BRAIN TUMOR DETECTION AND CLASSIFICATION BY MRI IMAGES USING DEEP LEARNING TECHNIQUES

A DISSERTATION

SUBMITTED IN PARTIAL FULFILLMENT OF THE REQUIREMENTS
FOR THE AWARD OF THE DEGREE
OF

MASTERS OF SCIENCE
IN
MATHEMATICS

Submitted by:
RASHIKA MANGLA
2K20/MSCMAT/23
CHETNA
2K20/MSCMAT/08

Under the supervision of
Ms. SUMEDHA SENIARAY

DEPTT. OF APPLIED MATHEMATICS
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi - 110042

MAY, 2022

MSc. MATHEMATICS
RASHIKA, CHETNA
2022

DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi - 110042

CANDIDATE'S DECLARATION

We, (Rashika Mangla and Chetna), 2K20/MSCMAT/23 and 2K20/MSCMAT/08, students of MSc. (Mathematics), hereby declare that the project Dissertation titled "Brain Tumor Detection and Classification by MRI Images using Deep Learning Techniques" which is submitted by us to the Department of Applied Mathematics, Delhi Technological University, Delhi in partial fulfillment of the requirements for the award of the degree of Master of Science, is original and not copied from any source without proper citation. This work has not previously formed the basis for the award of Degree, Diploma Associateship, Fellowship or other similar title or recognition.

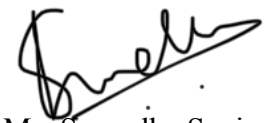
Place: Delhi
Date: 1 May, 2022

Rashika Mangla
Chetna

DEPTT. OF APPLIED MATHEMATICS
DELHI TECHNOLOGICAL UNIVERSITY
(Formerly Delhi College of Engineering)
Bawana Road, Delhi - 110042

CERTIFICATE

I hereby certify that the Project Dissertation titled "Brain Tumor Detection and Classification by MRI Images using Deep Learning Techniques" which is submitted by (Rashika Mangla and Chetna), 2K20/MSCMAT/23 and 2K20/MSCMAT/08, Department of Applied Mathematics, Delhi Technological University, Delhi in partial fulfillment of the requirements for the award of the degree of Master of Science, is a record of the project work carried out by the students under my supervision. To the best of my knowledge this work has not been submitted in part or full for any Degree or Diploma to this university or elsewhere.



Place: Delhi
Date: 1 May, 2022

Ms. Sumedha Seniaray
SUPERVISOR
Assistant Professor
Deptt. Of Applied mathematics
Delhi Technological University
(Formerly Delhi College of Engineering)
Bawana Road, Delhi – 110042

ACKNOWLEDGEMENT

We would like to express our sincere gratitude to our teacher “Ms. Sumedha Seniaray” for providing their invaluable guidance, comments, and suggestions throughout the course of the project.

Also, we gratefully acknowledge the valuable contributions of people around us involved in the development of this dissertation. The acknowledgements are also due to constant useful suggestions given by them in reviewing and then improvising it by providing helpful inputs.

Lastly, we thank the almighty and our parents and friends for their constant encouragement without which this project would not have been possible.

RASHIKA MANGLA

CHETNA

ABSTRACT

MRI images [8] play a significant influence in brain tumor classification and detection but instead of having detection and classification using the medical equipment which is a radiologists or clinical professionals do a time-consuming and laborious task where accuracy depends only on the experience only, it can be beneficial to detect and classify the brain tumor by deep learning techniques and algorithms. As a result, using computer-assisted technologies to circumvent these limits is becoming increasingly vital.

In this paper, the early detected and diagnosed brain tumor images along with their csv data has been used to find out the accuracy of the CNN algorithm for tumor detection and SVM algorithm for tumor classification into benign and malignant. HOG has been used for the feature extraction. After performing the experiment, it was observed that CNN achieved the detection accuracy of 87.02% and further tumor classification by employing SVM, the highest accuracy achieved was 96.35%. The experiment proved a very good accuracy of detection and classification even after using three different methods in the whole procedure.

CONTENTS

LIST OF FIGURES

LIST OF TABLES

LIST OF ABBREVIATIONS, SYMBOLS AND NOMENCLATURE

CNN (Convolutional neural network) [13] ,

HOG (Histogram of Oriented Gradients),

SVM (Support Vector Machine),

MRI (Magnetic Resonance Imaging),

CT (Computed Tomography)

CAD (Computed Aided Diagnosis)

ML (Machine Learning)

DL (Deep Learning)

ANN (Artificial Neaural Network)

KNN (K-Nearest Neighbour)

GLCM (Grey Level Co-occurrence Matrix)

HGG (High Grade Glioma)

LGG (Low Grade Glioma)

CHAPTER 1

INTRODUCTION

1.1 BRAIN TUMOR

A brain tumor is a malformation of the brain growth in the brain that can be harmful or even cancer-free. Uncontrolled cell proliferation and excessive brain damage cause tumors in the brain. There are two types of brain cancers: primary and secondary tumors that result in benign or malignant brain tumors. Gliomas are the most widely used sort of tumor. In order to collect relevant clinical data, such as the presence of a tumor, location, and type, computer-assisted devices can be used to perform the procedure automatically.

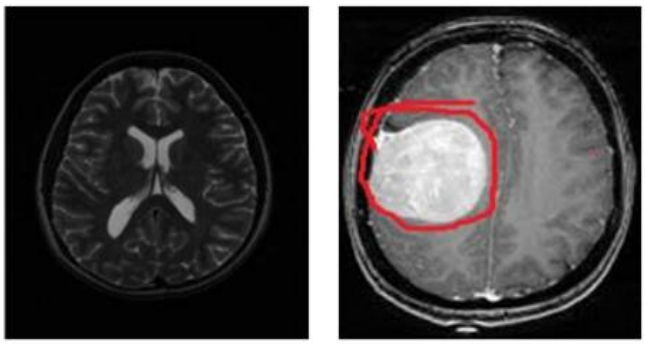


Fig 1: Sample images of a human brain where (a) represents a Healthy Brain and (b) represents a Brain affected with tumor

However, determining their composition, volume, parameters, detection, size and classification remains a difficult task. It is non-invasive and delivers high optical acuity to soft tissues, MRI [12] is preferred over other therapies and diagnostic methods. Signs and symptoms of a tumor in the brain vary depending on the type and location of the tumor. Because different parts of the brain control different aspects of body processes, some do not show symptoms until they are extremely large and cause severe and rapid loss of health. Hearing problems, problems with balance, speech impairment, changes in vision, memory problems, mobility problems, personality changes, lack of concentration, and a weak point in one part of the body are some of the most common symptoms.

Gliomas are divided into three categories, including Oligodendroglioma, Astrocytoma, and Glioblastoma, regardless of the WHO classification or classification of tumor. Due to the remarkable diversity and structure of MRI data, Brain Tumor is a difficult task.

The brain tumor is one of the most common and, as a result, one of the most lethal brain ailments, affecting and destroying countless lives around the world. Cancer is a brain disease in which cancer cells spread throughout the tissues of the brain. According to new cancer research, more than 1000 people worldwide are diagnosed with brain tumors each year.Despite consistent efforts to overcome the problems of brain tumors, statistics reveal unfavorable outcomes for tumor patients. Researchers are using computer vision to obtain a better understanding of malignancies in their early stages and how to treat them using modern therapy choices to combat this.

1.2 MAGNETIC RESONANCE IMAGING

MRI and CT scans of the brain are the two most popular techniques for detecting the presence of a tumor and determining its location for treatment considerations. These two scans are still widely utilized due to their convenience and capacity to produce high-definition pictures of diseased tissues. Currently, a variety of additional treatments for tumors are available, including surgery, radiation therapy, and chemotherapy.

The size, type, and grade of the tumor displayed in the MRI imaging all play a role in determining which treatment is best. It's also in charge of determining whether cancer has spread to other parts of the body. For therapy operations with a resolution to reduce diagnostic errors, precise sighting of the type of brain problem is really important. Using computer-aided diagnostic (CAD) technologies, the precision is frequently improvised.

The main goal of computer vision is to generate a trustworthy output, which is an association estimate that may help doctors interpret images and reduce picture reading time. These advancements increase medical diagnosis consistency and accuracy; yet, segmenting an MRI image of a tumor and its surrounding area is a

tough task. Another issue that makes computerized brain tumor detection and segmentation challenging is the emergence of tumors in specific positions within the brain image without distinguishing picture intensities.

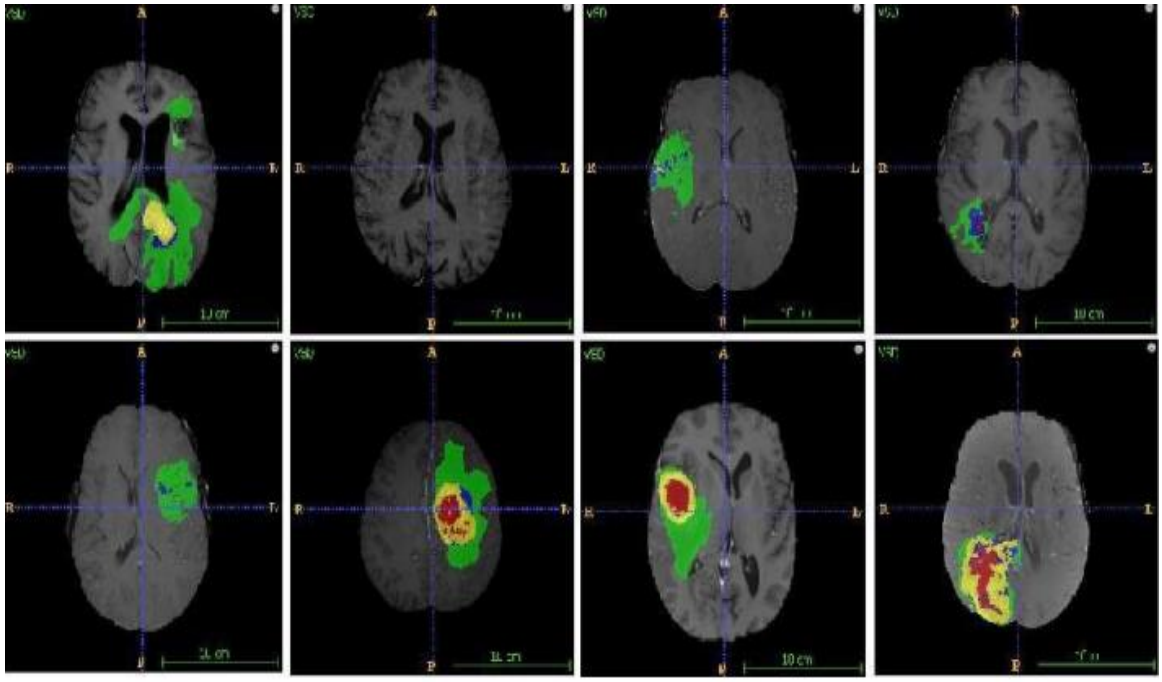


Fig 2: Location of tumors in eight different images.

MRI is preferred over other therapeutic and diagnosis methods because it gives better picture contrast in soft tissues and is non-invasive. High variability and inherent MRI data properties, such as variability in tumor sizes or shapes, tumor identification, area computation, segmentation, classification, and discovering ambiguity in segmented regions, make Brain Tumor a difficult task. Hence, there is a strong need for automation of brain tumor segmentation and classification with the help of computer vision and deep learning algorithms.

Sound and image clarity are limits on MRI imaging. Unwanted information in pictures is called sound. Noise can sometimes affect the edges and details, reducing the brightness adjustment. Because it is difficult to find specific boundaries and to distinguish a tumor from noise, it is an emerging topic of image processing

research. In this study, we examine and contrast two methods for brain differentiation, detection, and differentiation that are both effective and efficient.

1.3 DEEP LEARNING TECHNIQUES

Many strategies for brain tumor classification have recently been presented, which may be divided into ML and DL techniques based on feature selection and learning process. For classification in ML techniques, feature selection and extraction are critical.

DL techniques, on the other hand, directly extract and learn the features from the image. The image processing application's ultimate purpose is to extract relevant elements from the image. Picture display and printing, image editing and manipulation, image enhancement, feature identification, image compression, image segmentation, and image registration are some of the features included. Biomedical, forensics, remote sensing, agricultural, food management, automobiles, and communications are just a few of the applications for image processing.

Image processing is currently one of the most rapidly growing study topics in medical image processing. Recent deep learning algorithms, particularly CNN, have shown to be highly accurate and are frequently employed in medical image analysis. Support vector machine (SVM), artificial neural network (ANN), Random Forest (RF), Decision Tree (DT), and K-Nearest Neighbor (KNN), Otsu's Thresholding, Histogram of gradient (HOG), Gray level Co-Occurrence matrix are some of the learning-based classifiers that have been used for brain tumor classification and detection (GLCM).

In this study, we have used CNN for tumor detection, HOG for feature extraction and SVM for tumor classification. These three are defined as follows.

1.3.1 CONVOLUTIONAL NEURAL NETWORK

A CNN is a Deep Learning gadget that could take a picture as input, deliver precedence to one of a kind aspects/gadgets withinside the picture, and differentiate

among them. When compared to different type methods, ConvNet calls for significantly much less pre-processing.

1.3.2 SUPPORT VECTOR MACHINE

The aim of the SVM technique is to decide the high-quality line or selection boundary for categorizing n-dimensional area into instructions in order that next statistics factors may be without problems located withinside the proper category. SVM is used to pick out the acute factors/vectors that assist in creating the hyperplane. The technique is named Support Vector Machine due to the fact that it helps vectors replicate excessive cases.

1.3.3 HISTOGRAM OF ORIENTED GRADIENTS

In pc imaginative and prescient and photograph processing, the HOG is a characteristic descriptor for item detection. The approach counts how usually a gradient orientation seems in a specific segment of a photograph. Edge orientation histograms, scale-invariant characteristic remodel descriptors, and form contexts are all comparable methods, however this one stands proud considering its miles computed on a dense grid of often spaced cells and makes use of overlapping neighborhood evaluation normalization for advanced accuracy.

1.4 SCOPE

Our intention is to broaden an automatic technique for improving, segmenting, and classifying mind tumors. The device may be utilized by neurosurgeons and healthcare specialists. The device employs picture processing, sample analysis, and laptop imaginative and prescient strategies to enhance the sensitivity, specificity, and performance of mind tumor screening. Medical imaging projects' important intention is to extract as a good deal applicable and correct records as viable from those photos with the least quantity of error. The suitable mixture and parameterization of the degrees lets in for the advent of adjunct equipment which could assist with early diagnosis, tumor detection, and region monitoring.

In this study, we have tried to increase the accuracy of tumor detection and classification using three different methods for different steps of the procedure. Earlier detection, feature extraction and classification using a single method has been done and compared for all the techniques like CNN, SVM, HOG, OSTU' S METHOD etc. and the accuracy was also good. But we have tried to use more appropriate methods for different steps so that accuracy can be further increased and we get better results in Brain Tumor disease as earlier and better detection and classification is the key to treat this life threatening disease.

CHAPTER 2

LITERATURE REVIEW

Machine learning techniques have been widely used in a variety of fields, including medical diagnosis and prevention. However, few studies have focused on the diagnosis of brain tissue using MRI. Most DL methods use MRI data to train and test old DL algorithms. DL has recently been used to diagnose brain tumors in a number of techniques.

G. Hemanth et al. [1] suggested Automated Brain Tumors Automatic Detection System. CNN is used for classifying and identifying tumors. Then a pixel-based detection method is used to obtain a picture of the brain and other affected areas. Then the separation using the extracted features is done using CNN. They found 91% accuracy in sections.

F.P. Polly et al. [8] suggested a Deep Learning-based technique for detecting and classifying HGG and LGG brain cancers. The study's goal was to distinguish between normal and abnormal brain tumors, as well as to classify abnormal brain tumors into HGG and LGG tumors. To transform the input image to a binary image, the suggested method uses otsu binarization. The tumor is then segmented using the k-means clustering approach. SVM is used to categorize both stage 1 and stage 2 cancers.

Praveena Pedapati and Rama Vaishnavi Tanneedi [2] propose to use SVM and in-depth study to detect and differentiate brain cancer automatically. However, in this case, SVM failed to provide the highest level of accuracy. The accuracy of the features extracted from high glioma (Malignant) images without the use of histogram of the targeted gradients was 97 percent in high glioma images. Low glioma (Benign) images have an accuracy of 68.75 percent.

In [3] the authors performed Brain tumor CT imaging classification into malignant and malignant images using SVM with kernel function. The best performance of the categories is obtained through WSVM, based on test data. Moreover, these findings suggest that the proposed method is effective and effective in predicting malignant

and malignant tumors from CT scans of the brain, with 74% accuracy of SVM and 76% of WSVM. The proposed method may be consistent with other types of imaging, such as MRI, in the future, and it may be used to classify and differentiate tumors from other parts of the body.

M.H.O Rashid et al. [9] proposed employing anisotropic filtering, morphological procedures, and SVM to detect brain cancers from MR images. SVM is used to segment cancers from filtered images. SVM divides pixels into two groups. The tumor's location was then retrieved utilizing morphological techniques from a divided area. For tumor detection, the proposed technique has an accuracy of 83 percent.

Reema Mathew et al. [10] proposed employing Wavelet Transform and SVM to detect and categorize a brain tumor. Morphological operations are next done to determine the tumor's location. Then, to extract features from segmented images, the Discrete Wavelet Transform is used. These characteristics were fed into the SVM classifier, which classified brain tumors as malignant or benign. For brain tumor categorization, the proposed technique has an accuracy of 86%.

The authors of another study [4] used the effectiveness of pressure strategies to improve accuracy as well as timing of CNN executions in differentiation by brain MRI. Before they were divided into sections, they presented the first step in processing. The method first uses the Probabilistic Neural Network (PNN) to differentiate the Genital Region (ROI) (especially the area of the brain tissue), and then uses Back Propagation Neural Networks to narrow the ROI (BPNN). Finally, compressed images are uploaded to CNN segregation. For comparison, results were obtained for three different grades of formats and the accuracy levels displayed were more than 90%.

R-CNN's rapid method of successful detection and classification of brain MRI images was proposed by the authors of another paper [5]. Images are first embedded in a simple CNN, which produces a dynamic feature map, which is then changed to regional suggestions, which are, after that, resized into a feature vector by the integration stage of ROI. Finally, by categories, this ROI vector feature is integrated

into the fast R-CNN. SVM was also used to create the largest margin possible between classes, allowing the algorithm to classify images into categories with greater accuracy. Their algorithm had an accuracy rate of 95%.

[6] proposes a flexible approach based on in-depth study of Generative Adversarial Networks (GANs). Their plan was to previously train CNN as a GAN racist using two databases. The productive part of GAN was aided by the development of data on the production of more realistic MRI scans of the brain. To separate, the final bed of CNN discrimination on GAN is changed by SoftMax. In the Figshare database, this precise CNN separator is fine-tuned over time. The outcomes reveal 88 percent precision.

Mircea Gurbina et al. [11] proposed using Wavelet transform and SVM to detect brain cancers from MR images. The thresholding algorithm developed by Otsu is used to recognise objects in the image. The SVM Classifier was then used to classify tumors into malignant and benign categories using the given features.

[7] Due to the complexity of imaging and there aren't any anatomical models that accurately represent the possible deformities in each and every component, the separation of medical imaging is a difficult task. When it comes to the original size of the collection, the proposed method is quite effective and collection centers. This function recommends a system that requires non-invasive human intervention to isolate tissue of the brain. The prime purpose of this suggested program is to assist human specialists or surgeons in locating patients in less time.

CHAPTER 3

METHODOLOGY

The proposed work has been implemented in four phases. The experimental setup involved a system with 4GB Ram, Windows 10 and implemented via Python.

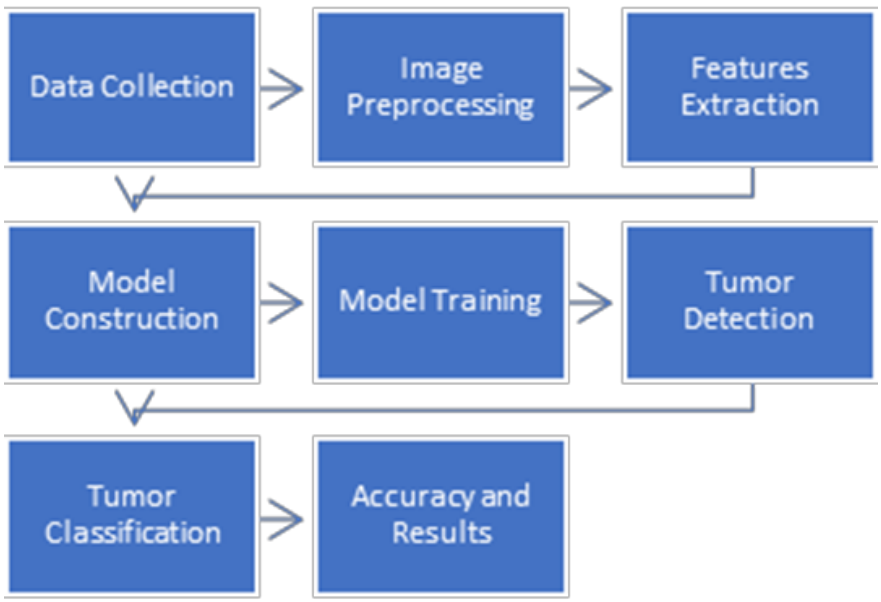


Fig 3: Steps followed for tumor detection and classification

3.1 PHASE-I

DATA COLLECTION AND IMAGE PREPROCESSING

Data collection -

The standard available dataset [17] was collected for the implementation. There are 2212 brain MRI images in the collection, divided into two folders: yes and no. Yes, the folder contains 1112 tumorous brain MRI images, and no, the folder has 1100 non-tumorous brain MRI images.

Data Preprocessing -

Preprocessing transforms raw data into a format that computers and machine learning can comprehend and evaluate. Text, images, videos, and other real-world data are disorganized. It is typically incomplete and lacks a coherent design, not to mention faults and contradictions. Machines prefer to work with data that is neat and tidy, therefore they read data in 1s and 0s. Calculating structured data such as whole numbers and percentages is therefore simple. Before being evaluated, unstructured data such as text and photographs must be cleaned and processed.

The information we gathered came in a variety of shapes, sizes, and orientations. As a result, the photos had to be resized to a shape of (240, 240, 3) = (image width, image height, number of channels). As a result, all photographs should have the same shape to transmit as an input into the neural network.

Normalization was the next step in the process.In image processing, normalization is a technique for changing the range of pixel intensity values. Histogram stretching, or contrast stretching, is another term for normalisation., can be used on photos with poor contrast due to glare. The photos were scaled to the range 0-1 so that they could be viewed regularly. To do so, multiply all of the pixel values by the largest pixel. This equals 255.

3.2 PHASE-II

FEATURES EXTRACTION USING HOG

HOG [15] is a feature descriptor that is used in image processing to recognize objects. The feature descriptor's goal is to generalize an object in an image so that it offers the same feature descriptors in all images that contain that object, regardless of angle, illumination, distance, or other circumstances. In a short area of a picture, the HOG descriptor approach counts occurrences of 22 gradient orientations in a detection window, or region of interest (ROI).

The histogram of directed gradient descriptors is based on the premise that the distribution of intensity gradients or edge directions can be used to describe the appearance and shape of local objects in an image. A histogram of gradient directions is constructed for the pixels within each cell when the image is segmented into small connected parts called cells. The descriptor is formed by joining these histograms together. To improve the accuracy of local histograms, generate an intensity measure across a wider section of the image, known as a block, and then use this value to normalize all cells within the block. The invariance to differences in illumination and shading is improved as a result of this normalization.

The HOG descriptor has a few significant advantages over other descriptors. Because it operates on local cells, it is insensitive to geometric and photometric changes except for object orientation. Such changes would only occur in larger spatial regions. Furthermore, as long as pedestrians maintain a roughly upright position, individual body movement can be ignored using coarse spatial sampling, fine orientation sampling, and strong local photometric normalization. As a result, the HOG descriptor is particularly useful for spotting humans in photographs.

One of the most significant topics in computer vision is the gradient vector. A gradient vector is sometimes known as a picture gradient. Feature extraction and edge detection are two applications of gradient vectors. The difference of neighborhood values in both the horizontal and vertical axes is used to calculate the gradient vector for a certain pixel. In HOG, orientation is crucial. The term "orientation" refers to a shift in direction when the pixel intensity value changes. The direction of change can be along the X-axis or the Y-axis.

Orientation = arctan (Y/X)

is the formula for determining the orientation of a gradient vector.

The foundation for histogram grouping and normalization is the grouping of cells into blocks. The normalized group of cells is called a block histogram. The feature descriptor is represented as a set of block histograms. A feature descriptor is a technical term that refers to a picture or a portion of an image that simplifies the

original form of the image by removing the key data. In HOG, the distribution of gradient orientation is used as a feature.

The picture, orientations, pixels per cell, cells per block, visualize, and multichannel parameters are used as input to the Hog() function.

3.3 PHASE-III

TUMOR DETECTION USING CNN

A CNN is a Deep Learning system that can take an image as input and assign meaning to different aspects/objects in the image while also distinguishing between them. Other partitioning techniques pale in contrast, ConvNet just requires a little amount of computation. Despite the fact that the fundamental approaches need manual filter manipulation,With enough training, ConvNets can learn these filters / symbols.Using the correct filters, ConvNet can successfully capture Local and Temporary dependencies on a picture. Architecture creates a high degree of image databases due of the reusability of weights and the reduced number of parameters involved

In comparison to other classification methods, a CNN requires substantially less preprocessing. While basic techniques require hand-engineering of filters, CNN can learn these filters/characteristics with enough training. A CNN may properly capture the spatial and temporal dependencies in a picture by using appropriate filters.

In other words, the network may be taught to detect certain patterns, higher-quality image technologies. In order to detect brain cancers in multiple MRI scans of the brain we performed detection using various algorithms Like CNN, SVM, OTSU'S THRESHOLDING, KNN and Random Forest. But among all algorithms the best one found out with highest accuracy [16] was CNN. The formula used to determine the model's accuracy is –

$$\text{ACCURACY} = \frac{TP + TN}{FP + FN + TP + TN}$$

Where,

TP = True Positive

TN = True Negative,

FP = False Positive

FN = False Negative.

TP and TN means the samples which have been detected accurately and

FP and FN means the samples which have not been detected accurately.

We'll need to import Keras and the other packages to build the CNN model. The following packages should be imported:

- The neural network is initialized using sequential.
- Convolution2D is a technique for creating the image-processing convolutional network.
- The pooling layers are added using the MaxPooling2D layer.
- Flatten transforms a pooled feature map into a single column that will be sent to a fully linked layer
- The High density layers provide layers that are fully connected to the neural network.

3.4 PHASE-IV

TUMOR CLASSIFICATION USING SVM

An SVM algorithm's purpose is to find a hyperplane that clearly separates data points in an N-dimensional space. There are various hyperplanes to choose from to separate the two types of data points.. Our objective is to locate a plane with a large

limit, or both groups' data points are separated by a great distance. Increasing the gene range adds some support, making it simpler to succeed in distinguishing the following data points.

SVM [14] has the following advantages:

- It works really well with a large separation margin.
- It works well in three-dimensional areas. If the maximum size exceeds the sample size, This strategy is effective.

Memory functions successfully because it makes decisions based on a set of training points (called supporting vectors).The HOG characteristics that we retrieved from the MRI images were saved in a .csv file. We employed a radial basis function to accomplish the processing, which involves mapping data into a higher-dimensional space. Kernelling is the term for this process, and the The kernel function is a mathematical function used to change the data. Then we have fitted our model and predicted the new values.

3.5 SYSTEM CONFIGURATION

SOFTWARE REQUIREMENTS

Windows: 10, Python: Jupyter Notebook

Python:Python is a high-level general-purpose programming language developed by Guido Van Rossum and first published in 1991. Python's design philosophy focuses on promoting readability, the code has a lot of whitespace. Its features and object-oriented approach are intended to assist programmers in writing clear, logical code for both small and large-scale projects. Python is garbage collected and dynamically typed. The programming paradigms supported include procedural, object-oriented, and functional programming.

Jupyter Notebook: The Anaconda distribution includes the conda package management and the virtual environment manager, as well as 1,500 PyPI items. Anaconda Navigator, a graphical user interface, is included as an alternative to the

CLI. A Jupyter Notebook document is a JSON file that has an ordered list of input/output cells that can contain code, text mathematics, graphs, and rich media, and commonly ends in ". ipynb."

NumPy:NumPy is a Python package for array operations. Includes high performance multidimensional array objects and manipulation tools. It is the most significant Python package for scientific computing. It has various qualities, including the following: This object contains a large N-dimensional array. Broadcasting capabilities (advanced) C/C++ and Fortran code integration tools Linear algebra, Fourier transform, and random number skills.

TensorFlow: Flow of tensors Dataflow and differentiable programming are two programming techniques that may be used to tackle a number of problems, and is a free and open-source application software library for them. It's a symbolic math library that's also used in machine learning programmes like neural networks. Google uses it for both research and manufacturing.

Keras: Keras is a Python-based open-source neural network library. It supports TensorFlow、Microsoft Cognitive Toolkit、R、Theano und Plaid ML . It's designed to be user-friendly, modular, and expandable in order to facilitate quick deep neural network research. Keras includes many implementations of standard neural-network building blocks such as layers, objectives, activation functions, optimizers, and a variety of other tools to make working with image and text data easier while also reducing the amount of coding required to write deep neural network code.

OpenCV: OpenCV is a computer vision library that is primarily utilized in real-time. After Intel produced it, Itseez and Willow Garage promoted it (which was later acquired by Intel). The open source BSD license makes the library cross-platform and free to use. According to a stated list of supported layers, OpenCV supports models from deep learning frameworks such as TensorFlow, Torch, and PyTorch. (after converting to an ONNX model), and Caffe. It promotes Open Vision Capsules, a portable, lightweight, cross-platform format.

HARDWARE CONFIGURATION

- Processor: Intel core i5 or above
- 64-bit, quad-core, 2.5 GHz minimum per core
- Ram: 4 GB
- Hard disk: 10 GB of available space
- Operating system: Windows

CHAPTER 4

RESULTS AND CONCLUSION

4.1 RESULTS

For the CNN algorithm the data collected and the detection done by our algorithm is as follows -

	No. of MRI Images	Positive MRI Images	Negative MRI Images
RAW DATA	2212	1112	1100

Table 1: Positive and Negative brain tumor raw data

From all the images of dataset, the accuracy of detection of images is as follows -

	TRUE POSITIVE	TRUE NEGATIVE	FALSE POSITIVE	FALSE NEGATIVE
No. of MRI mages detected	983	942	58	129

Table 2: Positive and Negative brain tumor detected data

So, using this result the accuracy of our model CNN is 87.02%.

MRI IMAGES:

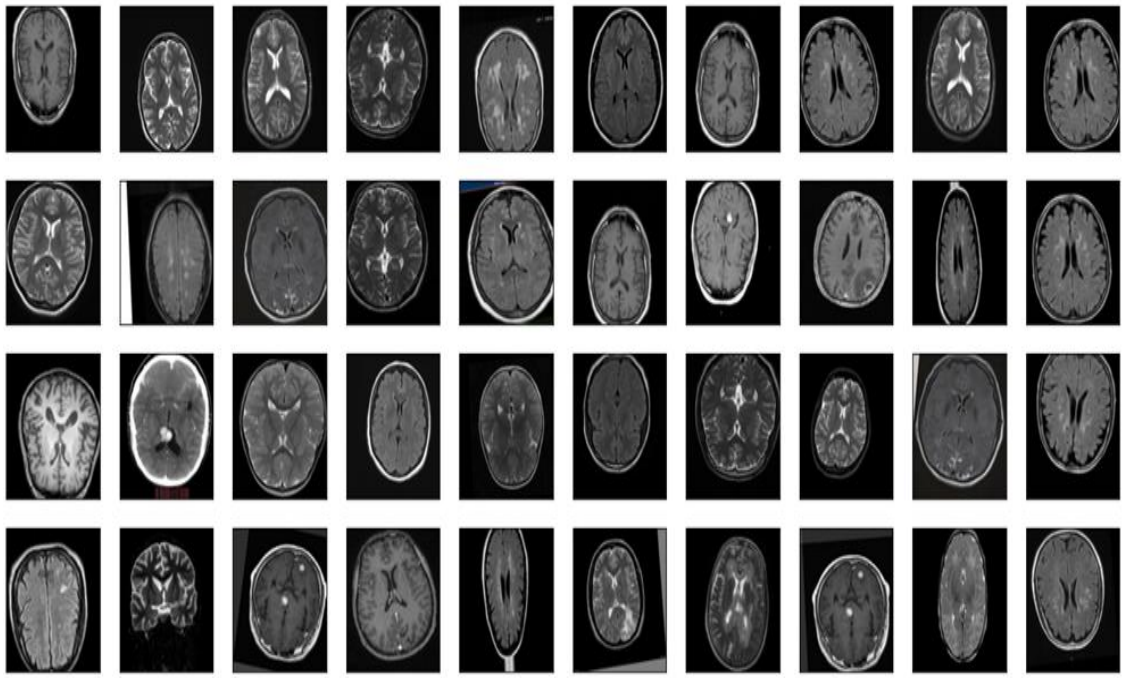


Fig 4: MRI Images of the data collected

For the SVM algorithm the data taken and the classification into benign and malignant done by our algorithm is as follows –
In the dataset that we took for our CNN model - 983 were truly detected positive for brain tumor, this data was then split into training and testing data. 137 were kept into testing data and 846 were kept into the training dataset. Out of these 137-testing dataset, the data of 90 brain tumors were benign and 47 were malignant.

But after evaluating the data in our SVM algorithm, out of these 137 tumors, only 85 were detected to be benign positively and 47 were detected to be malignant positively whereas 5 were detected to be false malignant and there were no false benign tumors.

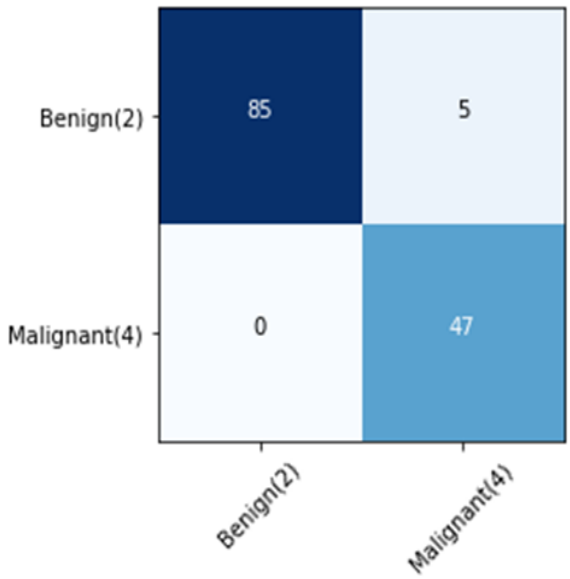


Fig 5: Confusion matrix

From this matrix it is clear that the benign tumors truly detected are 85 and the malignant tumors truly detected are 47. While the false malignant tumors detected are 5 in number however there are no benign tumors detected falsely. So, the accuracy of our SVM model is 96.35%.

4.2 CONCLUSION

The most important aspect of our proposed work was how to use a CNN to find out if a tumor is present and a HOG to extract brain characteristics tumor MRI images, after that a classification method, SVM, to differentiate between benign and malignant brain tumors. An approach based on a combination of the feature extraction algorithm (HOG), CNN, and SVM for tumor identification and classification from brain pictures is provided.

CNN has the ability to identify tumors. When it comes to picking an auto-feature in medical photographs, the CNN comes in handy. Clinicians classified the images obtained at the centers, and then tumor screenings were divided into two categories: normal and patient. The proportion of sick to healthy subjects was proportionate to

the proportion of image categorization in two classes. After pre-processing, the images were fed into the CNN. CNN's accuracy is 87.02 percent for appropriately categorizing images into two normal and patient classes.

The accuracy of the proposed algorithm increased to 96.35 percent on the test data while using the proposed approach of feature extraction and using SVM for categorization into benign and malignant tumors, which is an improvement over the CNN. Due to the importance of the physician's diagnosis, the model's accuracy can assist doctors in diagnosing the tumor and treating the patient, resulting in a high level of medical accuracy when utilizing the proposed technique. This model can be expanded in the future to further classify benign and malignant tumors into grades of different kinds (Grade 1, Grade 2, Grade 3, Grade 4). This could aid the medical system in providing more effective and timely treatments.

4.3 FUTURE SCOPE

Experimentation shows that the recommended strategy requires a large gathering of medical data in the realm of medical image processing; training sets for more accurate results is a time-consuming task, and datasets may be unavailable in some circumstances. The proposed algorithm must be sufficiently resilient to accurately recognise tumor regions from MR images in all of these cases. The proposed method can be improved further by combining weakly trained algorithms that can detect irregularities with little or no training data, as well as self-learning algorithms that can improve the algorithm's accuracy while reducing computing time.

CHAPTER 5

CODES -

5.1 PHASE I: DATA COLLECTION AND IMAGE PREPROCESSING

```
from google.colab import drive
drive.mount('/content/gdrive')
```

```
Mounted at /content/gdrive
```

```
import tensorflow as tf
from tensorflow.keras.layers import Conv2D, Input, ZeroPadding2D, BatchNormalization, Activation, MaxPooling2D, Flatten, Dense, Dropout
from tensorflow.keras.models import Model, load_model
from tensorflow.keras.callbacks import TensorBoard, ModelCheckpoint
from sklearn.model_selection import train_test_split
from sklearn.metrics import f1_score
from sklearn.utils import shuffle
import cv2
import imutils
import numpy as np
import matplotlib.pyplot as plt
import time
from os import listdir
from keras.models import Sequential
%matplotlib inline
import os
from tensorflow.keras.models import Sequential
```

```
[ ] def augment_data(file_dir, n_generated_samples, save_to_dir):
        """
        Arguments:
```

```
[ ]         file_dir: A string representing the directory where images that we want to augment are found.
            n_generated_samples: A string representing the number of generated samples using the given image.
            save_to_dir: A string representing the directory in which the generated images will be saved.
        """

        #from keras.preprocessing.image import ImageDataGenerator
        #from os import listdir

        data_gen = ImageDataGenerator(rotation_range=10,
                                      width_shift_range=0.1,
                                      height_shift_range=0.1,
                                      shear_range=0.1,
                                      brightness_range=(0.3, 1.0),
                                      horizontal_flip=True,
                                      vertical_flip=True,
                                      fill_mode='nearest'
                                     )


        for filename in listdir(file_dir):
            # load the image
            image = cv2.imread(file_dir + '\\' + filename)
            # reshape the image
            image = image.reshape((1,)+image.shape)
            # prefix of the names for the generated sampels.
            save_prefix = 'aug_' + filename[:-4]
            # generate 'n_generated_samples' sample images
            i=0
```

```
[ ]         for batch in data_gen.flow(x=image, batch_size=1, save_to_dir=save_to_dir,
                                       save_prefix=save_prefix, save_format='jpg'):
                i += 1
                if i > n_generated_samples:
                    break
```

```
[ ] augmented_path = '/content/gdrive/MyDrive/augmented data/'
```

```
[ ] def load_data(dir_list, image_size):
        """
        Read images, resize and normalize them.
        Arguments:
            dir_list: list of strings representing file directories.
        Returns:
            X: A numpy array with shape = (#_examples, image_width, image_height, #_channels)
            y: A numpy array with shape = (#_examples, 1)
        """
        # load all images in a directory
        X = []
        y = []
        image_width, image_height = image_size
        for directory in dir_list:
            for filename in listdir(directory):
                # load the image
                image = cv2.imread(directory + '/' + 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=(image_width, image_height), interpolation=cv2.INTER_CUBIC)
            # normalize values
            image = image / 255.
            # convert image to numpy array and append it to X
            X.append(image)
            # append a value of 1 to the target array if the image
            # is in the folder named 'yes', otherwise append 0.
            if directory[-3:] == 'yes':
                y.append([1])
            else:
                y.append([0])

    X = np.array(X)
    y = np.array(y)

    # Shuffle the data
    X, y = shuffle(X, y)

    print(f'Number of examples is: {len(X)}')
    print(f'X shape is: {X.shape}')
    print(f'y shape is: {y.shape}')

    return X, y
```

```
[ ] augmented_yes = augmented_path + 'yes'
    augmented_no = augmented_path + 'no'

    IMG_WIDTH, IMG_HEIGHT = (240, 240)
```

5.2 PHASE II: FEATURES EXTRACTION USING HOG

```
[ ] from PIL import Image
    import glob
    import cv2
    from skimage.io import imread
    from skimage.transform import resize
    from skimage.feature import hog
    from skimage import exposure

    def CalcFeatures(img):
        fd, hog_image = hog(img, orientations=9, pixels_per_cell=(8, 8),
                            cells_per_block=(2, 2), visualize=True, multichannel=True)
        return hog_image

    image_list = []
    for filename in glob.glob('/content/gdrive/MyDrive/augmented data/yes/*.jpg'): #assuming gif
        #im=Image.open(filename)
        keys=[]
        img = cv2.imread(filename)
        kp=CalcFeatures(img)
        if kp is not None:
                keys.append(kp)
```

5.3 PHASE III: TUMOR DETECTION USING CNN

```
[ ] X, y = load_data([augmented_yes, augmented_no], [IMG_WIDTH, IMG_HEIGHT])
```

```
[ ] def split_data(X, y, test_size=0.2):
        """
        Splits data into training, development and test sets.
        Arguments:
            X: A numpy array with shape = (#_examples, image_width, image_height, #_channels)
            y: A numpy array with shape = (#_examples, 1)
        Returns:
            X_train: A numpy array with shape = (#_train_examples, image_width, image_height, #_channels)
            y_train: A numpy array with shape = (#_train_examples, 1)
            X_val: A numpy array with shape = (#_val_examples, image_width, image_height, #_channels)
            y_val: A numpy array with shape = (#_val_examples, 1)
            X_test: A numpy array with shape = (#_test_examples, image_width, image_height, #_channels)
            y_test: A numpy array with shape = (#_test_examples, 1)
        """

        X_train, X_test_val, y_train, y_test_val = train_test_split(X, y, test_size=test_size, shuffle=True)
        X_test, X_val, y_test, y_val = train_test_split(X_test_val, y_test_val, test_size=0.5, shuffle=True)

        return X_train, y_train, X_val, y_val, X_test, y_test
```

```
[ ] X_train, y_train, X_val, y_val, X_test, y_test = split_data(X, y, test_size=0.3)
```

```
[ ] print ("number of training examples = " + str(X_train.shape[0]))
    print ("number of development examples = " + str(X_val.shape[0]))
    print ("number of test examples = " + str(X_test.shape[0]))
```

```
print ("X_train shape: " + str(X_train.shape))
print ("Y_train shape: " + str(y_train.shape))
print ("X_val (dev) shape: " + str(X_val.shape))
print ("Y_val (dev) shape: " + str(y_val.shape))
print ("X_test shape: " + str(X_test.shape))
print ("Y_test shape: " + str(y_test.shape))
```

```
[ ] def compute_f1_score(y_true, prob):
        # convert the vector of probabilities to a target vector
        y_pred = np.where(prob > 0.5, 1, 0)
        score = f1_score(y_true, y_pred)

        return score
```

```
[ ] model = Sequential()
    model.add(Conv2D(64,(3,3), input_shape=(240, 240, 3)))
    model.add(Activation('relu'))
    model.add(MaxPooling2D(pool_size=(2,2)))

    model.add(Conv2D(128,(3,3)))
    model.add(Activation('relu'))
    model.add(MaxPooling2D(pool_size=(2,2)))

    model.add(Conv2D(256,(3,3)))
    model.add(Activation('relu'))
```

```
    model.add(MaxPooling2D(pool_size=(2,2)))

    model.add(Conv2D(256,(3,3)))
    model.add(Activation('relu'))
    model.add(MaxPooling2D(pool_size=(2,2)))

    model.add(Flatten())
    model.add(Dense(128))
    model.add(Activation('relu'))
    model.add(Dropout(0.5))

    model.add(Dense(1))
    model.add(Activation('sigmoid'))
```

```
[ ] model.compile(optimizer='adam', loss='binary_crossentropy', metrics=['accuracy'])
    print(model.summary())
```

```
[ ] model.fit(x=X_train, y=y_train, batch_size=64, epochs=10, validation_data=(X_val, y_val))
    print(model.summary())
```

```
[ ] model.evaluate(X_test, y_test)
```

```
[ ] y_pred = model.predict(X_test)
    f1score = compute_f1_score(y_test, y_pred)
    print(f"F1 score: {f1score}")
```

```
[ ] def summary():
        yes = augmented_path + 'yes/'
        no = augmented_path + 'no/'

        nyes=len(os.listdir(yes))
        nno=len(os.listdir(no))
        ntotal=nyes+nno

        print('Total Images : ', ntotal)
        print('Yes Images : {} ( {}% )'.format(nyes, np.round((nyes/ntotal*1.0)*100), 3))
        print('No Images : {} ( {}% )'.format(nno, np.round((nno/ntotal*1.0)*100), 3))

    summary()
```

5.4 PHASE IV: TUMOR CLASSIFICATION USING SVM

```
[ ] !pip install wget
    import pandas as pd
    import pylab as pl
    import numpy as np
    import scipy.optimize as opt
    import wget
    from sklearn import preprocessing
    from sklearn.model_selection import train_test_split
    %matplotlib inline
    import matplotlib.pyplot as plt
```

```
[ ] cell_df = pd.read_csv('/content/gdrive/MyDrive/Predicting-Cancer-using-Support-Vector-Machines-master/Predicting-Cancer-using-Support-Vector-Machines-master/cell_sa
    cell_df.head()
```

```
[ ] ax = cell_df[cell_df['Class'] == 4][0:100].plot(kind='scatter', x='Clump', y='UnifSize', color='DarkBlue', label='malignant');
    cell_df[cell_df['Class'] == 2][0:100].plot(kind='scatter', x='Clump', y='UnifSize', color='Orange', label='benign', ax=ax);
    plt.show()
```

```
[ ] cell_df.dtypes
```

```
[ ] cell_df = cell_df[pd.to_numeric(cell_df['BareNuc'], errors='coerce').notnull()]
    cell_df['BareNuc'] = cell_df['BareNuc'].astype('int')
    cell_df.dtypes
```

```
[ ] feature_df = cell_df[['Clump', 'UnifSize', 'UnifShape', 'MargAdh', 'SingEpiSize', 'BareNuc', 'BlandChrom', 'NormNucl', 'Mit']]
    X = np.asarray(feature_df)
    X[0:5]
```

```
[ ] cell_df['Class'] = cell_df['Class'].astype('int')
    y = np.asarray(cell_df['Class'])
    y [0:20]
```

```
[ ] X_train, X_test, y_train, y_test = train_test_split( X, y, test_size=0.2, random_state=4)
    print ('Train set:', X_train.shape,  y_train.shape)
    print ('Test set:', X_test.shape,  y_test.shape)
```

```
[ ] from sklearn import svm
    clf = svm.SVC(kernel='rbf')
    clf.fit(X_train, y_train)
```

```
[ ] yhat = clf.predict(X_test)
    yhat [0:5]
```

```
[ ] from sklearn.metrics import classification_report, confusion_matrix
    import itertools
```

```
[ ] def plot_confusion_matrix(cm, classes,
                              normalize=False,
                              title='Confusion matrix',
```

```
[ ]                             cmap=plt.cm.Blues):
        """
        This function prints and plots the confusion matrix.
        Normalization can be applied by setting `normalize=True`.
        """
        if normalize:
            cm = cm.astype('float') / cm.sum(axis=1)[:, np.newaxis]
            print("Normalized confusion matrix")
        else:
            print('Confusion matrix, without normalization')

        print(cm)

        plt.imshow(cm, interpolation='nearest', cmap=cmap)
        plt.title(title)
        plt.colorbar()
        tick_marks = np.arange(len(classes))
        plt.xticks(tick_marks, classes, rotation=45)
        plt.yticks(tick_marks, classes)

        fmt = '.2f' if normalize else 'd'
        thresh = cm.max() / 2.
        for i, j in itertools.product(range(cm.shape[0]), range(cm.shape[1])):
            plt.text(j, i, format(cm[i, j], fmt),
                     horizontalalignment="center",
                     color="white" if cm[i, j] > thresh else "black")

        plt.tight_layout()
        plt.ylabel('True label')
        plt.xlabel('Predicted label')
```

```
[ ] # Compute confusion matrix
    cnf_matrix = confusion_matrix(y_test, yhat, labels=[2,4])
    np.set_printoptions(precision=2)

    print (classification_report(y_test, yhat))

    # Plot non-normalized confusion matrix
    plt.figure()
    plot_confusion_matrix(cnf_matrix, classes=['Benign(2)','Malignant(4)'],normalize= False,  title='Confusion matrix')
```

```
from sklearn.metrics import f1_score
f1_score(y_test, yhat, average='weighted')
```

CHAPTER 6

REFERENCES

[1] G. Hemanth, M. Janardhan and L. Sujihelen, "Design and Implementing Brain Tumor Detection Using Machine Learning Approach," *2019 3rd International Conference on Trends in Electronics and Informatics (ICOEI)*, 2019, pp. 1289-1294

[2] Pedapati, Praveena, and Rama Vaishnavi Tannedi. "Brain Tumor Detection using HOG by SVM." (2018).

[3] Rezaei, Kimia, Hamed Agahi, and David C. Wyld. "Segmentation and Classification of Brain Tumor CT Images Using SVM with Weighted Kernel Width." *Computer Science & Information Technology* 39 (2017).

[4] El Boustani, Abdelhakim, and Essaid El Bachari. "MRI Brain Images Compression and Classification Using Different Classes of Neural Networks." In *International Conference on Model and Data Engineering*, pp. 122-134. Springer, Cham, 2019.

[5] Poonguzhali, N., Kagne Raveena Rajendra, T. Mageswari, and T. Pavithra. "Heterogeneous Deep Neural Network for Healthcare Using Metric Learning." In *2019 IEEE International Conference on System, Computation, Automation and Networking (ICSCAN)*, pp. 1-4. IEEE, 2019.

[6] Ghassemi, Navid, Afshin Shoeibi, and Modjtaba Rouhani. "Deep neural network with generative adversarial networks pre-training for brain tumor classification based on MR images." *Biomedical Signal Processing and Control* 57 (2020): 101678.

[7] Arif, Muhammad, F. Ajesh, Shermin Shamsudheen, Oana Geman, Diana Izdrui, and Dragos Vicoveanu. "Brain Tumor Detection and Classification by MRI Using Biologically Inspired Orthogonal Wavelet Transform and Deep Learning Techniques." *Journal of Healthcare Engineering* 2022 (2022).

[8] Polly, F.P., Shil, S.K., Hossain, M.A., Ayman, A. and Jang, Y.M., 2018, January. Detection and classification of HGG and LGG brain tumor using machine learning. In 2018 International Conference on Information Networking (ICOIN) (pp. 813-817). IEEE.

[9] Rashid, M.H.O., Mamun, M.A., Hossain, M.A. and Uddin, M.P., 2018, February. Brain Tumor Detection Using Anisotropic Filtering, SVM Classifier and Morphological Operation from MR Images. In 2018 International Conference on Computer, Communication, Chemical, Material and Electronic Engineering (IC4ME2) (pp. 1-4). IEEE.

[10] Mathew, A. Reema, and P. Babu Anto. "Tumor detection and classification of MRI brain image using wavelet transform and SVM." *2017 International conference on signal processing and communication (ICSPC)*. IEEE, 2017.

[11] Gurbină, Mircea, Mihaela Lascu, and Dan Lascu. "Tumor detection and classification of MRI brain image using different wavelet transforms and support vector machines." In *2019 42nd International Conference on Telecommunications and Signal Processing (TSP)*, pp. 505-508. IEEE, 2019.

[12] https://www.nibib.nih.gov/science-education/science-topics/magnetic-resonance-imaging-mri

[13] https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53

[14] https://towardsdatascience.com/support-vector-machine-introduction-to-machine-learning-algorithms-934a444fca47

[15] https://towardsdatascience.com/hog-histogram-of-oriented-gradients-67ecd887675f

[16] https://deepchecks.com/how-to-check-the-accuracy-of-your-machine-learning-model/

[17] https://www.kaggle.com/jakeshbohaju/brain-tumor