Automated Detection of Brain Tumor and Brain Stroke

Graduation Project

For Faculty of Computers and Information Technology

The Egyptian E-learning University-Beni Suef University (Joint program)

By

Abdallah Shehath madeh
Yasmin Osama Abdelmnem
Kloud tarek fathy Hassan
Abeer Yousief Abdelsamed
Beshoy Nashat wadie
Asmaa Samir Mawad
Youmna Mustafa
Anas Ahmed Elgarhy

Supervised By

Dr. Wafaa Samy

Assistant Professor at The Egyptian E-learning University

TA. Hussein Mohamed

Teaching Assistant at The Egyptian E-learning University

2024

Acknowledgement

First and foremost, we thank God for His grace and success. If it were not for His success and generosity toward us, we would not have reached this stage, achieved this success, and completed this work.
we would like to express our heartfelt gratitude and appreciation to our supervisors Dr. Wafaa Samy and TA Hussein Mohamad For their guidance, continuous support and encouragement For us all the time and for giving us a lot of their invaluable experience, which helped us to continue and always try to provide the best and achieve this work and also We extend our sincere thanks and appreciation to them for the time they invested in us and benefited us from their knowledge and experience.

We would also like to express our heartfelt, thanks to our families for their efforts They support and encourage us, bear our pressures, and believe in us and they motivate us.

Once again, thank for dedication and commitment to our academic success, and for helping us achieve our goals. We will always be grateful for the role you played in shaping our future.

Abstract

A brain tumor is a growth of cells in the brain or near it. Brain tumors can happen in the brain tissue. Brain tumors also can happen near the brain tissue. Nearby locations include nerves, the pituitary gland, the pineal gland, and the membranes that cover the surface of the brain.

Your skull, which encloses your brain, is very rigid. Any growth inside such a restricted space can cause problems. Brain tumors can be of two types i.e. cancerous (malignant) or non-cancerous (benign).

Brain cancer is a life-threatening disease, and it affects all the diagnosed people severely. Precise brain tumor classification helps to diagnose brain cancer early, which increases the survival rate of brain cancer patients, but it is quite hard to detect early. It is difficult to evaluate the magnetic resonance imaging images manually. Therefore, there is a need for optimized and fast digital methods for tumor diagnosis with better accuracy.

In this work we are using a deep learning model based on convolutional neural networks to identify and classify brain cancer and its type from MRI images of patients using publicly available dataset in kaggle.

For brain stroke it is a brain attack and it is a sudden interruption of continuous blood

flow to the brain and a medical emergency. A stroke occurs when a blood vessel in the brain becomes blocked or narrowed, or when a blood vessel bursts and spills blood into the brain.

Stroke is a major contributor to death and disability worldwide. Prior to modern therapy, post-stroke mortality was approximately 10% in the acute period, with nearly one-half of the patients developing

moderate-to-severe disability. The most fundamental aspect of acute stroke management is “time is brain”. In acute ischemic stroke, the primary therapeutic goal of reperfusion therapy, including intravenous recombinant tissue plasminogen activator (IV TPA) and/or endovascular thrombectomy, is the rapid restoration of cerebral blood flow to the salvageable ischemic brain tissue at risk for cerebral infarction. Several landmark endovascular thrombectomy trials were found to be of benefit in selecting patients with acute stroke caused by occlusion of the proximal anterior circulation, which has led to a paradigm shift in the management of acute ischemic strokes. In this modern era of acute stroke care, more patients will survive with varying degrees of disability post-stroke. A comprehensive stroke rehabilitation program is critical to optimize post-stroke outcomes. Understanding the natural history of stroke recovery, and adapting a multidisciplinary approach, will lead to improved chances for successful rehabilitation.

Here we are using a CNN model for deep learning multi-image classification which classifies input image into tumor, no tumor, stroke or no stroke.

Table of Contents

List of Figures

ABBREVIATIONS

AI:	Artificial Intelligence
CSF:	Cerebral Spinal Fluid
CT:	Computerized Tomography
MRI:	Magnetic Resonance Imaging
CNS:	Central Nervous System
PNS:	Peripheral Nervous System
CSF:	Cerebrospinal Fluid
DIP:	Digital Image Processing
ML:	Machine Learning
CNN:	Convolutional Neural Network
FEM:	Finite Element Method
DCE:	Dynamic Contrast Enhanced
IDE:	Integrated Development Environment
UI:	User Interface
UX:	User Experience

Chapter 1

Introduction

Introduction

1.1 Overview

Brain tumor segmentation and identification are the focus of this research.
To assess the anatomy of the brain, MRI or CT scans are frequently used.
MRI scanned image is utilized throughout this investigation. The MRI scan is less intrusive than the CT scan for assessing a patient's health.
Magnetic fields and radio waves are at the core of this technology.
Numerous algorithms have been developed to detect brain tumors. However, detection and extraction may be impeded.
This research includes two alternative segmentation approaches.
Fuzzy C mean and K-means clustering techniques. For this reason, it offers an accurate tumor segmentation result.
Any part of the body may become a tumor if its tissues develop out of control.
Depending on its location, a tumor could be either primary or secondary [1].
The word "primary" refers to everything that has a beginning. It is stated to be secondary if a piece of the tumor spreads and develops in a new place.
CSF is generally altered by brain tumors (Cerebral Spinal Fluid) (Cerebral Spinal Fluid). Strokes are caused by it.
Instead of treating cancer, the doctor offers treatment for strokes. Consequently, early detection of malignancies is important to the success of this treatment.
If found early enough, a brain tumor may be more efficiently treated, and the patient's life expectancy may be prolonged.
That will increase the life of the gadget by about a year or two.
T-cells and B-cells are the two most prevalent forms of cancer cells. Both the frequency and the severity of these tumors suggest their malignancy. Mass tumors make the detection of malignant tumor more a particular tumor location and detect a malignancy in the brain using this technology.

Treatment in most situations is focused on removing or eradicating the tumor [2].
If discovered and treated early, most brain tumors may be cured.
The number of persons with brain tumors is growing at an alarming pace due to their incapacity to speak.
This project's major purpose is to enable people to learn to build tools. A tool that can tell people whether or whether they are at danger of having a brain tumor, as well as how great of a risk they have.
Java is the programming language utilized to develop the detecting system.
As a penultimate step, we are designing technologies that can assess the tumor's morphology and stage from a particular tumor location.
The illustrations below illustrate the different parts of the brain. Two types of the nervous system are the central and peripheral nervous systems.
Most of the nervous system consists of the brain and spinal cord (CNS). Spinal cord and skull are both vital nerves (PNS) [3] of the Peripheral Nervous System.
The spinal cord and the brain give rise to the spinal and cranial nerves, respectively.
Previously, it described how the Peripheral Nervous Systems (PNS) regulate a variety of biological activities, including respiration, digestion, heart rate, and hormone secretion (PNS). It consists of the brainstem, cerebrum, and cerebellum as its three principal regions. The cerebrum and cerebellum are connected to the spinal cord via the brain stem. The cerebrum is the biggest area of the brain. It has a right as well as a left hemisphere. Cerebrum oversees all elements of a person's mental and emotional health, as well as their ability to communicate.

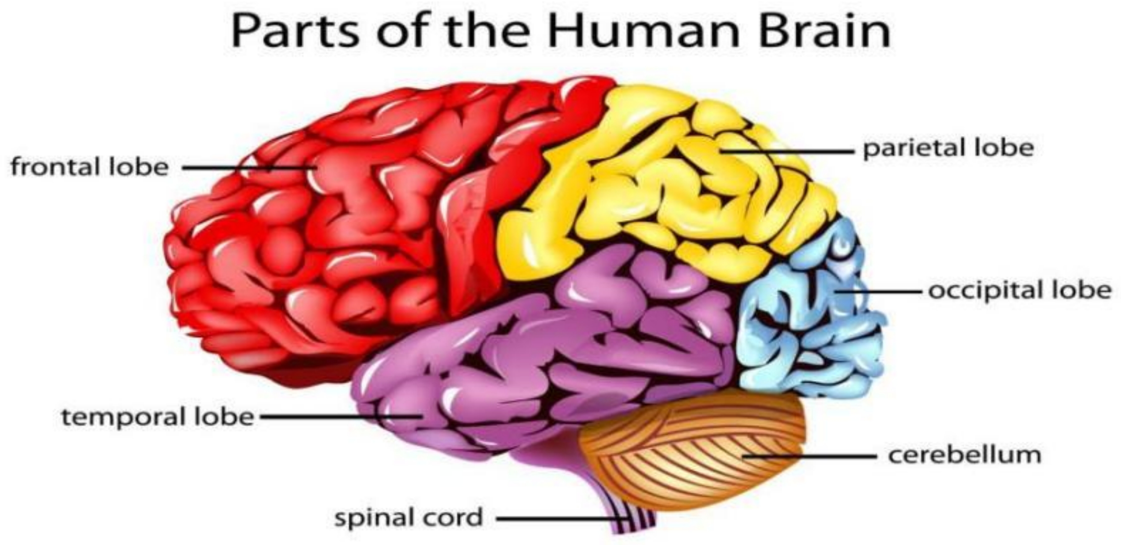


Figure1.1: Human Brain Part [3].

The cerebellum is the brain's subsidiary structure. The cerebrum must coordinate the movement of the muscles. Many regions of the brainstem included the midbrain, Pons, and medulla. The brain stem was responsible for numerous autonomic functions of the body, such as waking and sleeping cycles, digestion, sneezing, heart rate, breathing, body temperature, coughing, vomiting, and swallowing. The folds on the surface of the cerebrum are referred to as the cortex. In the 100 billion nerve cells in the brain, the cortex contains around 70% of them. Grey matter, the component that gives the cortex its gray-brown tone,
is composed of nerve cell bodies. As seen in Figure 1.2, the white matter is composed of the long, linked fibers known as axons found in the cortex.

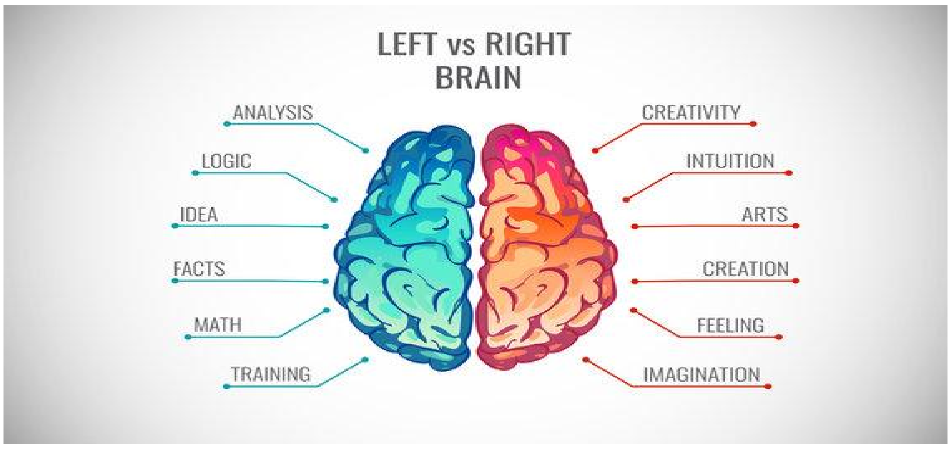


Figure1.2: Left Brain & Right Brain [4].

The corpus callosum links the brain's two hemispheres. It facilitates the flow of communication between the two parties. Each hemisphere of the brain is responsible for controlling a distinct side of the body. Weakness or paralysis of the left arm or leg could result from a tumor on the right side of the brain. The left hemisphere of the brain is responsible for thinking, mathematics, and speaking and writing. The right hemisphere oversaw imagination, three-dimensional ability, creativity, and musical aptitude. The left hemisphere dominates hand use and language in around 92% of individuals [4].

1.2Various Lobes of the Brain

As seen in the image below, it described the several lobes of the human brain. Each lobe has a specific purpose. The fissures in the hemisphere represent the brain's lobes. Hemispheric structure is composed of the frontal, temporal, parietal, and occipital lobes. The lobes of the brain show connections between the left and right hemispheres. The frontal lobe controls an individual's personality, behaviors, and emotions. It is responsible for a vital choice.

Planning and problem-solving are the subsequent crucial duties. The frontal lobe of the human brain was responsible for both speech and writing. The frontal lobes of our brain were responsible for human movement, as well as intellect and attentiveness. It is also important to highlight the parietal lobe, which assists with language and word comprehension, as well as the identification of signals from the senses of sight, hearing, touch, temperature, and 3-D and visual perception. Color vision is handled by the occipital lobe (color, light, movement). The temporal lobe of the brain is responsible for memory, sequencing, and Organization

1.3 Brain Tumor & its Types

Brain tumors affect millions of people on every continent. Li Hong Juang and Ming Ni Wub (2010) provide a speedy and accurate approach for detecting the location and size of a tumor
in a short amount of time using clustering. Figure 1.3 depicts a brain tumor, which is the accumulation of abnormal cells in brain tissue. Because our bodies contain so many cells, they are vulnerable to aging-induced cell death. To replace these dead cells, the body must produce new ones. When the body needs an abnormal or tumorous cell, it develops rapidly and does not die as normal cells do. Because of this technique, the tumor cells will continue to multiply. As demonstrated in Figure 3, gliomas and astrocytic tumors afflict adults more often than children. Meningeal tumor is a prevalent kind of tumor in adults.
The treatment of a patient with a tumor is determined by the tumor's size, location, and degree of metastasis to adjacent tissues and cells. The age of the patient, as well as the kind and grade of the tumor, are other important factors in cancer treatment. Patients diagnosed with a tumor at an early stage should expect to live longer.

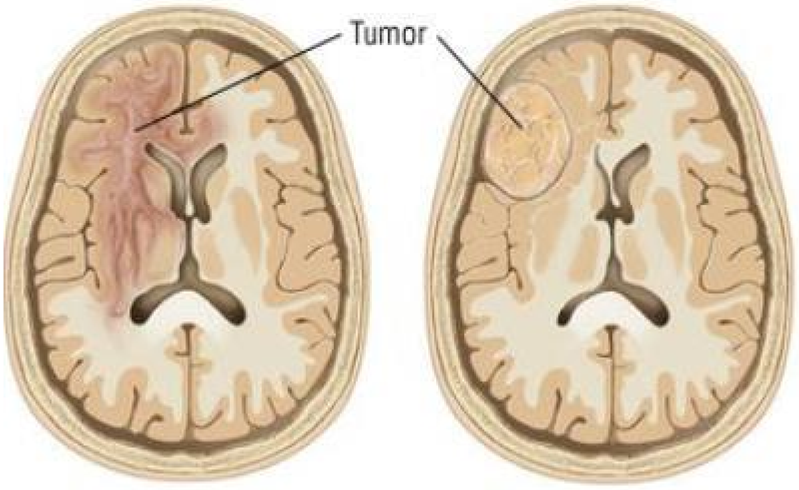


Figure1.3: Brain tumor example [6].

There are two basic forms of brain tumor, the primary and secondary varieties. A primary brain tumor is characterized by the proliferation of aberrant cells. As the formation of cells starts in the brain, the illness cannot spread across the body. A noncancerous brain tumor is referred to as a benign tumor. The growth of the tumor is slow, but it may be eliminated surgically.

When this lesion pushes on a specific region of the brain, it will create significant issues. The location of the tumor will affect the patient's life expectancy.

The most common kind of tumor, according to 2015 research by Madhukumar and Santhiyakumari, is a secondary brain tumor or a malignant brain tumor. It is contaminated with malignant cells.

This type of cancer can spread to the brain after affecting any section of the body. Skin, lungs, breasts, and kidneys are just a few of the organs that may be affected by cancer.

Regardless matter whether a tumor is benign or malignant, it poses a danger to life. Tumors are not benign because the brain lacks sufficient space to tolerate their growth. This kind of tumor damages the brain tissue. CSF (CSF) circulates around and nourishes the brain. Due to the blockage of CSF induced by some tumors, intracranial pressure increases. Edema is a sign of some types of brain tumors.

1.4 Steps of the Brain Tumor

Tumors are classified according to the apparent normalcy or abnormality of their cells. Using this tumor grade, the physician will be able to develop a more effective treatment plan.

Based on cell proliferation and rate of metastasis, tumors are categorized.

Step a: The tumor cells in this group seem to be normal cells, and their development is likewise gradual. Providing the correct therapy for a tumor-infected patient extends their life expectancy.

Step b: There is a problem with the cell's look and growth rate. The local tissue is affected by the tumor cells. This tumor is very harmful and shortens life expectancy.

Step c: The appearance of the cells is aberrant, and the cells are aggressively expanding into the brain tissue that is immediately in front of them.

Step d: In a short time later, the aberrant cell development spreads to other cells. For example, pesticides, industrial solvents, and other chemicals that we are exposed to on a regular basis are the primary causes of brain tumors. Neurofibromatosis is one example of a hereditary condition that may manifest as a brain tumor, as described by Bjoern et al.

Tumor cells in the brain destroy normal brain cells and raise intracranial pressure, both of which occur naturally when the brain is afflicted. The location of the tumor, the kind of tumor, and the tumor's size are all indicators of a brain tumor.

The following are signs of a brain tumor: morning headaches, seizures, giddiness, hesitancy, difficulty walking, communication difficulties (trouble finding the proper term, visual anomalies, unusual eye movements, and weakness on one side of the body are all symptoms of this condition. It is possible to get brain pictures utilizing a variety of medical imaging modalities. CT scan and MRI, as illustrated in the following Segmentation of MRI brain images is carried out as part of this study.

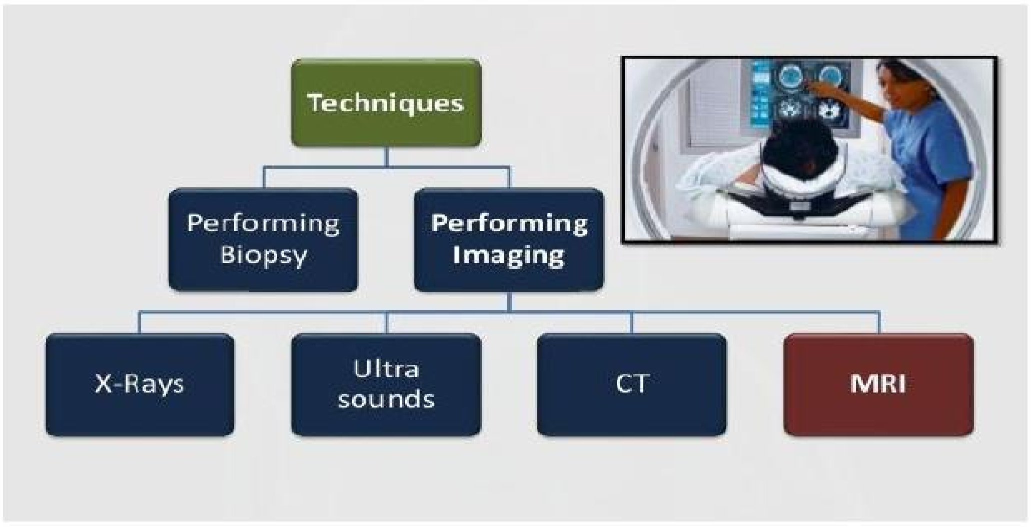


Figure1.4: Brain Tumor Diagnostics Methods [6].

Computed Tomography (CT-Scan)

There are several imaging techniques for the brain, one of which is CT tomography, a name derived from the Greek words tomos and agraphia for the imaging process. Tomos denotes "portion" or "slice," whereas agraphia is a visual depiction. This demonstrates that a CT scan provides a comprehensive picture of the internal organs of the human body. CT utilizes X-rays to reconstruct the internal structure of the human body. The profile of X-ray absorption effects the reconstruction of a picture after CT imaging. X-rays, a kind of electromagnetic radiation, are used to obtain information about the human body. The X-ray absorption profile will be distinct for each tissue. Dense tissue appears white when seen using a CT scanner, whereas soft tissue appears grey. During CT imaging, the air-filled hollow area inside the lungs gives the Lungs a black appearance. CT, unlike X-rays, does not expose the human body to potentially harmful radiation [6]. CT scan is one of the greatest medical imaging techniques, and it may be used to diagnose a broad variety of illnesses in a lot of areas of the body, like the brain and spine. Researchers and physicians must employ brain imaging equipment to identify the root of the problem. Currently, both invasive and noninvasive imaging modalities are available. CT and MRI are two of the most often utilized technologies for brain imaging. During a head injury, one of the most up-to-date technologies was used to capture the brain problem. As a result of strokes and brain tumors, the brain state resembled bleeding, skull damage, and a blood clot. This imaging approach is rapid and accurate, and it provides a plethora of information on the brain's anatomy [9]. Magnetic Resonance Imaging (MRI) The MRI is among the most versatile and dynamic electromagnetic imaging techniques.

Human internal organs are susceptible to electromagnetic radiation.

Using a non-invasive technique known as MRI, it is possible to detect abnormalities in internal body structures such as bone and soft tissue. In contrast to X-ray, MRI imaging technique does not use any potentially harmful radiation. When applied to the human body, radio frequency waves rearrange the hydrogen atoms. By altering the parameters for longitudinal relaxation time (T1) and transverse relaxation time (T2) in the MRI technique, images of different intensity may be generated (T2). Using an MRI to image the brain, chest, abdomen, and pelvis offers information on their anatomy. In addition, it assists in the detection of several diseases. MRI imaging enables the detection of cerebral infections and brain tumor infections at an early stage. MRI is the method most frequently used for diagnosing white matter disorders, other than CT imaging [10].

Digital Image Processing Brain tumor identification and segmentation are made easier using advanced image processing techniques. Image processing foundations must be learned before considering

segmentation [11]. Image - The intensity of an image is calculated at each pixel level by Functioning in two dimensions f (x, y). A picture's current intensity is determined by the magnitude of 'f' at a particular set of coordinates (x, y). Digital image- The term "digital image" describes a picture that consists of discrete and finite amplitudes of pixels. Digital Image Processing (DIP) is the process of using digital computers to enhance previously captured digital photographs.

Pixel - There are many individual pixels that make up the final picture.

An 8-bit picture has 256 possible values, and each pixel has a unique value within that range. The intensity of a pixel in an instant is determined by the quantity of photons that strike it.

In essence, image processing is used for numerous medical or industrial applications, including:

Visualization: It is used to identify objects that are not able to view through naked eyes.

Image Sharpening and restoration: Used to create a better-quality image

Image Retrieval: Used to estimate the region of interest

Pattern Measurement: It is used to categorize an input image into the objects based on certain features.

Image recognition: To find or locate a certain object in a digital image.

In the medical image processing is often used when MRI or CT scans of different body sections are obtained. To enhance the quality of the photographs, several processes such as Image Enhancement have been performed on the recorded data. An MRI picture may be segmented such that the lesion (tumor affected region) can be identified. Image processing methods may also be used to obtain secure information from faraway places [12].

1.5 Brain Stroke & its Types

A stroke is a life-threatening condition that happens when part of your brain doesn't have enough blood flow. This most commonly happens because of a blocked artery or bleeding in your brain. Without a steady supply of blood, the brain cells in that area start to die from a lack of oxygen.

IMPORTANT: A stroke is a life-threatening emergency condition where every second counts. If you or someone with you has symptoms of a stroke, **IMMEDIATELY** call 911 (or your local emergency services number). The quicker stroke is treated, the more likely you'll recover without disability.

To recognize the warning signs of a stroke, remember to think **BE FAST**:

- **B.** Be watchful for a sudden loss of balance.
- **E.** Look out for sudden loss of vision in one or both eyes. Are they experiencing double vision?
- **F.** Ask the person to smile. Look for a droop on one or both sides of their face, which is a sign of muscle weakness or paralysis.
- **A.** A person having a stroke often has muscle weakness on one side. Ask them to raise their arms. If they have one-sided weakness (and didn't have it before), one arm will stay higher while the other will sag and drop downward.
- **S.** Strokes often cause a person to lose their ability to speak. They might slur their speech or have trouble choosing the right words.
- **T.** Time is critical, so don't wait to get help! If possible, look at your watch or a clock and remember when symptoms start. Telling a healthcare provider when symptoms started can help the provider know what treatment options are best for you.

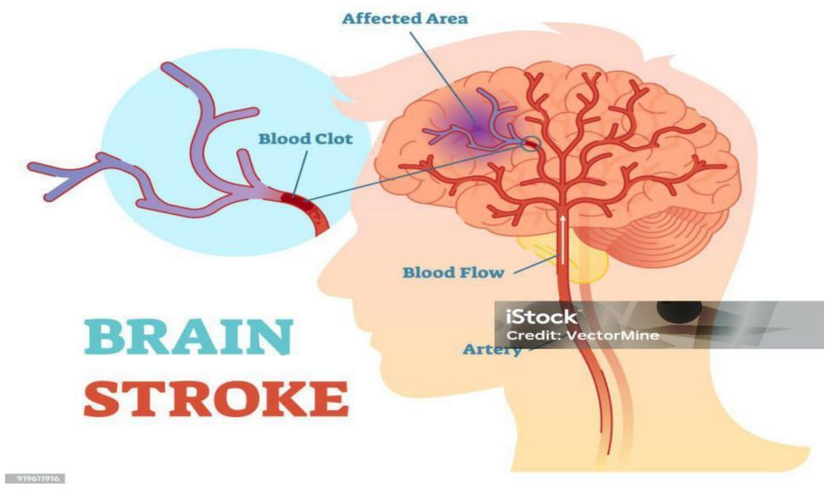


Figure1.5: Human stroke Parts [7].

Who does it affect?

Anybody can have a stroke, from children to adults, but there are some people who have a greater risk than others. Strokes are more common later in life (about two-thirds of strokes happen in people over age 65).

There are also certain medical conditions that increase the risk of stroke, including high blood pressure (hypertension), high cholesterol (hyperlipidemia), Type 2 diabetes, and people who have a history of stroke, heart attack or irregular heart rhythms like atrial fibrillation.

How common is a stroke? Strokes are very common. Worldwide, strokes rank second among the top causes of death. In the United States, stroke is the fifth cause of death. Strokes are also a leading cause of disability worldwide.

How does a stroke affect my body?

Strokes are to your brain what a heart attack is to your heart. When you have a stroke, part of your brain loses its blood supply, which keeps that brain area from getting oxygen. Without oxygen, the affected brain cells become oxygen-starved and stop working properly.

If your brain cells go too long without oxygen, they'll die. If enough brain cells in an area die, the damage becomes permanent, and you may lose the abilities that area once controlled. However, restoring blood flow may prevent that kind of damage or at least limit how severe it is. That's why time is critical in treating a stroke.

Types of brain stroke

Learn the various types of stroke and related treatments. Proper care can save lives and improve quality of life.

1. Ischemic Stroke (Clots)

Occurs when a blood vessel supplying blood to the brain is obstructed. It accounts for 87% of all strokes.

Ischemic strokes usually happen in one of the following ways:

- Formation of a clot in your brain (thrombosis).
- A fragment of a clot that formed elsewhere in your body that breaks free and travels through your blood vessels until it gets stuck in your brain (embolism).
- Small vessel blockage (lacunar stroke), which can happen when you have long-term, untreated high blood pressure (hypertension), high cholesterol (hyperlipidemia) or high blood sugar (Type 2 diabetes).
- Unknown reasons (these are cryptogenic strokes; the word "cryptogenic" means "hidden origin").

2. Hemorrhagic stroke

These happen when bleeding in your brain damages nearby cells.
Causes and risk factors for hemorrhagic stroke
The common factors are:

- You are over age 65
- Have high cholesterol, high blood pressure, or diabetes that isn't under control
- Are obese
- Have had a stroke in the past
- Have a family history of strokes
- Smoke
- Eat unhealthy foods
- Don't exercise
- Have been Injured
- Have a bleeding disorder
- Use cocaine
- Have abnormal blood vessels (AVMs)
- Have an aneurysm (a weak area in a blood vessel that breaks open)

- There are two types of hemorrhagic stroke. Which kind you have is based on where the bleeding happens.
- Subarachnoid hemorrhage means it happened in the area between your brain and skull.
- Intracerebral hemorrhage is bleeding inside the brain.

3.Transient Ischemic Attack or Mini-Stroke

Ischemic strokes also include something called a "mini stroke" or a TIA (transient ischemic attack). This is a temporary blockage in blood flow to your brain. The symptoms usually last for just a few minutes or may go away in 24 hours.

Symptoms of TIA

The symptoms may be like an ischemic stroke. You might have:

- Numbness/weakness on one side of your body
- Confusion
- Dizziness or loss of balance

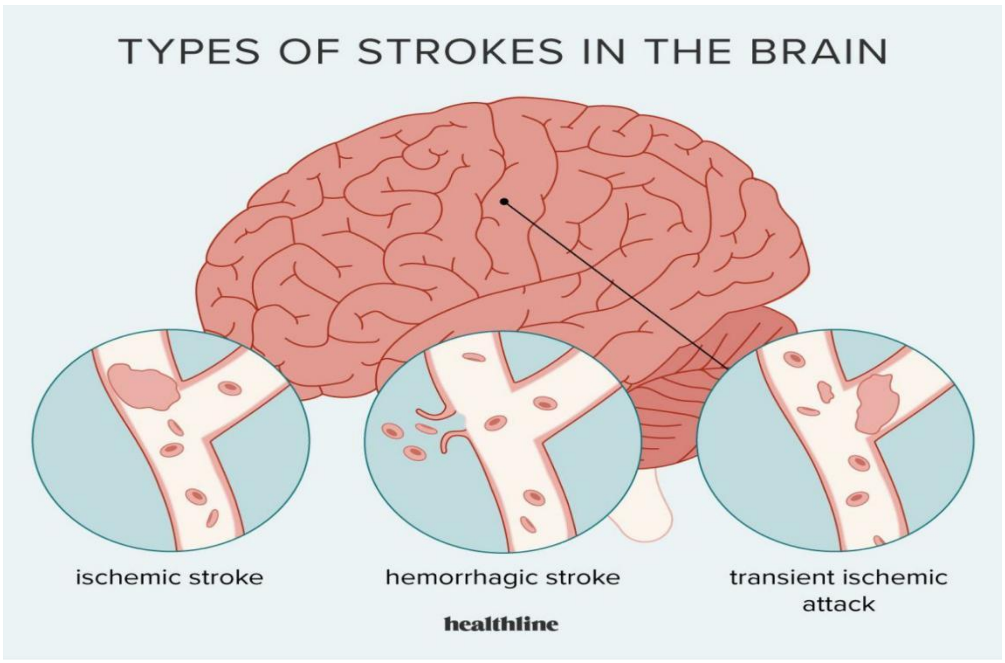


Figure1.6: Human stroke Part [6].

- Trouble talking or understanding

- Problems with your vision
- Severe headaches

1.6 Steps of the Brain Stroke

A stroke on the left side of the brain affects the right side of the body and you may experience some of the following:

• Speech and language problems

• Inability to read, write and learn new information

• Impaired ability to do math or to organize, reason and analyze things A stroke on the right side of the brain affects the left side of the body and you may experience some of following:

• Problems with depth perception or directions, such as up or down, and front and back

• Inability to be creative, such as painting a picture, or to appreciate art and music

• Failure to recognize the emotion in someone's voice

Stroke symptoms

The loss of blood flow to the brain damages tissues within the brain. Symptoms of a stroke show up in the body parts controlled by the damaged areas of the brain.

The sooner a person having a stroke gets care, the better their outcome is likely to be. For this reason, it's helpful to know the signs of a stroke so you can act quickly. Stroke symptoms can include:

- paralysis
- numbness or weakness in the arm, face, and leg, especially on one side of the body
- trouble speaking or understanding others
- slurred speech
- confusion, disorientation, or lack of responsiveness
- sudden behavioral changes, especially increased agitation

- vision problems, such as trouble seeing in one or both eyes with vision blackened or blurred, or double vision
- trouble walking
- loss of balance or coordination
- dizziness
- severe, sudden headache with an unknown cause
- seizures
- nausea or vomiting

A stroke requires immediate medical attention. If you think you or someone else is having a stroke, call 911 or local emergency services right away. Prompt treatment is key to preventing the following outcomes:

- brain damage
- long-term disability
- death
- It's better to be overly cautious when dealing with a stroke, so don't be afraid to get emergency medical help if you think you recognize the signs of a stroke.

What causes ischemic strokes?

Ischemic strokes usually involve certain processes. These are:

- Formation of a clot in your brain (thrombosis).
- A fragment of a clot that formed elsewhere in your body that breaks free and travels through your blood vessels until it gets stuck in your brain (embolism).
- Small vessel blockage (lacunar stroke).
- Unknown reasons (these are cryptogenic strokes; the word "cryptogenic" means "hidden origin").

Blood clots and other forms of ischemia can happen for many reasons, such as:

- Atherosclerosis.

- Clotting disorders.
- Atrial fibrillation (especially when it happens due to sleep apnea).
- Heart defects (atrial septal defect or ventricular septal defect).
- Microvascular ischemic disease (which can block smaller blood vessels in your brain).
- Fat emboli, which are clusters of fat particles circulating in your blood that get stuck in blood vessels in your brain.
- Infected tissue that gets into your bloodstream and travels to your brain, where it gets stuck and blocks a blood vessel (this is a major complication of sepsis, a deadly overreaction of your immune system to an infection spreading in your body).

Therefore, we work to help the patient in early detection of the disease through mri.

Where MRI is a vital tool for measuring acute stroke and has been used to visualize changes in activation patterns during stroke recovery. There is an emerging interest in using MRI to monitor the structural substrates of spontaneous recovery and neurorestorative treatment of stroke.

1.7 Problem Statement

a. Brain tumor lesions can be detected using image data than just symptoms.

b. It is a tool that will take MRI images as input and predict tumor type possibilities using deep learning.

c. Social Relevance of the project Accurate and faster detection of the brain tumor through MRI.

d. Helping the doctors, patient or radiologist using modern computational technology. Improved & efficient screening of Brain tumor at early stages can increase the chances of the patient's recovery after timely treatment.

1.8 Scope

Due to its soft tissue contrast and non-invasiveness, MR imaging is a popular option for significant medical imaging in the brain. MR imaging can also be employed to monitor how the size of a brain tumor varies as a result of treatment. It is obvious that a better method for separating tumors is required. Quantifying tumor volumes using MR images is currently impossible as a result of commonly accepted clinical practice procedures.

1.9 Objective

The objectives of this work can be summarized in the set of the following points:

a) The primary goal is to create a framework that will assist clinical specialists in cross confirming their predicted brain tumor type analysis results.

b) Because the current diagnosis process is time-consuming, labor-intensive, and expensive, this deep learning-based tool can detect tumor growth and predict stage.

c) Because this is an automated tool that uses image processing and AI, it reduces the amount of human effort required to predict the existence of tumor type from images.

1.10 Understanding Brain Tumor Risk

People who have had past brain tumors are more likely to get a new one. If you have two or more close relatives who have brain tumors, your risk of developing one is increased. Even though it is very unusual, it is possible for members of the same family to be born with a disease that renders the brain more susceptible to growing a tumor. Around 5% of all brain tumors may have a genetic component.

1.11 Motivation

a) It's obvious that a technique for accurately segmenting tumors would be valuable. Quantitating tumor regions using MR images is currently impossible due to a lack of well-acknowledged clinical practice methods.

b) It is our major goal in this research to detect if there is a Brain tumor or not and classified the tumor into its types if tumor is present, such as Glioma, Meningioma, and Pituitary.

Chapter 2

Literature Review

Literature Review

In order to begin research, it is necessary to write a review. It's critical to have access to old publications on topics like space and application, both of which are associated with the kind of work we do. It's not difficult to determine the real demand, the drawbacks of previous labor, and the weight of work after conducting a thorough examination into the past. The use of reference papers helps students comprehend the work, calculations, models, and previous research used, as well as pointing them in the right direction for future study. Deep learning and machine learning frameworks are reviewed in this section.

2.1 Fundamental Concepts

2.1.1 Data

Fusion is nothing but combination of data from various sources; these sources can be homogeneous or heterogeneous. It provides the result in some sense better than the individual input. Input from various sources gives a more accurate result than depending on only single input in case of various automatic applications [13].

2.1.2 Data Gathering

a) Information fusion is employed to define already processed data.

b) Information fusion is a process used to combine data from multiple sources [heterogeneous or homogeneous] such that, in some way, the outcome is superior to the individual inputs.

c) The input data to the system may be Imperfect, Correlated, and Inconsistent

d) The main objective of Information Fusion:

To improve accuracy in results, support intelligent Decision Making the Benefits of information fusion can be listed as follows:

a) To improve accuracy.

b) To reduce uncertainty.

c) To provide better analysis.

d) To improve decision making.

e) To improve reliability.

f) It’s more precise.

g) It’s faster.

h) To improve the performance of the task.

i) To reduce or to filter input information through its integration.

j) It covers up the limitation of single sensor giving output based on multiple sensors
information incorporation.

k) To reduce imperfection.

l) To reduce incompleteness.

m) To reduce ambiguity in results.

n) To reduce conflicts.

2.1.3 Sensor Fusion

a) Sensor fusion technology is widely used in automatic applications. Collection of separate data from various sensors is done in sensor fusion. This technology leads to accurate and reliable results than depending on the not that accurate result from single sensor. This technology uses microcontrollers for this work.

b) Sensor technology is evolving day by day and similarly trying to act like an actual human being. Because of precision and ability to perform well this technology is used in various applications such as climate monitoring, healthcare, automotive systems, smart mobile devices, industrial control and oil exploration.

c) In this technology every part is considered and instead of calculating individual part; it combines all the parts.

d) This is related to the internet of objects; this sensor fusion is highly used, and it shows real wonders of this technology with Internet of Things.

2.1.4 Relationship

Relationship among the Fusion Terms: Data fusion, Information fusion and Sensor fusion.

a) Sensor fusion is subset of Information Fusion.

b) Information fusion is subset of Data fusion.

2.1.5 Machine Learning

a) Throughout the 1980s and 1990s, the use of information-driven Artificial Intelligence (AI) tactics in many building fields, such as voice and picture analysis, grew in popularity. As indexes of structured knowledge grow, using robots to extract relevant instances from the data may be a game-changer.

b) An inquiry requires a large quantity of information, and it's difficult to examine all the potential linkages and connections with distinct connections of information sources that might produce many outcomes and can be investigated in detail to get desired results.

c) Artificial intelligence (AI) and Machine learning (ML) are ideal to unlock the full potential of massive data. Artificial intelligence (AI) is fully integrated within machine learning to help systems to learn and grow without needing to be directly programmed.

d) Machine learning occurs when a computer is given a large amount of data, and the computer uses that data to figure things out for itself. Artificial intelligence may benefit from it because of the improved method it gives for its construction. Automated decision-making is a key feature of machine learning, which is why it is so commonly utilized [14].

2.1.6 Supervised Learning

a) Data must be prepared for the train machine. This data is known as training data. The technique in which the training data is learned from supervised algorithms and give some results depending on them was nominated as supervised learning algorithm.

b) Supervised learning is of two types; these types are regression and classification.
c) Sometimes there is a risk of receiving some unrelated input from training data. This type of input causes imperfect results. To handle this issue is challenging.
d) Once the machine learns from supervised learning it keeps the record and whenever some data output is needed it gives that output using such previous work.
e) Kind of data set need to use in machine learning must be decided first to consider best supervised learning working.

2.1.7 Unsupervised Learning

a) There is no requirement to supervise the model in unsupervised machine learning.
b) If there are unknown patterns present in data then these unknown patterns are recognized by unsupervised machine learning.
c) There are two types of unsupervised learning first is clustering and second is association.
d) In this unsupervised learning technique four types of clustering methods are used. These clustering methods are exclusive, agglomerative, overlapping, probabilistic.
e) There are large databases that are involved these databases have data objects in it. These data objects need association between themselves. To do this association rules are required.
f) Whenever there is data sorting in process it is not possible to get precise information is the drawback of this unsupervised learning.

2.1.8 Reinforcement Learning

a) Sequence is very important in reinforcement learning. Simply output is dependent on current input and for the input of next stage; output of previous stage is required.
b) Dependency is there in reinforcement learning so there is need of assigning label to each and every dependent decision.

c) This technique broadens its area of training by taking help from the environment instead of just depending on the data set provided so that it can increase the chances of better output.

2.1.9 Statistics and Probability in Machine Learning

a) Machine learning is mainly involved in building intelligent applications. To make intelligent application statistics and probability algorithms are used in terms of providing better vision by learning from data.
b) To predict upcoming events probability is used and to analyze the events in past statistics is used.

2.1.10 Decision Making

Decision making is very important as it provides one conclusive stage to the system. Decision tree has branches and leaves. Branches denote observations and leaves represent conclusions, so the overall decision tree is used as a predictive model. Data mining, statistics and machine learning needs accurate decision making so it uses this approach of predictive modelling. Every leaf of the tree has a class label, and branches connect the leaves in the tree. These branches tend to reach to the class label of the leaves. Decision analysis needs decision tree for decision making. Decision tree illustrates the decisions which make it easy for decision making [15].

2.1.11 Deep Learning

In machine learning and artificial intelligence (AI), a technique known as "deep learning" mimics human learning processes. As part of data science, deep learning is a critical feature.
Data scientists benefit greatly from deep learning. must gather, analyze, and interpret massive, large amounts of data; it speeds up and simplifies the process. Predictive analytics may be made more efficient by using deep learning. While

most machine learning algorithms are linear, deep learning algorithms are stacked in a hierarchical framework of increasing complexity and abstraction [16].

Chapter 3

System Requirements

SOFTWARE REQUIREMENTS

3.1 Introduction and Purpose

The documents to be searched, as well as the software needed to do the search, are all listed in this document. In order to create the system, a list of the different software frameworks is necessary. The following topics are covered in the document.

3.2 Work Scope

Today's ailments are on the rise as a result of our hectic and stressful lifestyles. All age groups are susceptible to illness, thus early detection is critical. Pneumonia, lung cancer and brain tumor diseases by using symptoms or reports. Most people throughout the world still lack access to the necessary instruments for the early identification of these illnesses. One of the most hazardous diseases among both genders, early diagnosis and treatment are critical to the health of those sick. For the future, we're going to invent strong system which can accurately work on a large medical data set with a number of lung cancer and brain tumor diseases.

3.2.1 User Classes and Characteristics

In this system contain mainly 2 classes the User and the Admin, the characteristic of this classes can be listed as follows:

a) Characteristics of User

i. Login/Register

ii. Upload images

iii. Get prediction results

iv. Logout

b) Characteristics of Admin

i. Admin login with authentication

ii. Stage evaluation

iii. Prediction result

3.2.2 Assumptions and Dependencies

Assumptions:

Pathological tests, such as needle biopsy specimens and analysis by experienced pathologists, are necessary for accurate identification and classification of skin diseases because they require a human judgement of many factors, experiences, and a system of decision support is eligible in this case. Doctors are unable to use the current system since it lacks a diagnostic system [51].

Dependencies: Both structured data and unstructured data are used by the algorithm (from hospitals). To the best of our knowledge, there is an absence of studies on any data type in the field of big data analytics for medicine [52]. Providing a correct diagnosis, it lowers the mortality rate caused by misdiagnosis.

3.3 Functional Requirements

3.3.1 First feature of The System (Functional Requirement)

a) Images of lung cancer can be used to identify the condition.

b) Image-based detection of brain tumors.

3.3.2 System Feature (Functional Requirement)

a) Database

b) Database Requirements SQLite3

3.4 Connectivity to the Outside World Requirements

3.4.1 User Interfaces

Python: Python interface is currently being worked on. Many algorithms, as well as the functions that make them up or support them, may be found. With an STL container-friendly template interface, Open CV is developed entirely in C++.

Image Processing: Images may be read and written. Images and their characteristics are detected. In a picture, the detection of forms like circles, rectangles, and coins may be accomplished. Recognizing text in photographs is an emerging technology (e.g., taking a look at the plates).

Changing the color and resolution of the picture.

3.4.2 Hardware Interfaces

• In order to operate our project, we needed a hardware system such as an Intel I3 CPU with 4 GB of RAM and a 20GB hard drive.

• Standard keyboard, mouse, and LED monitor are also required**.**

3.4.3 Software Interfaces

Microsoft can be used as the operating system platform for the system. Some GUI tools are also employed by the system. PyCharm and higher is required as a python platform to execute this application. An SQLite3 database is required for data storage.

3.4.4 Communication Interfaces

a) Diseases detection System

b) User disease image data set

c) Pre-processing unit

d) Feature vector generation using CNN

e) Classified results in the form of predictions

f) Open-CV for image processing

3.5 Non-functional Requirements

3.5.1 Performance Requirements

a) Performance: Performance of our system is fast as compared to other system and response time is quick.

b) Availability: Availability of data is also a requirement for performing any operations.

c) Maintainability: In this system we can maintain data of user's images.

d) Security: In this system user information is stored in the form of images, so our system is secure.
e) Usability: This system is very useful as an assistive tool for the medical sector.

3.5.2 Safety Requirements

This study is carried out to examine the economic impact which the system is going to have on organization. The funding available to the company for system research and development is limited. Costs must be for clear reasons. As a result, the system was developed within the budget, which was accomplished because almost all the technologies used were free.

3.5.3 Security Requirements

The main thing in our system is, we must provide end to end security for user and provider signs by using proper authentication login credentials. Users have been given the right to upload images and only view the results. The system is fully secure as well as eco- friendly. We implemented this system by considering security aspects, so we divide our system into four different modules for achieving integrity.

3.5.4 Software Quality Attributes

a) Capacity: The project capacity according to data is very small.
b) Availability: Proposed system is available on python application.
c) Reliability: System is reliable for multi-disease prediction.
d) Security: When users log-in to the system that time user's mail ID and Password accurately match.

3.6 System Requirements

3.6.1 Requirements of Database

3.6.2 Software Requirements (Platform Choice)

The requirements for the software (platform choice) can be summarized as follows:

a) Operating System: Windows

b) Front End: Python3x

c) Database: SQLite3

d) IDE: PyCharm

3.6.3 Hardware Requirements

The requirements for the hardware can be summarized as follows:

a) Processor: I3

b) Speed: 1.1 GHz

c) RAM: 2 GB (min)

d) Hard Disk: 20GB

e) Floppy Drive: 1.44MB

f) Camera: System camera

g) Key Board: Standard windows keyboard

h) Mouse: Two or three button mouse

i) Monitor: SVGA

3.7 The SDLC Model

Waterfall Model: The waterfall model is a consecutive model that is used in the software development processes, where the process slowly descending spotted to be phase of requirements, gathering, analyses, design of systems, implementation, testing, Deploying and finally maintenance [53].

a) Requirement analysis: Here requirements are gathered means which kind of dataset is required. Then what are the functional requirements of system. The document is prepared, and then use cases are designed. In our system, we gather all information of Admin and user and the functionality of each module.

b) System Design: at this stage, the hardware (HW) and software (SW) required to design A system are decided. It uses the above-mentioned hardware and software requirements.

We design the Admin and user modules. Design the according to the functionality of each module.

c) Implementation: In this stage, the system is developed module wise. This system consist of mainly 2 modules is (1. Admin 2. User).

d) Testing: In this stage, all developed software is installed, and it is tested in different ways against the system requirements. In this stage, we check whether all these modules are working properly or not with proper authentication. Disease prediction proper or not as well as stage prediction proper or not.

e) Deployment: in this deployment stage we deployed the new functionality of each module like Crop Yields Predictions, Crop Suggestion to the farmer; Dynamic Assistance and Online E-Mart modules. We deploy all systems with proper functions.

f) Maintenance: According to software's new version and their use, they need to be updated some predefined machine learning libraries need to be used. This system is easy to maintain.

Chapter 4
SYSTEM DESIGN

System Design

4.1 Architectural Diagram

In this chapter of the thesis the proposed system design will be clearly presented and discussed. The architecture of the proposed system design is illustrated in [Figure 4.1].

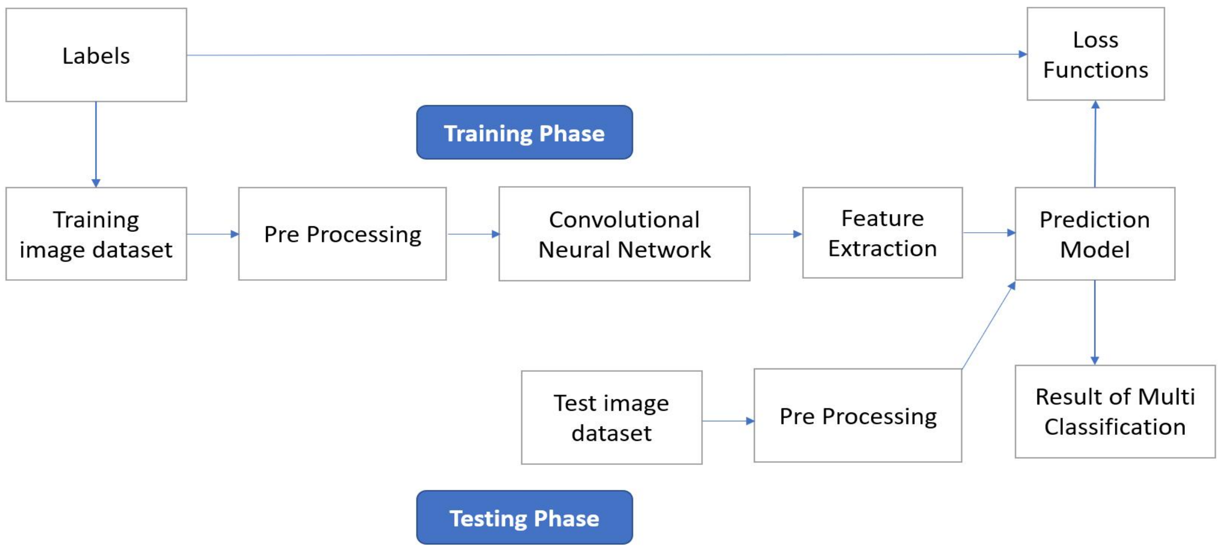


Figure 4.1: The System Architecture.

4.2 Processing Steps

The first step is pre-processing which is the stage converts the picture according to the requirements of the following level. It cleans up the picture by removing noise and other blemishes and sharpening the image's edges. RGB to grey conversion and reshaping are carried out through this stage also, it has a noise-removal feature that uses a median filter. Modern MRI scans have very low noise arrival probabilities. It's possible that it'll show up as a result of the heat effect. In this study, the primary focus is on identifying and classifying tumor cells. Noise reduction is required for the full system, though.

4.2.1 K-Means Clustering for Segmentation

The steps fir the k-mean clustering is illustrated as below:

a) K is the number of clusters.

b) Pick the k cluster centers at random

c) Calculate the cluster's average or center.

d) Determine the separation between the center of each cluster and each pixel.

e) If the distance is near the center, go to the closest cluster to the center.

f) Move on to the next cluster if you can't find what you need.

g) Calculate the center of gravity again.

h) It's time to repeat the operation till the center doesn't shift.

4.2.2 Approximate Reasoning

To estimate the tumor size, the linearization approach is used in the first stage of deductive reasoning. In other words, it's a picture with just two possible colors: black or white (0 or 1). After this, define the tumor's stage and forecast its prognosis from the supplied tumor area.

4.3 UML Diagrams

4.3.1 Use Case Diagram

The use case diagram, as shown in Figure 4.2, can be used to summarize the data about your system's users (also referred to as actors) and their interaction with it. A useful use case diagram can help your team represent and discuss [54]:

a) Representing the objectives of user-system interactions

b) Defining and structuring a system's functional needs

c) Defining a system's context and needs

d) The basic flow of events in a use case is modelled.

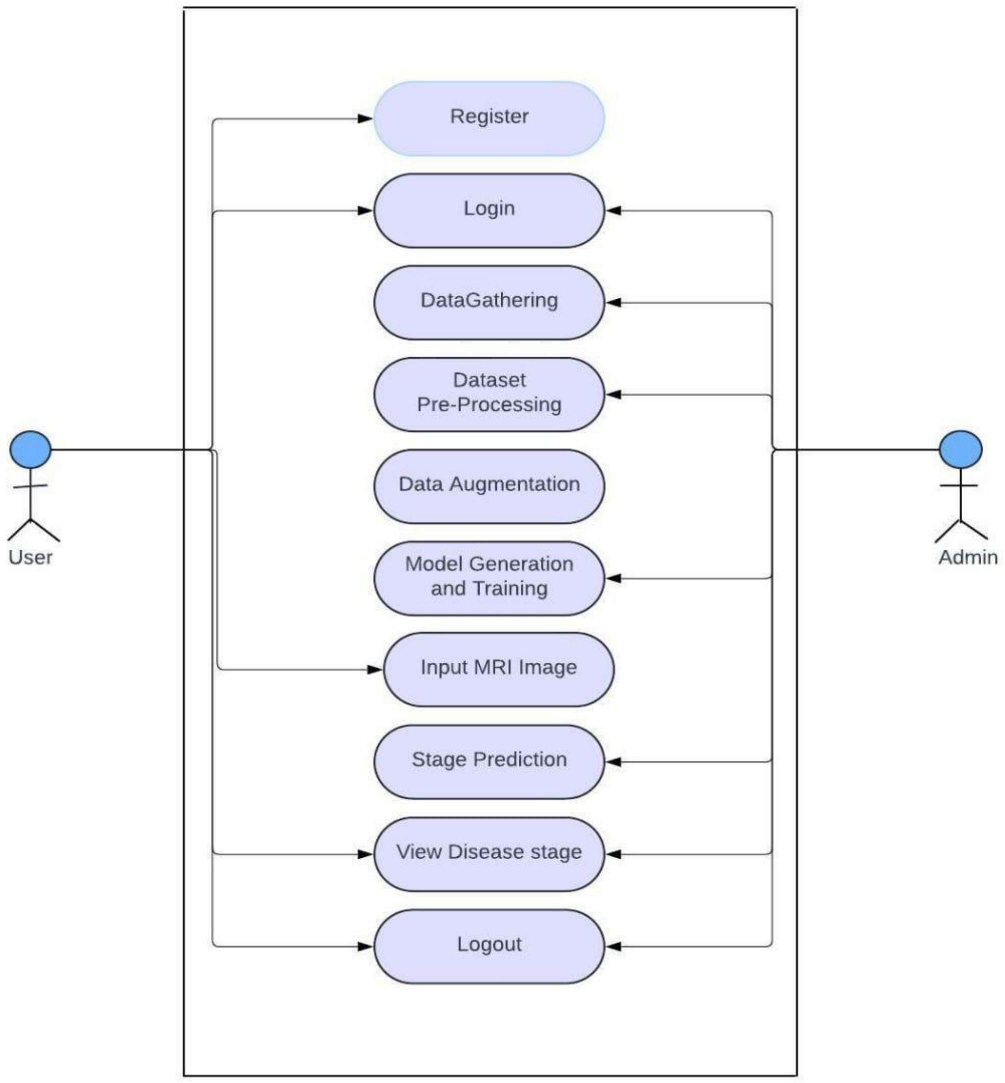


Figure 4.2: Use case diagram [54].

The connection between a system and a user is graphically depicted in a use case diagram. So many user types and their interactions with a system could be depicted In a use case diagram. Internal and external factors are taken
into account while drawing up use case diagrams. The majority of these criteria are design specifications. Usage cases are produced, and actors are identified when a system's capability is examined in order to prepare them for use. The following are some possible uses for use case diagrams:

a) Gathers system requirements.

b) To acquire a different perspective on a system.

c) Identify the external and internal elements that affect the system.

d) Display the performers' interplay.

4.3.2 Sequence Diagram

A sequence diagram is an interaction diagram that shows how and in what order the processes interact. This is the construction of message sequence diagrams, sometimes called event diagrams, event scenarios, and sequence diagrams.

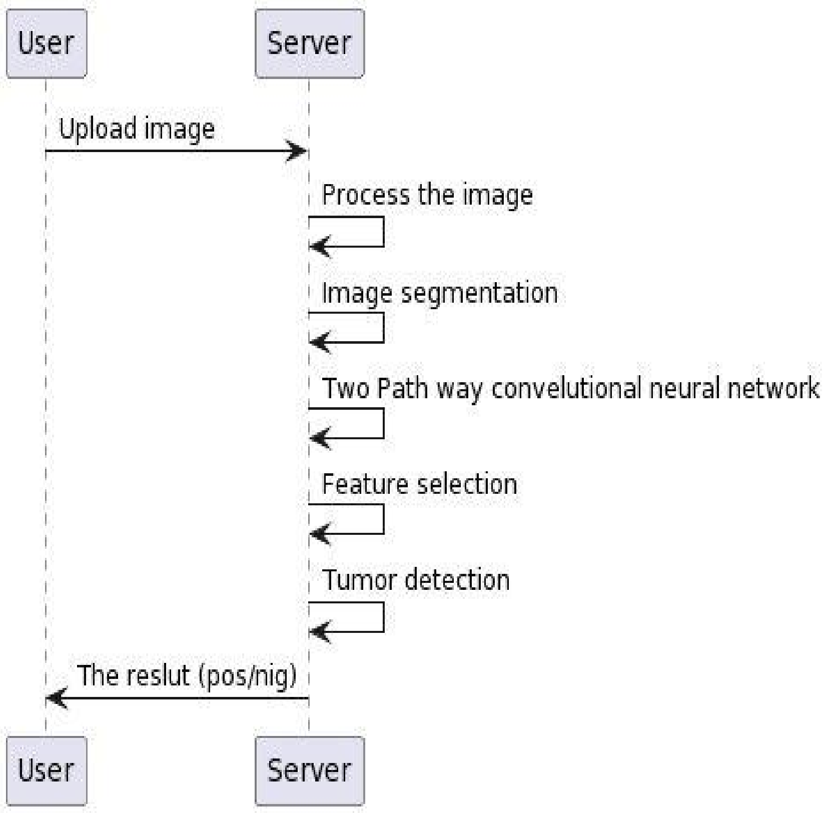


Figure 4.3: Sequence diagram.

4.3.3 Activity Diagram:

Activity diagram is a graphical representation of workflows of stepwise activities and actions with support for choice, iteration and concurrency. An activity diagram shows the overall flow of control.

The most important shape types:

● Rounded rectangles represent activities.

● Diamonds represent decisions.

● Bars represent the start or end of concurrent activities.

● A black circle represents the start of the workflow.

● An encircled circle represents the end of the workflow.

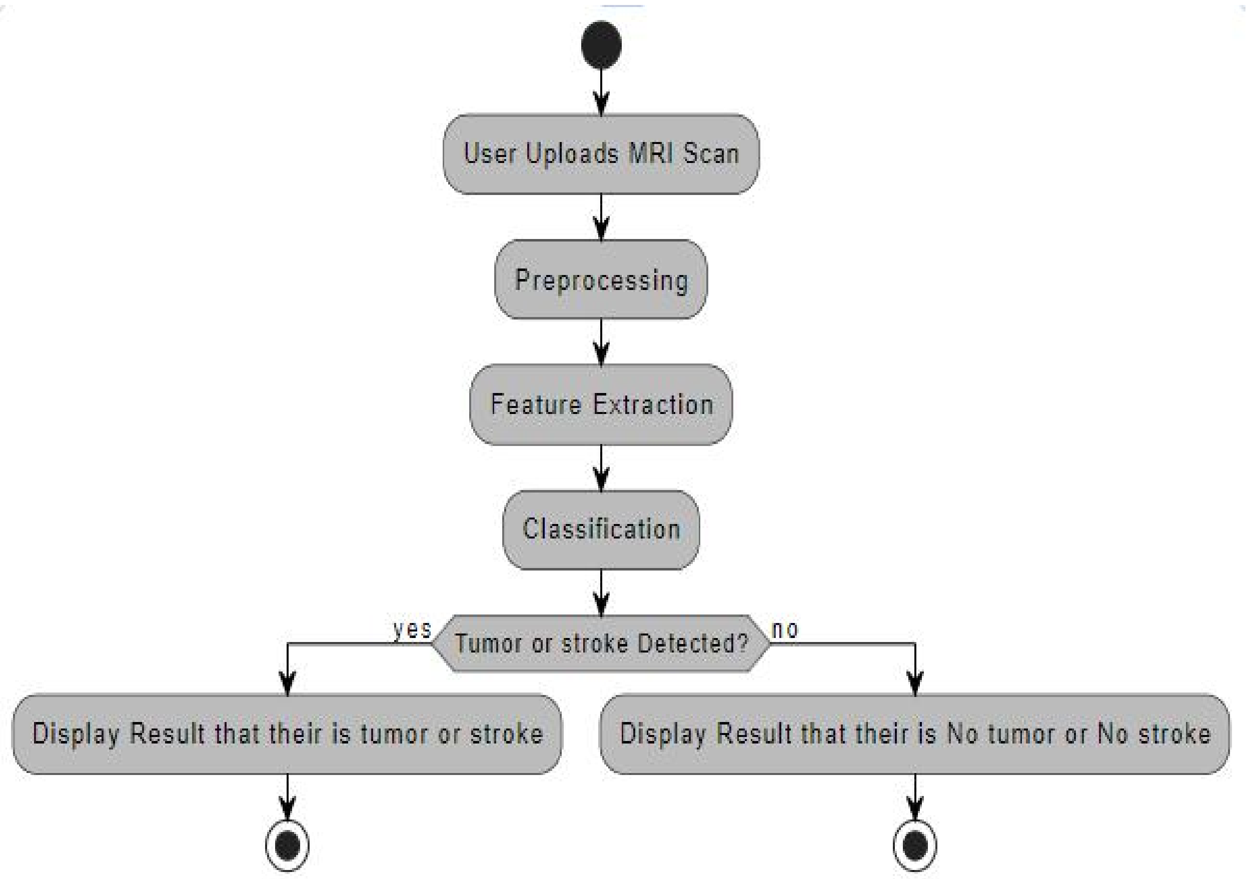

User Uploads MRI Scan
Preprocessing
Feature Extraction
Classification
yes
Tumor or stroke Detected?
no
Display Result that their is tumor or stroke
Display Result that their is No tumor or No stroke

Figure 4.4: Activity Diagram.

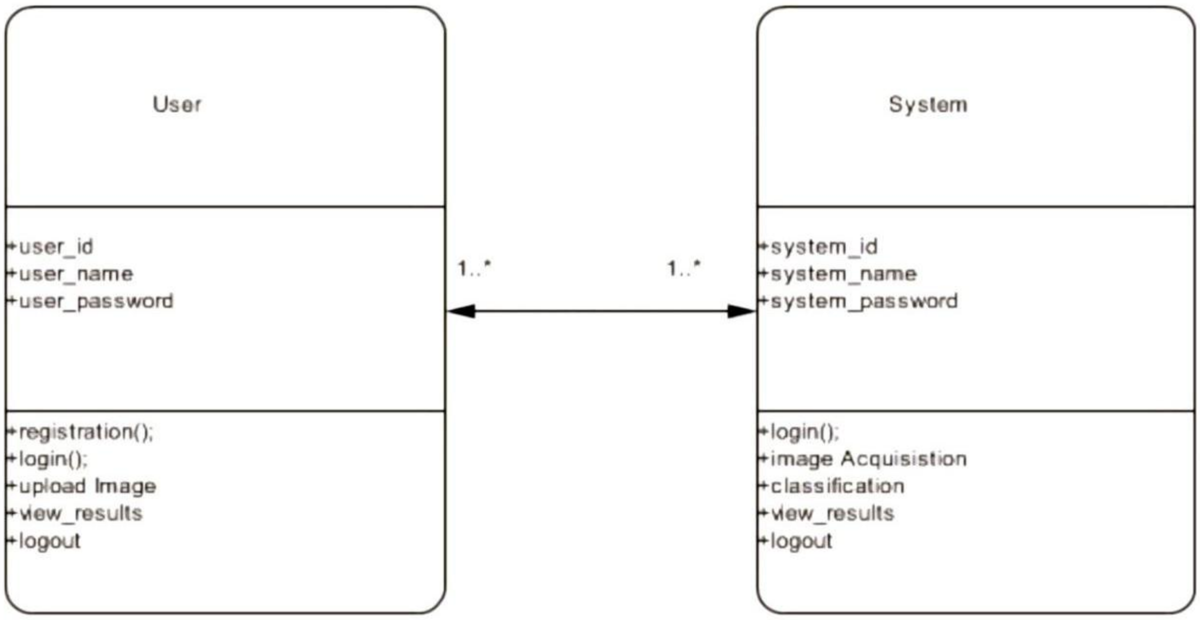
The order of events is shown by arrows that go from the beginning to the finish. As a result, they might be seen as a type of flowchart. It is difficult to express concurrency in flowcharts because of the absence of structures. This can only be resolved for basic

circumstances by using the join and split symbols in activity diagrams; the model's meaning is obscured when they are arbitrarily coupled with choices or loops.
Here we upload the image, whether it is of a brain tumor or stroke. It goes through the process of preprocessing, then feature extraction, then classification. Here, in the case of a brain tumor or stroke, the result appears as the presence of the disease, and in the case of the absence of the disease, the result appears as the absence of the disease.

4.4.1 Class Diagram

Static diagrams, such as the class diagram, are used in textbooks. It's a representation of the application's "static" state. It is possible to create executable code from a class diagram in addition to visualizing, explaining, and documenting many parts of a system. The class diagram is a graphic depiction of a class's attributes, capabilities, and constraints. Class diagrams are frequently used in the design of object-oriented systems as they are the only UML diagrams that may be immediately converted to object-oriented languages. In class diagrams, relationships, classes, and interfaces may all be viewed. Also displayed are restrictions. Another term for it is a structure diagram. The class diagram serves as a visual representation of an application's static view. Figure 4.5 shows a diagram shows how processes interact with each other and in what order they do so. As the name suggests, it's based on a Message Sequence Chart construct. The interactions between items are displayed chronologically in a sequence diagram. We can identify the objects and classes that are involved as well as the order of messages that need to be passed between them in this diagram to achieve the objectives of the scenario. Logical View use case realizations are generally depicted in sequence diagrams. Events and scenarios are sometimes used
interchangeably with sequence diagrams [56].

Figure



4.5: class Diagram.

4.4.2_Use Case Diagram

A use case diagram is a graphical representation of the interaction between actors (users or external systems) and a system to accomplish specific goals. It shows various use cases and different types of users the system has.

Key Elements:

- **Actors:** External entities that interact with the system.
- **Use Cases:** Specific tasks users perform within the system.
- **Relationships:** Connections between actors and use cases.

Benefits:

- Visualizes system functionality
- Identifies and documents requirements
- Communicates system design

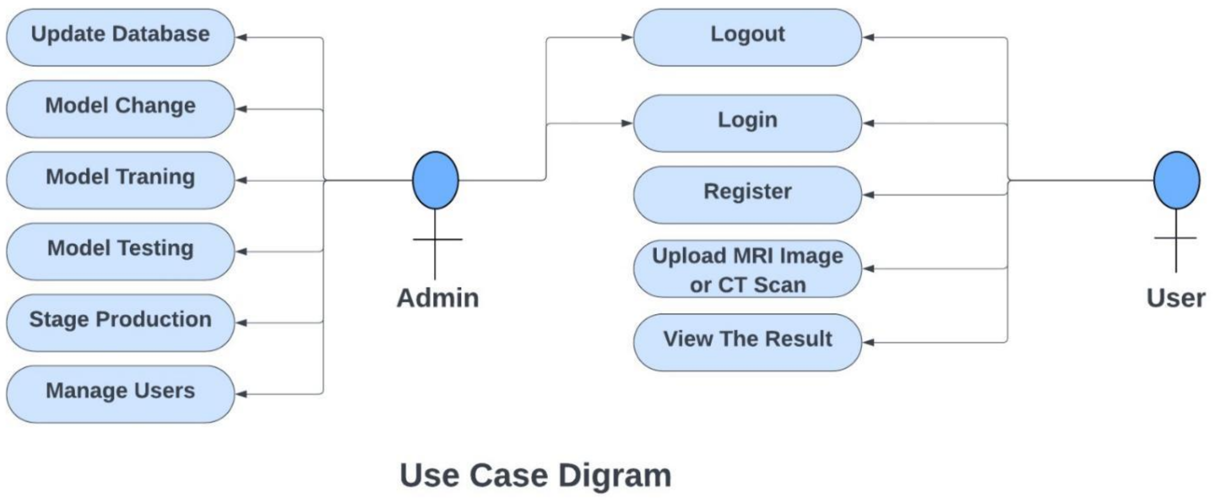


Figure 4.6: use case Diagram.

4.4.2 System Architecture

the system architecture facilitates the analysis of brain MRI or CT scans by employing machine learning techniques to extract meaningful information from medical imaging data. This information can aid in the diagnosis, treatment, and monitoring of various neurological conditions.

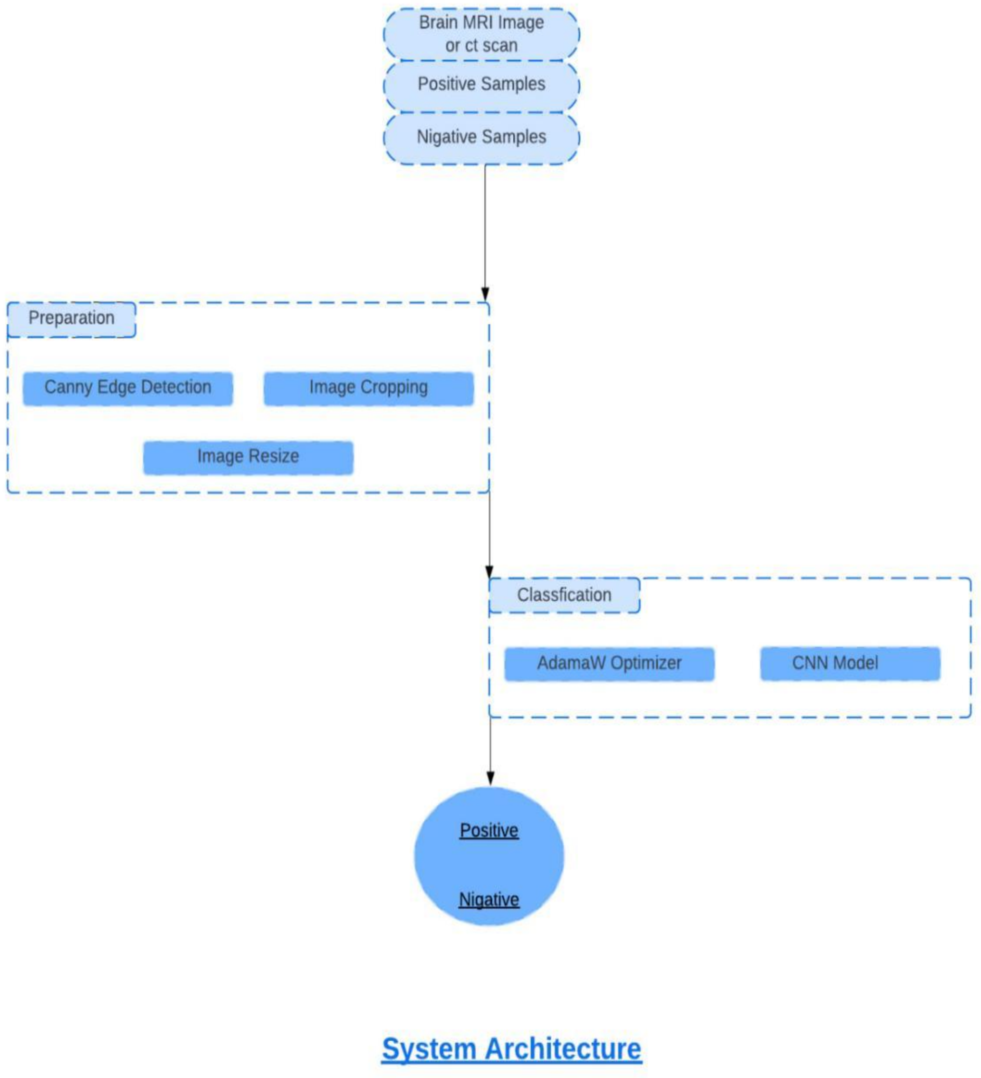


Figure 4.7: System Architecture.

Chapter5

Methodology and Implementation

Methodology

5.1 Algorithmic Approach

Convolutional Neural Networks (CNN):

Convolutional Neural Networks (also known as CNN/ConvNets) are a type of Artificial Neural Network that is known to be extremely powerful with in sector of distinguishing proof as well as image order. Figure 5.1 captures four major operations in Convolutional Neural Networks.

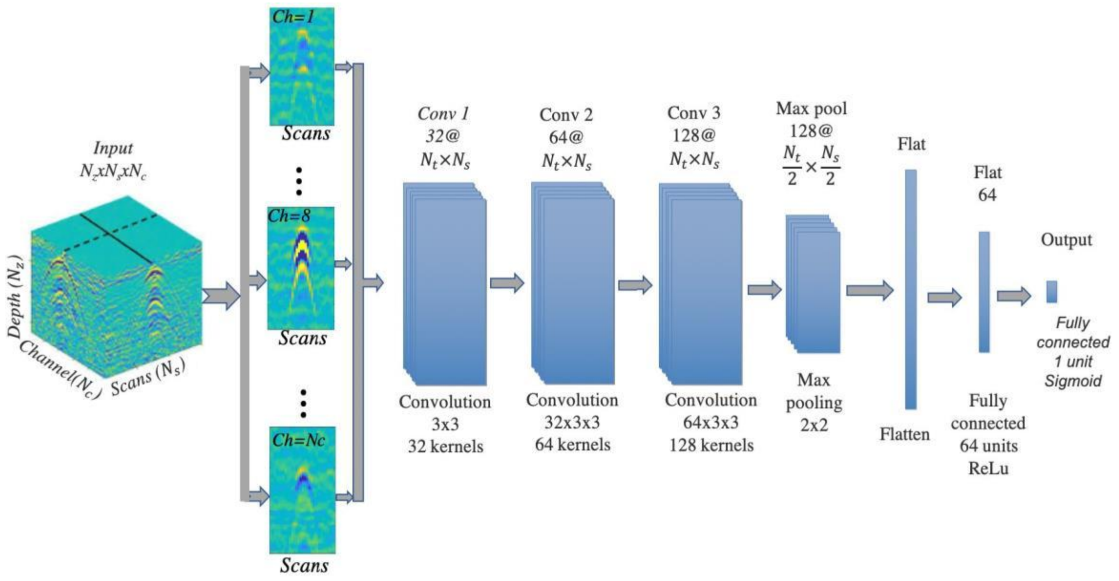


Figure 5.1: CNN Architecture.

(a) Convolution:

If a CNN occurs, the primary use of the Convolution activity is to recognize fitting features from the image, which serves as an ability to contribute to the primary layer. Convolution maintains the spatial interrelationship of the pixels by completing picture highlights with tiny squares of the image. Convolution equation. Every image is viewed as a network of pixels, with each having its own value. The pixel is the smallest unit in this image grid. For better agreement, let us consider a

5 by 5(5*5) framework with only two qualities (for example, 0 or 1). It should be noted that most images are RGB, with pixel values ranging from 0--255, ie 256 pixels.

(b) ReLU

ReLU follows it up on a basic level. Overall, it is a per-pixel activity that has no effect on any of the non-positive advantages of any pixel inside the component map.

(c) Pooling or sub-sampling

Spatial pooling, usually referred to as sub sampling or down sampling, helps keep one of the most crucial pieces of information from the guide while decreasing the elements of every element map. Following pooling, in the long term, our three-dimensional element map is reduced to a one-dimensional component vector.

(d) Fully Connected layer

The output of the convolution & pooling activities produces visible highlights that are removed from the image. These highlights then were used by the Fully Connected layer to categorize the information picture based on the preparation dataset.

❖ Brain Tumor Detection model

5.2 Datasets Details

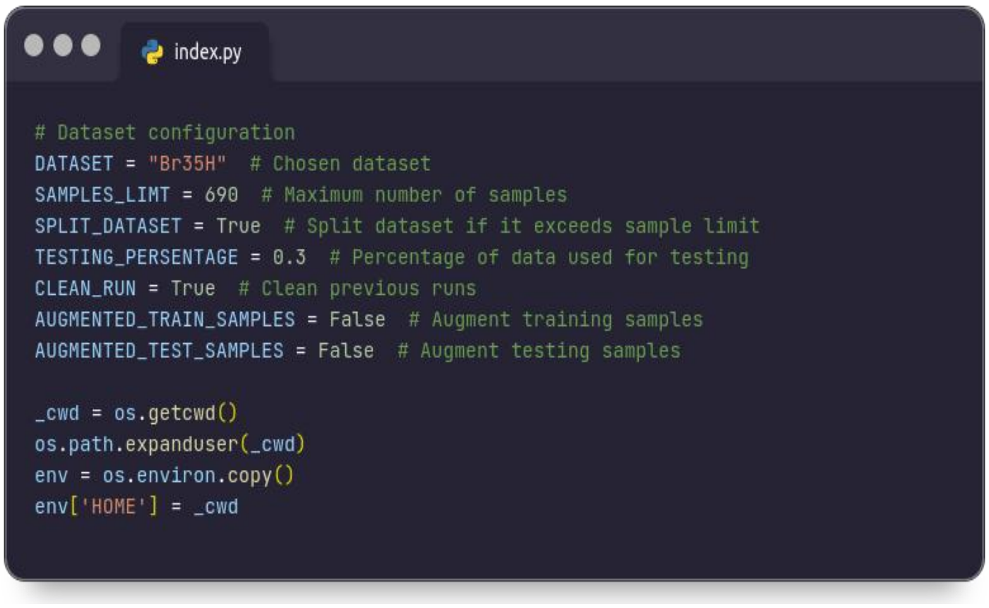
The datasets available for this project are:

1.Br35h: This dataset consists of 1500 negative and 1500 positive samples.

2.Huggingface: This dataset comprises 1312 negative and 4057 positive samples.

5.3. Dataset Configuration Parameters

The training script allows the customization of various dataset parameters through configurable options. These parameters include the choice of the dataset, the limit on the number of samples, the option to split large datasets, the percentage of data used for testing, and whether to clean previous runs and augment samples.



```python
# Dataset configuration
DATASET = "Br35H"  # Chosen dataset
SAMPLES_LIMT = 690  # Maximum number of samples
SPLIT_DATASET = True  # Split dataset if it exceeds sample limit
TESTING_PERSENTAGE = 0.3  # Percentage of data used for testing
CLEAN_RUN = True  # Clean previous runs
AUGMENTED_TRAIN_SAMPLES = False  # Augment training samples
AUGMENTED_TEST_SAMPLES = False  # Augment testing samples

_cwd = os.getcwd()
os.path.expanduser(_cwd)
env = os.environ.copy()
env['HOME'] = _cwd
```

Figure 5.2: Dataset configuration.

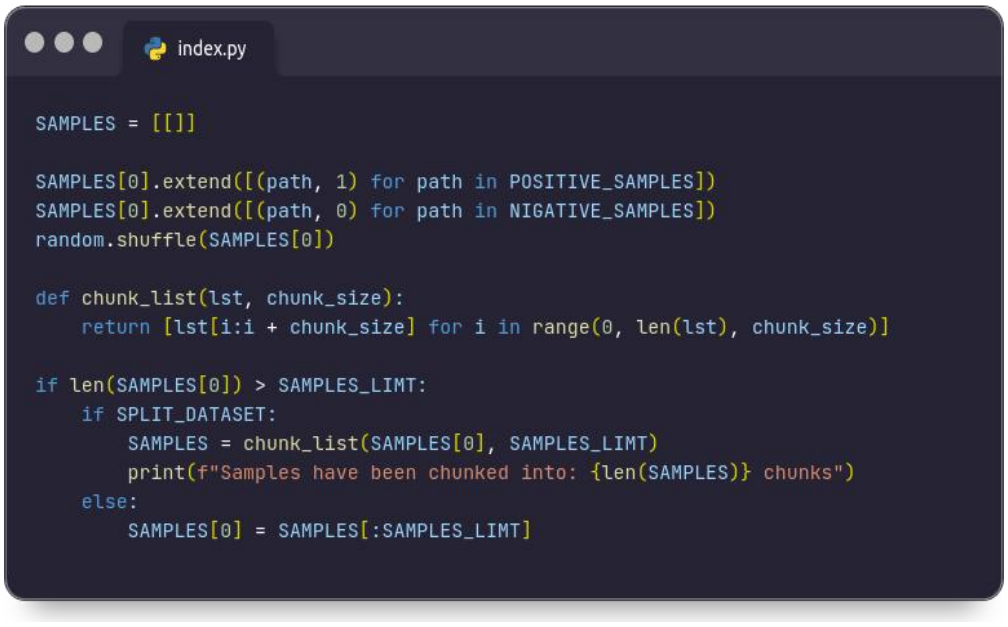
5.4. Dataset Preparation

The dataset preparation process includes downloading the chosen dataset, copying samples, and organizing them into positive and negative categories. Based on the dataset selected (Br35H or Huggingface), appropriate functions are called to fetch

and prepare the data.

5.4.1 Data Shuffling and Splitting

The samples are shuffled, and if the total number of samples exceeds the limit, they are either split into chunks or truncated based on the configuration, to improve the learning performance.



```python
SAMPLES = [[]]

SAMPLES[0].extend([(path, 1) for path in POSITIVE_SAMPLES])
SAMPLES[0].extend([(path, 0) for path in NIGATIVE_SAMPLES])
random.shuffle(SAMPLES[0])

def chunk_list(lst, chunk_size):
    return [lst[i:i + chunk_size] for i in range(0, len(lst), chunk_size)]

if len(SAMPLES[0]) > SAMPLES_LIMT:
    if SPLIT_DATASET:
        SAMPLES = chunk_list(SAMPLES[0], SAMPLES_LIMT)
        print(f"Samples have been chunked into: {len(SAMPLES)} chunks")
    else:
        SAMPLES[0] = SAMPLES[:SAMPLES_LIMT]
```

Figure 5.3: Data Shuffling.

This concludes the data set settings and preparation chapter. The next steps involve utilizing these prepared datasets to train and evaluate our AI model for brain tumor detection.

5.4.2. Edges Cropping

To enhance the accuracy of brain tumor detection, it is essential to focus on the region of interest (ROI), which in this case is the brain itself. We implemented a cropping technique to isolate the brain from the rest of the image by finding the extreme points of the brain contour. This section details the cropping method, which is based on the approach described in [Finding extreme points in contours with OpenCV.

5.4.2.1. Cropping Methodology

1. Convert to Grayscale and Blur:

- The input image is converted to grayscale to simplify the processing.

- A Gaussian blur is applied to the grayscale image to reduce noise and detail, which helps in thresholding.

2. Thresholding:
- A binary threshold is applied to the blurred image, converting it into a binary image where the brain region becomes white (255) and the background becomes black (0).

- Erosion and dilation operations are performed to remove small noise and smooth the object boundaries.

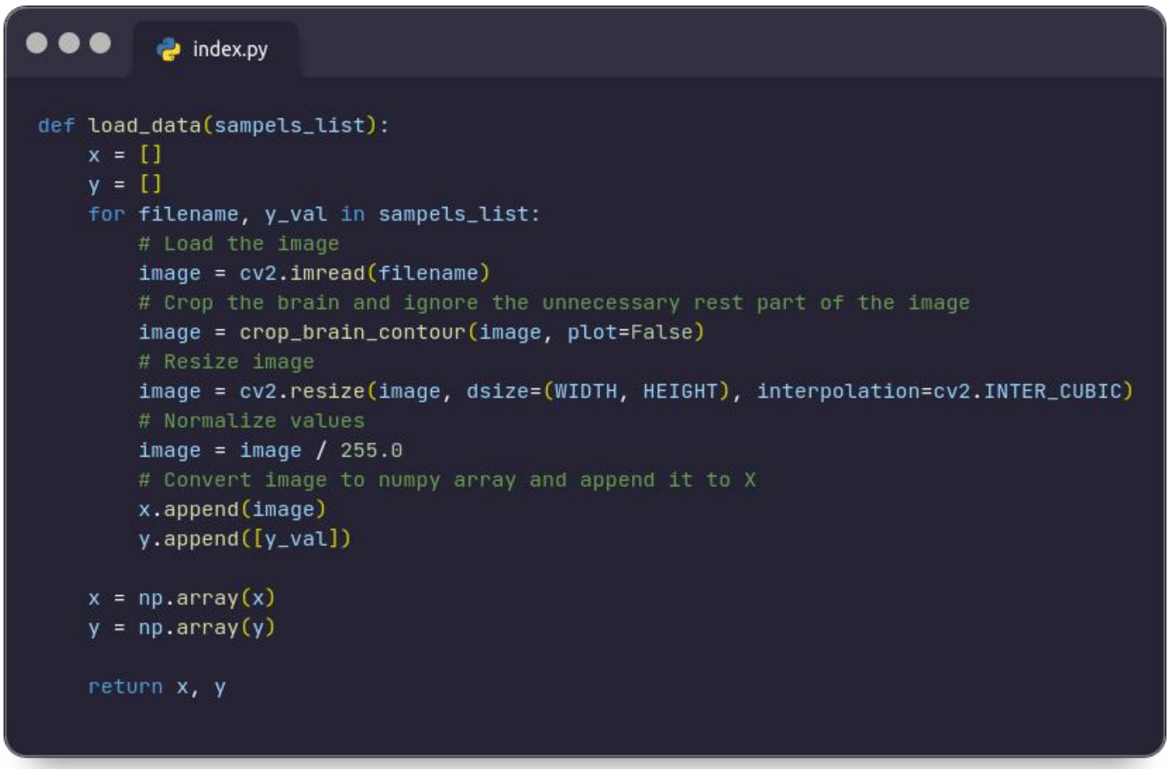
3. Finding Contours: Contours are identified in the binary image, and the largest contour is assumed to be the brain region.

4. Extreme Points Calculation: The extreme left, right, top, and bottom points of the brain contour are identified.

5. Cropping the Image: Using the extreme points, the original image is cropped to extract the brain region.

5.4.3. Load and Resizing

In order to feed the images into our AI model, it is crucial to prepare and resize them to a consistent shape. The `load_data` function takes a list of image file paths and labels, preprocesses the images, and returns the data in a format suitable for training.



```python
def load_data(sampels_list):
    x = []
    y = []
    for filename, y_val in sampels_list:
        # Load the image
        image = cv2.imread(filename)
        # Crop the brain and ignore the unnecessary rest part of the image
        image = crop_brain_contour(image, plot=False)
        # Resize image
        image = cv2.resize(image, dsize=(WIDTH, HEIGHT), interpolation=cv2.INTER_CUBIC)
        # Normalize values
        image = image / 255.0
        # Convert image to numpy array and append it to X
        x.append(image)
        y.append([y_val])

    x = np.array(x)
    y = np.array(y)

    return x, y
```

Figure 5.4: The load_data function.

5.4.3.1. Explanation

1. Loading the Image: Each image is loaded using `cv2.imread`.

2. Cropping the Brain Region: The function `crop_brain_contour` is called to crop the brain region from the image.

3. Resizing the Image: The cropped image is resized to the specified `WIDTH` and `HEIGHT` using cubic interpolation.

4. Normalization: The pixel values of the image are normalized to the range [0, 1] by dividing by 255.0.

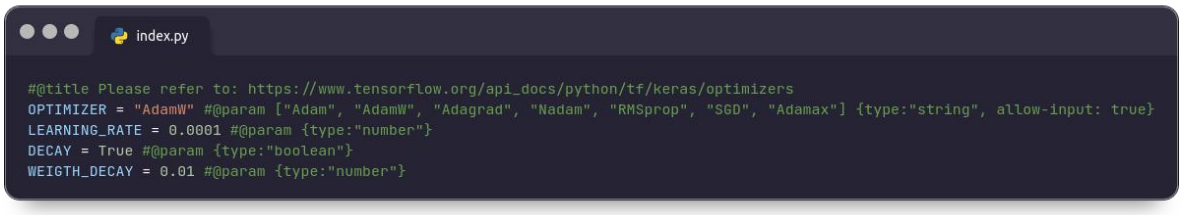
5. Appending to Lists: The processed image and its corresponding label are appended to lists `x` and `y`, respectively.

6. Conversion to NumPy Arrays: The lists `x` and `y` are converted to numpy arrays for efficient processing and handling in machine learning models.

5.5. Model Building

5.5.1. Model parameters

We can select the optimizer and set its parameters through the following code snippet:



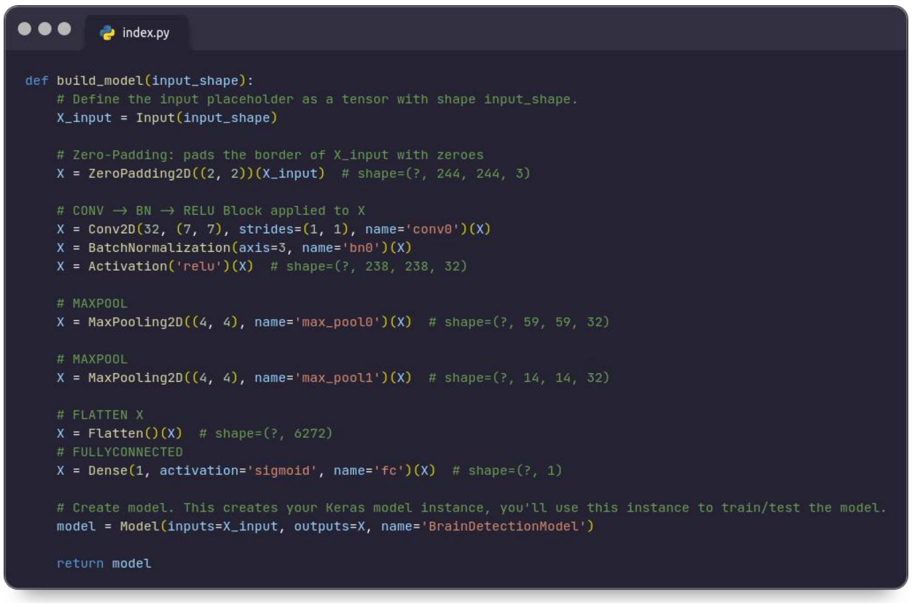
index.py

```python
#@title Please refer to: https://www.tensorflow.org/api_docs/python/tf/keras/optimizers
OPTIMIZER = "AdamW" #@param ["Adam", "AdamW", "Adagrad", "Nadam", "RMSprop", "SGD", "Adamax"] {type:"string", allow-input: true}
LEARNING_RATE = 0.0001 #@param {type:"number"}
DECAY = True #@param {type:"boolean"}
WEIGTH_DECAY = 0.01 #@param {type:"number"}
```

Figure 5.5:The CNN model parameters.

5.5.2. The building function

The `build_model` function defines the architecture of the CNN:



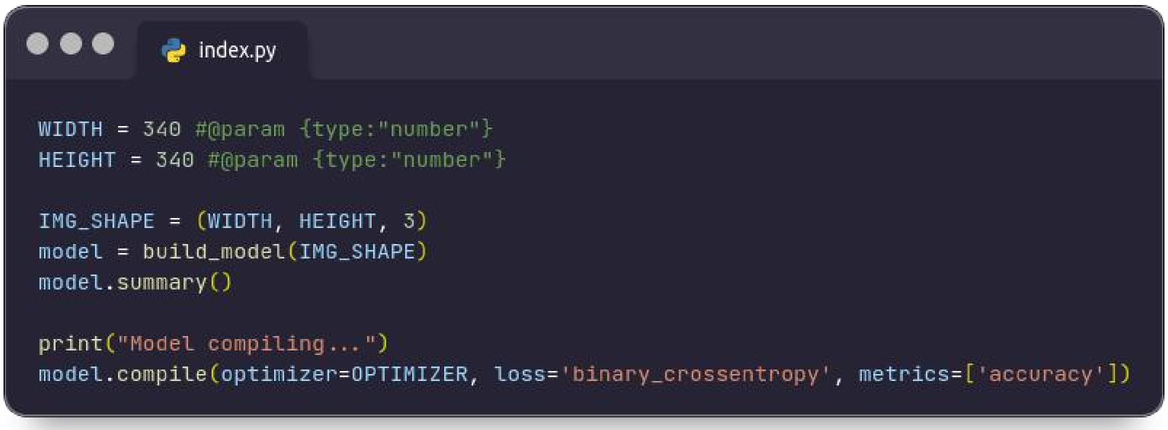
index.py

```python
def build_model(input_shape):
    # Define the input placeholder as a tensor with shape input_shape.
    X_input = Input(input_shape)

    # Zero-Padding: pads the border of X_input with zeroes
    X = ZeroPadding2D((2, 2))(X_input)  # shape=(?, 244, 244, 3)

    # CONV → BN → RELU Block applied to X
    X = Conv2D(32, (7, 7), strides=(1, 1), name='conv0')(X)
    X = BatchNormalization(axis=3, name='bn0')(X)
    X = Activation('relu')(X)  # shape=(?, 238, 238, 32)

    # MAXPOOL
    X = MaxPooling2D((4, 4), name='max_pool0')(X)  # shape=(?, 59, 59, 32)

    # MAXPOOL
    X = MaxPooling2D((4, 4), name='max_pool1')(X)  # shape=(?, 14, 14, 32)

    # FLATTEN X
    X = Flatten()(X)  # shape=(?, 6272)
    # FULLYCONNECTED
    X = Dense(1, activation='sigmoid', name='fc')(X)  # shape=(?, 1)

    # Create model. This creates your Keras model instance, you'll use this instance to train/test the model.
    model = Model(inputs=X_input, outputs=X, name='BrainDetectionModel')

    return model
```

Figure 5.6: The model architecture.

5.5.3. Model Compilation

After building the model, it needs to be compiled with the chosen optimizer, loss function, and metrics:



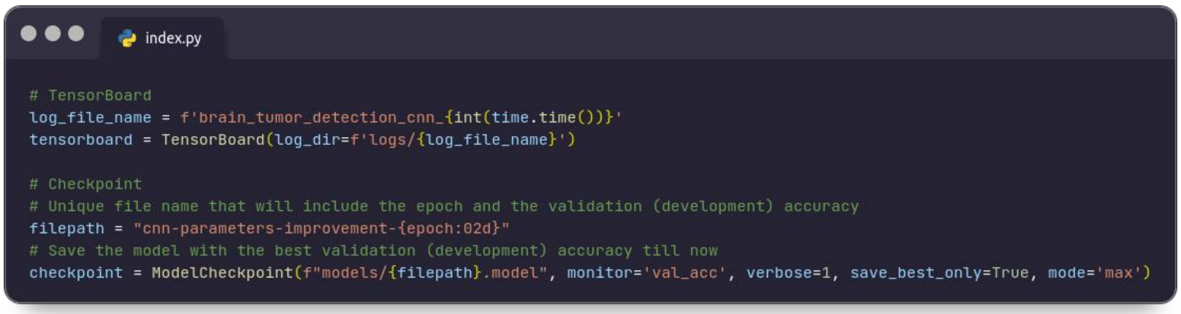
index.py

```python
WIDTH = 340 #@param {type:"number"}
HEIGHT = 340 #@param {type:"number"}

IMG_SHAPE = (WIDTH, HEIGHT, 3)
model = build_model(IMG_SHAPE)
model.summary()

print("Model compiling...")
model.compile(optimizer=OPTIMIZER, loss='binary_crossentropy', metrics=['accuracy'])
```

Figure 5.7: CNN model compilation.

5.5.4. Callbacks

To monitor the training process and save the best model, we use TensorBoard and ModelCheckpoint callbacks:



index.py

```python
# TensorBoard
log_file_name = f'brain_tumor_detection_cnn_{int(time.time())}'
tensorboard = TensorBoard(log_dir=f'logs/{log_file_name}')

# Checkpoint
# Unique file name that will include the epoch and the validation (development) accuracy
filepath = "cnn-parameters-improvement-{epoch:02d}"
# Save the model with the best validation (development) accuracy till now
checkpoint = ModelCheckpoint(f"models/{filepath}.model", monitor='val_acc', verbose=1, save_best_only=True, mode='max')
```

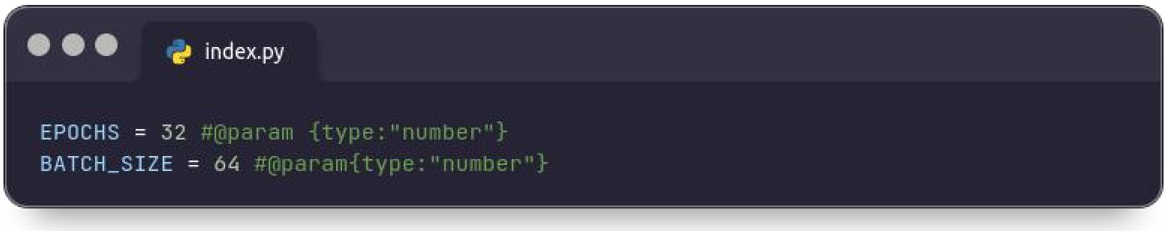
Figure 5.8.:Setup the callbacks.

5.5.5. Summary

This chapter covers the crucial steps of building and compiling the CNN model for brain tumor detection. By allowing us to select the optimizer and its parameters, we ensure flexibility in training. Additionally, callbacks are set up to monitor and save the best-performing model during training. This setup ensures that the model is efficiently trained and can be evaluated using the validation dataset.

5.6. Training

5.6.1. Training Configuration

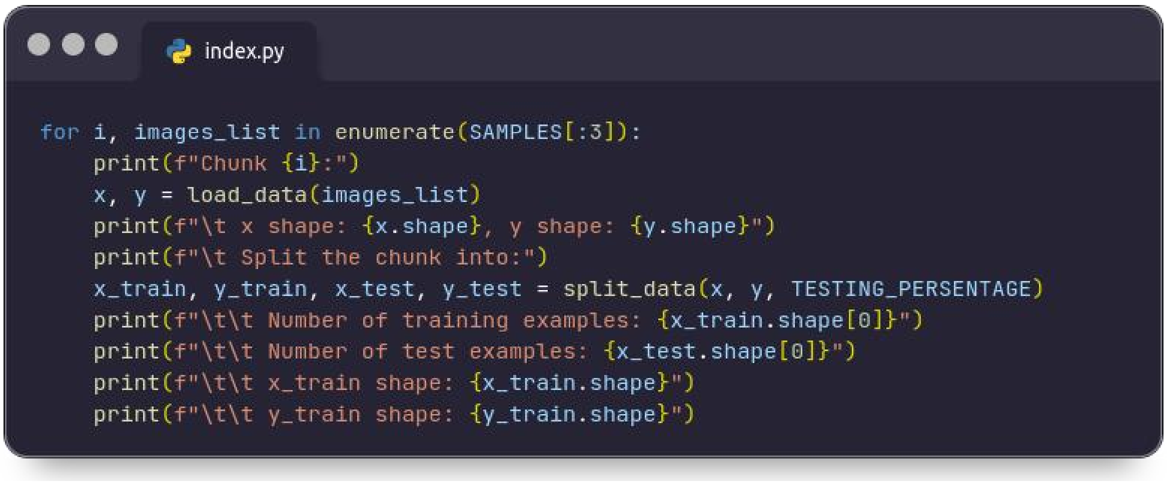

index.py

```
EPOCHS = 32 #@param {type:"number"}
BATCH_SIZE = 64 #@param{type:"number"}
```

Figure 5.9: Training parameters.

5.6.2. Loading and Splitting Data

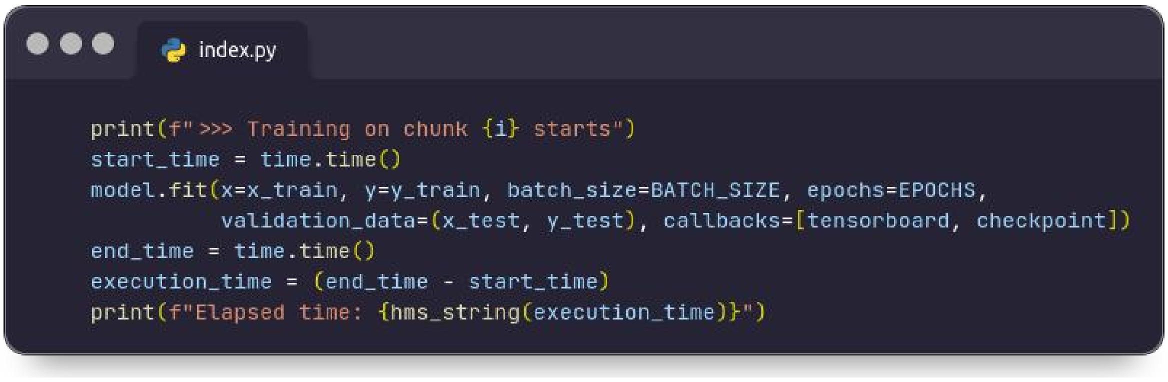
We load and split the data for training and testing. This is done for each chunk of data to handle large datasets efficiently:


index.py

```
for i, images_list in enumerate(SAMPLES[:3]):
    print(f"Chunk {i}:")
    x, y = load_data(images_list)
    print(f"\t x shape: {x.shape}, y shape: {y.shape}")
    print(f"\t Split the chunk into:")
    x_train, y_train, x_test, y_test = split_data(x, y, TESTING_PERSENTAGE)
    print(f"\t\t Number of training examples: {x_train.shape[0]}")
    print(f"\t\t Number of test examples: {x_test.shape[0]}")
    print(f"\t\t x_train shape: {x_train.shape}")
    print(f"\t\t y_train shape: {y_train.shape}")
```

Figure 5.10: Loading and Splitting the data.

5.6.3. Training Loop


index.py

```python
print(f">>> Training on chunk {i} starts")
start_time = time.time()
model.fit(x=x_train, y=y_train, batch_size=BATCH_SIZE, epochs=EPOCHS,
          validation_data=(x_test, y_test), callbacks=[tensorboard, checkpoint])
end_time = time.time()
execution_time = (end_time - start_time)
print(f"Elapsed time: {hms_string(execution_time)}")
```

Figure 5.11: The Training Loop.

5.6.4. Summary

In this chapter, we detailed the process of training the CNN model for brain tumor detection. We covered data loading, splitting, model training, and performance monitoring. By using callbacks, we ensured that the best model is saved, and the training process is logged for further analysis. The model's performance can be evaluated on the test set, and additional metrics such as the F1 score can be computed to assess its effectiveness.

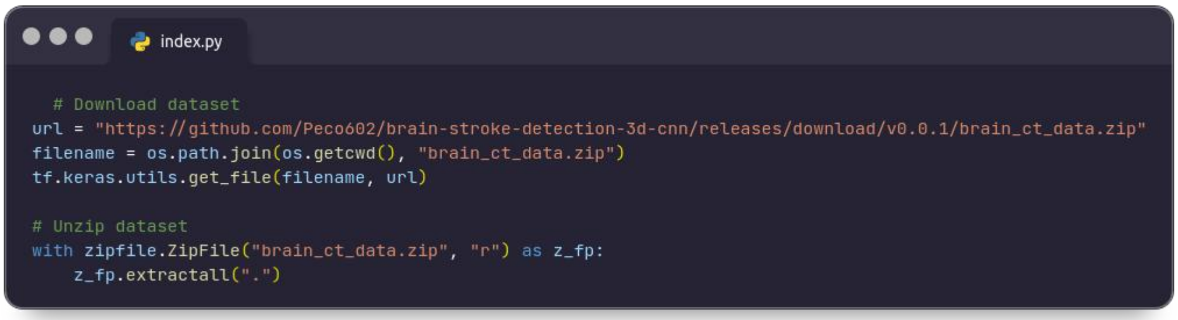
❖ Brain Stroke detection model

5.7.1. Introduction

In this chapter, we present the initial setup for our Brain Stroke Detection model, detailing the steps taken to prepare and process the dataset. This setup includes downloading the dataset, unzipping the files, and organizing the data into appropriate directories for normal and stroke CT scans. Additionally, we introduce sorting configurations to ensure the CT scan slices are correctly ordered for analysis.

5.7.2. Dataset Acquisition and Preparation

To begin our model, we downloaded a dataset of brain CT scans from a public repository. This dataset is essential for training and evaluating our brain stroke detection model. The following code snippet illustrates the process of downloading and extracting the dataset:



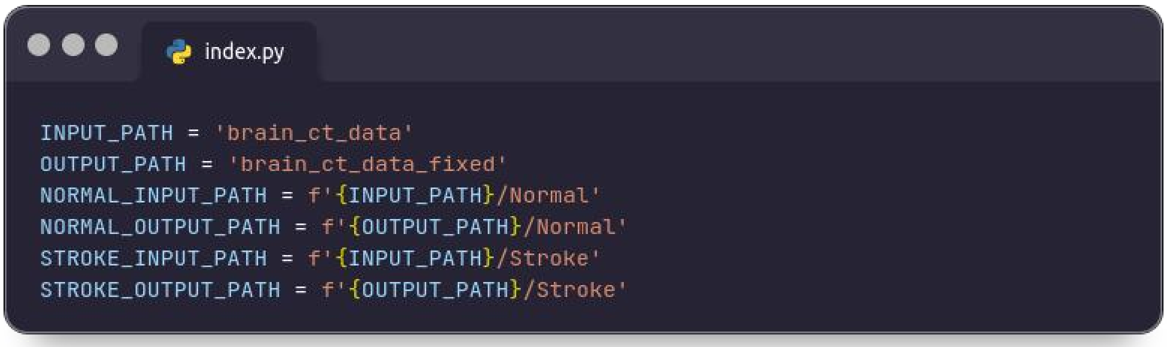
```
  # Download dataset
url = "https://github.com/Peco602/brain-stroke-detection-3d-cnn/releases/download/v0.0.1/brain_ct_data.zip"
filename = os.path.join(os.getcwd(), "brain_ct_data.zip")
tf.keras.utils.get_file(filename, url)

# Unzip dataset
with zipfile.ZipFile("brain_ct_data.zip", "r") as z_fp:
    z_fp.extractall(".")
```

Figure 5.12: Dataset Acquisition and Preparation.

This script utilizes TensorFlow's get_file method to download the dataset from a specified URL. The dataset is then extracted into the current working directory using Python's `zipfile` module.

5.7.3. Directory Structure

Post extraction, the dataset is organized into separate directories for normal and stroke cases. The directory paths are defined as follows:

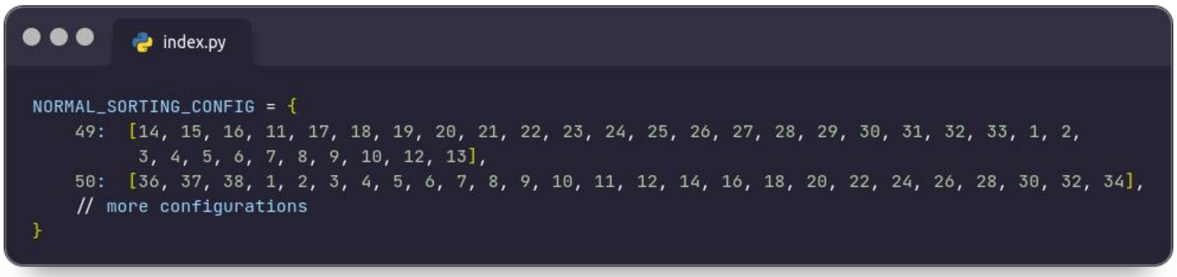

index.py

```
INPUT_PATH = 'brain_ct_data'
OUTPUT_PATH = 'brain_ct_data_fixed'
NORMAL_INPUT_PATH = f'{INPUT_PATH}/Normal'
NORMAL_OUTPUT_PATH = f'{OUTPUT_PATH}/Normal'
STROKE_INPUT_PATH = f'{INPUT_PATH}/Stroke'
STROKE_OUTPUT_PATH = f'{OUTPUT_PATH}/Stroke'
```

Figure 5.13: Data Directory Structure.

5.7.4. Sorting Configuration

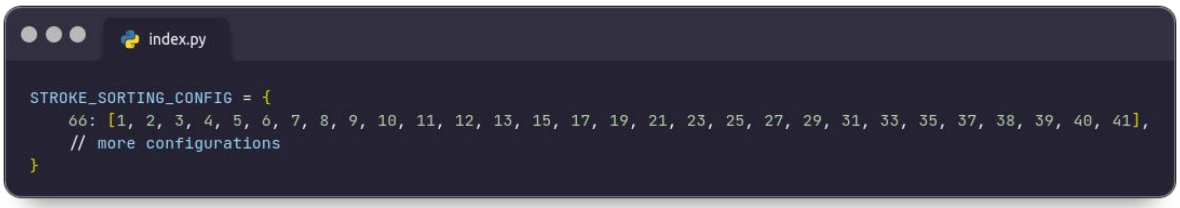
Properly sorting the CT scan slices is crucial for accurate model training. The slices need to be ordered correctly to preserve the spatial information of the scans. We have implemented specific sorting configurations for both normal and stroke datasets. Below is an example of the sorting configuration for normal cases:


index.py

```
NORMAL_SORTING_CONFIG = {
    49:  [14, 15, 16, 11, 17, 18, 19, 20, 21, 22, 23, 24, 25, 26, 27, 28, 29, 30, 31, 32, 33, 1, 2,
          3, 4, 5, 6, 7, 8, 9, 10, 12, 13],
    50:  [36, 37, 38, 1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 14, 16, 18, 20, 22, 24, 26, 28, 30, 32, 34],
    // more configurations
}
```

Figure 5.14:the normal sorting configuration.

The `NORMAL_SORTING_CONFIG` dictionary maps each patient identifier to a list of slice numbers in the correct order. Similarly, a sorting configuration is maintained for the stroke cases:


index.py

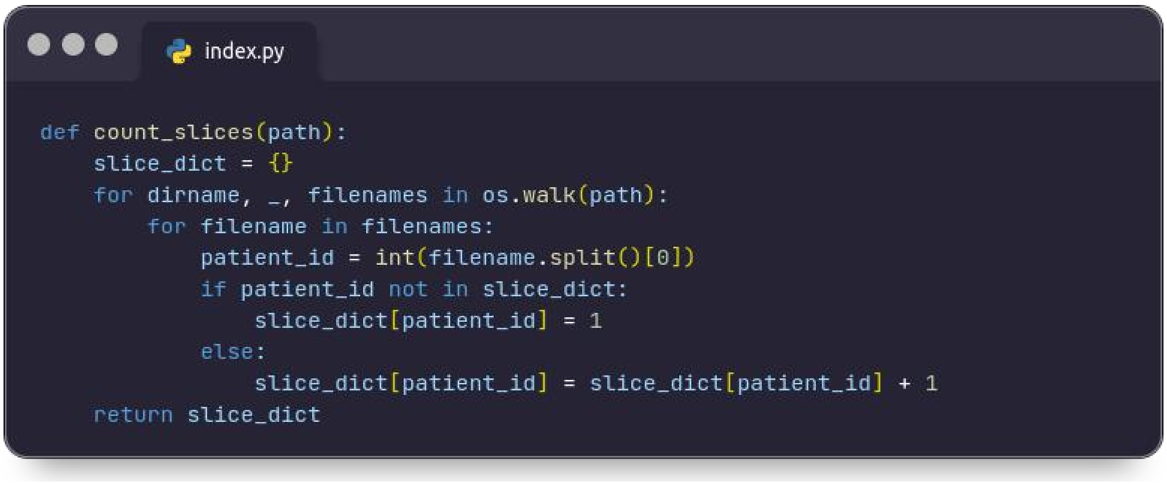
```
STROKE_SORTING_CONFIG = {
    66: [1, 2, 3, 4, 5, 6, 7, 8, 9, 10, 11, 12, 13, 15, 17, 19, 21, 23, 25, 27, 29, 31, 33, 35, 37, 38, 39, 40, 41],
    // more configurations
}
```

Figure 5.15 the normal sorting configuration.

Each configuration ensures that the slices for a given patient are processed in the correct sequence, which is vital for maintaining the integrity of the 3D structure of the brain scans.

5.7.5. Dataset Preparation

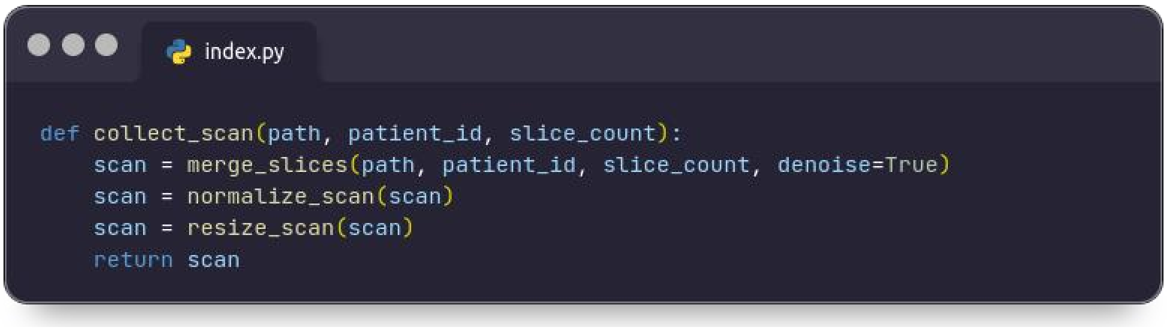
We have the count_slices to analyze the slices path and returns a dictionary with the slices count associated to each patient:


index.py

```python
def count_slices(path):
    slice_dict = {}
    for dirname, _, filenames in os.walk(path):
        for filename in filenames:
            patient_id = int(filename.split()[0])
            if patient_id not in slice_dict:
                slice_dict[patient_id] = 1
            else:
                slice_dict[patient_id] = slice_dict[patient_id] + 1
    return slice_dict
```

Figure 5.16: count_slices.

and collect_scan to load the scan per patient ID and prepare it:

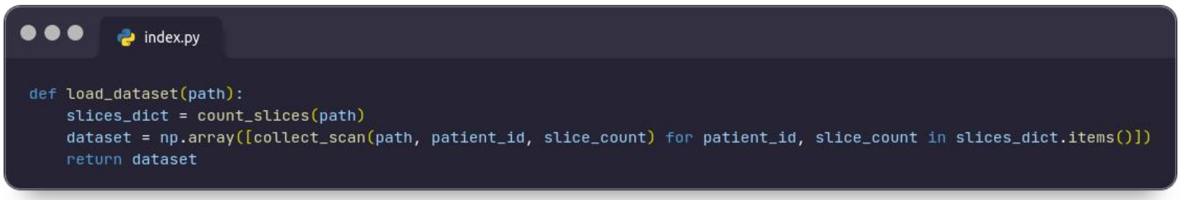

index.py

```python
def collect_scan(path, patient_id, slice_count):
    scan = merge_slices(path, patient_id, slice_count, denoise=True)
    scan = normalize_scan(scan)
    scan = resize_scan(scan)
    return scan
```

Figure 5.17: collect_scan.

5.7.6. Loading the Dataset

We defines a function to load the entire dataset into memory as a 4D array. Then we'll labeling the it as stroke (1) or normal (0) and splitting the dataset into training and validation sets.

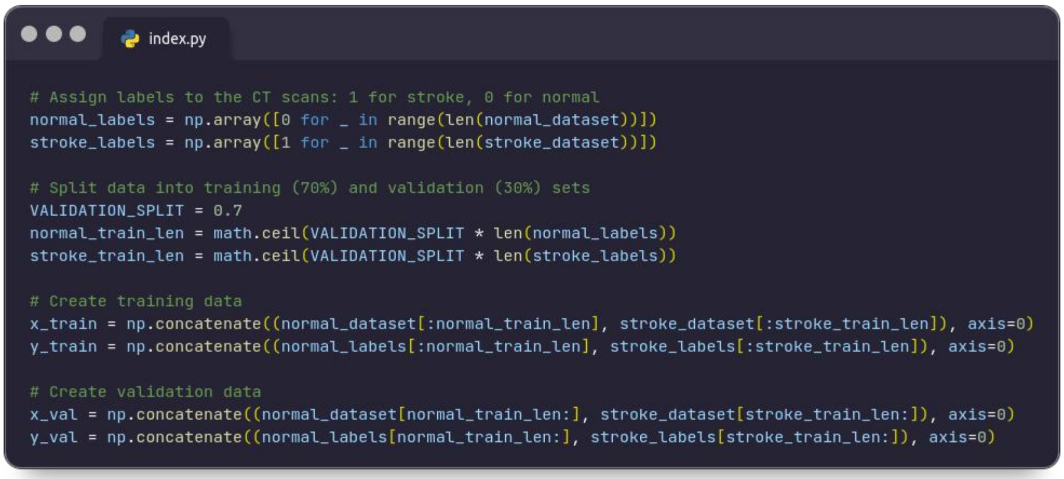


index.py

```python
def load_dataset(path):
    slices_dict = count_slices(path)
    dataset = np.array([collect_scan(path, patient_id, slice_count) for patient_id, slice_count in slices_dict.items()])
    return dataset
```

Figure 5.18: Load the dataset.

5.7.7. Labeling and Splitting the Dataset



index.py

```python
# Assign labels to the CT scans: 1 for stroke, 0 for normal
normal_labels = np.array([0 for _ in range(len(normal_dataset))])
stroke_labels = np.array([1 for _ in range(len(stroke_dataset))])

# Split data into training (70%) and validation (30%) sets
VALIDATION_SPLIT = 0.7
normal_train_len = math.ceil(VALIDATION_SPLIT * len(normal_labels))
stroke_train_len = math.ceil(VALIDATION_SPLIT * len(stroke_labels))

# Create training data
x_train = np.concatenate((normal_dataset[:normal_train_len], stroke_dataset[:stroke_train_len]), axis=0)
y_train = np.concatenate((normal_labels[:normal_train_len], stroke_labels[:stroke_train_len]), axis=0)

# Create validation data
x_val = np.concatenate((normal_dataset[normal_train_len:], stroke_dataset[stroke_train_len:]), axis=0)
y_val = np.concatenate((normal_labels[normal_train_len:], stroke_labels[stroke_train_len:]), axis=0)
```

Figure 5.19: Labeling the data.

5.7.8. Data Augmentation

We will use data augmentation techniques to increase the diversity of the training data without collecting new data. Augmentation helps improve the generalization capability of the model by artificially enlarging the dataset using random transformations.

The used techniques:

1. **Rotation:** Slightly rotates images to simulate different angles of view
2. **Flipping:** Vertically flips images to increase variability.

3. **Shifting:** Translates images along the x and y axes, creating different viewpoints.
4. **Zooming:** Zooms in on images to simulate different focal lengths
5. **Shear Layer:** Distorts the image by slanting its shape, which can simulate perspective distortions.

5.7.9. Building the model

This chapter details the architecture of the 3D Convolutional Neural Network (CNN) model, including the augmentation layers and the process of compiling the model with appropriate loss functions, optimizers, and performance metrics.

5.7.9.1. Model Parameters

In our model, we have defined several key parameters that govern the architecture and training of our Convolutional Neural Network (CNN). These parameters are crucial for understanding how the model processes input data and how it learns from the data during training. Below is a detailed explanation of each parameter:

1. Image Dimensions (WIDTH and HEIGHT)

- WIDTH = 128

- HEIGHT = 128

- **Explanation**: These parameters define the spatial dimensions of the input images fed into the CNN. Each image is resized or cropped to 128 pixels in width and 128 pixels in height. This standardization ensures that the input data is consistent in size, which is necessary for batch processing in the network. By using a fixed size, we also reduce computational complexity and memory usage.

2. Input Depth (DEPTH)

- DEPTH = 64

- **Explanation**: This parameter specifies the number of channels in the input images. For typical RGB images, the depth would be 3 (corresponding to the red, green, and blue channels). However, in our project, the depth is set to 64, indicating that each input image consists of 64 feature channels. This could result from specific pre-processing steps or the use of specialized input data.

3. Initial Learning Rate (INITIAL_LEARNING_RATE)

  - INITIAL_LEARNING_RATE = 0.0001

- **Explanation**: The learning rate is a critical hyperparameter that determines the step size during the optimization process. An initial learning rate of 0.0001 means that during the initial stages of training, the model's weights will be adjusted by this factor with respect to the loss gradient. A small learning rate helps in making gradual updates, which can lead to more stable convergence.

4. Learning Rate Decay Steps (DECAY_STEPS)

  - DECAY_STEPS = 100000

- **Explanation**: The decay steps parameter specifies the number of training steps after which the learning rate is decayed. In our model, after every 100000 steps, the learning rate is updated. This mechanism helps in gradually reducing the learning rate, allowing the model to fine-tune the weights as it approaches convergence.

5. Learning Rate Decay Rate (DECAY_RATE)

  - DECAY_RATE = 0.96

- **Explanation**: The decay rate defines the factor by which the learning rate is multiplied after each decay step. A decay rate of 0.96 means that the learning rate will be reduced by 4% after every 100000 steps. This gradual reduction helps in maintaining a balance between convergence speed and the precision of the weight updates.

5.7.9.2. Model Building Function

This section describes the architecture and implementation details of our CNN model designed for [your specific task, e.g., image classification, medical imaging, etc.]. The model leverages 3D convolutional layers to effectively capture spatial features from the input data.

Model Parameters

- Input Dimensions

- Width: 128 pixels

- Height: 128 pixels

- Depth: 64 channels

- Initial Learning Rate: 0.0001

- Learning Rate Decay:
- Steps: 100000 - Rate: 0.96

Model Architecture

Our model consists of several layers designed to extract and process features from the input data:

1. **Input Layer**: Accepts input images of shape (128, 128, 64).
2. **Reshape Layer**: Adds an extra dimension to make the input suitable for 3D convolutions.
3. **Convolutional Layers**: Four 3D convolutional layers with increasing filter sizes and ReLU activations.
4. **Pooling Layers**: 3D max pooling layers to reduce spatial dimensions.
5. **Batch Normalization Layers**: Normalize activations to improve training stability.
6. **Global Average Pooling**: Reduces each feature map to a single value.
7. **Fully Connected Layer**: Dense layer with 512 units and ReLU activation, followed by dropout for regularization.
8. **Output Layer**: Single unit with sigmoid activation for binary classification.

Learning Rate Schedule

The learning rate decreases exponentially based on the specified decay steps and rate, allowing the model to fine-tune its weights as training progresses.

Compilation

The model is compiled using the Adam optimizer with the defined learning rate schedule and binary cross-entropy loss. The performance is evaluated using specified metrics.

Conclusion

This model is designed to effectively learn from high-dimensional input data and perform accurate binary classification. Further details on training procedures, data preprocessing, and evaluation metrics are discussed in subsequent sections.

5.7.10. Model Training and Evaluation

In this section, we cover the final step of the machine learning pipeline: training the model. This involves setting up callbacks for saving the best model, defining the training function, and training the model with different augmentation strategies. We aim to ensure the model learns effectively and generalizes well to unseen data.

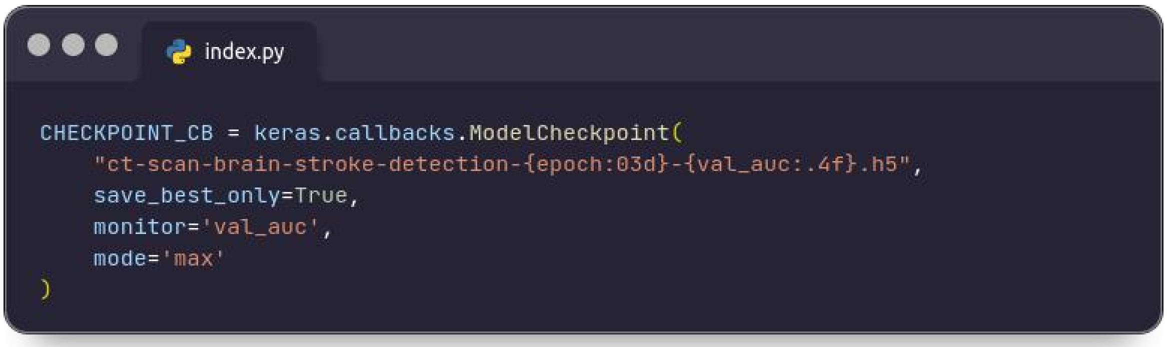
5.7.10.1. Setting the Number of Epochs

The number of epochs determines how many times the model will iterate over the entire training dataset. Training for too few epochs might lead to underfitting, while too many epochs might cause overfitting. After careful consideration, we define the number of epochs as follows:

EPOCHS = 150

5.7.10.2. Callback for Model Checkpointing

To ensure we save the best version of our model during training, we use a callback for model checkpointing. This callback monitors the validation AUC (Area Under the Curve) and saves the model weights whenever there is an improvement. This strategy helps in retaining the model that performs best on the validation set, preventing overfitting and ensuring better generalization.

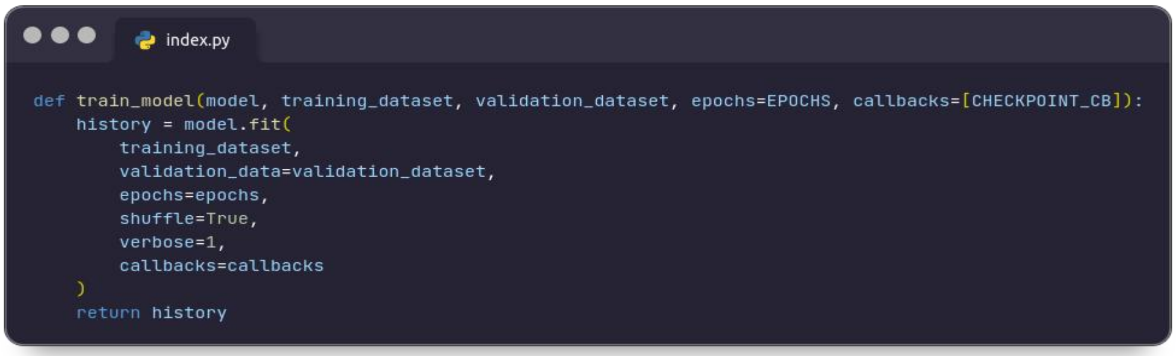


```python
CHECKPOINT_CB = keras.callbacks.ModelCheckpoint(
    "ct-scan-brain-stroke-detection-{epoch:03d}-{val_auc:.4f}.h5",
    save_best_only=True,
    monitor='val_auc',
    mode='max'
)
```

Figure 5.20. Setup the model checkpoint.

5.7.10.3. Training Function

We define a function to encapsulate the training process. This function takes the model, training dataset, validation dataset, number of epochs, and any callbacks as input parameters. It then trains the model, performs validation at the end of each epoch, and returns the training history for further analysis.



```python
def train_model(model, training_dataset, validation_dataset, epochs=EPOCHS, callbacks=[CHECKPOINT_CB]):
    history = model.fit(
        training_dataset,
        validation_data=validation_dataset,
        epochs=epochs,
        shuffle=True,
        verbose=1,
        callbacks=callbacks
    )
    return history
```

Figure 5.21: The training function.

5.7.10.3. Conclusion

This chapter outlines the comprehensive steps involved in training and evaluating our CNN model. By setting up proper callbacks, defining a robust training function, and applying data augmentation techniques, we aim to develop a model that not only performs well on the training data but also generalizes effectively to new, unseen data. The next chapter will delve into the results of our training process and discuss the model's performance on the test dataset.

5.8 Overview of the Website

5.8.1 Tools, Technologies and Techniques

There are numerous tools and techniques available to aid in the completion of tasks and the execution of processes.

UI (User Interface) and UX (User Experience) are two crucial aspects of product design that focus on enhancing the interaction between users and a product. While often used interchangeably, they represent distinct yet complementary disciplines.

User Interface (UI)

UI refers to the visual elements and interactive components that users directly engage with. It encompasses the layout, colors, typography, icons, buttons, and other visual components that shape the user's perception and interaction with the product. A well-designed UI should be aesthetically pleasing, intuitive, and easy to navigate, guiding users seamlessly through their tasks.

Key aspects of UI design:

- **Visual hierarchy:** Arranging elements to create a clear visual flow and prioritize important information.
- **Color theory:** Using colors effectively to convey emotions, establish branding, and enhance usability.
- **Typography:** Choosing fonts that are legible, consistent with the brand, and appropriate for the content.
- **Layout:** Structuring elements in a way that is organized, balanced, and optimized for different screen sizes.
- **Responsiveness:** Ensuring that the UI adapts gracefully to different devices and screen resolutions.

User Experience (UX)

UX delves into the overall experience a user has when interacting with a product. It considers the user's emotions, thoughts, and behaviors throughout their journey with the product. A positive UX should be enjoyable, efficient, and satisfying, leaving users with a sense of accomplishment and loyalty.

Key aspects of UX design:

- **User research:** Understanding user needs, goals, and pain points through research methods like surveys, interviews, and usability testing.
- **Interaction design:** Designing interactions that are intuitive, consistent, and provide clear feedback to user actions.
- **Usability testing:** Evaluating the product's usability through testing with real users to identify and address usability issues
- **Accessibility:** Ensuring that the product is accessible to users with disabilities, considering factors like visual impairments, motor limitations, and cognitive differences.

The Relationship Between UI and UX

UI and UX are not isolated disciplines; they work in tandem to create a cohesive and successful product. A great UI complements the UX by providing a visually appealing and user-friendly interface that supports the overall user experience. Conversely, a well-crafted UX guides the UI design, ensuring that the visual elements align with the user's needs and expectations.

Importance of UI and UX

In today's competitive landscape, UI and UX play a crucial role in the success of digital products. A well-designed UI and UX can:

- **Increase user engagement:** Keep users coming back and spending more time using the product.
- **Improve user satisfaction:** Create a positive user experience that leads to loyalty and positive word-of-mouth.
- **Boost conversion rates:** Encourage users to take desired actions, such as making purchases or signing up for services

Integrated Development Environment (IDE) in which the source code can be implemented, such as Android Studio IDE, which will be used to develop the system using the Java programming language, as well as to design the Graphical User Interface (GUI) allowing interaction processes among system components.

- Adobe Photoshop (UI) for designing the app's user interface, buttons, and layout.
- Sketch (UX) to improve the user experience.

5.8.1.1 GUI System

GUI:

A GUI (graphical User Interface) is a system of interactive visual components for computer software. A GUI displays objects that convey information and represent actions that can be taken by the user. The objects change color, size or visibility when the user interacts with them.

GUI overview:

A GUI includes GUI objects, like icons, cursors, and buttons. These graphical elements are sometimes enhanced with sounds, or visual effects like transparency and drop shadows. Using these objects, a user can use the computer without having to know commands.

What are the elements of a GUI?

To make a GUI as user-friendly as possible, there are different elements and objects that the user use to interact with the software. Below is a list of each of these with a brief description.

- **Button**: A graphical representation of a button that performs an action in a program when pressed
- **Dialog box**: A type of window that displays additional information and asks a user for input.
- **Icon**: Small graphical representation of a program, feature, or file.
- **Menu**: List of commands or choices offered to the user through the menu bar.
- **Menu bar**: Thin, horizontal bar containing the labels of menus.
- **Ribbon**: Replacement for the file menu and toolbar that groups programs activities together.
- **Tab**: Clickable area at the top of a window that shows another page or area.
- **Toolbar**: Row of buttons, often near the top of an application window that controls software functions.
- **Window**: Rectangular section of the computer's display that shows the program currently being used.

How does a GUI work?

A GUI uses windows, icons, and menus to carry out commands, such as opening, deleting, and moving files. Although a GUI operating system is primarily

navigated using a mouse, a keyboard can also be used via keyboard shortcuts or the arrow keys.
For example, if you want to open a program on a GUI system, you would move the mouse pointer to the program's icon and double-click it. With a command line interface, you need to know the commands to navigate to the directory containing the program, list the files, and then run the file.

What are the benefits of GUI?
A GUI is more user-friendly than a text-based command-line interface, such as MS-DOS, or the shell of Unix-like operating systems.
Unlike a command-line operating system or CUI, like Unix or MS-DOS, GUI operating systems are easier to learn and use because commands do not need to be memorized. Additionally, users do not need to know any programming languages. Because of their ease of use and more modern appearance, GUI operating systems have come to dominate today's market.

5.8.1.2 Technologies

Front End.

HTML

- HTML is essential for creating the structure of web pages. By using tags and attributes, you can create headings, paragraphs, links, images, and lists. It forms the foundation of web development, often used with CSS for styling and JavaScript for interactivity.

CSS

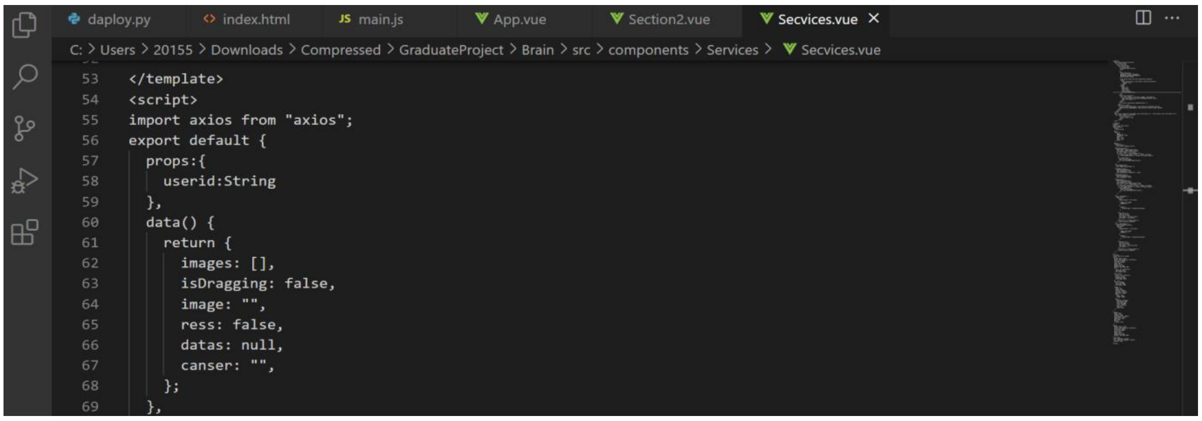
- CSS is used to style HTML documents by defining rules that specify how elements should look. It enhances the visual presentation of web pages, allowing for separation of content (HTML) from design (CSS). This separation makes it easier to maintain and update web pages.

- The framework used is **Bootstrap**, a versatile and powerful front-end framework that simplifies the process of designing responsive and modern web pages. Its grid system, pre-designed components, and utility classes allow developers to create attractive layouts quickly, while its JavaScript plugins add interactivity and enhance the user experience.

JavaScript

- JavaScript is a versatile language essential for adding interactivity to web pages. It works alongside HTML and CSS to create dynamic user experiences, handle events, manipulate the DOM, and perform various client-side tasks. It allows developers to enhance the user experience by making web pages more engaging and responsive.

- **Vue.js** is a flexible and powerful framework for building interactive web interfaces. Its reactivity system, component-based architecture, and support for single-file components make it a popular choice for developers looking to create maintainable and efficient applications.



daploy.py | index.html | main.js | App.vue | Section2.vue | Secvices.vue

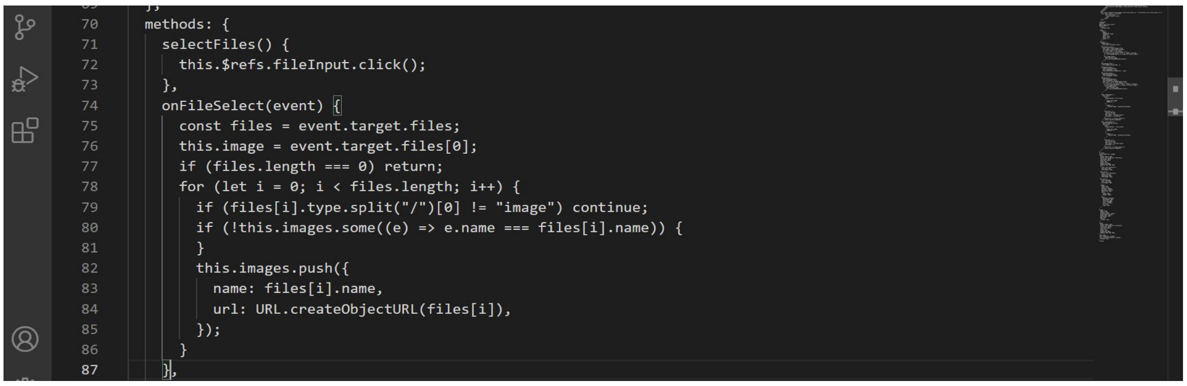
C: > Users > 20155 > Downloads > Compressed > GraduateProject > Brain > src > components > Services > Secvices.vue

```
</template>
<script>
import axios from "axios";
export default {
  props:{
    userid:String
  },
  data() {
    return {
      images: [],
      isDragging: false,
      image: "",
      ress: false,
      datas: null,
      canser: "",
    };
  },
```

Figure 5.22: Java script code with explanation.

- **import axios from "axios";**: This line imports the Axios library, which is a popular JavaScript library used to make HTTP requests.
- **export default { ... }**: This is the default export of the Vue component. Everything inside the curly braces defines the properties and behavior of the component.
- **props**: This is an object where we define the properties that the component accepts from its parent component.
- userid**: String**: This line defines a prop named userid that is expected to be a string. This prop allows the parent component to pass a userid value to this component.
- **data()**: This is a function that returns an object. This object contains the reactive data properties of the component.
- **images: []**: An empty array to hold images.
- **isDragging: false**,: A boolean flag that can be used to indicate if an item is being dragged (likely used in drag-and-drop functionality).
- **image: ""**: A string to hold the URL or data of a single image.
- **ress: false**: A boolean flag, its purpose isn't clear from this snippet alone but it's likely used to indicate some sort of state.
- **datas: null**: A variable to hold some data, initially set to null.

- **canser: ""**: A string variable, possibly meant to be cancel or something similar, but it's unclear without further context.

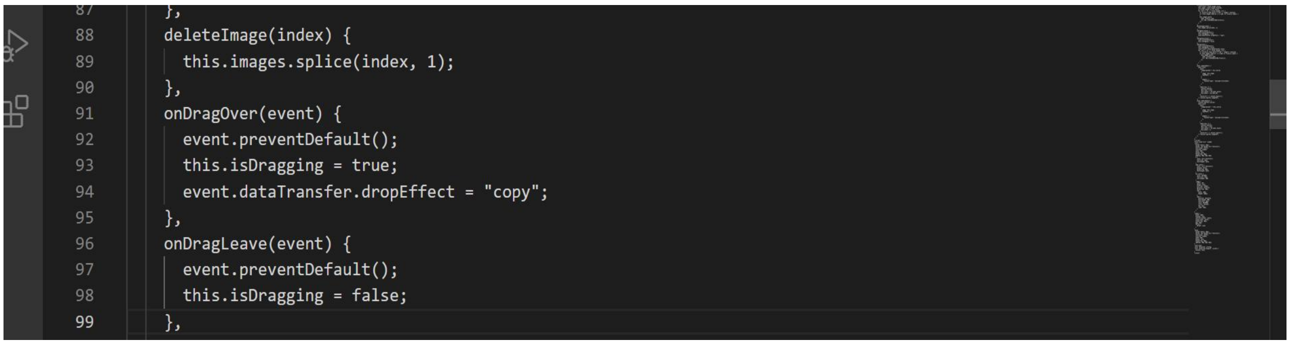


```
  methods: {
    selectFiles() {
      this.$refs.fileInput.click();
    },
    onFileSelect(event) {
      const files = event.target.files;
      this.image = event.target.files[0];
      if (files.length === 0) return;
      for (let i = 0; i < files.length; i++) {
        if (files[i].type.split("/")[0] != "image") continue;
        if (!this.images.some((e) => e.name === files[i].name)) {
        }
        this.images.push({
          name: files[i].name,
          url: URL.createObjectURL(files[i]),
        });
      }
    },
```

Figure 5.23: Java script code with explanation.

- **Purpose**: This method programmatically triggers a click event on a file input element.
- **Usage**: It simulates a user clicking on a file input element to open the file picker dialog.
- Details:
- this.$refs.fileInput: References the file input element using Vue's $refs system.
- .click(): Triggers the click event on the referenced file input element.
- **Purpose**: This method handles the event when files are selected in the file input.
- **Usage**: It processes the selected files, filters out non-image files, and adds image files to the images array.
- Details:
- event.target.files: Accesses the list of files selected by the user.
- this.image = event.target.files[0];: Sets the first selected file to the image data property.
- if (files.length === 0) return;: If no files are selected, the method returns early.
- for (let i = 0; i < files.length; i++) { ... }: Iterates over the selected files.

- if (files[i].type.split("/")[0] != "image") continue;: Skips files that are not images.
- if (!this.images.some((e) => e.name === files[i].name)) { ... }: Checks if the file is already in the images array to avoid duplicates.
- this.images.push({ name: files[i].name, url: URL.createObjectURL(files[i]) });: Adds the image file to the images array with a name and a URL created using URL.createObjectURL().

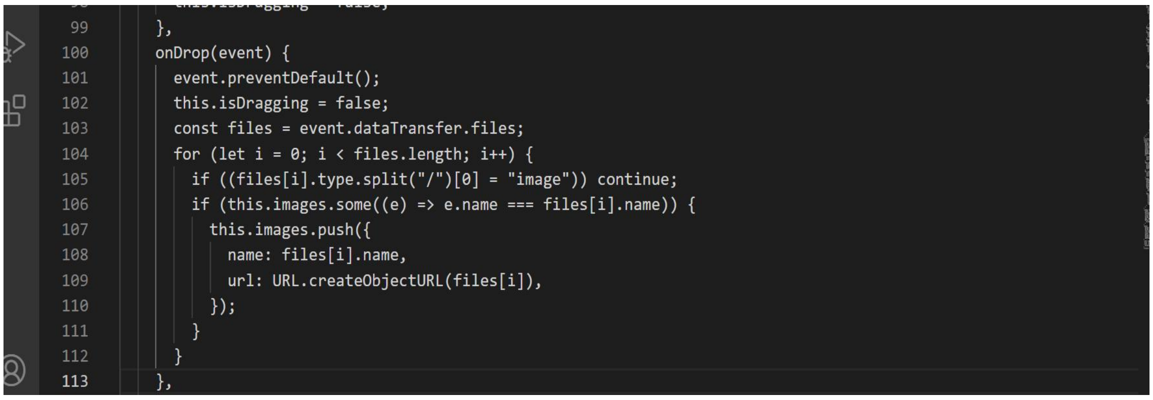


```
    },
    deleteImage(index) {
      this.images.splice(index, 1);
    },
    onDragOver(event) {
      event.preventDefault();
      this.isDragging = true;
      event.dataTransfer.dropEffect = "copy";
    },
    onDragLeave(event) {
      event.preventDefault();
      this.isDragging = false;
    },
```

Figure 5.24: Java script code with explanation.

- **Purpose**: This method removes an image from the images array at the specified index.
- **Usage**: It allows users to delete an image from the list of selected images.
- Details:
- this.images.splice(index, 1);: Removes one element from the images array at the given index.
- **Purpose**: This method handles the drag-over event when a file is being dragged over the drop area.
- **Usage**: It provides visual feedback and prepares the drop area to accept the dragged file.
- Details:
- event.preventDefault();: Prevents the default behavior to allow the drop.

- this.isDragging = true;: Sets the isDragging flag to true, which can be used to apply visual styles indicating the drop area is active.
- event.dataTransfer.dropEffect = "copy";: Changes the cursor to indicate that a copy action will occur if the file is dropped.
- **Purpose**: This method handles the drag-leave event when a file is dragged away from the drop area.
- **Usage**: It removes the visual feedback indicating the drop area is active.
- Details:
- event.preventDefault();: Prevents the default behavior.
- this.isDragging = false;: Sets the isDragging flag to false, which can be used to revert the visual styles indicating the drop area is no longer active.



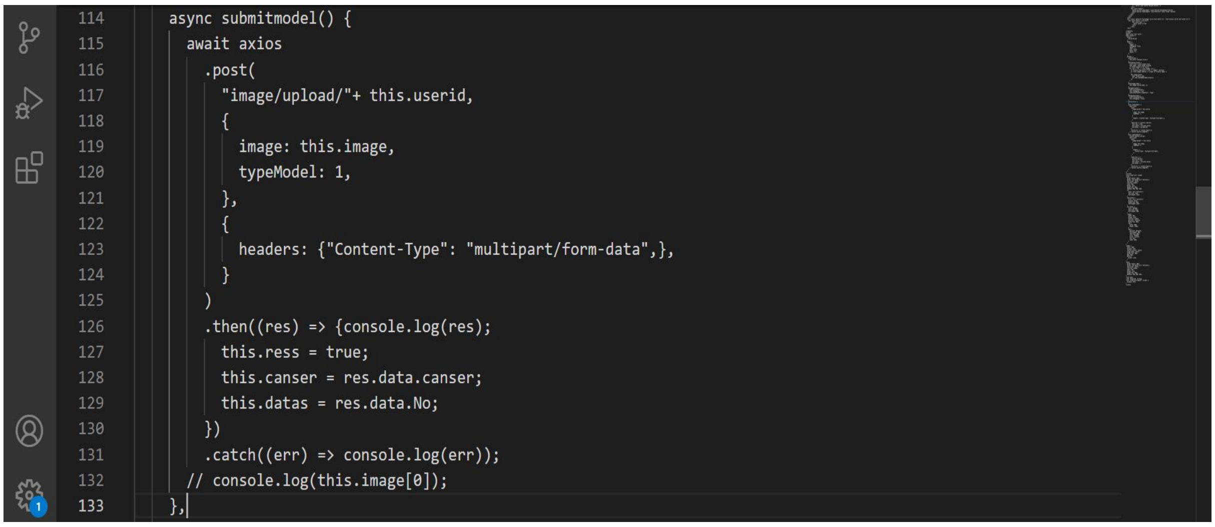
```
  },
  onDrop(event) {
    event.preventDefault();
    this.isDragging = false;
    const files = event.dataTransfer.files;
    for (let i = 0; i < files.length; i++) {
      if ((files[i].type.split("/")[0] = "image")) continue;
      if (this.images.some((e) => e.name === files[i].name)) {
        this.images.push({
          name: files[i].name,
          url: URL.createObjectURL(files[i]),
        });
      }
    }
  },
```

Figure 5.25: Java script code with explanation.

The onDrop method handles the event when files are dropped onto a designated area:

- It prevents the default browser behavior.
- Updates the isDragging state to false.
- Iterates through the dropped files.
- Filters out non-image files.
- Checks for duplicate image names.
- Adds new images to the images array with a URL for previewing.

This method, combined with the previous ones, provides a complete drag-and-drop interface for uploading images in the Vue.js component.



```
async submitmodel() {
  await axios
    .post(
      "image/upload/"+ this.userid,
      {
        image: this.image,
        typeModel: 1,
      },
      {
        headers: {"Content-Type": "multipart/form-data",},
      }
    )
    .then((res) => {console.log(res);
      this.ress = true;
      this.canser = res.data.canser;
      this.datas = res.data.No;
    })
    .catch((err) => console.log(err));
  // console.log(this.image[0]);
},
```

Figure 5.26: Java script code with explanation.

Declares an asynchronous function that allows the use of await within it to handle asynchronous operations like HTTP requests.

- Sends a POST request to the server to upload an image.
- **URL**: "image/upload/" + this.userid constructs the URL by appending the userid prop to the base URL "image/upload/".
- **Data**: The request payload contains an object with image and typeModel.
- image: this.image: The image file to be uploaded.
- typeModel: 1: Some type identifier, likely indicating the type of model being submitted.
- **Headers**: Sets the content type to multipart/form-data, which is necessary for file
- Handles the response from the server.
- **Logging**: console.log(res); logs the response to the console for debugging purposes.
- Updating State:

- this.ress = true;: Sets the ress flag to true, indicating that the request was successful.
- this.canser = res.data.canser;: Updates the canser data property with the value from the response.
- this.datas = res.data.No;: Updates the datas data property with the value from the response.

Catches any errors that occur during the request and logs them to the console.
This line is commented out. If uncommented, it would log the first element of this.image to the console, but since this.image is a single file, not an array, this line may be unnecessary or incorrect.

Back End.

Java

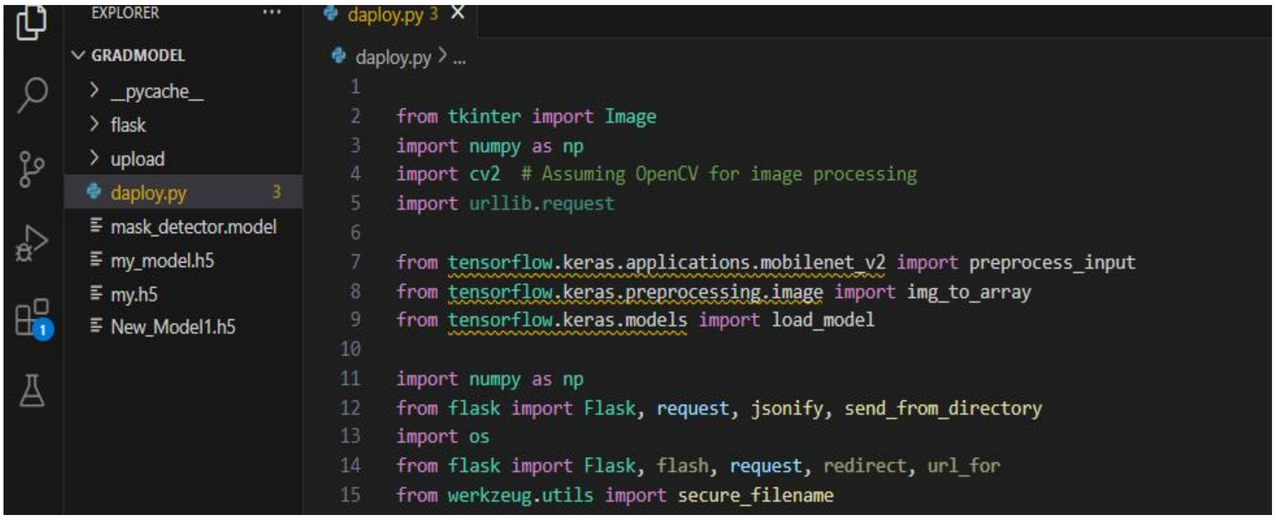
- Java is a widely used, object-oriented programming language designed for building platform-independent applications. Its object-oriented nature, platform independence, and extensive libraries make it a popular choice for developers in various domains, from web and mobile development to enterprise systems.
- The framework used is **Spring Boot** is a powerful framework for building Spring-based applications with minimal setup and configuration. Its features like embedded servers, dependency management, and production-ready tools make it a popular choice for developing scalable and robust applications quickly.

Python

- Python is a versatile and powerful language known for its simplicity and readability. It is widely used across various domains due to its extensive libraries, ease of use, and support for multiple programming paradigms.
- The framework used is **Flask**, a versatile and easy-to-use web framework for Python that allows for quick development of web applications, designed for building web applications quickly and with minimal setup. Its minimalist approach provides flexibility, while its support for extensions ensures that additional features can be easily integrated when needed.

Database (MYSQL)

- MySQL is a popular, open-source relational database management system known for its high performance and reliability. It is widely used in web development due to its ease of integration with various programming languages and its strong support for handling large datasets efficiently.

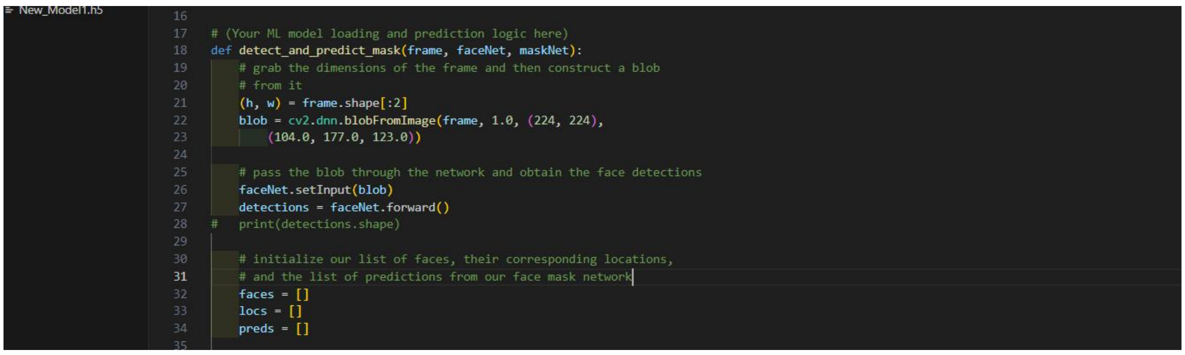


```python

from tkinter import Image
import numpy as np
import cv2  # Assuming OpenCV for image processing
import urllib.request

from tensorflow.keras.applications.mobilenet_v2 import preprocess_input
from tensorflow.keras.preprocessing.image import img_to_array
from tensorflow.keras.models import load_model

import numpy as np
from flask import Flask, request, jsonify, send_from_directory
import os
from flask import Flask, flash, request, redirect, url_for
from werkzeug.utils import secure_filename
```

Figure 5.27:API (Application Programming Interface) Code.

- **tkinter (Image)**: Although Image is not typically imported directly from tkinter, this library is generally used for creating graphical user interfaces (GUIs). You might actually be referring to PIL.Image from the Pillow library for image processing.

- **numpy (np)**: A fundamental library for scientific computing in Python, used here for handling arrays and matrices.
- **cv2 (OpenCV)**: An open-source computer vision library. It provides tools for image processing and computer vision tasks.
- **urllib.request**: A module for opening and reading URLs. Often used to fetch data from the web.
- **tensorflow.keras.applications.mobilenet_v2 (preprocess_input)**: Part of TensorFlow's Keras module, this provides functions to preprocess input data for the MobileNetV2 model.
- **tensorflow.keras.preprocessing.image (img_to_array)**: Utility for converting a PIL Image instance to a Numpy array.
- **tensorflow.keras.models (load_model)**: Provides functions to load pre-trained Keras models.
- **Flask**: A lightweight WSGI web application framework in Python, used for creating web applications and APIs.
- **os**: A module providing a way of using operating system-dependent functionality like reading or writing to the file system.
- **werkzeug.utils (secure_filename)**: A utility for securing a filename before storing it directly on the filesystem.



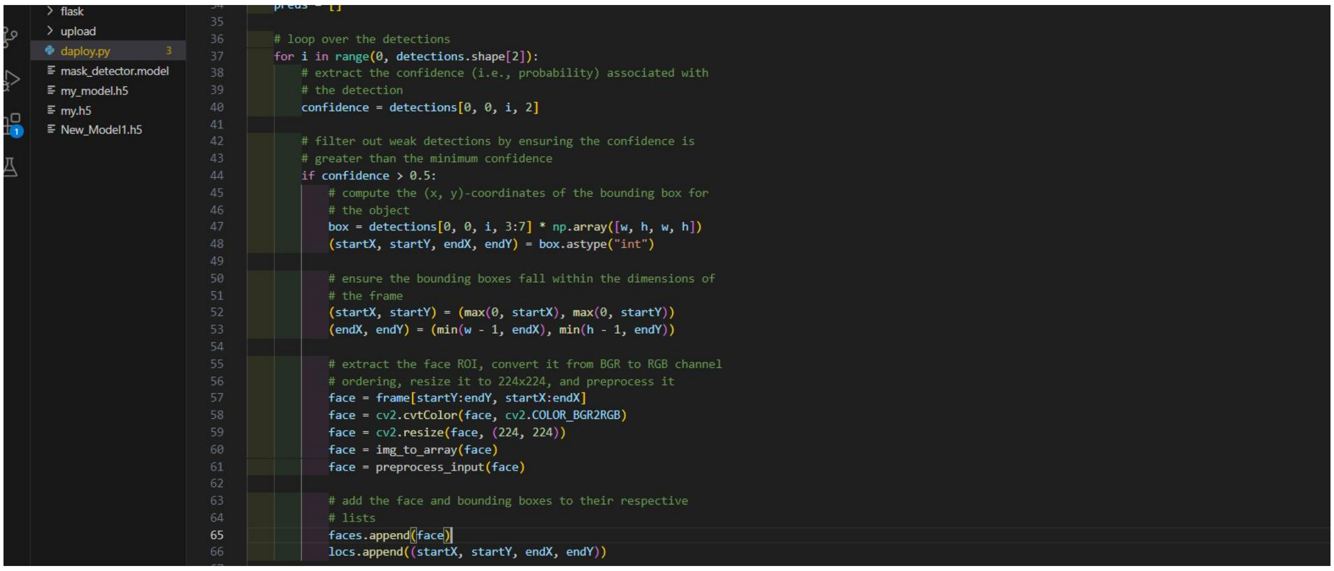
```python

# (Your ML model loading and prediction logic here)
def detect_and_predict_mask(frame, faceNet, maskNet):
    # grab the dimensions of the frame and then construct a blob
    # from it
    (h, w) = frame.shape[:2]
    blob = cv2.dnn.blobFromImage(frame, 1.0, (224, 224),
        (104.0, 177.0, 123.0))

    # pass the blob through the network and obtain the face detections
    faceNet.setInput(blob)
    detections = faceNet.forward()
#   print(detections.shape)

    # initialize our list of faces, their corresponding locations,
    # and the list of predictions from our face mask network
    faces = []
    locs = []
    preds = []

```

Figure 5.28: API (Application Programming Interface) Code.

- frame: The input image frame where face masks need to be detected.

- faceNet: The pre-trained deep learning model used for face detection.
- maskNet: The pre-trained deep learning model used for face mask classification.

  It extracts the height and width of the input frame.
- blobFromImage function of OpenCV's dnn module is used to preprocess the frame:
- It resizes the image to (224, 224) pixels.
- Normalizes the pixel intensities.
- Converts the image to a blob format suitable for input to a neural network.
- The pre-trained faceNet model is used to detect faces in the image.
- The preprocessed blob is set as the input to the faceNet model.
- The forward method is called to perform forward pass inference, obtaining face detections.



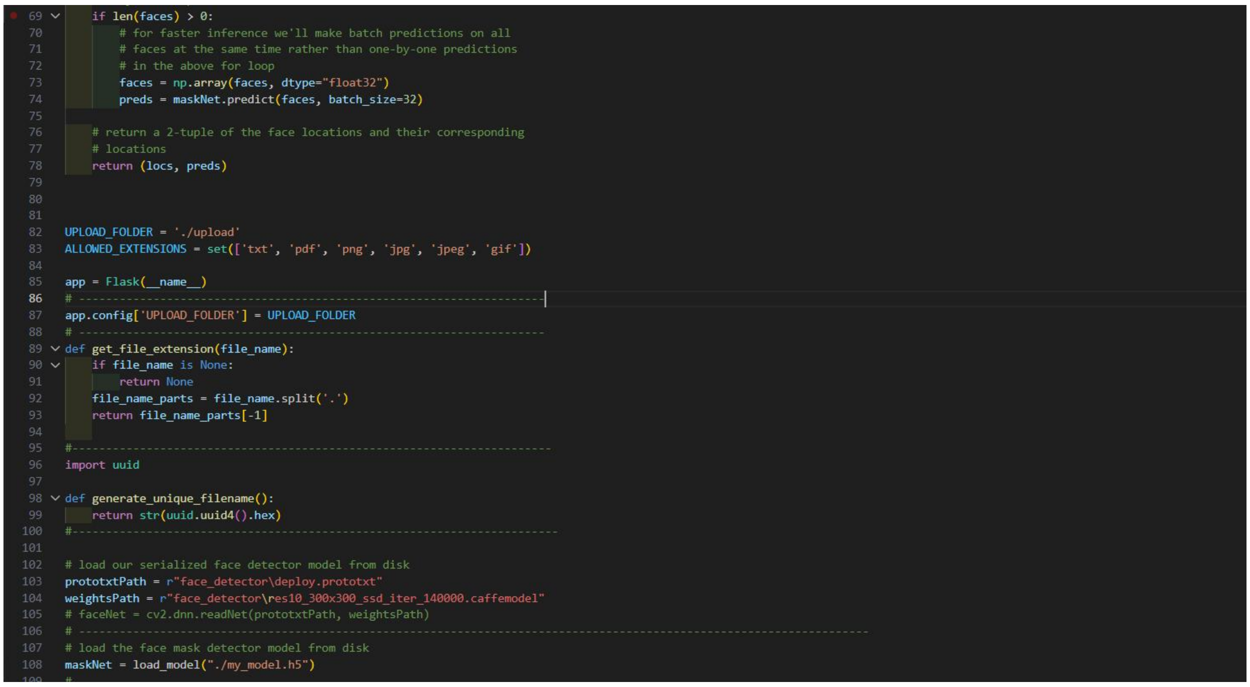
```python
preds = []

# loop over the detections
for i in range(0, detections.shape[2]):
    # extract the confidence (i.e., probability) associated with
    # the detection
    confidence = detections[0, 0, i, 2]

    # filter out weak detections by ensuring the confidence is
    # greater than the minimum confidence
    if confidence > 0.5:
        # compute the (x, y)-coordinates of the bounding box for
        # the object
        box = detections[0, 0, i, 3:7] * np.array([w, h, w, h])
        (startX, startY, endX, endY) = box.astype("int")

        # ensure the bounding boxes fall within the dimensions of
        # the frame
        (startX, startY) = (max(0, startX), max(0, startY))
        (endX, endY) = (min(w - 1, endX), min(h - 1, endY))

        # extract the face ROI, convert it from BGR to RGB channel
        # ordering, resize it to 224x224, and preprocess it
        face = frame[startY:endY, startX:endX]
        face = cv2.cvtColor(face, cv2.COLOR_BGR2RGB)
        face = cv2.resize(face, (224, 224))
        face = img_to_array(face)
        face = preprocess_input(face)

        # add the face and bounding boxes to their respective
        # lists
        faces.append(face)
        locs.append((startX, startY, endX, endY))
```

Figure 5.29:API (Application Programming Interface) Code.

- Iterates over the detections found by the face detection model.
- Extracts the confidence (probability) associated with the detection.

- Filters out weak detections by ensuring the confidence is greater than a minimum threshold (here, 0.5).
- Computes the (x, y)-coordinates of the bounding box for the detected face. The bounding box coordinates are relative to the dimensions of the input frame.
- Ensures that the bounding boxes fall within the dimensions of the frame to prevent errors.
- Extracts the face region from the frame.
- Converts the color space of the face region from BGR to RGB.
- Resizes the face region to the required input size (224x224) for the mask classification model.
- Converts the face region to a NumPy array and preprocesses it for the model.

Adds the preprocessed face and its bounding box coordinates to their respective lists for further processing.

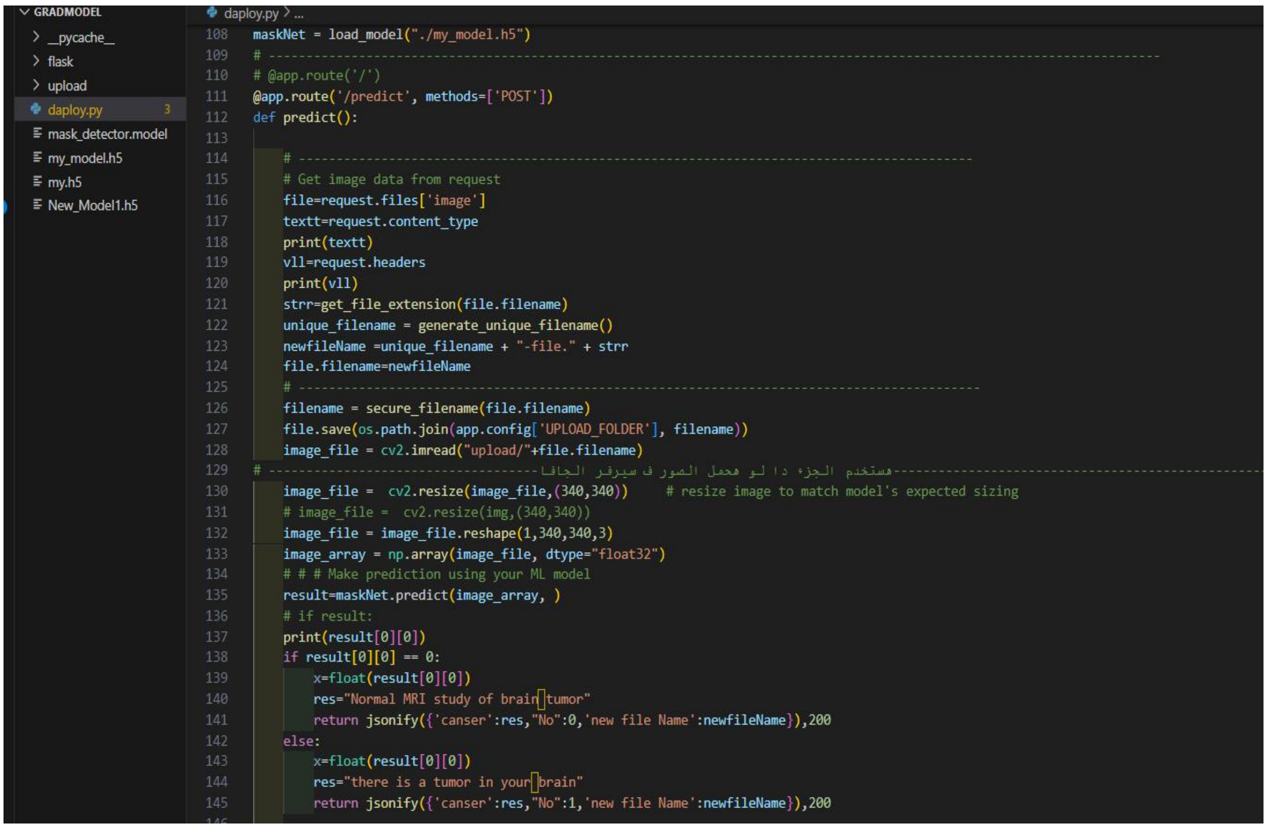


```
    if len(faces) > 0:
        # for faster inference we'll make batch predictions on all
        # faces at the same time rather than one-by-one predictions
        # in the above for loop
        faces = np.array(faces, dtype="float32")
        preds = maskNet.predict(faces, batch_size=32)

    # return a 2-tuple of the face locations and their corresponding
    # locations
    return (locs, preds)



UPLOAD_FOLDER = './upload'
ALLOWED_EXTENSIONS = set(['txt', 'pdf', 'png', 'jpg', 'jpeg', 'gif'])

app = Flask(__name__)
# -------------------------------------------------------------------
app.config['UPLOAD_FOLDER'] = UPLOAD_FOLDER
# -------------------------------------------------------------------
def get_file_extension(file_name):
    if file_name is None:
        return None
    file_name_parts = file_name.split('.')
    return file_name_parts[-1]

#--------------------------------------------------------------------
import uuid

def generate_unique_filename():
    return str(uuid.uuid4().hex)
#--------------------------------------------------------------------

# load our serialized face detector model from disk
prototxtPath = r"face_detector\deploy.prototxt"
weightsPath = r"face_detector\res10_300x300_ssd_iter_140000.caffemodel"
# faceNet = cv2.dnn.readNet(prototxtPath, weightsPath)
# ---------------------------------------------------------------------------------------------------
# load the face mask detector model from disk
maskNet = load_model("./my_model.h5")
```

Figure 5.30:API (Application Programming Interface) Code.

- If there are detected faces, it converts the list of faces into a NumPy array.

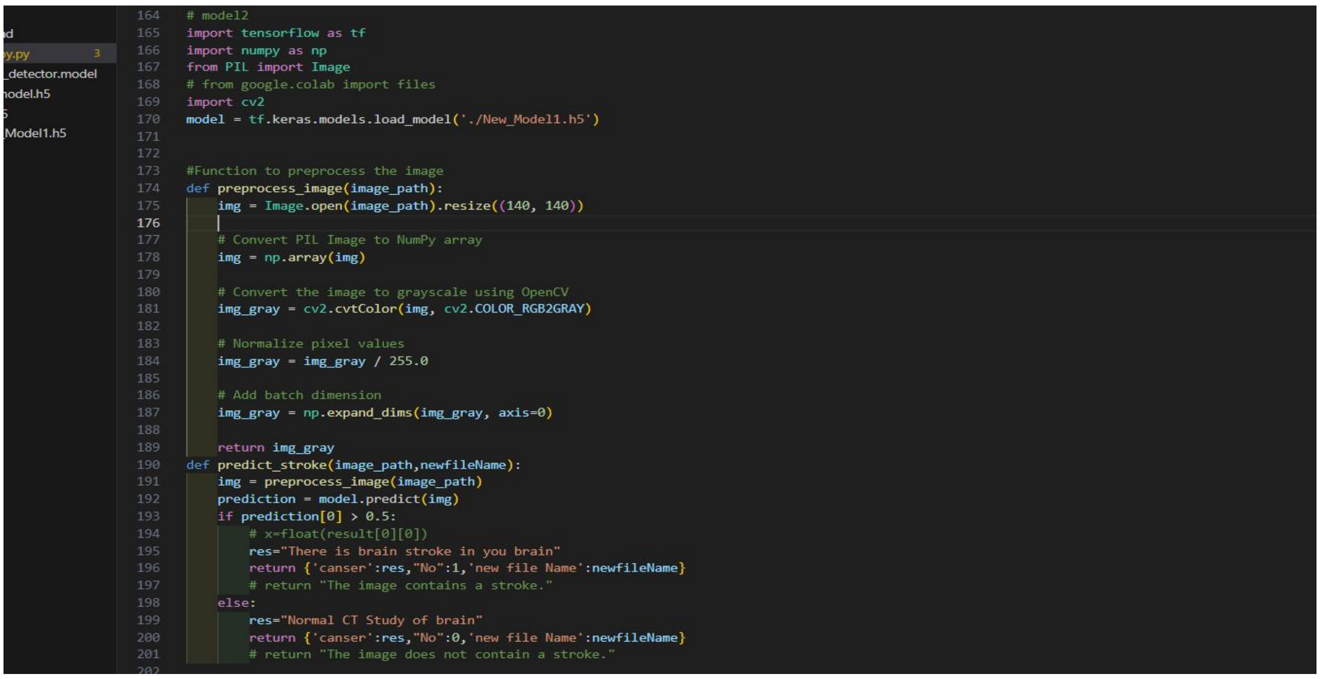
- It then uses the predict method of the mask detection model (maskNet) to predict whether each face is wearing a mask or not. This is done in batch mode for faster processing.
- Returns a tuple containing the locations of the detected faces and their corresponding predictions regarding whether they are wearing masks or not.
- Sets up configurations for handling uploaded files, including the allowed file extensions and the upload folder.
- Defines a utility function to extract the file extension from a given file name.
- Defines a utility function to generate a unique filename using the UUID module.
- Loads the face detection model (prototxtPath and weightsPath) and the face mask detection model (maskNet) from disk.
- The face detection model seems to be based on the Caffe framework, and the face mask detection model is loaded using Keras's load_model function.



```python
maskNet = load_model("./my_model.h5")
# ------------------------------------------------------------------------------------------------------------
# @app.route('/')
@app.route('/predict', methods=['POST'])
def predict():

    # ----------------------------------------------------------------------------------
    # Get image data from request
    file=request.files['image']
    textt=request.content_type
    print(textt)
    vll=request.headers
    print(vll)
    strr=get_file_extension(file.filename)
    unique_filename = generate_unique_filename()
    newfileName =unique_filename + "-file." + strr
    file.filename=newfileName
    # ----------------------------------------------------------------------------------
    filename = secure_filename(file.filename)
    file.save(os.path.join(app.config['UPLOAD_FOLDER'], filename))
    image_file = cv2.imread("upload/"+file.filename)
# ------------------------------------مستخدم الجزء دا لو هحمل الصور ف سيرفر الجافا------------------------------------
    image_file =  cv2.resize(image_file,(340,340))     # resize image to match model's expected sizing
    # image_file =  cv2.resize(img,(340,340))
    image_file = image_file.reshape(1,340,340,3)
    image_array = np.array(image_file, dtype="float32")
    # # # Make prediction using your ML model
    result=maskNet.predict(image_array, )
    # if result:
    print(result[0][0])
    if result[0][0] == 0:
        x=float(result[0][0])
        res="Normal MRI study of brain tumor"
        return jsonify({'canser':res,"No":0,'new file Name':newfileName}),200
    else:
        x=float(result[0][0])
        res="there is a tumor in your brain"
        return jsonify({'canser':res,"No":1,'new file Name':newfileName}),200
```

Figure 5.31: API (Application Programming Interface) Code.

- Retrieves the image file from the request.
- Collects information about the content type and headers of the request.
- Generates a unique filename for the uploaded image.
- Saves the uploaded image with the generated filename in the upload folder specified in the application's configuration.
- Reads the uploaded image using OpenCV (cv2).
- Resizes the image to match the expected input size of the model.
- Prepares the image array for prediction.
- Uses the trained machine learning model (maskNet) to predict the class of the input image.
- Based on the prediction result, it returns a JSON response indicating whether the image depicts a normal MRI study or if there's a tumor in the brain. It also includes the new filename generated for the uploaded image.



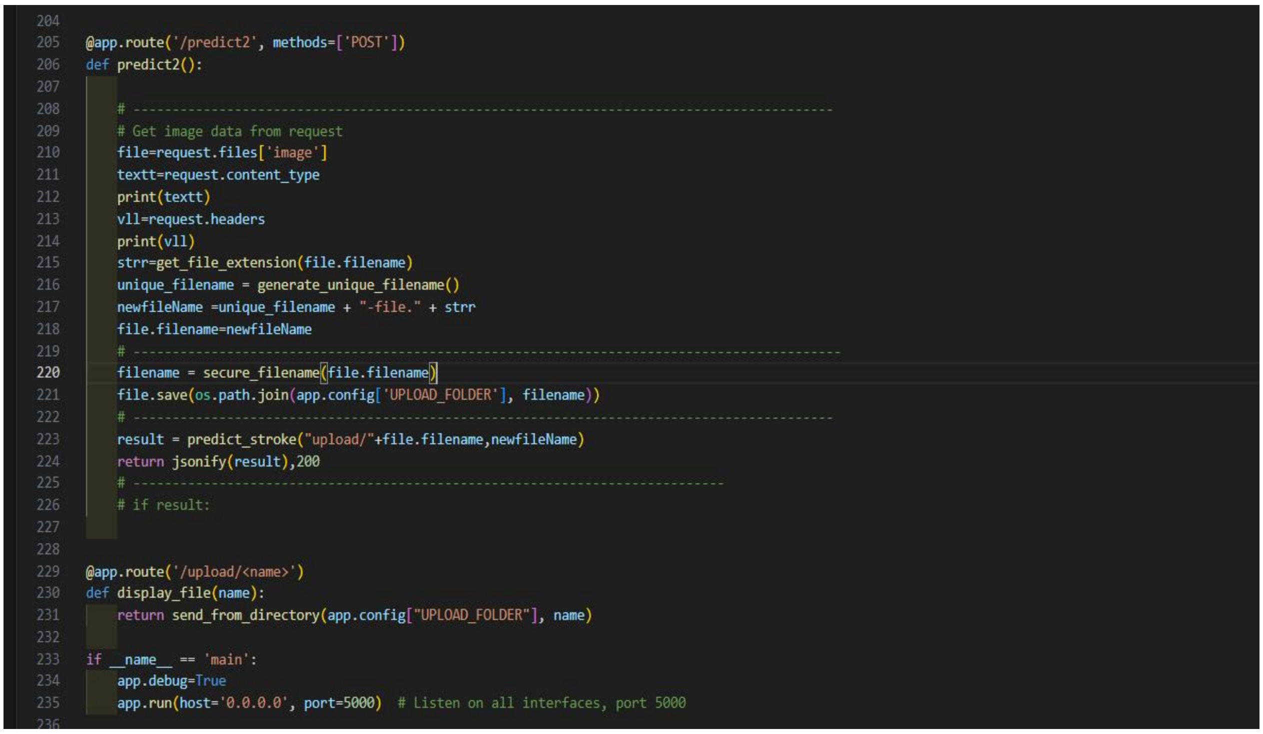
```python
# model2
import tensorflow as tf
import numpy as np
from PIL import Image
# from google.colab import files
import cv2
model = tf.keras.models.load_model('./New_Model1.h5')


#Function to preprocess the image
def preprocess_image(image_path):
    img = Image.open(image_path).resize((140, 140))

    # Convert PIL Image to NumPy array
    img = np.array(img)

    # Convert the image to grayscale using OpenCV
    img_gray = cv2.cvtColor(img, cv2.COLOR_RGB2GRAY)

    # Normalize pixel values
    img_gray = img_gray / 255.0

    # Add batch dimension
    img_gray = np.expand_dims(img_gray, axis=0)

    return img_gray
def predict_stroke(image_path,newfileName):
    img = preprocess_image(image_path)
    prediction = model.predict(img)
    if prediction[0] > 0.5:
        # x=float(result[0][0])
        res="There is brain stroke in you brain"
        return {'canser':res,"No":1,'new file Name':newfileName}
        # return "The image contains a stroke."
    else:
        res="Normal CT Study of brain"
        return {'canser':res,"No":0,'new file Name':newfileName}
        # return "The image does not contain a stroke."
```

Figure 5.32.:API (Application Programming Interface) Code.

- TensorFlow (tf) for loading the machine learning model.
- NumPy (np) for numerical operations.

- PIL (Image) for image manipulation.
- OpenCV (cv2) for image processing.
- Loads a trained Keras model from the specified file (New_Model1.h5).
- Resizes the image to (140, 140) pixels.
- Converts the image to grayscale using OpenCV.
- Normalizes pixel values to the range [0, 1].
- Adds a batch dimension to the image.
- Preprocesses the input image using the preprocess_image function.
- Uses the loaded model to predict whether the image contains a stroke.
- Returns a dictionary indicating the prediction result along with the new filename.
- Checks if the model prediction is greater than 0.5.
- If the prediction is greater, it suggests that the image contains a stroke; otherwise, it suggests the image is normal.

Returns a dictionary containing the prediction result (res), a flag (No) indicating the presence of stroke (1) or absence (0), and the new filename for the uploaded image.



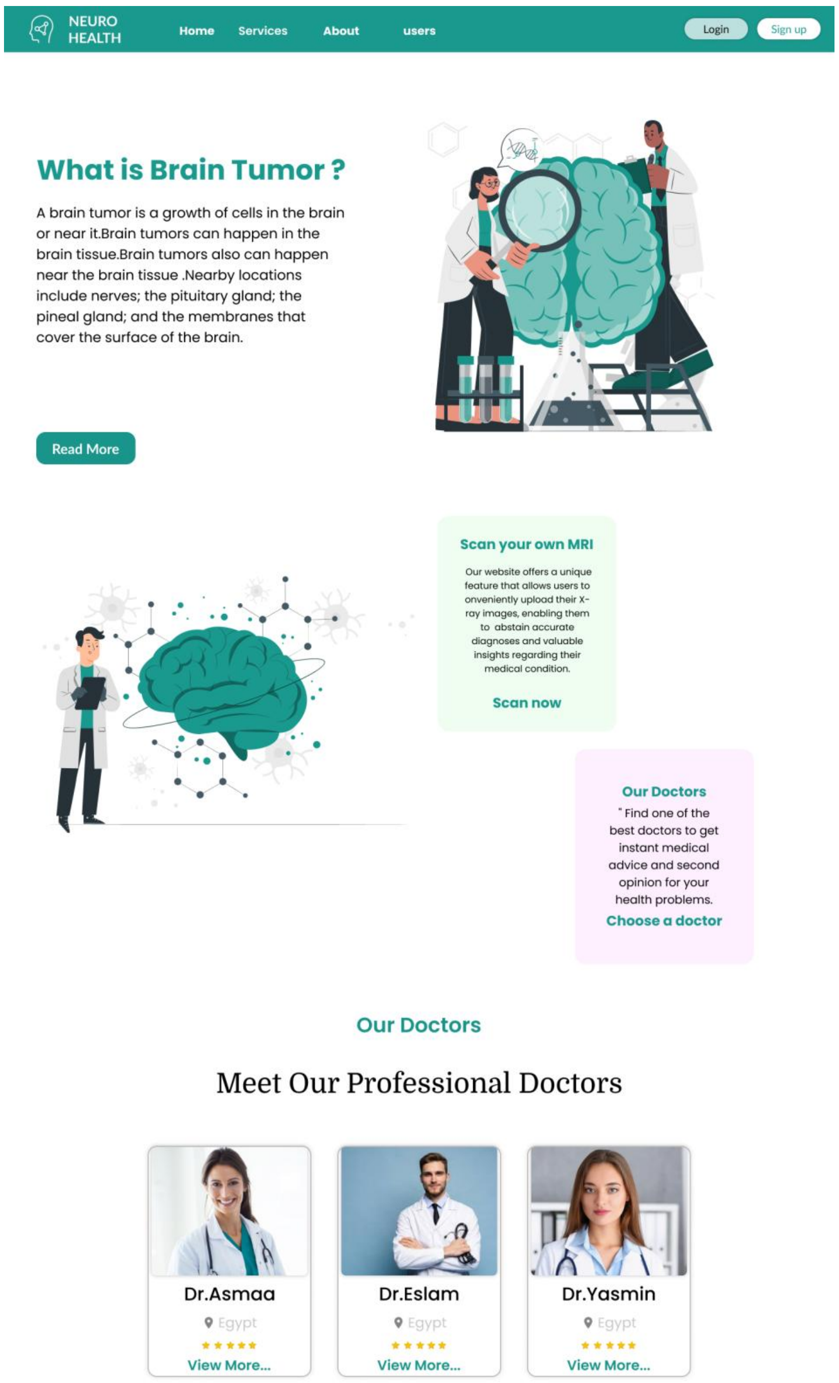
```python

@app.route('/predict2', methods=['POST'])
def predict2():

    # ---------------------------------------------------------------------------------------
    # Get image data from request
    file=request.files['image']
    textt=request.content_type
    print(textt)
    vll=request.headers
    print(vll)
    strr=get_file_extension(file.filename)
    unique_filename = generate_unique_filename()
    newfileName =unique_filename + "-file." + strr
    file.filename=newfileName
    # ---------------------------------------------------------------------------------------
    filename = secure_filename(file.filename)
    file.save(os.path.join(app.config['UPLOAD_FOLDER'], filename))
    # ---------------------------------------------------------------------------------------
    result = predict_stroke("upload/"+file.filename,newfileName)
    return jsonify(result),200
    # -------------------------------------------------------------------------
    # if result:


@app.route('/upload/<name>')
def display_file(name):
    return send_from_directory(app.config["UPLOAD_FOLDER"], name)

if __name__ == 'main':
    app.debug=True
    app.run(host='0.0.0.0', port=5000)  # Listen on all interfaces, port 5000

```

Figure 5.33.:API (Application Programming Interface) Code.

- Defines a new route /predict2 that listens for POST requests.
- Retrieves the image file from the request.
- Collects information about the content type and headers of the request.
- Generates a unique filename for the uploaded image.
- Saves the uploaded image with the generated filename in the upload folder specified in the application's configuration.
- Calls the predict_stroke function to predict whether the image contains a brain stroke.
- Returns the prediction result as a JSON response along with an HTTP status code indicating success.
- Defines a route /upload/<name> to display uploaded files.
- The function display_file retrieves the file with the given filename from the upload folder and sends it to the client.

Runs the Flask application on the specified host and port (0.0.0.0:5000).

❖ Overview of the website



What is Brain Tumor ?

A brain tumor is a growth of cells in the brain or near it.Brain tumors can happen in the brain tissue.Brain tumors also can happen near the brain tissue .Nearby locations include nerves; the pituitary gland; the pineal gland; and the membranes that cover the surface of the brain.

Scan your own MRI

Our website offers a unique feature that allows users to onveniently upload their X-ray images, enabling them to abstain accurate diagnoses and valuable insights regarding their medical condition.

Our Doctors

" Find one of the best doctors to get instant medical advice and second opinion for your health problems.

Our Doctors

Meet Our Professional Doctors

Figure 5.34: Home page.

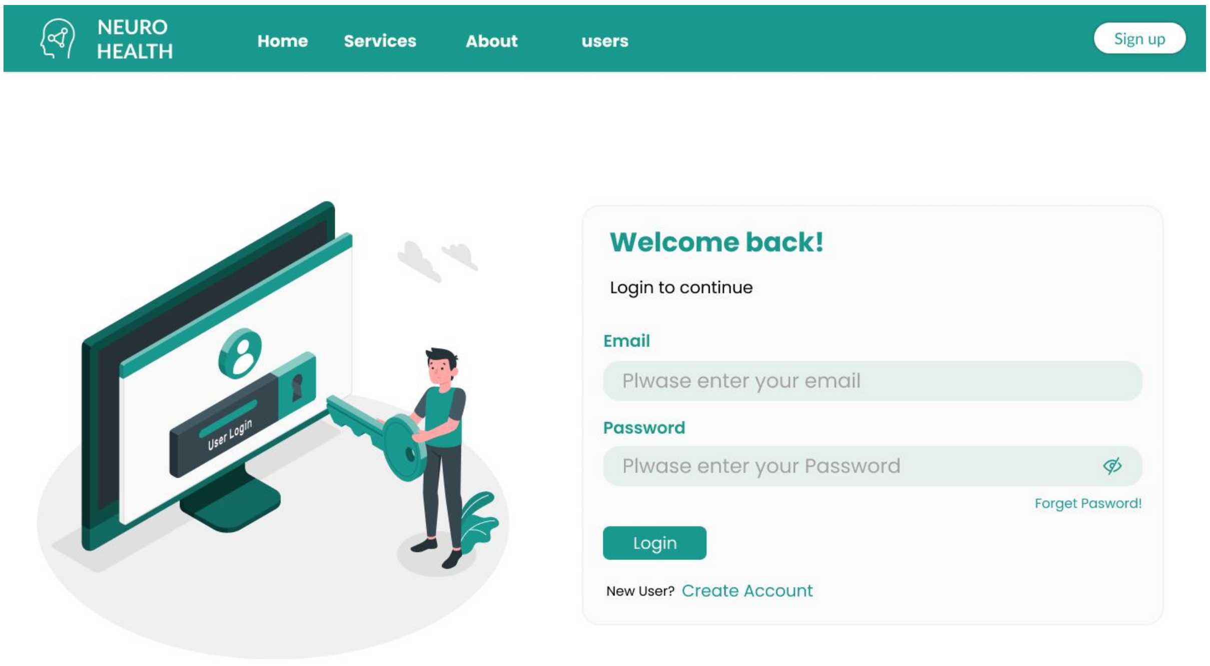
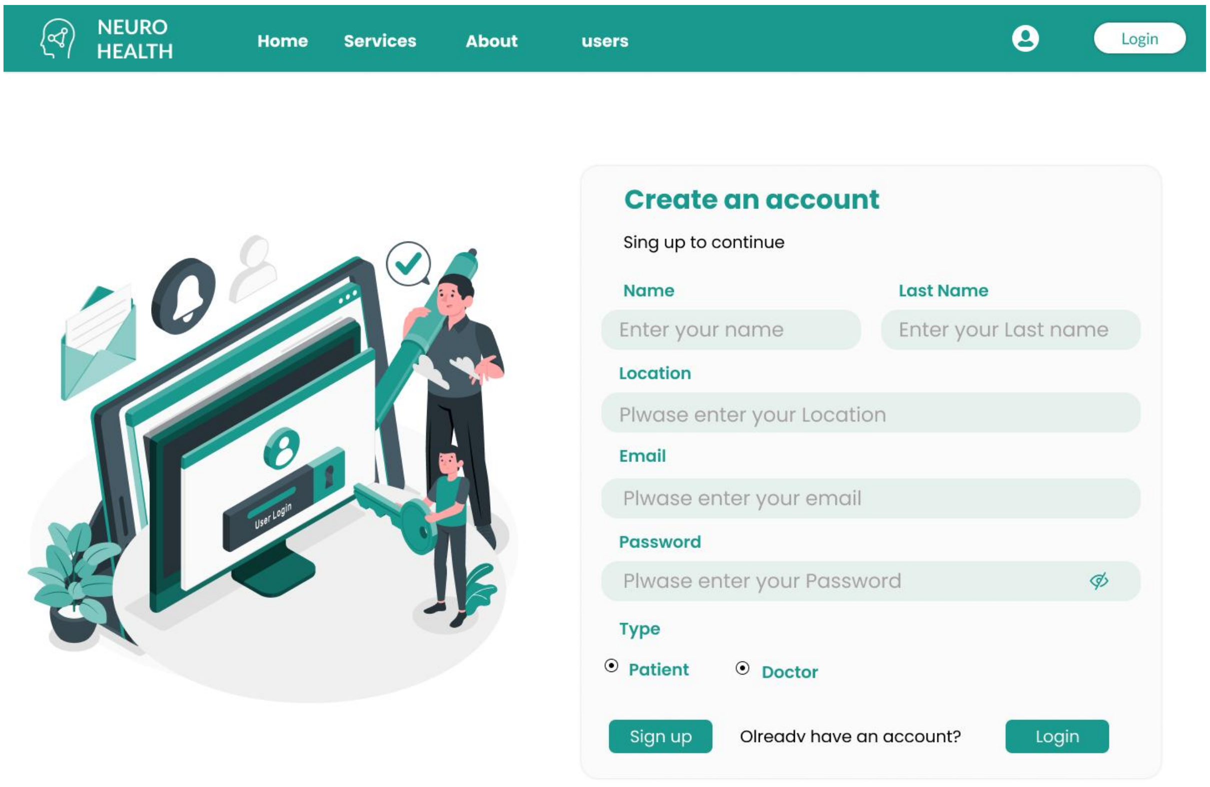


Figure 5.35:Sign in Page.



Create an account

Sing up to continue

Name
Enter your name

Last Name
Enter your Last name

Location
Plwase enter your Location

Email
Plwase enter your email

Password
Plwase enter your Password

Type
◉ Patient ◉ Doctor

Sign up

Olreadv have an account?

Login

Figure 5.36:Sign up Page.

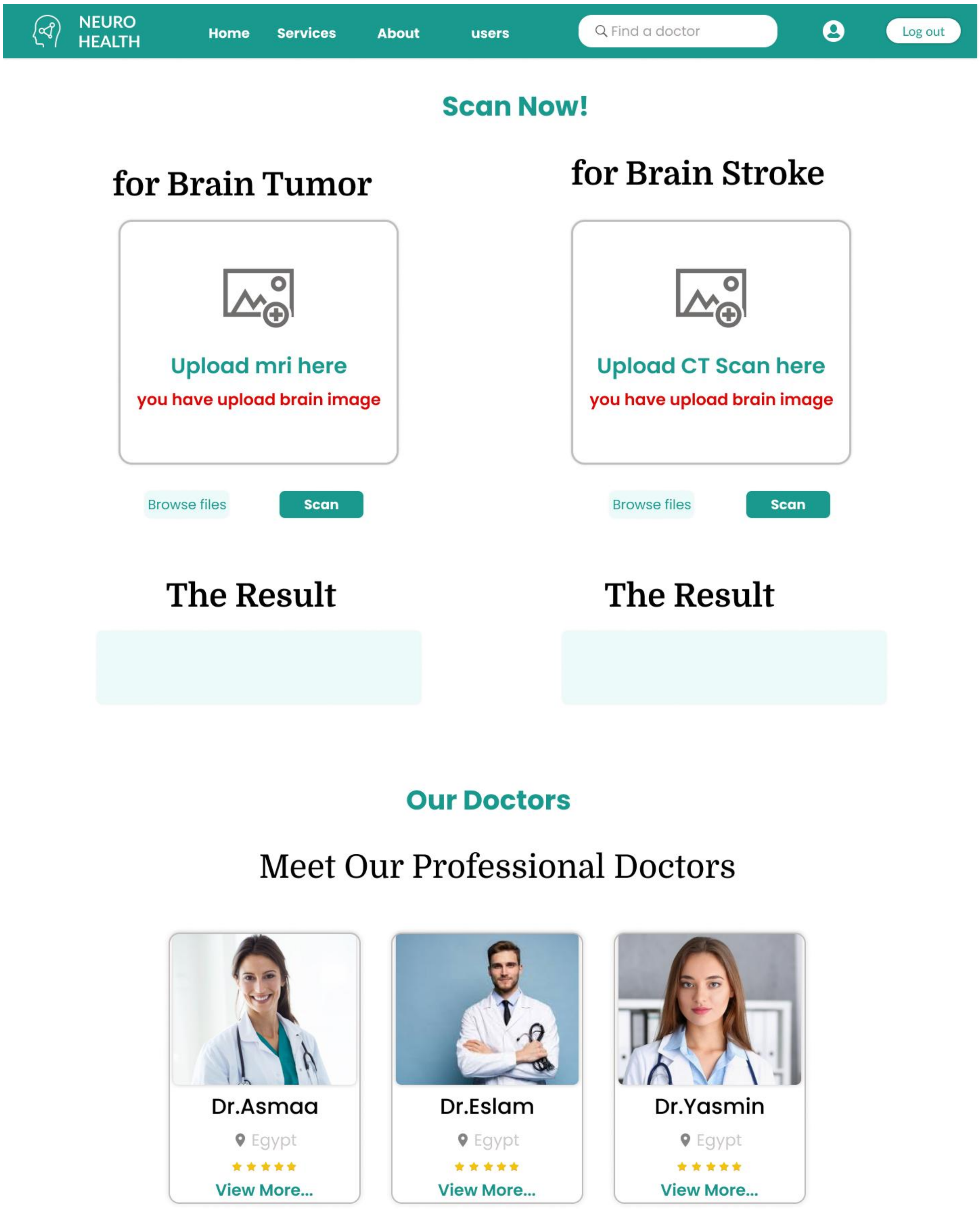
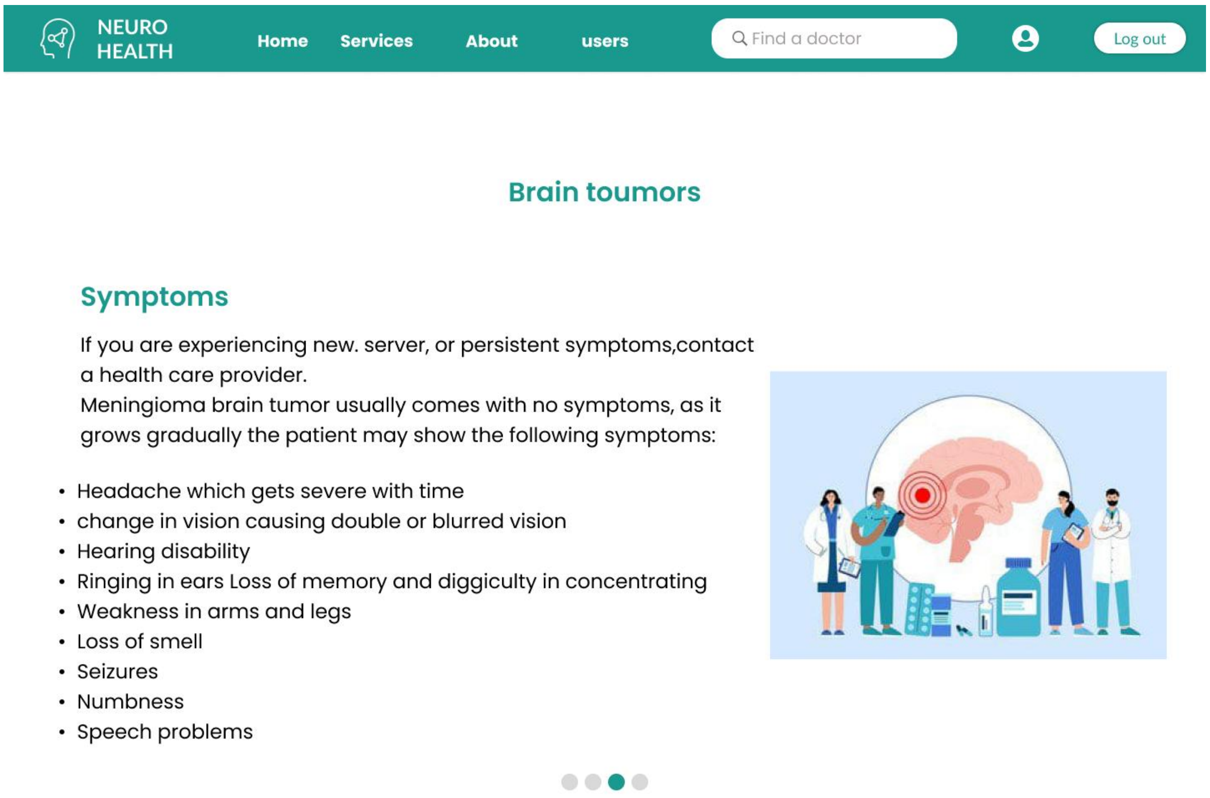


Figure 5.37 :Scan Page.



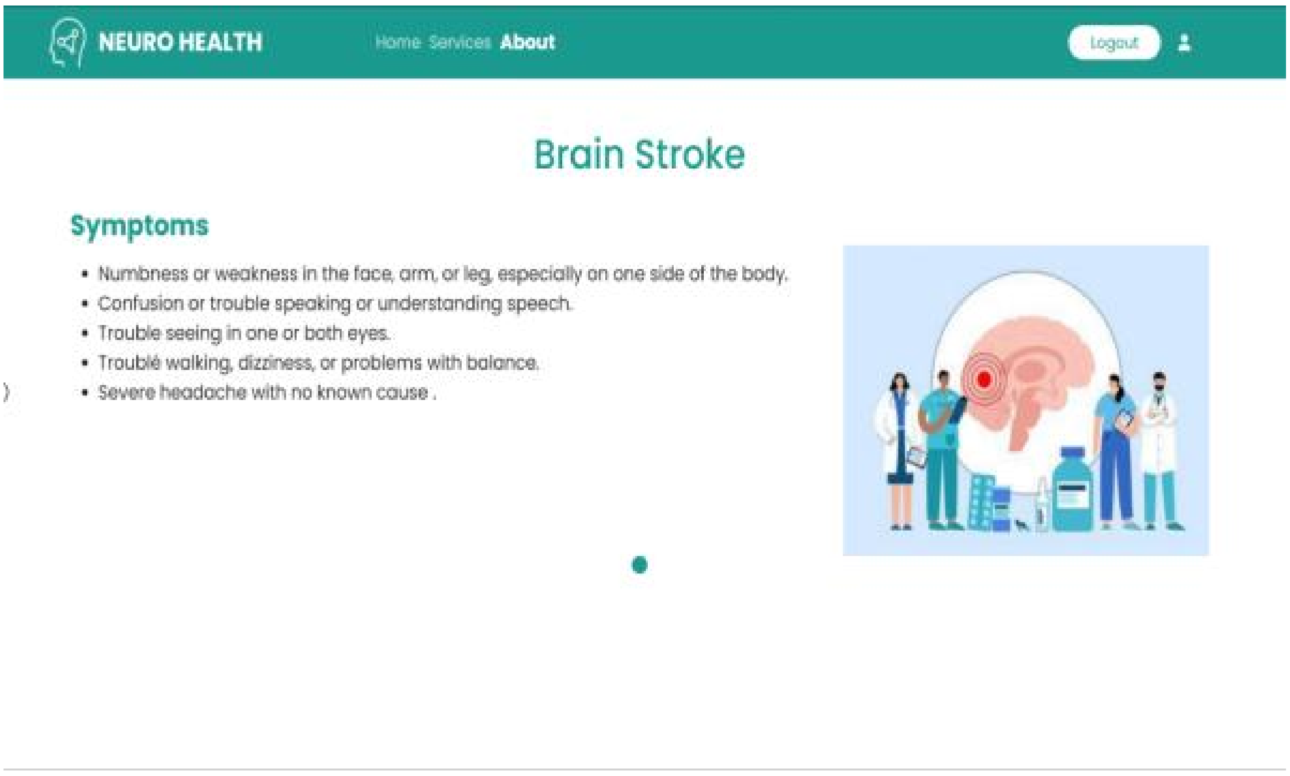
Brain toumors

Symptoms

If you are experiencing new. server, or persistent symptoms,contact a health care provider.
Meningioma brain tumor usually comes with no symptoms, as it grows gradually the patient may show the following symptoms:

- Headache which gets severe with time
- change in vision causing double or blurred vision
- Hearing disability
- Ringing in ears Loss of memory and diggiculty in concentrating
- Weakness in arms and legs
- Loss of smell
- Seizures
- Numbness
- Speech problems

Figure 5.38 : Page about Brain Tumor.

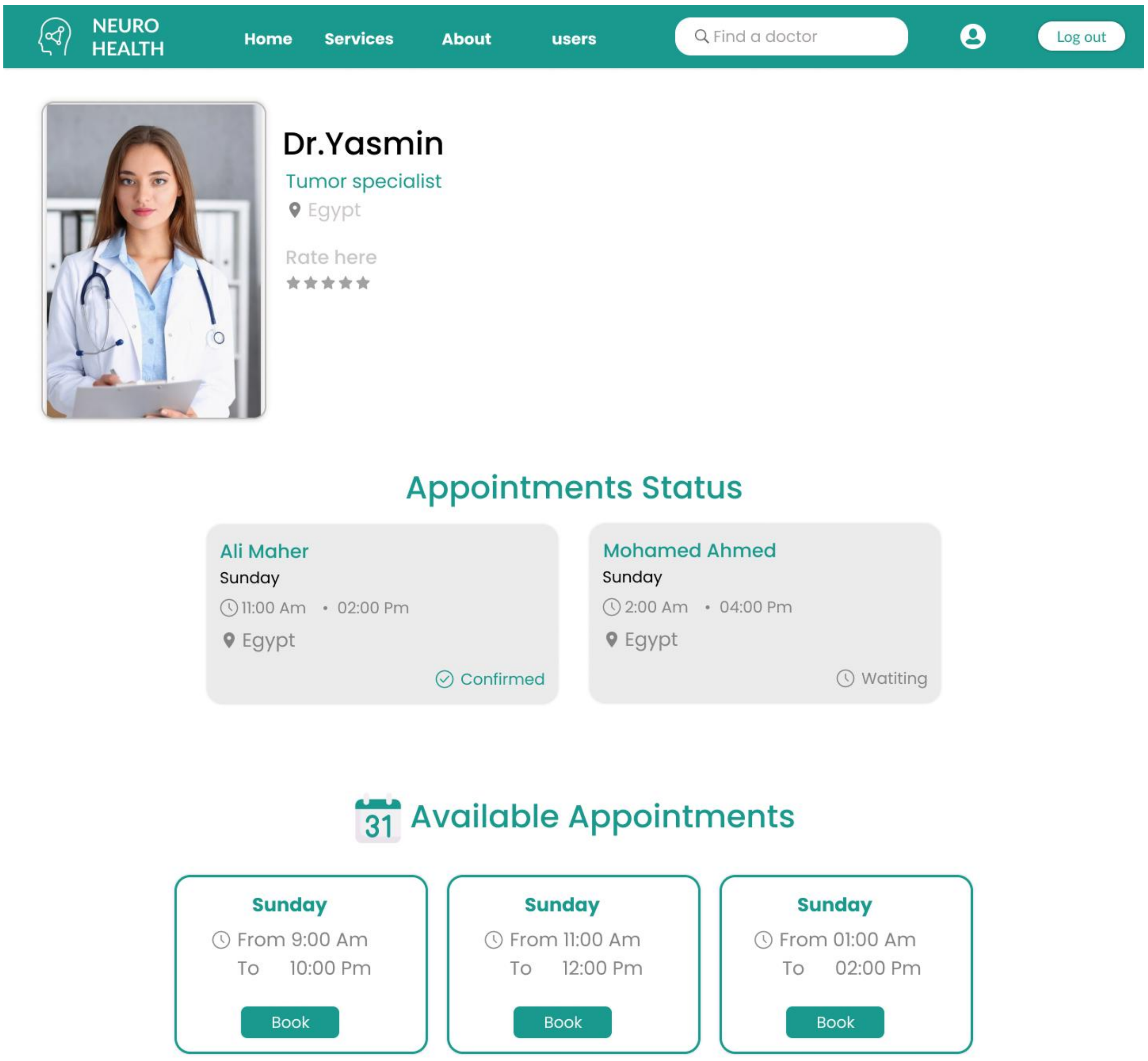


Brain Stroke

Symptoms

- Numbness or weakness in the face, arm, or leg, especially on one side of the body.
- Confusion or trouble speaking or understanding speech.
- Trouble seeing in one or both eyes.
- Trouble walking, dizziness, or problems with balance.
- Severe headache with no known cause .

Figure 5.39 : Page about Brain Stroke.



Dr.Yasmin
Tumor specialist
Egypt
Rate here
★★★★★

Appointments Status

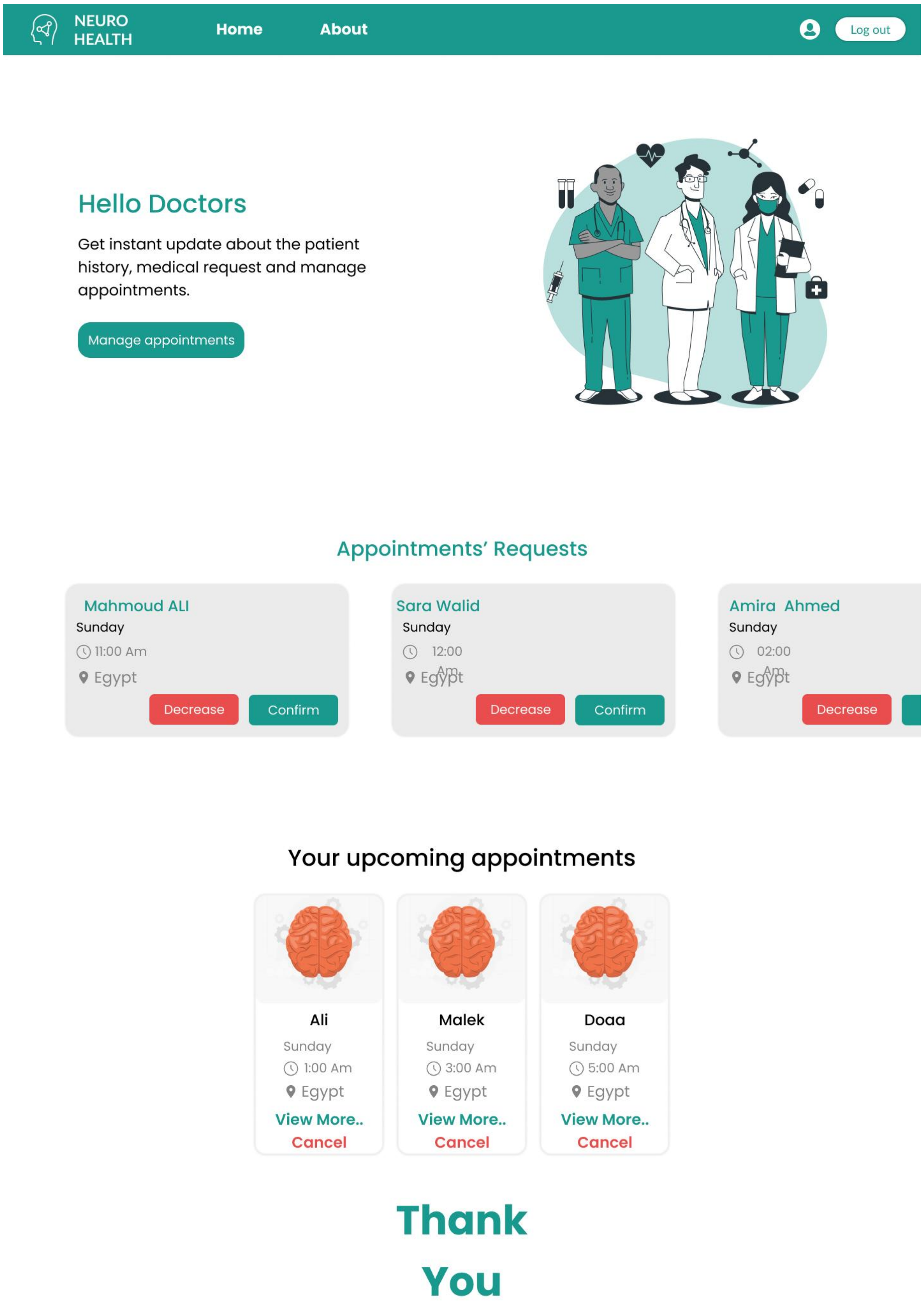
Ali Maher
Sunday
11:00 Am • 02:00 Pm
Egypt
Confirmed

Mohamed Ahmed
Sunday
2:00 Am • 04:00 Pm
Egypt
Watiting

31 Available Appointments

Sunday	Sunday	Sunday
From 9:00 Am	From 11:00 Am	From 01:00 Am
To 10:00 Pm	To 12:00 Pm	To 02:00 Pm
Book	Book	Book

Figure 5.40: the Reservations page.


NEURO HEALTH
Home
About
Log out

Hello Doctors

Get instant update about the patient history, medical request and manage appointments.

Manage appointments

Appointments' Requests

Mahmoud ALI
Sunday
11:00 Am
Egypt
Decrease Confirm

Sara Walid
Sunday
12:00 Am
Egypt
Decrease Confirm

Amira Ahmed
Sunday
02:00 Am
Egypt
Decrease

Your upcoming appointments

Ali
Sunday
1:00 Am
Egypt
View More..
Cancel

Malek
Sunday
3:00 Am
Egypt
View More..
Cancel

Doaa
Sunday
5:00 Am
Egypt
View More..
Cancel

Thank You

Figure 5.41: the Doctor Page.

Chapter6

CONCLUSION AND FUTURE WORK

6.1 CONCLUSION

We researched a variety of pertinent studies on this condition and made an effort to employ the best feature extraction, image pre-processing, and deep learning mechanisms to produce more precise prediction outcomes. In the end, a profound neural system approach is used to Discover brain tumors and brain strokes with greater accuracy and precision. We implemented a computer-aided approach based on CNN for classifying brain Tumor MRI Images and CT scans into tumor or no tumor and stroke or no stroke. Our proposed system has achieved the highest accuracy of 92 % for tumor and 90.43 % for stroke on validation dataset. Once disease is diagnosed .

6.2 FUTURE WORK

Our long-term goal is to Designing a system capable of determining the type of brain tumor Not just identifying the disease
Also we would like to provide an easy system for all users by developing it to a mobile application to becomes available for all users
create a system which able to conducting an accurate analysis of a large medical dataset, which will contain a significant number of cases involving brain tumors and brain strokes. In addition, one of our goals is to improve the algorithm that is advised by include a wider range of picture sizes in a number of different formats, as well as the capacity to determine the kind of tumour. We recommend the development of a machine learning algorithm that uses image processing technology to increase the effectiveness of tumour detection in the human body. This algorithm would reduce the time which needed for accurate diagnosis of tumors and would assist the doctor in determining the treatment that would be most effective for the tumor or stroke.

.

REFERENCES

[1] Tandel, G. S., Biswas, M., Kakde, O. G., Tiwari, A., Suri, H. S., Turk, M., & Suri, J. S., "*A review on a deep learning perspective in brain cancer classification*", Cancers, vol.11, no.1, pp.111, 2019.

[2] Polat, Ö, & Güngen, C., "*Classification of brain tumors from MR images using deep transfer learning*", The Journal of Supercomputing, vol.77, no.7, pp.7236-7252, 2021.

[3] Sjeklocha, L., & Gatz, J. D., "*Traumatic Injuries to the Spinal Cord and Peripheral Nervous System*", Emergency Medicine Clinics, vol.39, no.1, pp.1-28, 2021.

[4] C. Ma, G. Luo, and K. Wang, "*Concatenated and connected random forests with multiscale patch driven active contour model for automated brain tumor segmentation of MR images*," IEEE Trans. Med. Imag., vol. 37, no. 8, pp. 1943–1954, 2018.

[5] Havaei, M., Davy, A., Warde-Farley, D., Biard, A., Courville, A., Bengio, Y., & Larochelle, H., "*Brain tumor segmentation with deep neural networks*", Medical image analysis, vol.35, pp.18-31, 2017.

[6] C.-M. Feng, Y. Xu, J.-X. Liu, Y.-L. Gao, and C.-H. Zheng, "*Supervised discriminative sparse PCA for com-characteristic gene selection and tumor classification on multiview biological data*," IEEE Trans. Neural Netw. Learn. Syst., vol. 30, no. 10, pp. 2926–2937, 2019.

[7] Jain, R. K., Di Tomaso, E., Duda, D. G., Loeffler, J. S., Sorensen, A. G., & Batchelor, T. T., "*Angiogenesis in brain tumours*", Nature Reviews Neuroscience, vol.8, no.8, pp.610-622, 2018.

[8] Goldman, R. D., Cheng, S., & Cochrane, D. D., "*Improving diagnosis of pediatric central nervous system tumours: aiming for early detection*", CMAJ, vol.189, no.12, pp.459-463, 2017.

[9] Sheppard, J. P., Nguyen, T., Alkhalid, Y., Beckett, J. S., Salamon, N., & Yang, I., "*Risk of brain tumor induction from pediatric head CT procedures: a systematic literature review*", Brain tumor research and treatment, vol.6, no.1, pp.1-7, 2018.
[10] Ozdemir, O., Russell, R. L., & Berlin, A. A., "*A 3D probabilistic deep learning system for detection and diagnosis of lung cancer using low-dose CT scans*", IEEE transactions on medical imaging, vol.39, no.5, pp.1419-1429, 2019.

[11] Fatahi, M., & Speck, O., "*Magnetic resonance imaging (MRI): A review of genetic damage investigations*", Mutation Research/Reviews in Mutation Research, vol.764, pp.51-63, 2015.
[12] Roth, H. R., Shen, C., Oda, H., Oda, M., Hayashi, Y., Misawa, K., & Mori, K., "*Deep learning and its application to medical image segmentation*", Medical Imaging Technology, vol.36, no., pp.63-71, 2018.
[13] Beddad, B., Hachemi, K., & Vaidyanathan, S., "*Design and implementation of a new cooperative approach to brain tumour identification from MRI images*", International Journal of Computer Applications in Technology, vol.59, no.1, pp.1-10, 2019.
[14] Anum Masood, Bin Sheng, Po Yang, Ping Li "*Automated Decision Support System for Lung Cancer Detection and Classification via Enhanced RFCN with Multilayer Fusion RPN*", IEEE Transactions on Industrial Informatics, vol.16, no.12, pp.123- 130, 2020.
[15] Xia, Y., Liu, C., Li, Y., & Liu, N., "*A boosted decision tree approach using Bayesian hyper-parameter optimization for*", credit scoring. Expert Systems with Applications, vol.78, pp.225-241, 2017.
[16] Munappy, A., Bosch, J., Olsson, H. H., Arpteg, A., & Brinne, B., "*Data management challenges for deep learning*", In 2019 45th Euromicro Conference on Software Engineering and Advanced Applications (SEAA), pp. 140-147, 2019.
[17] Khan Muhammad; Salman Khan; Javier Del Ser; Victor Hugo C. de Albuquerque. "*Deep Learning for Multigrade Brain Tumor Classification in Smart Healthcare*

Systems: A Prospective Survey", IEEE Transactions on Neural Networks and Learning Systems, vol.32, no.2, pp.208-214, 2021.

[18] J.selvakumar, A.Lakshmi and T.Arivoli, "*Brain Tumor Segmentation and Its Area Calculation in Brain MR Images using K-Mean Clustering and Fuzzy C-Mean Algorithm*", IEEE-International Conference On Advances In Engineering, Science And Management (ICAESM -2012), vol.30, pp.131-142, 2012.

[19] Samir Kumar Bandhyopadhyay and Tuhin Utsab Paul, "*Automatic Segmentation of Brain Tumor from Multiple Images of Brain MRI*" International Journal of Application or Innovation in Engineering & Management (IJAIEM), vol. 2, no. 1, pp.22-35 2013.

[20] Meena, A., & Raja, R., "*Spatial fuzzy c means pet image segmentation of neurodegenerative disorder*", arXiv preprint arXiv: 1303.0647, 2013.

[21] Suman Tatirajua and Avi Mehta, "*Avoiding energy holes in wireless sensor networks with nonuniform node distribution*," IEEE Trans. Parallel Distrib. Syst., vol. 19, no. 5, pp. 710–720, 2018.

[22] Ajala, A., Oke O.A, Adedeji T.O & Alade O.M, "*Fuzzy k-c-means Clustering Algorithm for Medical Image Segmentation*", Journal of Information Engineering and Applications, vol.2, no.6, pp.22-30, 2012

[23] M.H. Fazel Zarandia, M. Zarinbal and M. Izadi, "*Systematic image processing for diagnosing brain tumors*", Department of Industrial Engineering, Amirkabir University of Technology, vol.3, no.3, pp.15875-4413, 2020.

[24] Samarjit Das, "*Pattern Recognition using the Fuzzy c-means Technique*" International Journal of Energy, Information and Communications, vol. 4, no.1, pp.87-66, 2013.

[25] Selvakumar, J., Lakshmi, A., & Arivoli, T., "*Brain tumor segmentation and its area calculation in brain MR images using K-mean clustering and Fuzzy C-mean algorithm*", In IEEE-international conference on advances in engineering, science and management (ICAESM-2012), IEEE, pp. 186-190, 2012.

[26] Krishna Kant Singh1 and Akansha Singh, "*A Study of Image Segmentation*

Algorithms for Different Types of Images", IJCSI International Journal of Computer Science Issues, vol. 7, no. 5, pp.223-231, 2019.

[27] Beshiba Wilson and Julia Punitha Malar Dhas, "*An Experimental Analysis of Fuzzy C-Means and K-Means Segmentation Algorithm for Iron Detection in Brain SWI using Matlab*", International Journal of Computer Applications, vol. 104, no.15, pp.21-33, 2014.

[28] Mircea G. & Mihaela L. "*Tumor Detection and Classification of MRI Brain Image using Different Wavelet Transforms and Support Vector Machines*"42nd International Conference on Telecommunications and Signal Processing (TSP), IEEE, 2019

[29] Ishita Maiti and Monisha Chakraborty" *A new method for brain tumor segmentation based on watershed and edge detection algorithms in HSV colour model*"2012

[30] Riries Rulaningtyas and Khusnul Ain "*Edge Detection For Brain Tumor Pattern Recognition*" International Conference on Instrumentation, Communication, Information Technology, and Biomedical Engineering, vol.9, pp.41-52, 2019.

[31] Nadim Mahmud Dipu and Sifatul Alam Shohan and K. M. A. Salam"*Deep Learning Based Brain Tumor Detection and Classification*"2021 International Conference on Intelligent Technologies (CONIT), vol.4, pp.23-33, 2021.

[32] M. Usman Akram and Anam Usman" *Computer aided system for brain tumor detection and segmentation*"International Conference on Computer Networks and Information Technology, 15 September 2011.

[33] R. Tamilselvi and A. Nagaraj and M. Parisa Beham and M.Bharkavi Sandhiya "*BRAMSIT: A Database for Brain Tumor Diagnosis and Detection*" 2020 Sixth International Conference on Bio Signals, Images, and Instrumentation (ICBSII), vol.26, no.8, pp.112-119, 2020.

[34] Atiq Islam and Syed M. S. Reza and Khan M. Iftekharuddin "*Multifractal Texture Estimation for Detection and Segmentation of Brain Tumors*" IEEE Transactions on Biomedical Engineering, Vol.60, no.11, pp.1123-1133, 2013.
[35] Andac Hamamci and Nadir Kucuk;Kutlay Karaman and Kayihan Engin and Gozde Unal " *Tumor-Cut: Segmentation of Brain Tumors on Contrast Enhanced MR Images for Radiosurgery Applications*" IEEE Transactions on Medical Imaging, vol.31, no.3, pp. 112-119, 2017.
[36] Neelum Noreen and Sellappan Palaniappan and Abdul Qayyum and Iftikhar Ahmad and Muhammad Imran and Muhammad Shoaib "*A Deep Learning Model Based on Concatenation Approach for the Diagnosis of Brain Tumor*", IEEE Access, vol.8, no.5, 2020.
[37] Ming Li and Lishan Kuang and Shuhua Xu and Zhanguo Sha "*Brain Tumor Detection Based on Multimodal Information Fusion and Convolutional Neural Network*" IEEE Access, vol.12, no.0, pp.180134 – 180146, 2019.
[38] T. Uchiyama and K. Mohri and M. Shinkai and A. Ohshima and H. Honda and T. Kobayashi and T. Wakabayashi and J. Yoshida "*Position sensing of magnetite gel using MI sensor for brain tumor detection*", IEEE Transactions on Magnetics, vol. 121, pp.4266 – 4268, 2020.
[39] Qiaobo Hao and Yu Pei and Rong Zhou and Bin Sun and Jun Sun and Shutao Li and Xudong Kang "*Fusing Multiple Deep Models for In Vivo Human Brain Hyperspectral Image Classification to Identify Glioblastoma Tumor*", IEEE Transactions on Instrumentation and Measurement, vol.70, pp.2223-2233, 2021.
[40] Amran Hossain and Mohammad Tariqul Islam and Mohammad Shahidul Islam and Muhammad E. H. Chowdhury and Ali F. Almutairi and Qutaiba A. Razouqi;Norbahiah Misran*" A YOLOv3 Deep Neural Network Model to Detect Brain Tumor in Portable Electromagnetic Imaging System*" IEEE Access , vol.9, pp.1232-1240, 2021.

[41] Seong-Ik Kim and Kwanghoon Lee and Jae-Kyung Won, "*Revisiting vimentin: a negative surrogate marker of molecularly defined oligodendroglioma in adult type diffuse glioma*", Brain Tumor Pathology, vol.38, pp.271–282, 2021.
[42] Eiichi Ishikawa, and Tsubasa Miyazaki, and Shingo Takano and Hiroyoshi Akutsu "*Anti-angiogenic and macrophage-based therapeutic strategies for glioma immunotherapy*", Brain Tumor Pathology, vol.38, pp.149–155, 2021.
[43] Rajinikanth, V.; Fernandes, S.L.; Bhushan, B.; Harisha. Sunder, N.R. Segmentation and Analysis of Brain Tumor Using Tsallis Entropy and Regularised Level Set. In Proceedings of 2nd International Conference on Micro-Electronics, Electromagnetics and Telecommunications; Springer Singapore, 7 September 2017; pp. 313–321.
[44] Almahfud, M.A.; Setyawan, R.; Sari, C.A.; Setiadi, D.R.I.M.; Rachmawanto, E.H. An Effective MRI Brain Image Segmentation using Joint Clustering (K-Means and Fuzzy C-Means). In Proceedings of the 2018 International Seminar on Research of Information Technology and Intelligent Systems (ISRITI), Yogyakarta, Indonesia, 21-22 Nov. 2018, pp. 11-16.
[45] Kaur, N.; Sharma, M. Brain tumor detection using self-adaptive K-means clustering. In Proceedings of the 2017 International Conference on Energy, Communication, Data Analytics and Soft Computing (ICECDS),Chennai, India, 1–2 August 2017, pp. 1861–1865.
[46] Bal, A.; Banerjee, M.; Sharma, P.; Maitra, M. Brain Tumor Segmentation on MR Image Using K-Means and Fuzzy-Possibilistic Clustering. In Proceedings of the 2018 2nd International Conference on Electronics, Materials Engineering & NanoTechnology (IEMENTech), Kolkata, India, 4–5 April 2018.
[47] Shanker, R.; Singh, R.; Bhattacharya, M. Segmentation of tumor and edema based on K-mean clustering and hierarchical centroid shape descriptor. In Proceedings of the 2017 IEEE International Conference on Bioinformatics and Biomedicine (BIBM), Kansas City, MO, USA, 13–16 November 2017, pp. 1105–1109.

[48] Mahmud, M.R.; Mamun, M.A.; Hossain, M.A.; Uddin, M.P. Comparative Analysis of K-Means and Bisecting K-Means Algorithms for Brain Tumor Detection. In Proceedings of the 2018 International Conference on Computer, Communication, Chemical, Material and Electronic Engineering (IC4ME2), Rajshahi, Bangladesh, 8– 9 February 2018, pp.1–4.
[49] Suraj, N.S.S.K.; Muppalla, V.; Sanghani, P.; Ren, H. Comparative Study of Unsupervised Segmentation Algorithms for Delineating Glioblastoma Multiforme Tumour. In Proceedings of the 2018 3rd International Conference on Advanced Robotics and Mechatronics (ICARM), Singapore, 18–20 July 2018, pp. 468–473.
[50] Rundo, L.; Militello, C.; Tangherloni, A.; Russo, G.; Vitabile, S.; Gilardi, M.C.; Mauri, G. NeXt for neuro-radiosurgery: A fully automatic approach for necrosis extraction in brain tumor MRI using an unsupervised machine learning technique. *Int. J. Imaging Syst. Technol.* **2017**, *28*, 21–37.
[51] Abdel-Maksoud, E.; Elmogy, M.; Al-Awadi, R. Brain tumor segmentation based on a hybrid clustering technique. Egypt. Informatics J. 2015, 16, 71–81.
[52] Bera, K., Schalper, K. A., Rimm, D. L., Velcheti, V., & Madabhushi, A., "*Artificial intelligence in digital pathology—new tools for diagnosis and precision oncology*", Nature reviews Clinical oncology, vol.16, no.11, pp.703-715, 2019.
[53] Chen, M., Hao, Y., Hwang, K., Wang, L., & Wang, L., "*Disease prediction by machine learning over big data from healthcare communities*", IEEE Access, vol.5, pp.8869-8879, 2017.
[54] Mohapatra, H., & Rath, A. K., "*Fundamentals of software engineering: designed to provide an insight into the software engineering concepts*", BPB Publications, 2020.
[55] Bittner, K., & Spence, I., "*Use case modeling*", Addison-Wesley Professional, 2009.
[56] Mishra, S. K., & Deepthi, V. H., "*Brain image classification by the combination of different wavelet transforms and support vector machine*

classification", Journal of Ambient Intelligence and Humanized Computing, vol.12, no.6, pp.6741-6749, 2021.
[57] Mellor, S. J., Mellor, S., & Balcer, M. J., "*Executable UML: a foundation for modeldriven architecture*", Addison-Wesley Professional, 2008.