# Brain-Tumor-Detection-Web-app

A Project Report

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

# Bachelor of Technology in

Department of Computer Science and Engineering

By

2010030024 – Mr. B VEGESH SAI

2010030031 – Mr. CH VARUN

2010030083 – Mr. K SAI ANIRUDH

2010030452 – Mr. FAUZAAN

Under the esteemed guidance of

# Dr. ARPITA GUPTA



Department of Computer Science and Engineering

K L University Hyderabad,

Aziz Nagar, Moinabad Road, Hyderabad – 500 075, Telangana, India.

March, 2022

**DECLARATION**

# The Project Report entitled “Brain-Tumor-Detection-Web-app” is a record of bonafide work of Mr. D Haranath (2010030044), Mr. K Pranay (2010030082), Mr. R V Rahul Krishna (2010030137) and Mr. S Rohit Jaiswal (2010030141) submitted in partial fulfillment for the award of B.Tech in the Department of Computer Science and Engineering to the K L University, Hyderabad. The results embodied in this report have not been copied from any other Departments/University/Institute.

2010030024 – Mr. B VEGESH SAI

2010030031 – Mr. CH VARUN

2010030083 – Mr. K SAI ANIRUDH

2010030452 – Mr. FAUZAAN

**CERTIFICATE**

# This is to certify that the Project Report entitled “Brain-Tumor-Detection-Web-app” is being submitted by Mr. B VEGESH SAI bearing Regd. No. 2010030024, Mr. CH VARUN bearing Regd. No. 2010030031, Mr. K SAI ANIRUDH bearing Regd. No. 2010030083 and Mr. FAUZAAN bearing Regd. No. 2010030452 submitted in partial fulfillment for the award of B.Tech in Computer Science and Engineering to the K L University, Hyderabad is a record of bonafide work carried out under our guidance and supervision.

The results embodied in this report have not been copied from any other department/ University/ Institute.

## Signature of the Supervisor

Dr. ARPITA GUPTA

(Associate Professor)

## Signature of the HOD Signature of the External Examination

**ACKNOWLEDGEMENT**

It is great pleasure for us to express our gratitude to our honorable President

**Sri. Koneru Satyanarayana**, for giving the opportunity and platform with facilities in accomplishing the project-based laboratory report.

We express our sincere gratitude to our Principal **Dr. L. Koteswara Rao** for his administration towards our academic growth.

We express our sincere thanks to HOD-CSE **Dr. M Chiranjeevi** for his leadership and constant motivation provided in successful completion of our academic semester. I record it as my privilege to deeply thank for providing us the efficient faculty and facilities to make our ideas into reality.

We express immense gratitude to our guide and Associate Professor **Dr. Arpita Gupta** for her novel association of ideas, encouragement, appreciation and intellectual zeal which motivated us to venture this project successfully.

Finally, it is pleased to acknowledge the indebtedness to all those who devoted themselves directly or indirectly to make this project report success.

**ABSTRACT**

In the recent times, the requirement for better and advanced healthcare facilities has grown to a very large extent. With this Pandemic, we not only need better healthcare facilities to handle Covid-19 but we also need advanced methods to make other healthcare facilities and diagnostics faster. With Machine learning and Artificial Intelligence taking over almost everything, we decided to tackle one such issue related to the health sector using Deep learning Techniques.

This is our Solution for Data Science and AI . A web app that helps in diagnosing whether a person has Brain Tumor or not by taking the MRI scan as the input from the user.

**INDEX**

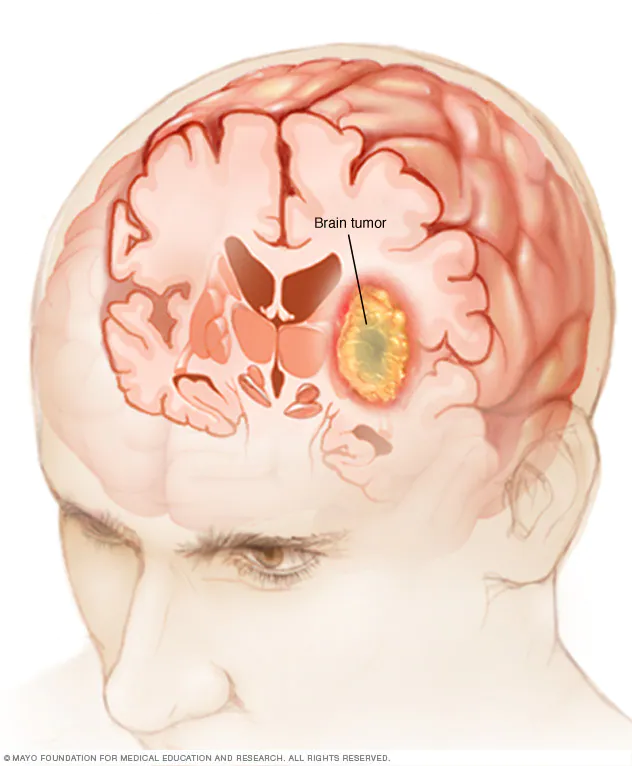
|  |  |  |
| --- | --- | --- |
| Chapter No. | Title | Page No. |
| 1. | Introduction | 1 |
| 1.1. | Convolutional Neural Network | 1 |
| 2. | Literature survey | 3-4 |
| 3. | Methodology | 5 |
| 3.1. | Image Segmentation | 5 |
| 3.2. | Otsu’s Thresholding | 6 |
| 3.3. | Gaussian Blur | 7-8 |
| 4. | System Requirements | 9 |
| 4.1. | Software requirements | 9 |
| 4.2. | Hardware requirements | 9 |
| 5. | Flowchart | 10 |
| 6. | Implementation | 11 |
| 7. | Git Setup | 17-19 |
| 8. | Dataset | 20 |
| 9. | Conclusion & Future work | 21 |
| 10. | References | 22 |

1. **INTRODUCTION**

A brain tumor is a mass or growth of abnormal cells in your brain.

Many different types of brain tumors exist. Some brain tumors are noncancerous (benign), and some brain tumors are cancerous (malignant). Brain tumors can begin in your brain (primary brain tumors), or cancer can begin in other parts of your body and spread to your brain as secondary (metastatic) brain tumors.

A brain tumor can vary growth greatly. The growth rate as well as the location of a brain tumor determines how it will affect the function of your nervous system.



### **1.1. Convolutional Neural Networks**

CNN is an advanced and high-potential type of the classic artificial neural network model. It is built for tackling higher complexity, pre-processing, and data compilation. It takes reference from the order of arrangement of neurons present in the visual cortex of an animal brain.

The CNNs can be considered as one of the most efficiently flexible models for specializing in image as well as non-image data. These have four different organizations:

* It is made up of a single input layer, which generally is a two-dimensional arrangement of neurons for analysing primary image data, which is similar to that of photo pixels.
* Some CNNs also consist of a single-dimensional output layer of neurons that processes images on their inputs, via the scattered connected convolutional layers.
* The CNNs also have the presence of a third layer known as the sampling layer to limit the number of neurons involved in the corresponding network layers.
* Overall, CNNs have single or multiple connected layers that connect the sampling to output layers.

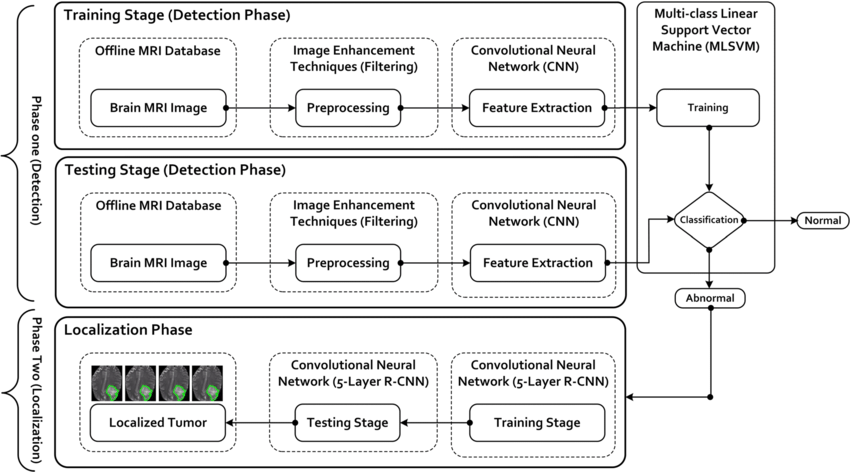
This network model can help derive relevant image data in the form of smaller units or chunks. The neurons present in the convolution layers are accountable for the cluster of neurons in the previous layer.

Once the input data is imported into the convolutional model, there are four stages involved in building the CNN:

* **Convolution:**The process derives feature maps from input data, followed by a function applied to these maps.
* **Max-Pooling:**It helps CNN to detect an image based on given modifications.
* **Flattening:**In this stage, the data generated is then flattened for a CNN to analyze.
* **Full Connection:**It is often described as a hidden layer that compiles the loss function for a model.

The CNNs are adequate for tasks, including image recognition, image analyzing, image segmentation, video analysis, and natural language processing. However, there can be other scenarios where CNN networks can prove to be useful like:

* Image datasets containing OCR document analysis
* Any two-dimensional input data which can be further transformed to one-dimensional for quicker analysis
* The model needs to be involved in its architecture to yield output.



1. **LITREARURE SURVEY**

Table

Description automatically generated

Table

Description automatically generated with medium confidence

Table

Description automatically generated

**3. METHODOLOGY**

**3.1. Image Segmentation**

Image Segmentation is the process by which a digital image is partitioned into various subgroups (of pixels) called Image Objects, which can reduce the complexity of the image, and thus analyzing the image becomes simpler.

We use various image segmentation algorithms to split and group a certain set of pixels together from the image. By doing so, we are actually assigning labels to pixels and the pixels with the same label fall under a category where they have some or the other thing common in them.

The concept of partitioning, dividing, fetching, and then labeling and later using that information to train various ML models have indeed addressed numerous business problems. In this section, let’s try to understand what problems are solved by Image Segmentation.

Image segmentation is a promising set of skills from Deep Learning as it has an important role to play in Medical Imaging.

**3.2. Otsu’s Thresholding**

[Image segmentation](https://learnopencv.com/image-segmentation/) refers to the class of algorithms that partition the image into different segments or groups of pixels. In that sense, image thresholding is the simplest kind of image segmentation because it partitions the image into two groups of pixels — white for foreground, and black for background. The core idea is separating the image histogram into two clusters with a threshold defined as a result of minimization the weighted variance of these classes denoted by  \sigma^{2}_w(t).

The whole computation equation can be described as:

\sigma^{2}_w(t) = w_1(t)\sigma^{2}_1(t) + w_2(t)\sigma^{2}_2(t),

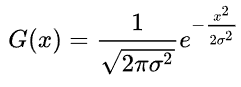
where w_1(t), w_2(t)  are the probabilities of the two classes divided by a threshold t, and the value is within the range from 0 to 255 inclusively.

**3.3. Gaussian Blur**

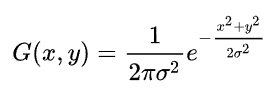
In [image processing](https://en.wikipedia.org/wiki/Image_processing), a Gaussian blur (also known as Gaussian smoothing) is the result of blurring an [image](https://en.wikipedia.org/wiki/Image) by a [Gaussian function](https://en.wikipedia.org/wiki/Gaussian_function) (named after mathematician and scientist [Carl Friedrich Gauss](https://en.wikipedia.org/wiki/Carl_Friedrich_Gauss)). It is a widely used effect in graphics software, typically to reduce [image noise](https://en.wikipedia.org/wiki/Image_noise) and reduce detail. The visual effect of this blurring technique is a smooth blur resembling that of viewing the image through a translucent screen, distinctly different from the [bokeh](https://en.wikipedia.org/wiki/Bokeh) effect produced by an out-of-focus lens or the shadow of an object under usual illumination.

Gaussian smoothing is also used as a pre-processing stage in [computer vision](https://en.wikipedia.org/wiki/Computer_vision) algorithms in order to enhance image structures at different scales—see [scale space representation](https://en.wikipedia.org/wiki/Scale_space_representation) and [scale space implementation](https://en.wikipedia.org/wiki/Scale_space_implementation). In Gaussian Blur operation, the image is convolved with a Gaussian filter instead of the box filter. The Gaussian filter is a low-pass filter that removes the high-frequency components are reduced.

The Gaussian blur is a type of image-blurring filter that uses a Gaussian function (which also expresses the [normal distribution](https://en.wikipedia.org/wiki/Normal_distribution) in statistics) for calculating the [transformation](https://en.wikipedia.org/wiki/Transformation_(mathematics)) to apply to each [pixel](https://en.wikipedia.org/wiki/Pixel) in the image. The formula of a Gaussian function in one dimension is



In two dimensions, it is the product of two such Gaussian functions, one in each dimension:



where x is the distance from the origin in the horizontal axis, y is the distance from the origin in the vertical axis, and σ is the [standard deviation](https://en.wikipedia.org/wiki/Standard_deviation) of the Gaussian distribution. When applied in two dimensions, this formula produces a surface whose [contours](https://en.wiktionary.org/wiki/contour) are [concentric circles](https://en.wikipedia.org/wiki/Concentric_circles) with a Gaussian distribution from the center point.

**4. SYSTEM REQUIREMENTS**

* 1. **Software requirements:**

The major software requirements of the project are as follows:

Language : Python

Operating system : Windows 10

Tools : Jupyter Notebook, Kaggle Datasets,VSC

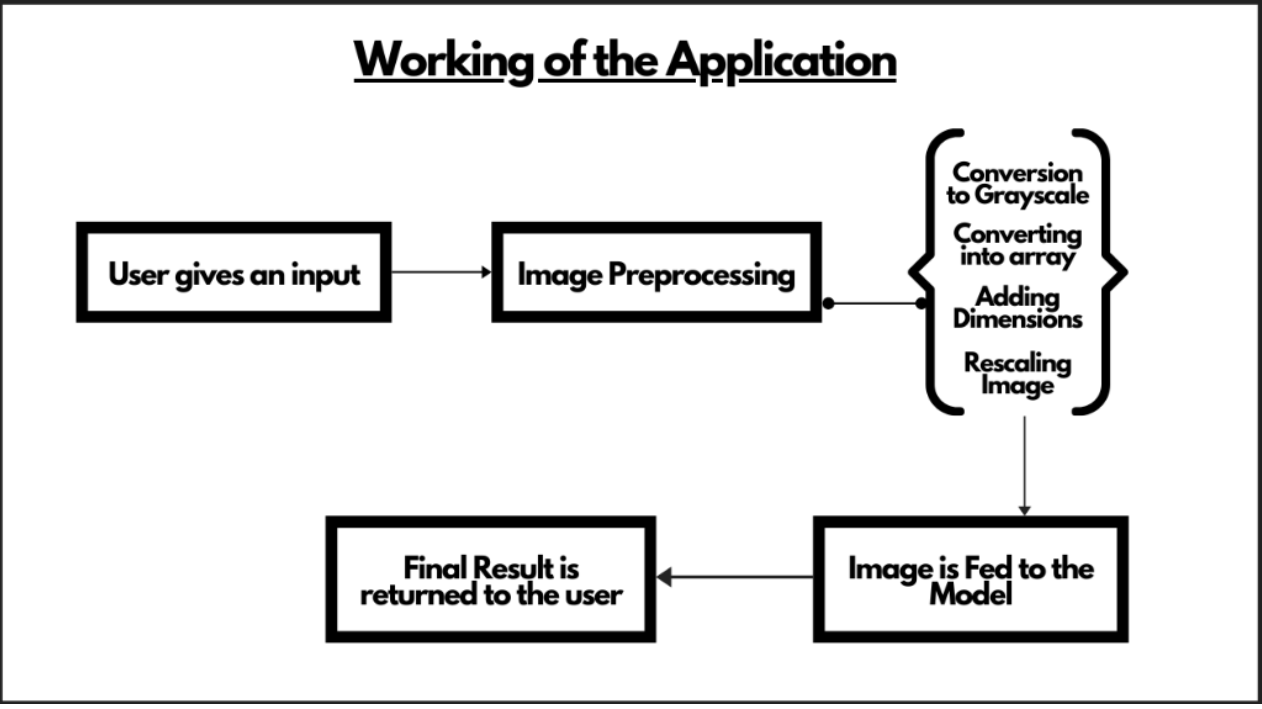
* 1. **Hardware requirements:**

The hardware requirements that map towards the software are as follows:

RAM : 4.00 GB

Processor : Intel(R) Core(TM) i5-4210U CPU @ 1.70GHz 1.70 GHz

1. **FLOWCHART**



1. **IMPLEMENTATION**
2. import numpy as np
3. import pandas as pd
4. import os
5. import tensorflow as tf
6. import matplotlib.pyplot as plt
7. import matplotlib.image as mpimg
8. from tensorflow.keras.preprocessing.image import ImageDataGenerator
9. from tensorflow.keras.optimizers import RMSprop,Adam
10. import cv2

DATA = r"D:/HackOff---Brain-Tumor-Detection-Web-app-main/archive/Training" #reading the data

CATEGORIES = ["glioma\_tumor","meningioma\_tumor","no\_tumor","pituitary\_tumor"] #defining the 4 categories that we have

for category in CATEGORIES:

    path = os.path.join(DATA,category)

    for img in os.listdir(path):

        img\_array = cv2.imread(os.path.join(path,img))

        plt.imshow(img\_array)

        plt.show()

        plt.axis("off")

        break

    break

IMG\_SIZE = 150 #defining our image size

new\_array = cv2.resize(img\_array,(IMG\_SIZE,IMG\_SIZE))#scaling down our images

plt.imshow(new\_array,cmap = "gray")

plt.axis("off")

**Background pattern

Description automatically generated**

training\_data = [] #manipulating our training data

def create\_training\_data():

    for category in CATEGORIES:

        path = os.path.join(DATA,category)

        class\_num = CATEGORIES.index(category) #defining the different categories of the images in our data

        for img in os.listdir(path):

            try:

                img\_array = cv2.imread(os.path.join(path,img),cv2.IMREAD\_GRAYSCALE) #loading the images in grayscale

                new\_array = cv2.resize(img\_array,(IMG\_SIZE,IMG\_SIZE))

                training\_data.append([new\_array,class\_num]) #adding our data in to the training\_data list which we will use to define our X and y for train-tets split

            except Exception as e:

                pass

create\_training\_data()

X = [] #used for storing  the features

y = [] #used for storing the labels

for features,label in training\_data:

    X.append(features)

    y.append(label)

X = np.array(X).reshape(-1,IMG\_SIZE,IMG\_SIZE)

#print(X.shape)

X = X/255.0

X = X.reshape(-1,150,150,1)

print(X.shape)

#defining our model

model = tf.keras.models.Sequential([

    tf.keras.layers.Conv2D(64, (3,3), activation='relu',padding = 'Same', input\_shape=(150, 150, 1)),

    tf.keras.layers.MaxPooling2D(2, 2),

    tf.keras.layers.Dropout(0.25),

    tf.keras.layers.Conv2D(128, (3,3), activation='relu',padding = 'Same'),

    tf.keras.layers.MaxPooling2D(2,2),

    tf.keras.layers.Dropout(0.25),

    tf.keras.layers.Conv2D(128, (3,3), activation='relu',padding = 'Same'),

    tf.keras.layers.MaxPooling2D(2,2),

    tf.keras.layers.Dropout(0.25),

    tf.keras.layers.Conv2D(128, (3,3), activation='relu',padding = 'Same'),

    tf.keras.layers.MaxPooling2D(2,2),

    tf.keras.layers.Dropout(0.25),

    tf.keras.layers.Conv2D(256, (3,3), activation='relu',padding = 'Same'),

    tf.keras.layers.MaxPooling2D(2,2),

    tf.keras.layers.Dropout(0.25),

    tf.keras.layers.Flatten(),

    tf.keras.layers.Dense(1024, activation='relu'),

    tf.keras.layers.Dropout(0.5),

    tf.keras.layers.Dense(4, activation='softmax')

])

optimizer = Adam(lr=0.001)

model.compile(loss='categorical\_crossentropy',

              optimizer = optimizer,

              metrics=['accuracy'])

epochs = 50

batch\_size = 40

datagen = ImageDataGenerator(

        rotation\_range=0,

        zoom\_range = 0,

        width\_shift\_range=0,

        height\_shift\_range=0,

        horizontal\_flip=True,

        vertical\_flip=False)

model.summary() #checking what our final model would look like

**Text

Description automatically generated**

datagen.fit(X\_train)

history = model.fit\_generator(datagen.flow(X\_train,Y\_train, batch\_size=batch\_size),

                              epochs = epochs, validation\_data = (X\_val,Y\_val))

**Text

Description automatically generated**

**Graphical user interface, text

Description automatically generated with medium confidence**

**A picture containing graphical user interface

Description automatically generated**

**7.GIT SETUP**

<https://github.com/K-L-H-CODERS/BRAIN-TUMOR/blob/main/README.md>

**Graphical user interface, text, application, email

Description automatically generated**

**8.DATASET**

The data set was taken from Kaggle which contains two folders, one is for training and the other is for testing. It has 3 types of Brain Tumor - Glioma, Meningioma and pituitary along with images of patients with no tumor.

A picture containing graphical user interface

Description automatically generated

**9.CONCLUSION & FUTURE WORK**

The objective of this work is to uprise a model for classifying the brain tumor mri images. hence, convolution neural network supported classification is used. This concept is able to detect the images using Keras, by building an artificial convolutional neural network. Pre-processing image is done by this is splitting segmenting and extracting the brain tumor from MR images. In this document, a new approach was presented to classify brain tumors. First, using the image edge detection technique, we find the region of interest in MRI images and cropped them then, we used the data augmentation technique for increasing the size of our training data. Second, we provide an efficient methodology for brain tumor classification by proposing a simple CNN network. For sophisticated and accurate results neural network requires a large amount of data to train on, but our experimental result shows that even on such a small dataset, we can attain full accuracy and our accuracy rate is very fine as compared to VGG-16, ResNet-50, and Inception-v3 model. Our model average training time per epoch is 205 sec while the VGG-16 takes 456 sec, ResNet-50 takes 606 sec and Inception-v3 takes 375 sec average training time per epoch. So, our model needs less computational specifications as it takes less execution time. Moreover, our model and Inception-v3. Our proposed system can play a prognostic significance in the detection of tumors in brain tumor patients. Our proposed system is for binary classification problems, however, in future work, the proposed method can be extended for categorical classification problems such as identification of brain tumor types such as Glioma, Meningioma, and Pituitary or may be used to detect other brain abnormalities. Also, our proposed system can play an effective role in the early diagnosis of dangerous disease in other clinical domains related to medical imaging, particularly lung cancer and breast cancer whose mortality rate is very high globally. We can prolong this approach in other scientific areas as well where there is a problem in the availability of large data or we can use the different transfer learning methods with the same proposed technique. The major drawback will be the computational time while working with larger data. However, working on a larger data set may improve the accuracy of the training model. As an extension to this work, the model can be modified to become compatible with 3 dimensional brain scans, in order to perform more efficient image segmentation and also to identify the stage of the tumor

**10. REFERENCES**

* ARTICLE LINK:-https://towardsdatascience.com/classification-of-brain-mri-as-tumor-non-tumor-d48838ccc162
* ARTICLE LINK:-<https://www.nature.com/articles/s41467-019-12527-5>
* ARTICLE LINK:- <https://www.medrxiv.org/content/10.1101/2020.02.24.20026955v1>
* ARTICLE LINK:- <https://arxiv.org/pdf/1810.11654v3.pdf>
* <https://paperswithcode.com/datasets?task=brain-tumor-segmentation&page=1>
* <https://plos.figshare.com/articles/dataset/Spatio-spectral_classification_of_hyperspectral_images_for_brain_cancer_detection_during_surgical_operations/6000767>
* <https://www.omicsdi.org/dataset/geo/GSE90496>
* <https://frontiersin.figshare.com/articles/dataset/DataSheet_1_Deep_Neural_Network_for_Differentiation_of_Brain_Tumor_Tissue_Displayed_by_Confocal_Laser_Endomicroscopy_docx/14573307/1>
* <https://journals.plos.org/plosone/article?id=10.1371/journal.pone.0161807#sec006>