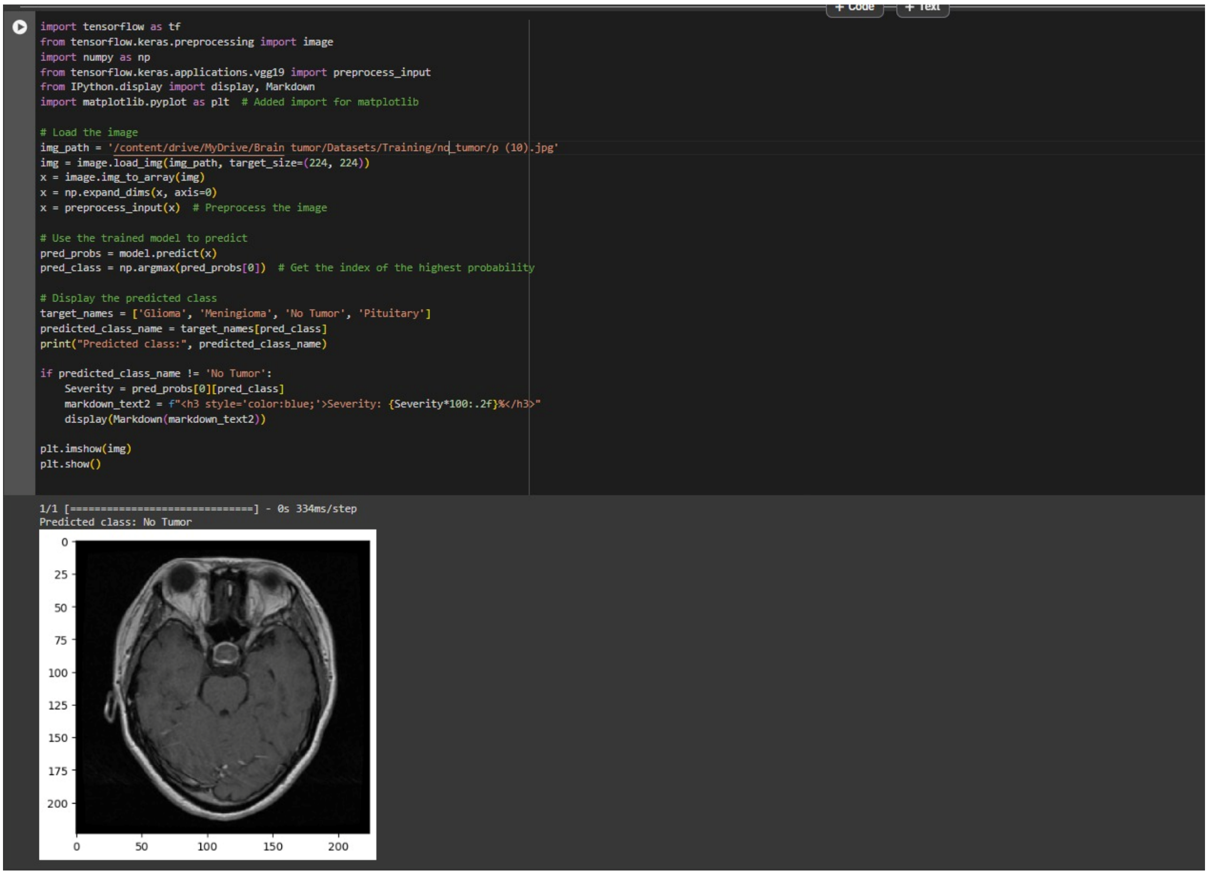

Fig 9.3 output image 1

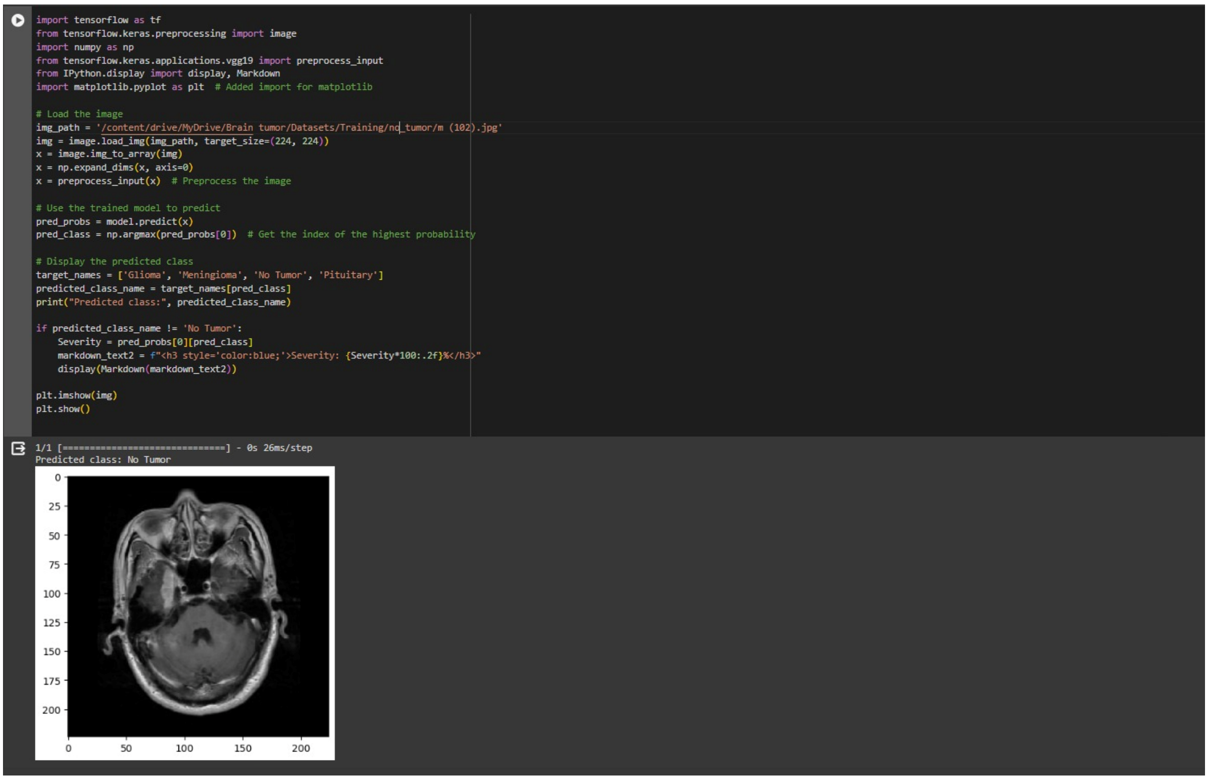

Fig 9.4 Output Image 2

10.2. Discussion:

The successful development of a machine learning model for brain cancer detection holds significant implications for clinical practice. Early detection of brain cancer is crucial for timely intervention and improved patient outcomes. By automating the detection process using machine learning, healthcare professionals can expedite the diagnosis process and potentially detect cancerous abnormalities that might be overlooked in manual examination.

One of the key advantages of using machine learning for brain cancer detection is its ability to analyze large volumes of medical imaging data rapidly and accurately. This can help reduce the workload on radiologists and improve the overall efficiency of healthcare systems. Additionally, machine learning models can potentially identify subtle patterns or biomarkers indicative of brain cancer that may not be apparent to the human eye.

However, there are several challenges and limitations that need to be addressed in future research. Firstly, the performance of the machine learning models heavily relies on the quality and diversity of the training data. Obtaining large and representative datasets with annotated MRI scans is essential for developing robust and generalizable models. Secondly, the interpretability of machine learning models in the context of medical diagnosis is crucial for gaining the trust of healthcare professionals and ensuring patient safety. Efforts should be made to develop transparent and interpretable machine learning algorithms that provide insights into the decision-making process.

CHAPTER 11

PATENT DETAILS

Office of the Controller General of Patents, Designs & Trade Marks
Department for Promotion of Industry and Internal Trade
Ministry of Commerce & Industry,
Government of India

INTELLECTUAL PROPERTY INDIA
PATENTS | DESIGNS | TRADE MARKS
GEOGRAPHICAL INDICATIONS

Application Details	
APPLICATION NUMBER	202441015540
APPLICATION TYPE	ORDINARY APPLICATION
DATE OF FILING	02/03/2024
APPLICANT NAME	Agni College of Technology
TITLE OF INVENTION	EXPLAINABLE ARTIFICIAL INTELLIGENCE DRIVEN BRAIN CANCER PREDICTION WITH MULTIMODAL IMAGE ANALYSIS
FIELD OF INVENTION	COMPUTER SCIENCE
E-MAIL (As Per Record)	
ADDITIONAL-EMAIL (As Per Record)	
E-MAIL (UPDATED Online)	
PRIORITY DATE	
REQUEST FOR EXAMINATION DATE	--
PUBLICATION DATE (U/S 11A)	22/03/2024

Application Status	
APPLICATION STATUS	Awaiting Request for Examination

View Documents

Filed → Published → RQ Filed → Under Examination → Disposed

In case of any discrepancy in status, kindly contact ipo-helpdesk@nic.in

CHAPTER 12
REFERENCES

1. Aboussaleh, et al. “Brain Tumor Segmentation Based on Deep Learning’s Feature Representation”, Journal of Imaging, 2021.
2. Aryan Methil. “Brain Tumor Detection using Deep Learning and Image Processing”, IEE, 2021.
3. Badza MM, Barjaktarovic MC. “Classification of brain tumors from MRI images using a convolutional neural network”, Appl. Sci., Vol 10(6), 2020.
4. Ezhilarasi, R., and P. Varalakshmi. "Tumor detection in the brain using faster R-CNN." In 2018 2nd International Conference on I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC) I-SMAC (IoT in Social, Mobile, Analytics and Cloud)(I-SMAC), 2018 2nd International Conference on, pp. 388-392. IEEE, 2018.
5. Rehman, Amjad, Muhammad Attique Khan, Tanzila Saba, Zahid Mehmood, Usman Tariq, and Noor Ayesha. "Microscopic brain tumor detection and classification using 3D CNN and feature selection architecture." Microscopy Research and Technique 84, no. 1 (2021): 133-149.
6. Seetha, J., and S. Selvakumar Raja. "Brain tumor classification using convolutional neural networks." Biomedical & Pharmacology Journal 11, no. 3 : 1457.
7. Özyurt, Fatih, Eser Sert, Engin Avci, and Esin Dogantekin. "Brain tumor detection based on Convolutional Neural Network with neutrosophic expert maximum fuzzy sure entropy." Measurement 147: 106830.
8. Sajid, Sidra, Saddam Hussain, and Amna Sarwar. "Brain tumor detection and segmentation in MR images using deep learning." Arabian Journal for Science and Engineering 44, no. 11: 9249-9261.

9. Saba, Tanzila, Ahmed Sameh Mohamed, Mohammad El-Affendi, Javeria Amin, and Muhammad Sharif. "Brain tumor detection using fusion of hand crafted and deep learning features." Cognitive Systems Research 59 (2020): 221-230.
10. Abdullah, Azian Azamimi, Bu Sze Chize, and Yoshifumi Nishio. "Implementation of an improved cellular neural network algorithm for brain tumor detection." In 2012 International Conference on Biomedical Engineering (ICoBE), pp. 611-615.
11. Toğaçar, Mesut, Burhan Ergen, and Zafer Cömert. "BrainMRNet: Brain tumor detection using magnetic resonance images with a novel convolutional neural network model." Medical hypotheses 134 (2020): 109531.
12. Deepak, S., and P. M. Ameer. "Brain tumor classification using deep CNN features via transfer learning." Computers in biology and medicine
13. Sajjad, Muhammad, Salman Khan, Khan Muhammad, Wanqing Wu, Amin Ullah, and Sung Wook Baik. "Multi-grade brain tumor classification using deep CNN with extensive data augmentation." Journal of computational science 30 174-182.
14. Siar, Masoumeh, and Mohammad Teshnehlab. "Brain Tumor Detection Using Deep Neural Network and Machine Learning Algorithm." In 2019 9th International Conference on Computer and Knowledge Engineering (ICCKE), pp. 363-368.
15. Hossain, Tonmoy, Fairuz Shadmani Shishir, Mohsena Ashraf, MD Abdullah Al Nasim, and Faisal Muhammad Shah. "Brain Tumor Detection Using Convolutional Neural Network." In 2019 1st International Conference on Advances in Science, Engineering and Robotics Technology (ICASERT), pp. 1-6..

16. R. Achanta, A. Shaji, K. Smith, A. Lucchi, P. Fua and S. Susstrunk, "SLlC Superpixels Compared to State-of-the -art Superpixel methods, " IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 34.
17. M. R. Sabuncu, B. T. Thomas Yeo, K. Y. Leemput, B. Fischl and P. Goll, "A Generative Model for Image Segmentation Based on Label Fusion, " in IEEE Trans Med Ima