BRAIN TUMOR CLASSIFICATION

A Mini Project Report

Submitted in partial fulfillment for the award of the degree of

BACHELOR OF TECHNOLOGY

in

ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

by

A. Sulakshana Sraavya	**22L31A5407**
K. Ramya Sri Sai	**22L31A5443**
K. Aasa Deepika	**22L31A5442**
D. John Wesley	**22L31A5423**
K. Deepika	**23L35A5412**

Under the Esteemed Guidance of

Mrs. N.R.S.L Prasanthi **M.Tech(PhD)**

Assistant Professor

DEPARTMENT OF ARTIFICIAL INTELLIGENCE AND DATA SCIENCE

VIGNAN'S INSTITUTE OF INFORMATION TECHNOLOGY VISAKHAPATNAM

(Autonomous)

Affiliated to JNTU GV & Approved by AICTE, NewDelhi ,Re-Accredited by NAAC (CGPA of 3.4/ 4.00)

ISO 9001:2008, ISO 14001:2004, OHSAS 18001:2007 Certified Institution

VISAKHAPATNAM – 530 039

VIGNAN'S INSTITUTE OF INFORMATION TECHNOLOGY

Department of Artificial Intelligence and Data Science

CERTIFICATE

This is to certify that this project report entitled **"Brain Tumor Classification"** is a bonafide record of the work done by **A. Sulakshana Sraavya 22L31A5407, K. Ramya Sri Sai 22L31A5443, K. Aasa Deepika 22L31A5442, D. John Wesley 22L31A5423, K. Deepika 23L35A5412** during the academic year 2022-2023, in partial fulfillment of the requirements for the award of the degree of Bachelor of Technology in Artificial Intelligence and Data Science of Jawaharlal Nehru Technological University, Vizianagaram. The results embodied in this project report have not been submitted to any other University or Institute for the award of any Degree or Diploma.

Project Guide
Mrs. N.R.S.L Prasanthi
Assistant Professor, AI&DS

Head of the Department
Dr. T.V Madhusudhana Rao
Professor,HoD-AI&DS

DECLARATION

We hereby declare that the project report entitled "**BRAIN TUMOR CLASSIFICATION**" has been written by us and has not been submitted either in part or whole for the award of any degree, diploma or any other similar title to this or any other university.

A.Sulakshana Sraavya	22L31A5407
K. Ramya Sri Sai	22L31A5443
K. Aasa Deepika	22L31A5442
D. John Wesley	22L31A5423
K. Deepika	23L35A5412

DATE: 23/08/2024

PLACE:VIIT,Duvvada

ACKNOWLEDGEMENT

It gives us a great sense of pleasure to acknowledge the assistance and cooperation we have received from several persons while undertaking this B. Tech. Mini Project.

We express our gratitude to our beloved and honourable principal **Dr.Sudhakar Jyothula**, for providing necessary departmental facilities in spite of his busy schedule and guiding us in every possible way

We also take the opportunity to acknowledge the contribution of **Prof. Dr. T. V. Madhusudhana Rao**, Head of the Department of artificial intelligence and data science, for his full support and assistance during the development of the project.

We owe special debt of gratitude to **Mrs. N.R.S.L Prasanthi**, Assistant Professor Department of Artificial Intelligence and Data Science, for her constant support and guidance throughout the course of our work. Her sincerity, thoroughness, and perseverance have been a constant source of inspiration for us.

We would like to thank **Vignan's Institute of Information Technology**, Duvvada for providing me with the facilities and resources necessary to complete this project. Your support has been invaluable.

We also do not like to miss the opportunity to acknowledge the contribution of all faculty members of the department for their kind assistance and cooperation during the development of our project. Last but not least, we acknowledge our friends for their contribution to the completion of the project.

ABSTRACT

The brain tumor is among the primary health challenges in the world today, as successful treatment relies heavily on early and accurate recognition. A brain tumor is a mass - non-cancerous - but can be in some or most cases a cancerous mass - and is characterized by its own growth which continues to multiply if left untreated. Magnetic Resonance Imaging (MRI) is the most commonly employed imaging procedure used in the diagnosis of brain tumors as MRIs provide both high contrast and high resolution imaging of organs. However, assessment of these scans on a manual basis is complex and time-consuming, as a necessary evaluation of the images will require radiologists to draw on great experience to pinpoint tumor presence. The accuracy of the diagnosis at that moment is woefully limited by the complexity and mastery of the backend features of the tumor, as well as the operator's experience.

Nonetheless, the increase in the number of patients in need of MRI scans has introduced large imaging data to analyze when screening. The volume of data is in many ways overwhelming to radiologists and making it much more difficult to accurately and quickly diagnose. Tumors can change considerably in volumes, shape and morphology, even within tumors of the same tumor type, and can often be mistaken for things or diseases completely different from each other. Therefore, this is problematic for classifying patients, as misdiagnoses only diminishes the quality of medical care that patients deserve to receive.

Brain tumor classification is a critical task in medical imaging, aiding in the accurate diagnosis and treatment of various tumor types. This study focuses on the classification of brain tumors into four categories: glioma, meningioma, pituitary tumors, and no tumor. Leveraging deep learning techniques, specifically the EfficientNetB0 model, the system processes MRI images to distinguish between these categories with high accuracy. The model's performance is evaluated using a dataset comprising labeled images representing each class. The results demonstrate the potential of this approach in automating brain tumor detection, thereby assisting radiologists in making informed decisions. This research underscores the significance of AI in enhancing diagnostic precision and optimizing patient outcomes.

CONTENTS

LIST OF FIGURES

NOMENCLATURE

- **MRI** – Magnetic Resonance Imaging
- **CNN** – Convulational neural networks
- **EfficientNetB0** – A deep learning model architecture used for image classification tasks
- **TensorFlow** – An open-source deep learning framework for building and training machine learning models
- **Scikit-Learn –** Machine Learning Algorithms Library
- **Confusion Matrix** – A table used to evaluate the performance of a classification model
- **Categorical Crossentropy** – A loss function used for multi-class classification problems
- **Global Average Pooling** – A layer that reduces the spatial dimensions of feature maps
- **Dropout** – A regularization technique to prevent overfitting in neural networks
- **Preprocessing** – Steps to prepare and clean MRI images for model input
- **Classification Report** – A summary of model performance metrics such as precision, recall, and F1-score
- **OS**- operating system
- **Shuffle** – A process of randomly reordering the training data to prevent model bias
- **Validation Split** – A portion of the training data set aside for evaluating model performance during training

CHAPTER I

INTRODUCTION

INTRODUCTION

The brain is the most complex and complicated part of the human body. Its structure is not easy to understand, as more than one hundred nerves communicate with each other to enable brain function. Brain-related diseases are spreading day by day. Some disorders can be detected at early stages, but some need proper diagnosis, like brain tumours, a type of brain-related disease that may lead to death. So, to overcome this we have introduced "Brain tumor classification using EfficientNetB0". In this documentation we provide comprehensive overview of classifying different tumours such as glioma, meningioma,pituitary and no tumour. To optimize we use EfficientNetB0 model to achieve the highest possible accuracy and efficiency in brain tumor classification.

1.1 MOTIVATION

The motivation for brain tumor classification lies in the need for accurate and reliable diagnosis and treatment planning for patients with brain tumors. Brain tumors can have a significant impact on a person's quality of life and can be life-threatening if not properly diagnosed and treated.

However, the diagnosis and treatment of brain tumors can be challenging, as the tumors can be difficult to distinguish from normal brain tissue using traditional imaging techniques. The use of advanced medical imaging techniques, such as MRI scans, has improved the ability to visualize brain tumors and make accurate diagnoses. However, the process of manually analyzing these images to classify and segment tumors can be time-consuming and prone to human error.

By using machine learning and image processing techniques to automate the process, the accuracy and reliability of diagnosis and treatment planning can be improved. The project is to improve the diagnosis and treatment of brain tumors by providing doctors with a fast and accurate way to classify tumors using MRI images. application, the accuracy and reliability of diagnosis and treatment planning can be improved. Additionally, the ability to classify tumors in 2 seconds or less, makes the use of the application in real-time practice easier and more efficient.

Accurate classification of brain tumors aids in early detection, which can significantly enhance the chances of successful treatment and survival. Early diagnosis can lead to interventions that may reduce the severity of symptoms and improve quality of life.

Automated classification systems can reduce the workload on radiologists and pathologists, speeding up the diagnostic process and allowing them to focus on more complex cases. Efficient classification methods can reduce the costs associated with diagnostic procedures and treatment planning, making healthcare more accessible and affordable.

Accurate classification of brain tumors is essential for early detection and personalized treatment, enabling clinicians to tailor interventions to the specific type and grade of the tumor. This precision not only optimizes treatment effectiveness but also minimizes potential side effects, ultimately contributing to better patient survival rates and quality of life.

1.2 PROBLEM DEFINTIONS

1. The old system Reduced accuracy in classification and potential misdiagnosis.
2. The old system Models trained on small or non-representative datasets may not perform well on real-world data.
3. MRI images can vary in resolution and may contain noise or artifacts that can interfere with tumor detection and classification.
4. Limited trust from medical professionals and challenges in integrating AI-driven insights into clinical workflows.
5. Difficulty in extracting and representing relevant features for accurate classification.

1.3 WHAT IS BRAIN TUMOR CLASSIFICATION?

Brain tumors are classified based on their origin, cell type, and malignancy. They can be broadly categorized into primary and secondary tumors. Primary tumors originate in the brain, while secondary tumors, also known as metastatic tumors, spread to the brain from other parts of the body. Tumors are further classified as benign or malignant. Benign tumors are non-cancerous, grow slowly, and have clear borders, whereas malignant tumors are cancerous, grow rapidly, and invade surrounding tissues.

Among primary brain tumors, gliomas are the most common and are classified based on the type of glial cells they originate from. Astrocytomas arise from astrocytes and range from low-grade (pilocytic astrocytoma) to high-grade (glioblastoma multiforme).

Other types of primary brain tumors include meningiomas, which arise from the meninges and can be benign, atypical, or malignant, and medulloblastomas, which are highly malignant tumors that occur in the cerebellum. Pituitary tumors, such as pituitary adenomas and carcinomas, originate in the pituitary gland.

1.4 OBJECTIVE OF THE PROJECT

The objective of brain tumor classification is to accurately identify and categorize brain tumors based on their type, grade, and other relevant characteristics, to guide effective treatment planning and prognosis. By determining the precise nature of a tumor, classification enables

clinicians to select the most appropriate treatment strategies, such as surgery, radiation, or chemotherapy, tailored to the tumor's specific features and the individual patient's needs. This classification also plays a crucial role in predicting patient outcomes, monitoring disease progression, and personalizing care. Moreover, it supports ongoing research into new therapies and diagnostic tools, enhances communication among healthcare providers, and ensures adherence to medical standards and regulatory requirements. Overall, brain tumor classification aims to improve patient care and outcomes through precise, informed decision-making.

ADVANTAGES

- **Accurate Diagnosis:** Provides a clear and precise identification of the tumor type
- **Effective Treatment Planning:** Helps in selecting the most suitable treatment options based on the tumor's characteristics**.**
- **Real-time monitoring:** The system allows for real-time monitoring, enabling immediate identification of attendance irregularities or unauthorized access.
- **Improved Clinical Decision-Making:** Ensures that treatment approaches align with the latest medical standards and practices, enhancing the overall quality of care.
- **Personalized Medicine**: Enhances treatment efficacy and reduces side effects by targeting therapies based on detailed tumor and patient characteristics**.**

CHAPTER II

LITERATURE SURVEY

LITERATURE SURVEY

The literature survey aims to provide an overview of development in brain tumor classification using EfficientNet B0, OpenCV, Python libraries, Tensorflow, CNN. By identifying relevant studies, methodologis, and performance evaluations, this survey shed light on the advancements and potentials of these technologies in the context of attendance tracking in educationl institutions and workplace.

EfficientNet-B0, introduced by Tan and Le in 2019, is part of the EfficientNet family of models designed to optimize both accuracy and computational cost. The key innovation behind EfficientNet is its compound scaling method, which uniformly scales all dimensions of depth, width, and resolution. This approach contrasts with previous methods that scaled dimensions individually, leading to suboptimal performance and increased computational burden.

OpenCV, an open-source computer vision library introduced by Bradski, G., & Kaehler, A. (2008), provides a robust framework for developing image processing and machine learning applications. This article examines how OpenCV can be employed to classify brain tumors effectively.

TensorFlow is a comprehensive open-source platform for machine learning developed by Google. It facilitates the development, training, and deployment of machine learning models, including deep learning neural networks. Its flexibility, scalability, and extensive library support make it an ideal tool for advancing medical image analysis.

Convolutional Neural Networks have significantly advanced brain tumor classification, offering automated and accurate solutions for analyzing medical images. While challenges remain, ongoing research and technological advancements continue to improve the effectiveness of CNNs in medical imaging. As CNNs evolve, they hold promise for further enhancing diagnostic processes and patient outcomes in neurology.

In conclusion ,Brain tumor classification using EfficientNet B0 offer a promising solution for accurate. By harnessing the power of advanced machine learning models, integrating multimodal data, and addressing interpretability and ethical considerations, the field will continue to evolve, offering more accurate, personalized, and efficient diagnostic solutions.

CHAPTER III

SYSTEM ANALYSIS

SYSTEM ANALYSIS

System analysis is a key aspect of the brain tumor classification project, involving a methodical approach to understanding, designing, and implementing a software solution that meets the needs of medical professionals and patients. In this phase, existing diagnostic processes and workflows are carefully analyzed to identify areas for improvement and to determine the essential functionalities and features required for an effective and accurate tumor classification system.

3.1 EXISTING SYSTEM

Existing systems for brain tumor classification utilized advanced techniques in medical imaging and machine learning to assist in diagnosing and categorizing brain tumors.

- **Radiomics-Based Systems**: These systems extract a large number of features from medical images (such as MRI) and use them for classification. Radiomics transforms images into data that can be analyzed to predict tumor type and grade.

- **Deep Learning Models**: Convolutional Neural Networks (CNNs) are widely used for brain tumor classification. These models are trained on labeled MRI datasets to classify tumor types, such as gliomas, meningiomas, and pituitary tumors

- **Support Vector Machines (SVM)**: SVMs are used in conjunction with feature extraction techniques to classify brain tumors based on MRI images. This approach is often combined with traditional image processing techniques.

- **Hybrid Models**: These systems combine multiple techniques, such as deep learning and traditional machine learning methods, to improve classification accuracy. For example, a hybrid system might use CNNs for feature extraction followed by an SVM for classification.

- **Cloud-Based AI Platforms**: Some modern systems utilize cloud-based platforms that provide AI-driven tools for brain tumor classification. These platforms often integrate with hospital information systems, offering scalability and remote access for analysis.

3.2 PROPOSED SYSTEM

Our brain tumor classification offers a vast range of advantages:

- **Precision and Dependability**: Utilizing the EfficientNetB0 model, our system ensures high accuracy and reliability in classifying brain tumors. The advanced convolutional neural network is trained to distinguish between glioma, meningioma, pituitary tumors, and no tumor with precision, reducing misclassifications and ensuring dependable diagnostic results.

- **Time and Cost Savings**: The system automates the classification process, significantly reducing the time required for manual diagnosis. By providing quick and accurate results, it alleviates the workload of medical professionals, leading to faster decision-making and cost-effective operations in clinical settings.

- **Enhanced Diagnostic Confidence**: The system's use of state-of-the-art deep learning algorithms ensures a high level of diagnostic confidence. With robust performance in identifying tumor types, it serves as a critical tool for radiologists and oncologists, supporting accurate treatment planning and improving overall patient outcomes.
- **Improved Patient Care**: By providing quick and accurate tumor classifications, the system facilitates timely and precise treatment decisions, enhancing patient care quality. Early and accurate diagnosis is critical for effective treatment planning, and this system aids in achieving that goal, contributing to better patient outcomes and potentially higher survival rates.

3.3 SOFTWARE REQUIREMENT SPECIFICATION

Software Requirement Specification (SRS) for Brain Tumor Classification Using Streamlit, Python, EfficientNetB0.

3.3.1. Introduction

3.3.1.1 Purpose

The purpose of this brain tumor classification project is to develop a machine learning system capable of accurately identifying and categorizing brain tumors from MRI scans.By leveraging the EfficientNetB0 model, the project aims to enhance diagnostic accuracy and efficiency in distinguishing between gliomas, meningiomas, pituitary tumors, and cases with no tumors. The system seeks to assist medical professionals by automating and streamlining the classification process, thereby improving patient outcomes through faster and more reliable diagnoses.

3.3.1.2 Scope

The Brain Tumor Classification will include the following features and functionalities:

- **Data Collection**: The system will gather MRI images of brain tumors from various sources, including different tumor types: glioma, meningioma, pituitary tumor, and no tumor.
- **Image Preprocessing**: It will preprocess the images by resizing and normalizing them to ensure consistent input for the model.
- **Model Training**: The system will utilize the EfficientNetB0 model, trained on the dataset, to classify the MRI images into the specified tumor categories.
- **Classification**: It will classify new MRI images in real-time, providing accurate predictions for the presence and type of brain tumors.
- **Evaluation and Reporting**: The system will evaluate its performance using metrics like accuracy, confusion matrix, and classification reports to assess the effectiveness of the classification.
- **Visualization:** It will include tools to visualize training progress, such as loss and accuracy plots, and provide user-friendly outputs for interpreting model predictions and confidence levels.

3.3.2. Functional Requirements

3.3.2.1 Data Collection and Preprocessing

- The system shall allow the loading of MRI images from specified directories for different brain tumor types.
- The system shall preprocess images by resizing them to a fixed size (150x150 pixels) to ensure uniform input for the classification model.
- The system shall categorize images into four labels: glioma tumor, meningioma tumor, pituitary tumor, and no tumor.
- The system shall convert image labels from strings to integer indices suitable for model training.

3.3.2.2 Model Training and Evaluation

- The system shall utilize the EfficientNetB0 model, pretrained on ImageNet, for feature extraction and build a custom classification model on top of it.
- The system shall compile the model using the Adam optimizer and categorical crossentropy loss function.
- The system shall train the model using the training dataset and validate its performance using a validation split.
- The system shall evaluate the model's performance using metrics such as accuracy, confusion matrix, and classification report.

3.3.2.3 Prediction and Visualization

- The system shall predict brain tumor types for new MRI images using the trained model.
- The system shall display the predicted label, actual label, and confidence of the prediction for random test images.
- The system shall provide visualizations of the training and validation loss and accuracy throughout the training process.

3.3.2.4 Performance Metrics and Reporting

- The system shall generate and display a confusion matrix to show the performance of the model across different tumor categories.

- The system shall provide a classification report detailing precision, recall, and F1-score for each tumor type.
- The system shall compute and display the test accuracy of the model on the test dataset.

3.3.3. Non-Functional Requirements

3.3.3.1 Performance

- The system should process MRI images and generate predictions within a specified time frame.
- It should handle a large number of images efficiently without significant delays.

3.3.3.2 Accuracy

- The classification model should achieve a high level of accuracy, precision, and recall for identifying and classifying brain tumors.

3.3.3.3 Reliability

- The system should be reliable and provide consistent performance, minimizing the chances of failure or errors during processing.

3.3.3.4 Usability

- The system should provide an intuitive and user-friendly interface for administrators to interact with, including uploading images and viewing results.

3.3.3.5 Maintainability

- The system should be easy to maintain and update, including the ability to incorporate new features or improvements as needed.

3.3.3.6 Scalability

- The system should be able to scale to accommodate an increasing number of images and users without degrading performance.

3.3.4 Glossary

- **EfficientNetB0**: A convolutional neural network architecture known for its efficiency and accuracy, pretrained on ImageNet and used as a feature extractor for image classification tasks.
- **OpenCV**: Open Source Computer Vision Library, providing tools and functions for image processing.
- **TensorFlow**: A Open source deep learning framework.
- **ImageNet**: : A large-scale image database used for training deep learning models, with a wide variety of labeled images across many categories.
- **Streamlit**: a Python library that creates interactive web applications for data science and machine learning projects.
- **OS**: Operating System, the software that manages computer hardware and software resources.

CHAPTER IV

SYSTEM DESIGN

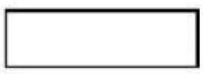
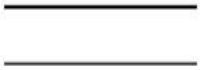
SYSTEM DESIGN

System design is a critical phase in the development of your project that focuses on translating the requirements gathered during the system analysis phase into a well-defined and structured solution. It involves designing the architecture, components, and interfaces of the system to ensure its functionality, reliability, and maintainability.

4.1 DATA FLOW DIAGRAM

A data flow diagram is a graphical view of how data is processed in a system in terms of input and output. Data flow diagram is a graphical representation of flow of data through an information system modelling as its process aspects. A DFD shows what kind of information will be input to and output from the system, how data will advance through the system and where the data will be stored. The Data flow diagram (DFD) contains some symbols for drawing the data flow diagram.

Data flow diagram symbol: -

Symbol	Description
	**Data Flow** – Data flow are pipelines through the packets of information flow.
	**Process:** A Process or task performed by the system.
	**Entity:** Entity is object of the system. A source or destination data of a system.
	**Data Store:** A place where data to be stored.

FLOW DIAGRAM

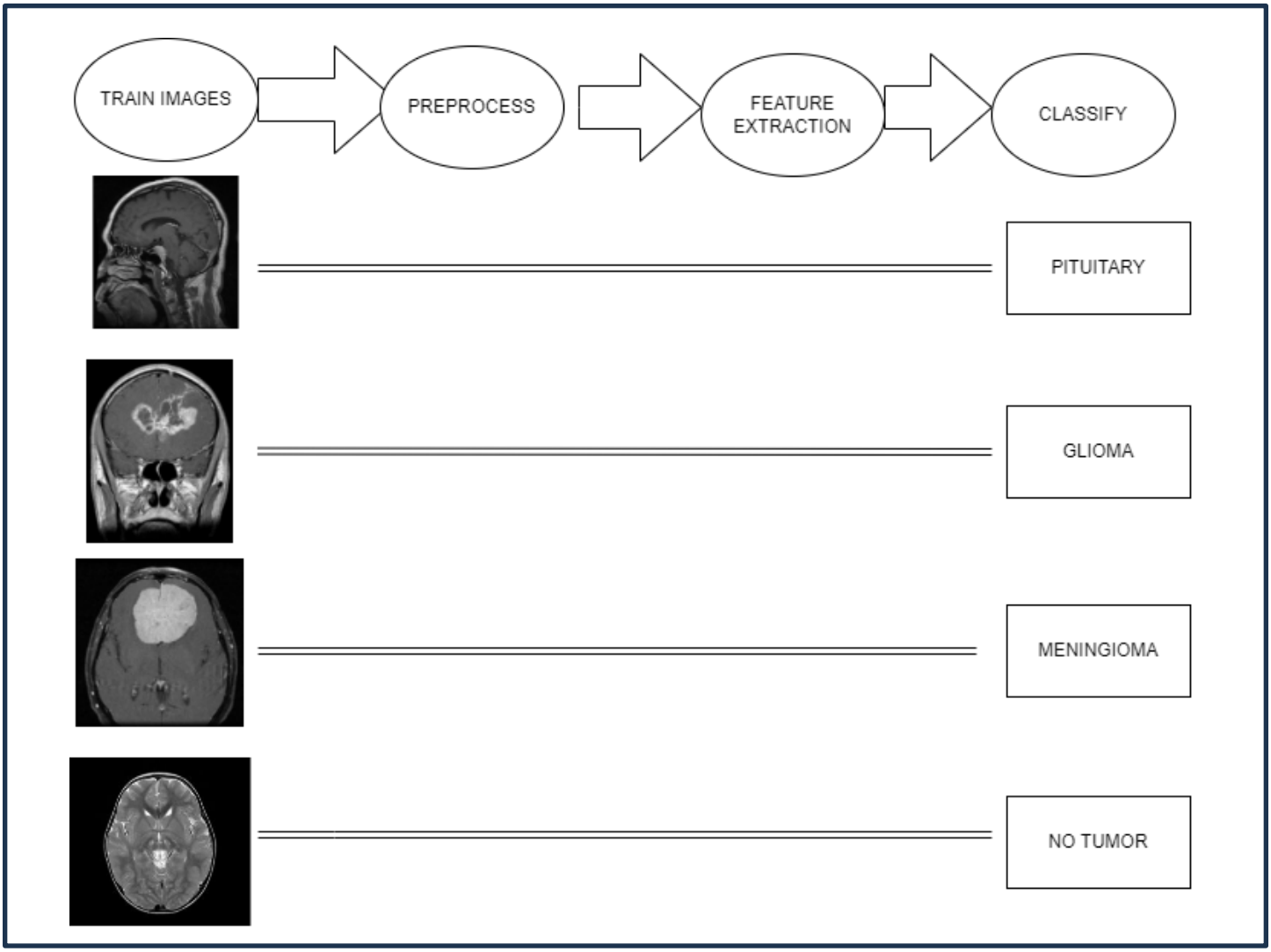


Fig - 4.2 : Flow Chart

4.2UML DIAGRAMS

UML means Unified Modelling Language. UML is used for visualizing, constructing and documenting the artifacts of software intensive systems. UML makes a clear conceptual distinction between models, views and diagrams. UML can be used to develop diagrams and provide users with ready-to-use, expressive modelling examples. Some UML tools generate program language code from UML.

UML can be used for modelling a system independent of a platform language. Unified Modelling Language (UML) is a non-proprietary specification language for object modelling. UML is a general-purpose modelling language that includes a standardized graphical notation used to create an abstract model of a system, referred to as a UML model.

UML is a standard language for specifying, visualizing, constructing, and documentingthe 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. OMG iscontinuously making efforts to create a truly industry standard.

- 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 systems.
- 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 flows in a manufacturing unit, etc.

4.2.1 USE CASE DIAGRAM

A use case diagram shows a set of use cases and actors and their relationships. Use case is represented as an eclipse/rectangle within a name inside it. Use cases are used to capture high level functionalities of a system .Each use case should provide some observable and valuable resultto the actors or other stakeholders of the system. Use case diagrams have a specialization of class diagrams and class diagrams are structured diagrams.

Use case diagrams are in fact two fold they are both behavior diagrams because they describe behavior of the system

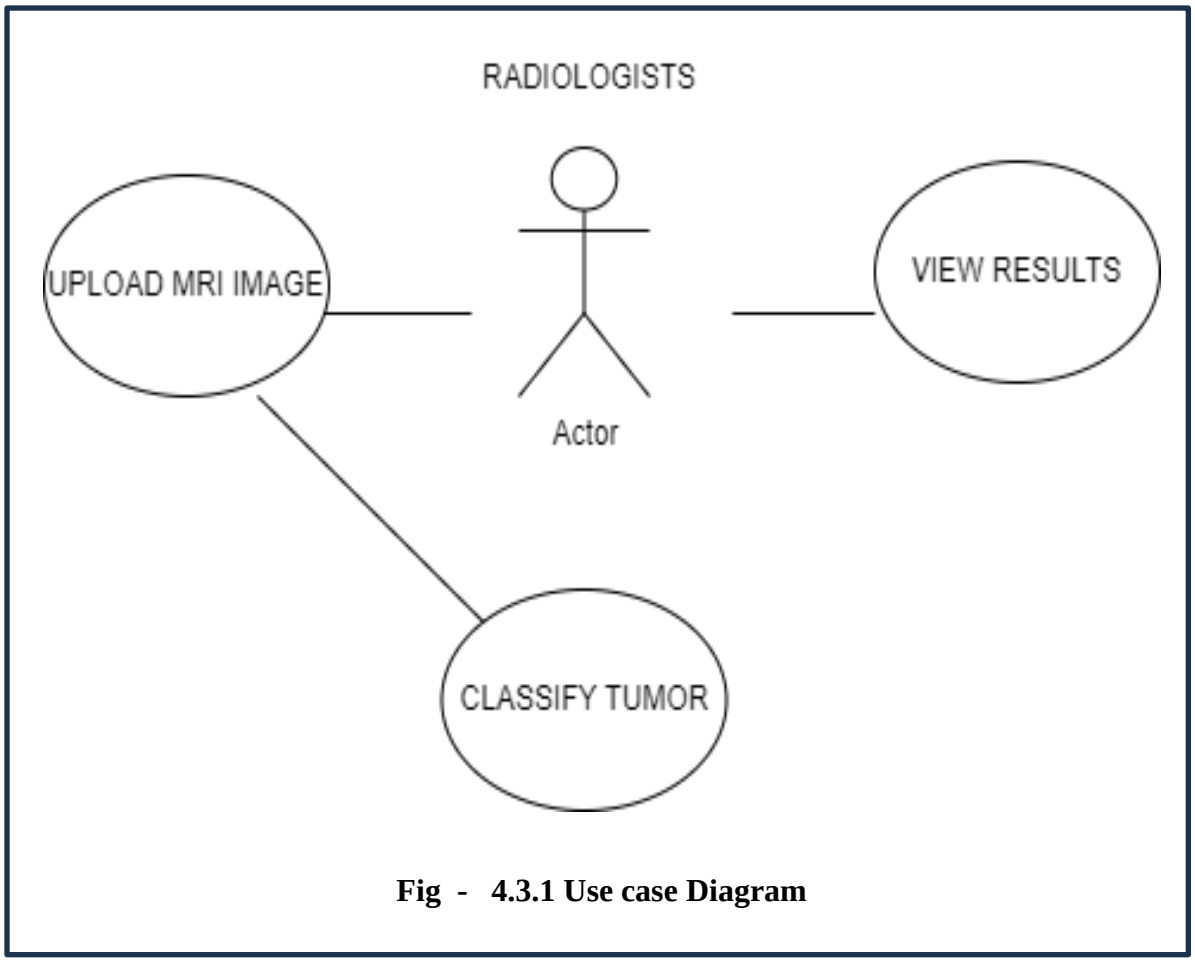


Fig - 4.3.1 Use case Diagram

4.3 EFFICIENTNETB0

EfficientNetB0 operates by processing an input image of size 150x150 through a series of convolutional layers, each designed to extract features at different levels of complexity. It starts with a simple 3x3 convolution followed by several blocks known as Mobile Inverted Bottle neck Convolution (MBConv) layers. These layers are efficient because they use depthwise separable convolutions, which significantly reduce the number of parameters while maintaining performance.The MBConv layers are repeated multiple times with different configurations of expansion factors, kernel sizes, and strides. This combination allows the model to capture spatial hierarchies and complex features from the image. As the image progresses through these layers, it is downsampled to reduce the spatial dimensions but increase the depth of feature maps.After the convolutional blocks, the features are passed through a 1x1 convolution layer to reduce the dimensionality, followed by a global average pooling layer that converts the feature maps into a single vector. This vector is then passed through a fully connected (dense) layer to produce the final classification output, which typically uses the Softmax activation function to generate probabilities for each class.

EfficientNetB0 is efficient because it maintains a balance between the model's accuracy and computational cost.

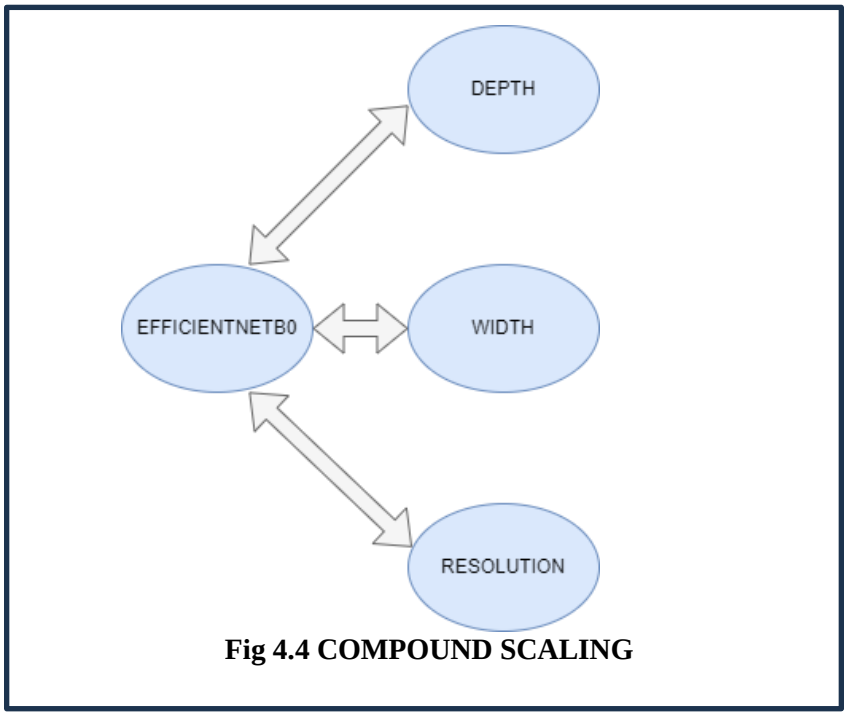


Fig 4.4 COMPOUND SCALING

CHAPTER V

IMPLEMENTATION

IMPLEMENTATION

During the implementation phase, the development team will translate the design specifications into actual code. They will follow software development best practices, such as writing clean and modular code, conducting unit tests, and ensuring proper documentation. The modules and implementation process play a crucial role in bringing the system design to life and creating a functional and reliable scheduling platform for doctors and patients.

5.1 INTRODUCTION TO TECHNOLOGIES USED

5.1.1 PYTHON

Python plays a crucial role in the above code for the Face Recognition Attendance System. Here are the key reasons why Python is important in this code:

- **Language Choice:** Python is the chosen programming language for implementing the Brain tumor classification. Python is known for its simplicity, readability, and extensive libraries.
- **OpenCV Integration:** The code integrates the OpenCV library. OpenCV provides a rich set of functions and algorithms for face detection, image processing. Python's compatibility with OpenCV allows seamless integration of these capabilities.

- **Machine Learning with scikit-learn:** Python’s scikit-learn library is utilized for training . Scikit-learn offers a comprehensive set of machine learning algorithms and tools, making it convenient to build and train models for.

- **Web Application Development with Streamlit:** Python's Streamlit web framework is used todevelop the web application for the Brain Classification. Flask provides a simple and lightweight framework for building web applications, making it suitable for this project.

- **Cyberpunk Aesthetic Enhancements:** The Matplotlib plots in the code are enhanced with a "cyberpunk" aesthetic, adding visual appeal to the data visualizations. Python's flexibility in customizing visual output makes it easy to integrate such stylistic elements.

In summary, Python's versatility, extensive libraries, and user-friendly syntax contribute to the successful implementation of the Face Recognition Attendance System. Python enables seamless integration of computer vision, machine learning, web development, and datahandling functionalities, making it a suitable choice for developing such applications.

5.1.2 STREAMLIT

- **Interactive Web Application Development:** Streamlit is employed to develop a web-based interface that allows users to interact with the trained machine learning model. This interface enables users to upload MRI images, visualize the data, and receive real-time predictions from the model.
- **User-Friendly Interface:** Streamlit simplifies the creation of user interfaces by providing an intuitive and straightforward API. It facilitates the development of interactive components such as file uploaders, buttons, and text displays, making it easy for users to engage with the system.
- **Real-Time Predictions:** Streamlit integrates with the model to provide real-time predictions based on user-uploaded images. This functionality allows users to quickly see the results of the model's analysis, including the predicted tumor type and the associated confidence score.
- **Visualization Capabilities:** Streamlit supports the integration of various visualizations, enabling users to view the MRI images along with the model's predictions and performance metrics. This helps in presenting the results in a clear and understandable manner.
- **Ease of Deployment:** Streamlit applications can be easily deployed to the web, making the MRI Brain Tumor Classification System accessible from any device with

internet access. This ease of deployment enhances the usability and accessibility of the system for users

5.1.3 EFFICIENTNETB0

EfficientNetB0 is a deep learning model architecture used in the MRI Brain Tumor Classification System for Image classification.

- **Pretrained Model**: EfficientNetB0 is employed as a pretrained model with weights initialized from training on the ImageNet dataset. This approach leverages transfer learning, allowing the system to benefit from the model's prior knowledge and reduce the need for extensive training from scratch.
- **Feature Extraction**: EfficientNetB0 serves as a feature extractor in the system. Its convolutional layers analyze and extract high-level features from MRI images, which are then used to identify and classify different tumor types.
- **Custom Classification Layers**: After the feature extraction, custom classification layers are added on top of EfficientNetB0. These layers include global average pooling, dense layers, and a final softmax activation layer to predict tumor types based on the extracted features.
- **Model Compilation and Training**: EfficientNetB0, along with the custom layers, is compiled and trained using TensorFlow. The model's architecture is optimized for accuracy and efficiency, making it well-suited for classifying MRI images into predefined tumor categories.
- **Performance Improvement**: EfficientNetB0's efficient design and balanced architecture enhance the performance of the classification system. Its ability to achieve high accuracy with relatively fewer parameters and computations contributes to the system's effectiveness in diagnosing brain tumors.

In summary, EfficientNetB0 is utilized in the project as a powerful and efficient pretrained model for

feature extraction and classification. Its advanced architecture and transfer learning capabilities significantly improve the accuracy and efficiency of the MRI Brain Tumor Classification System.

5.1.4 SCIKIT-LEARN

Scikit-learn plays an important role in the above code for implemention. Here are several points highlighting the importance of scikit-learn in thecode:

- **Data Preprocessing**: Scikit-learn provides functions for shuffling and splitting the dataset into training and testing sets. This ensures that the data is randomly distributed and properly divided, which is essential for training the model and evaluating its performance
- **Label encoding:** The library is used to convert categorical labels (e.g., tumor types) into numerical format. This transformation is necessary for feeding the data into the neural network, which requires numerical input
- **Performance Metrics**: Scikit-learn is utilized to compute performance metrics such as the confusion matrix and classification report. These metrics are crucial for evaluating the accuracy and effectiveness of the trained model in classifying MRI images.
- **Model Evaluation**: Scikit-learn's tools for metrics and evaluation assist in assessing how well the model generalizes to unseen data. The confusion matrix and classification report generated by Scikit-learn provide insights into the model's strengths and weaknesses.
- Integration with Data Processing: Scikit-learn seamlessly integrates with other data processing libraries, such as NumPy. In your project, scikit-learn works in conjunction with NumPy to handle the numerical computations involved in data preprocessing and feature extraction.

In summary, Scikit-learn enhances the MRI Brain Tumor Classification System by offering robust .Tools for data preprocessing, label encoding, and model evaluation. Its capabilities stream line the management and analysis of data, contributing to the overall effectiveness of the machine learning pipeline.

5.1.5 TENSORFLOW

TensorFlow is a deep learning framework used in the MRI Brain Tumor Classification System for building and training the neural network model. Here's how TensorFlow is utilized

- **Deep Learning Framework**: TensorFlow provides the foundational tools and libraries for developing and training deep learning models. It supports the creation of complex neural

network architectures, including the integration of EfficientNetB0 for image classification tasks.

- **Model Building**: TensorFlow's Keras API is used to define and construct the neural network model. This includes adding layers such as global average pooling, dense layers, and dropout layers to build a custom classification model on top of the EfficientNetB0 base.
- **Model Compilation**: TensorFlow handles the compilation of the model, specifying the optimizer, loss function, and evaluation metrics. In this project, the model is compiled with the Adam optimizer and categorical crossentropy loss function, which are crucial for training the classification model.
- **Training and Validation**: TensorFlow manages the training process, including fitting the model on the training data, validating performance on a separate validation set, and adjusting the learning rate based on the validation results. It also supports callbacks like `ReduceLROnPlateau` to optimize training performance.
- **Prediction and Evaluation**: TensorFlow facilitates the prediction of tumor types from MRI images and evaluates the model's performance using metrics like accuracy and loss. It provides functions for making predictions and computing performance metrics, such as confusion matrices and classification reports.
- **Model Saving and Deployment**: TensorFlow allows for saving the trained model to a file, which can be loaded and used for future predictions. This feature is used to save the `efficientnetB0.h5` model for deployment and further use.

5.1.6 NUMPY

NumPy is a fundamental library used in the MRI Brain Tumor Classification System for numerical operations and array handling. Here's how NumPy is utilized:

- **Array Handling**: NumPy provides efficient and flexible data structures for managing large arrays of image data. In the project, MRI images and their labels are stored and manipulated as NumPy arrays, facilitating efficient data processing and analysis.
- **Mathematical Operations**: NumPy supports a wide range of mathematical operations and functions. It is used for performing operations on the image data, such as resizing images, normalizing pixel values, and calculating statistics.

- **Data Manipulation**: NumPy assists in manipulating and transforming data. For example, it is used to convert lists of image data and labels into NumPy arrays, which are then fed into the machine learning model.
- **Performance Optimization:** NumPy's optimized performance for numerical computations enhances the efficiency of data processing. Its array operations are implemented in C, making them faster than equivalent operations using native Python lists.
- **Integration with Other Libraries**: NumPy integrates seamlessly with other libraries used in the project, such as TensorFlow and Matplotlib. It provides the necessary data structures and functions that these libraries depend on for model training, prediction, and visualization.

5.2 SAMPLE CODE

BACKENED

```
import os
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
plt.style.use("cyberpunk")
import cv2
import tensorflow as tf
from sklearn.utils import shuffle
from sklearn.model_selection import train_test_split
from tensorflow.keras.models import Sequential
from tensorflow.keras.layers import GlobalMaxPooling2D,GlobalAveragePooling2D, Dropout,
Dense, BatchNormalization
from tensorflow.keras.callbacks import  ReduceLROnPlateau,
from sklearn.metrics import classification_report,confusion_matrix
from tqdm import tqdm
import random
x_train=[]
```

```
y_train=[]
labels=['glioma_tumor','no_tumor','meningioma_tumor','pituitary_tumor']
image_size=150
for i in labels:
    folderPath = os.path.join('D:\Training',i) #two paths ni set chestundi D:\Training\no_tumor
    for j in tqdm(os.listdir(folderPath)):
        img=cv2.imread(os.path.join(folderPath,j))
        img=cv2.resize(img,(image_size,image_size))
        x_train.append(img)
        y_train.append(i)
for i in labels:
    folderPath = os.path.join('D:\Testing',i)
    for j in tqdm(os.listdir(folderPath)):
        img=cv2.imread(os.path.join(folderPath,j))
        img=cv2.resize(img,(image_size,image_size))
        x_train.append(img)
        y_train.append(i)
  x_train = np.array(x_train) #python lists are changed to numpy array
y_train = np.array(y_train) #python lists are changed to numpy arrays
a =len(x_train)
print(a)
b=random.randint(0,a) #a=3264
print(y_train[b])
print(plt.imshow(x_train[b]))
label_counts = {label: np.sum(label == y_train) for label in labels} #each label how many images it
contains
print(label_counts)
plt.figure(figsize=(8,6))
colors=["red","purple","blue","green"]
plt.subplot(2,1,1)
bars=plt.bar(label_counts.keys(),label_counts.values(),color=colors)
mplcyberpunk.add_bar_gradient(bars=bars)
```

```
#plt.xlabel('Types of Tumors')
plt.ylabel('Counts')
plt.title('Distribution of Labels')



k=0
for i in labels:
    j=0
    while True:
        if(y_train[j]==i):
            plt.subplot(2,4,k+5)
            plt.imshow(x_train[j])
            plt.axis('off')
            k+=1
            break
        j+=1
plt.tight_layout()
plt.show()
x_train , y_train = shuffle(x_train , y_train , random_state=101)
x_train , x_test , y_train , y_test = train_test_split(x_train , y_train, test_size=0.2, random_state=101)
#labels=['glioma_tumor','no_tumor','meningioma_tumor','pituitary_tumor']
y_train_new = [labels.index(i) for i in y_train] #strings to integers
y_test_new = [labels.index(i) for i in y_test]
print(y_train_new[0])
y_train = tf.keras.utils.to_categorical(y_train_new, num_classes=len(labels))
y_test = tf.keras.utils.to_categorical(y_test_new, num_classes=len(labels))
print(y_train[0])
print(np.argmax(y_train[0]))
# Load the EfficientNetB0 model pretrained on ImageNet without the top layers
efficientnetB0 = tf.keras.applications.EfficientNetB0(weights='imagenet',
                                                      include_top=False,
                                                      input_shape=(image_size, image_size, 3))
```

```
# Build the custom model on top of the EfficientNetB0 base
model = efficientnetB0.output
model = tf.keras.layers.GlobalAveragePooling2D()(model)
model = tf.keras.layers.Dense(1024,activation='relu')(model)
model = tf.keras.layers.Dropout(rate=0.4)(model)
model = tf.keras.layers.Dense(4,activation='softmax')(model)
model = tf.keras.models.Model(inputs=efficientnetB0.input, outputs = model)

# Compile the model
model.compile(optimizer='adam', loss='categorical_crossentropy', metrics=['accuracy'])
model.summary()
# ReduceLROnPlateau callback to reduce learning rate if validation accuracy plateaus
reduce_lr = ReduceLROnPlateau(monitor='val_accuracy', factor=0.1, patience=2,
                              min_delta=0.0001, verbose=1)
history = model.fit(x_train,y_train,validation_split = 0.1, epochs = 12, verbose = 1,
                    batch_size = 32, callbacks=[reduce_lr])
model.save('efficientnetB0.h5')
# Plotting training and validation loss
plt.figure(figsize=(12, 5))

plt.subplot(1, 2, 1)
plt.plot(history.history['loss'], label='Train Loss')
plt.plot(history.history['val_loss'], label='Validation Loss')
plt.title('Training and Validation Loss')
plt.xlabel('Epoch')
plt.ylabel('Loss')
plt.legend()
plt.grid(True)
mplcyberpunk.make_lines_glow()

# Plotting training and validation accuracy
```

```
plt.subplot(1, 2, 2)
plt.plot(history.history['accuracy'], label='Train Accuracy')
plt.plot(history.history['val_accuracy'], label='Validation Accuracy')
plt.title('Training and Validation Accuracy')
plt.xlabel('Epoch')
plt.ylabel('Accuracy')
plt.legend()
plt.grid(True)
mplcyberpunk.make_lines_glow()

plt.tight_layout()
plt.show()
y_true_test = np.argmax(y_test, axis=1)
y_pred_test = np.argmax(model.predict(x_test), axis=1)

conf = confusion_matrix(y_true_test, y_pred_test)
print(f'Confusion matrix:\n{conf}')
print(classification_report(y_true_test,y_pred_test))
x = len(x_test)
print(x)
random_index = np.random.randint(0, x)
random_img = x_test[random_index]
predictions = model.predict(random_img.reshape(1, 150, 150, 3))  # Reshape and preprocess the
image

# Interpret the model's predictions
predicted_class = np.argmax(predictions)  # Get the index of the class with the highest probability
predicted_label = labels[predicted_class]  # Convert class to label
confidence = predictions[0][predicted_class]

actual_index = y_test[random_index]  # Get the one-hot encoded actual class
actual_class = np.argmax(actual_index)
```

```
actual_label = labels[actual_class]

# Display the image and prediction information
print(f"\033[94mPredicted label: {predicted_label}\033[0m \n\033[92mActual label:
{actual_label}\033[0m \n\033[93mConfidence: {confidence*100:.2f}%\033[0m\n")
plt.figure(figsize = (3,3))
plt.imshow(random_img)
plt.axis('off')
plt.show()
test_loss, test_accuracy = model.evaluate(x_test, y_test, verbose=1)
print(f"Test Accuracy: {test_accuracy*100:.2f}%")
```

FRONTEND

```
import streamlit as st
import tensorflow as tf
import numpy as np
import cv2
from PIL import Image
import matplotlib.pyplot as plt
from gtts import gTTS
import os
import IPython.display as ipd
import pyttsx3

# Load the trained model
model = tf.keras.models.load_model("C:/Users/Sraavya/Downloads/efficientnetB0.h5")
```

```
# Define the labels
labels = ['glioma_tumor', 'no_tumor', 'meningioma_tumor', 'pituitary_tumor']

# Define the function to classify the uploaded image
def classify_image(image):
    if image.shape[-1] == 4:
        # Convert RGBA image to RGB
        image = cv2.cvtColor(image, cv2.COLOR_RGBA2RGB)
    image = cv2.resize(image, (150, 150))  # Resize image to the required input size
    image = np.expand_dims(image, axis=0)  # Add batch dimension
    image = tf.keras.applications.efficientnet.preprocess_input(image)  # Preprocess the image
    predictions = model.predict(image)  # Predict the class
    predicted_class = np.argmax(predictions)  # Get the class with the highest probability
    predicted_label = labels[predicted_class]  # Map the class index to the label
    confidence = predictions[0][predicted_class]  # Get the confidence of the prediction

    return predicted_label, confidence, predictions

# Streamlit interface
st.set_page_config(
    page_title="Brain Tumor Detection",
    page_icon="🧠",
    layout="centered",
    initial_sidebar_state="expanded",
)

st.title("Brain Tumor Detection using EfficientNetB0")
```

```
st.markdown('''
This app uses a pre-trained EfficientNetB0 model to classify MRI images of the brain into one of the
    following categories:
- **Glioma Tumor**
- **No Tumor**
- **Meningioma Tumor**
- **Pituitary Tumor**

Upload an MRI image to get started.
''')

# File uploader
uploaded_file = st.file_uploader("Choose an image...", type=["jpg", "jpeg", "png"])

if uploaded_file is not None:
    # Display the uploaded image
    image = np.array(Image.open(uploaded_file))
    st.image(image, caption='Uploaded Image', use_column_width=True)

    st.markdown("### Classifying...")

    # Classify the image
    label, confidence, predictions = classify_image(image)

    # Generate speech
    text=f"The predicted label is {label} with a confidence of {confidence*100:.2f} percent."
    # Display the results
    st.success(f"**Predicted Label:** {label}")
```

```
st.info(f"**Confidence:** {confidence*100:.2f}%")

# Display a matplotlib plot for confidence scores
fig, ax = plt.subplots()
ax.barh(labels, predictions[0], color='skyblue')
ax.set_xlim([0, 1])
ax.set_xlabel('Confidence')
ax.set_title('Prediction Confidence')
st.pyplot(fig)
```

CHAPTER VI

SCREENSHOTS

6.1 HOME PAGE

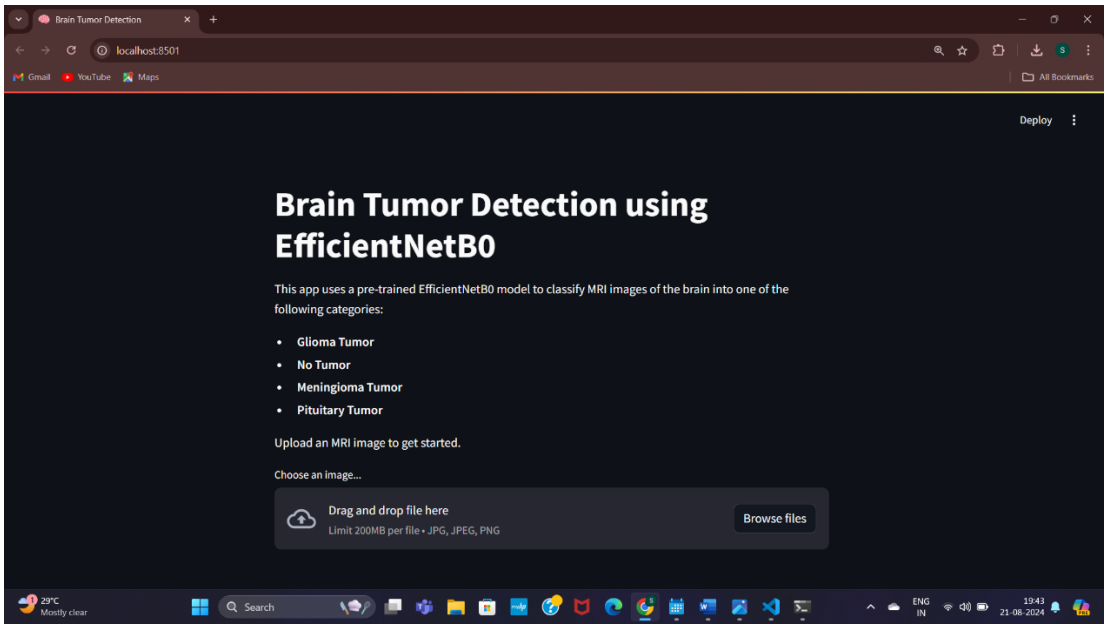


Fig - 6.1 : Home page

6.2 UPLOADING MRI SCAN

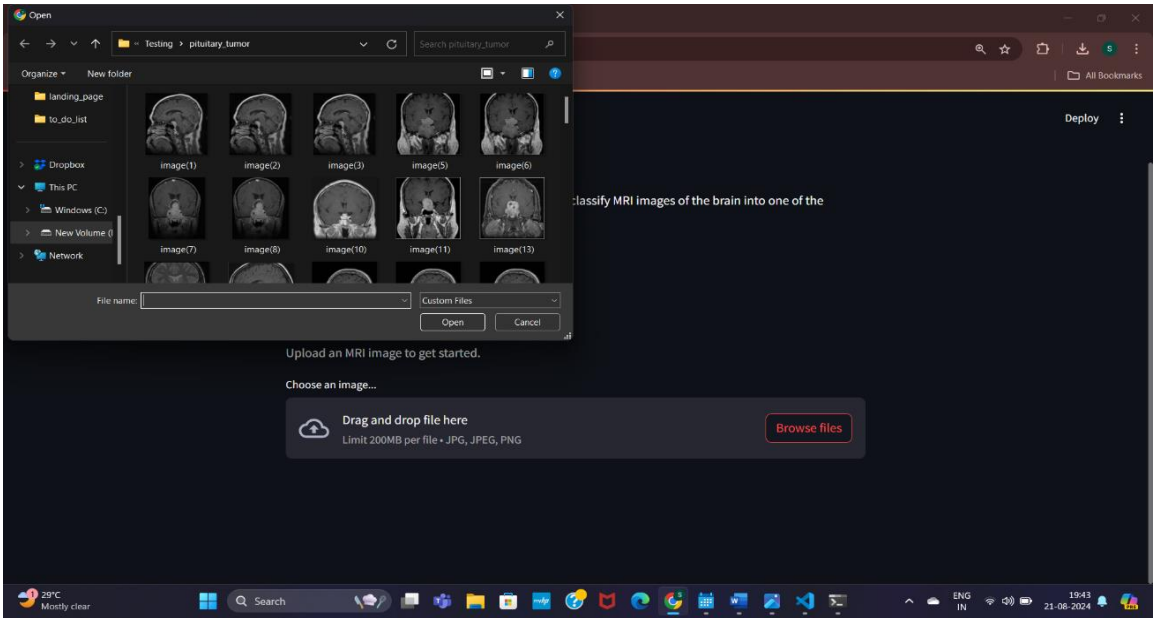


Fig- 6.2: Uploading MRI Scan

6.3 CLASSIFYING

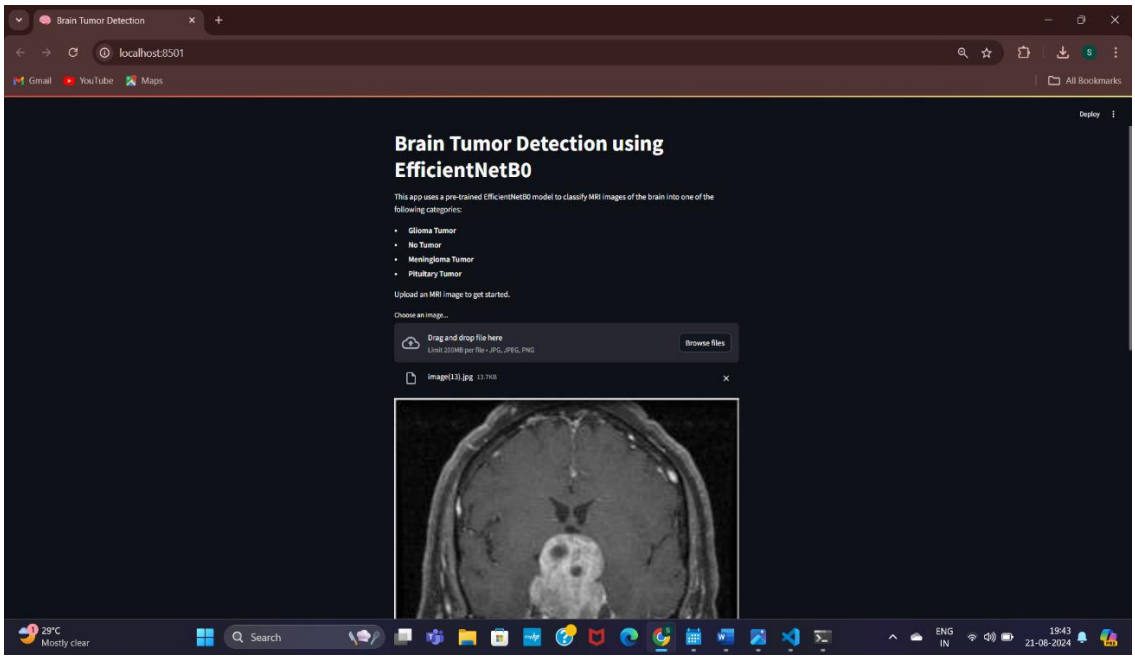


Fig 6.3- Classifying

6.3RESULTS

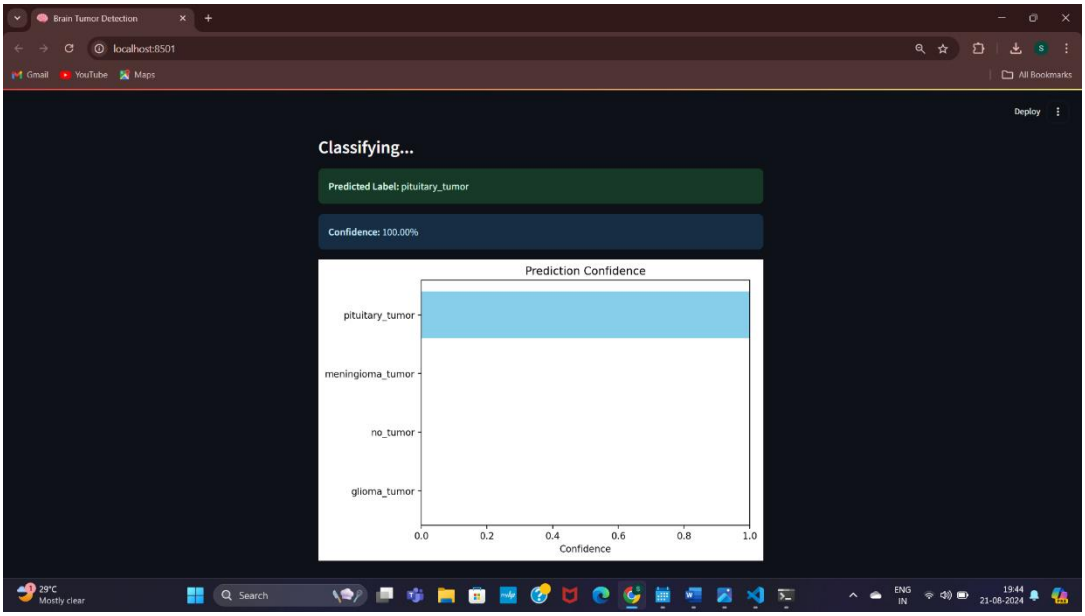


Fig 6.4- Results

CHAPTER VII

CONCLUSION AND FUTURE ENHANCEMENTS

7.1 CONCLUSION

In conclusion, the Brain tumor Classification offers a transformative approach to the critical task of diagnosing brain tumors using MRI scans. By harnessing the capabilities of deep learning and transfer learning, this system significantly advances the accuracy and efficiency of tumor detection and classification. The project integrates cutting-edge libraries and frameworks, including TensorFlow, EfficientNetB0, OpenCV, and Keras, to deliver precise and reliable results. The incorporation of machine learning algorithms further refines the system's ability to distinguish between different types of brain tumors, such as glioma, meningioma, pituitary tumors, and cases with no tumor, with exceptional accuracy. Additionally, the use of data visualization tools like matplotlib and seaborn ensures a clear presentation of the model's performance and outcomes, making the system both powerful and user-friendly for medical professionals.

The system's advantages are substantial. It streamlines the diagnostic process, reducing the time and effort required for manual interpretation of MRI scans. This automated approach not only enhances the accuracy of diagnoses but also helps in early detection, which is crucial for effective treatment planning. The model's scalability allows for continuous improvement as more data becomes available, ensuring that it remains relevant and effective in diverse clinical settings. Moreover, the deployment of EfficientNetB0 as the backbone for feature extraction brings state-of-the-art accuracy with reduced computational overhead, making it suitable for real-time applications

With its seamless integration of advanced technologies, the Brain Tumor Classification Project sets a new benchmark in medical diagnostics. Its potential applications extend beyond individual healthcare facilities to large-scale diagnostic centers and research institutions, where rapid and accurate analysis of brain MRI scans is essential. By automating and enhancing the diagnostic process, this system empowers healthcare providers to improve patient outcomes, optimize resource allocation, and focus on critical aspects of patient care.

The successful implementation of this project underscores the immense potential of deep learning and computer vision in medical diagnostics. As these technologies continue to

evolve, the Brain Tumor Classification Project paves the way for future innovations in automated healthcare, data-driven decision-making, and personalized medicine. By embracing these advancements, the medical community can move toward a future where early and accurate diagnosis of brain tumors is the standard, leading to better patient outcomes and more efficient healthcare delivery.

7.2FUTURE ENHANCEMENTS

- **Data Augmentation** : Further data augmentation techniques can be explored to increase the robustness of the model, such as rotation, zooming, and flipping, to simulate various real-world scenarios.
- **Hyperparameter Tuning :** Further data augmentation techniques can be explored to increase the robustness of the model, such as rotation, zooming, and flipping, to simulate various real-world scenarios.
- **Model Ensemble :** Consider implementing an ensemble of different models (e.g., VGG16, ResNet50) to improve accuracy and robustness by combining their predictions.
- **Transfer Learning with Fine-Tuning :** Fine-tuning the entire EfficientNetB0 model or selected layers can be explored to improve performance, especially if more labeled data becomes available.
- **Real-time Application :** Integrate the model into a real-time application, such as a web-based interface, where users can upload brain scans for instant tumor classification.
- **Explainability :** Implement explainability techniques like Grad-CAM to visualize and understand which parts of the brain scan the model focuses on while making predictions, enhancing the interpretability of the model.
- **Deployment :** Deploy the trained model into a production environment using cloud platforms like AWS, Azure, or GCP, making it accessible for clinical use or further research.

By addressing these future enhancements, the model's performance, usability, and applicability in real-world scenarios can be significantly improved.

CHAPTER VIII

REFERENCES

REFERENCES

[1] R. Singh, N. Sharma and R. Gupta, "Proposed CNN model for classification of brain tumor disease", 2023 International Conference on Distributed Computing and Electrical Circuits and Electronics.

[2] Badža MM, Barjaktarović MČ. Classifcation of brain tumors from MRI images using a convolutional neural network. Appl Sci. 2020;10(6):1999

[3] Gumaei A, Hassan MM, Hassan MR, Alelaiwi A, Fortino G. A Hybrid feature extraction method with regularized extreme learning machine for brain tumor classifcation. IEEE Access. 2019;7:36266–73

[4] Pashaei A, Sajedi H, Jazayeri N. Brain tumor classifcation via convolutional neural network and extreme learning machines. In: 2018 8th international conference on computer and knowledge engineering (ICCKE). IEEE; 2018 Oct 25. p. 314–9.

[5] Abiwinanda N, Hanif M, Hesaputra ST, Handayani A, Mengko TR. Brain tumor classifcation using convolutional neural network. In: World congress on medical physics and biomedical engineering 2018. Singapore: Springer; 2019. p. 183–9

[6] Swati, Z.N.K.; Zhao, Q.; Kabir, M.; Ali, F.; Ali, Z.; Ahmed, S.; Lu, J. Brain tumor classification for MR images using transfer learning and fine-tuning. Comput. Med Imaging Graph. **2019**, 75, 34–46.

[7] Bodapati, J.D.; Shaik, N.S.; Naralasetti, V.; Mundukur, N.B. Joint training of two-channel deep neural network for brain tumor classification. Signal Image Video Process. **2021**, 15, 753–760.

[8] Zulfiqar, F.; Bajwa, U.I.; Mehmood, Y. Multi-class classification of brain tumor types from MR images using EfficientNets. Biomed. Signal Process. Control **2023**, 84, 104777.

[9] Brindha, P.G.; Kavinraj, M.; Manivasakam, P.; Prasanth, P. Brain tumor detection from MRI images using deep learning techniques. In Proceedings of the IOP Conference Series: Materials Science and Engineering, Sanya, China, 12–14 November 2021; IOP Publishing: Bristol, UK, 2021; Volume 1055, p. 012115.

[10] Zar Nawab Khan Swati et al., "Brain tumor classification for MR images using transfer

learning and fine-tuning", Computerized Medical Imaging and Graphics, vol. 75, pp. 34-46, 2019.

[11] Mohamed Arbane et al., "Transfer learning for automatic brain tumor classification using MRI images", 2020 2nd International Workshop on Human-Centric Smart Environments for Health and Well-being (IHSH), 2021.

[12] R. C. Suganthe et al., "Convolutional Neural Network Based Multi Class Classification Model for Brain Tumor Diagnosis", 2022 International Conference on Computer Communication and Informatics (ICCCI), 2022.