Mini Project Synopsis

On

Brain Tumour Detection Using Deep Learning

Submitted by

212is009, Gudiwada Raghava

Under the Guidance of

Udayaprasad P K

Dept. of Computer Science and Engineering,

NITK, Surathkal

Date of Submission: 03-06-2022

Department of Computer Science and Engineering
National Institute of Technology Karnataka, Surathkal.
2021-2022

Abstract

In today's world fostering personal health care is one of the supremely important. The prevalence of health disorders has placed a significant strain on doctors around the world, making it difficult to treat each patient efficiently. There have been key innovative concepts that have revolutionised the healthcare business to relieve the pressure on doctors. Accurate and speedy outcomes can be generated because to advances in computer technology, and patients can be treated accordingly.

The term "tumour" refers to an unwanted mass in the body. A brain tumour is a mass of brain cells that has grown out of control. To assess cancers in the brain, lung, liver, breast, prostate, and other organs, image modalities such as computed tomography (CT), magnetic resonance imaging (MRI), and ultrasound pictures are used. This research uses X-ray pictures in particular to diagnose brain cancers. The proposed system is to classified the dataset of included the brain tumour images into two sub-types: healthy and brain tumor patients using convolutional neural network (CNN). This experiment was carried out using Python and Google Colab. These precisions enable in the early detection of cancers before they cause physical harm such as paralysis and other disabilities.

keywords: Brain Tumour, Convolutional Neural Network, MRI, Deep Learning, MobileNet

Declaration

I hereby declare that the Report of the P.G. Project Work entitled **Brain Tumour Detection Using Deep Learning** which is being submitted to the National Institute of Technology Karnataka Surathkal, in partial fulfillment of the requirements for the award of the Degree of Master of Technology (M.tech) in CS-IS in the Department of Computer Science, is a bonafide report of the work carried out by us. The material contained in this report has not been submitted to any University or Institution for the award of any degree.

Gudiwada Raghava (212is009)
Department of Computer Science

Certificate

This is to certify that the P.G. Project Work Report entitled **Brain Tumour Detection Using Deep Learning** submitted by Gudiwada Raghava the record of the work carried out is accepted as the P.G. Project Work Report submission in partial fulfillment of the requirements for the award of the degree of Master of Technology (M.Tech) in CS-IS in the Department of Computer Science.

Internal Guide
Udayaprasad P K
(Associate Professor)

Chairman - DPGC
(Signature with Date and Seal)

Contents

List of Figures

1 Introduction

The brain is a vital organ in the human body that regulates the operation of all other organs and aids decision-making. It is the primary control center of the central nervous system, and it is in charge of performing the body's daily voluntary and involuntary functions[1]. The tumour is a fibrous web of undesired tissue growth that proliferates uncontrollably inside our brain. In 2022, the United States is expected to see 1,918,030 new cancer cases and 609,360 cancer fatalities [2], with around 350 deaths per day from lung cancer, the main cause of cancer death. The correct understanding of a brain tumour and its stages is critical for preventing and treating the sickness.To accomplish so, radiologists frequently employ X-ray images to examine brain tumours.

A deep neural network (DNN) is an ANN with multiple hidden layers between the input and output layers. Similar to shallow ANNs, DNNs can model complex non-linear relationships. The main purpose of a neural network is to receive a set of inputs, perform progressively complex calculations on them, and give output to solve real world problems like classification. We restrict ourselves to feed forward neural networks. Content of a picture represent its structural qualities, qualities related to its colour and other important information of pictures. Gateways present across the network search pictures on the basis of extra details.

It would be impossible for health practitioners to digest these huge databases without the aid of a computer, especially when undertaking intensive data analysis. Furthermore, a precise classification of a malignant tumour may prevent people from getting the care they need. Deep learning is well-known for its use in the categorization and modelling of brain cancers. Figure 1 shows the applications of deep learning.

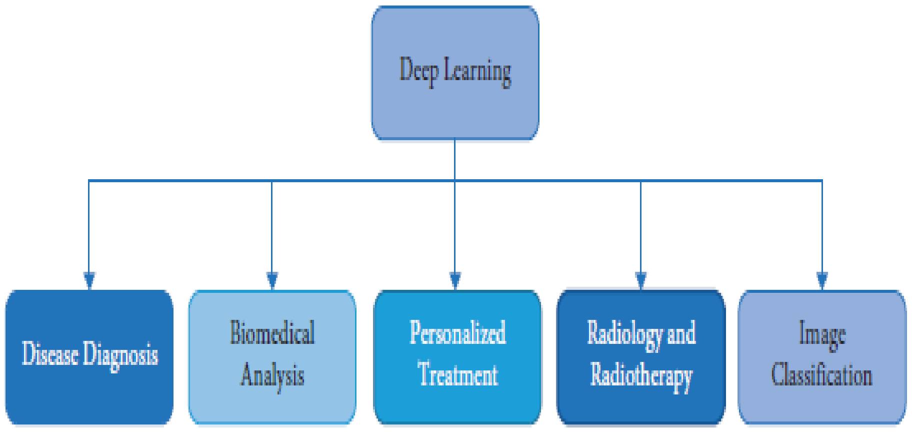


Figure 1: Primary applications of deep learning

The use of artificial intelligence (AI) tools in clinical research is rapidly expanding as a result of its accomplishments in prediction and categorization, particularly in clinical analysis to characterize brain tumors, and it is now widely used in biomedical exploration and constructing robust diagnostic systems for various diseases. Deep learning (DL) is a subset of machine learning that is typically involving data representations and hierarchical features. For descriptors extraction, DL algorithms use an arrangement of many layers of nonlinear processing techniques. Each successive layer's output becomes the input for the subsequent layer, which aids in data abstraction as we go deeper into the network [3]. Convolutional neural networks (CNNs) are a kind of deep learning that are often employed in visual image analysis and are intended to need little preparation. It is based on the biological functions of the human brain [4] and is used to organize data in a variety of arrays [5].

In this paper CNN and MobileNet is used in the classification of malignant and healthy brains. Pre-processing is the first step in the classification process and could be done by using data generators. Then, the images from each type of tumour are shuffled and divided into training (70%) and validation (15%) and testing (15%) of the dataset. Due to the difficulty of analysing bio-medical images, CNN and MobileNet were chosen for their categorization based on depth of feature extraction. Convolutional is the term of a mathematical linear operation used by CNN. At each layer of CNN, the image's dimension is compressed without sacrificing the information needed for training. The model is created using various processing techniques such as Convolve, Maxpooling, Droupout, flatten, and dense. This study focuses on developing a self-defined architecture for CNN and MobilNet models, and then comparing their performance on a brain tumour MRI dataset.

2 Literature Survey

In this paper a Brain Tumour Detection Using Deep Learning have been developed with the use of CNN[8]. The image processing techniques such as histogram equalization, pooling, image enhancement, and feature extraction have been used. The proposed approach using CNN as a classifier for classification of brain images provides a good classification efficiency as compared to other classifiers. The sensitivity, specificity and accuracy is also improved. The proposed approach is computationally effective and yields good result.

In 2020, A. Sadiq, I. F. Nizami et al. [6] assessed level related to Blind pictures. In order to obtain necessary requirement, information related to regular image of fixed wavelet transform was used. Human beings are the final user of picture. As a result, assessment of picture level from the prospective of human beings becomes important. Process in which level of pictures are assessed through human is a very time consuming process. As a result, it becomes necessary to use computational models for the examination of image quality using objective metrics.

In this research work , the Convolutional Neural Network (CNN) was implemented, which drives an overall accuracy of 98.98% in the detection of brain tumour, and healthy images respectively. Deep learning architecture by leveraging 2D convolutional neural networks for the classification of the different types of brain tumor from MRI image slices. In this paper techniques like data acquisition, data preprocessing, pre –model, model optimization and hyper parameter tuning are applied. Moreover the validation was performed on the complete dataset to check for the generalizability of the model.

The brain is an essential organ in the human body which control and coordinates the tasks carried out by the other parts of the body. It is primarily the control center of the central nervous system and is responsible for performing the daily voluntary and involuntary activities in the human body. The tumor is a fibrous mesh of unwanted tissue growth inside our brain that proliferates in an unconstrained way. To prevent and to cure the tumor, magnetic resonance imaging (MRI) is widely used by radiologists to analyze stages of brain tumors. The result of this analysis reveals the presence of the brain tumor.

3 Stages of Brain tumour and stroke lesions

Brain tumours are graded as slow-growing or aggressive. A benign (slow-growing) tumour does not invade the neighboring tissues; in contrast, a malignant (aggressive) tumor propagates itself from an initial site to a secondary site. According to WHO, a brain tumor is categorized into grades *I–IV*. Grades I and II tumors are considered as slow-growing, whereas grades III and IV tumors are more aggressive, and have a poorer prognosis [7]. In this regard, the detail of brain tumor grades is as follows.

Grade I: These tumors grow slowly and do not spread rapidly. These are associated with better odds for long-term survival and can be removed almost completely by surgery. An example of such a tumor is grade 1 pilocyticastrocytoma.

Grade II: These tumors also grow slowly but can spread to neighboring tissues and become higher grade tumors. These tumors can even come back after surgery. Oligodendroglioma is a case of such a tumor.

Grade III: These tumors develop at a faster rate than grade II, and can invade the neighboring tissues. Surgery alone is insufficient for such tumors, and post-surgical radiotherapy or chemotherapy is recommended. An example of such a tumor is anaplastic astrocytoma.

Grade IV: These tumors are the most aggressive and are highly spreadable. They may even use blood vessels for rapid growth. Glioblastoma multiforme is such a type of tumor [8].

Ischemic stroke: Ischemic stroke is an aggressive disease of brain and it is major cause of disability and death around the globe . An ischemic stroke occurs when the blood supply to the brain is cut off, resulting underperfusion (in tissue hypoxia) and dead the advanced tissues in hours [9]. Based on the severity, stroke lesions are categories into different stages such as acute (0–24 h), sub-acute (24 h–2 weeks) and chronic ($\geq 2weeks$)[10].

4 Methodology

The data was collected from the Kaggle database, which is open-source[11]. X-ray images of both healthy and brain tumour patients were included in the collection. A CNN is used for feature extraction. Four Conv2D layers, three Maxpooling2D levels, Dropout, one flatten layer, two dense layers, and a ReLu activation function are included in the model. The Sigmoid is used as an activation function for the last dense layer. Pretrained models were generated using Mobile-NetV2 and other architectures, with small alterations in the last layers, and a head model was created from the basic model. The configurable final layers are Max Pooling, Flatten, Dense, and Dropout. When it comes to extracting visual information, the CNN model comes in handy. The algorithm extracts the features of the input images and learns to differentiate them based on these qualities.

4.1 System Description

The suggested system is depicted in Figure 2 as a block diagram, in which raw photos are tagged and pre-processed to improve the model's performance and robustness. The dataset was then divided into three sections: training (70%), validation (15%), and testing (15%). Certain hyper-parameters are tweaked to improve model efficiency, and the model is eventually trained and tested. Precision, recall, and accuracy are used to evaluate performance computations.

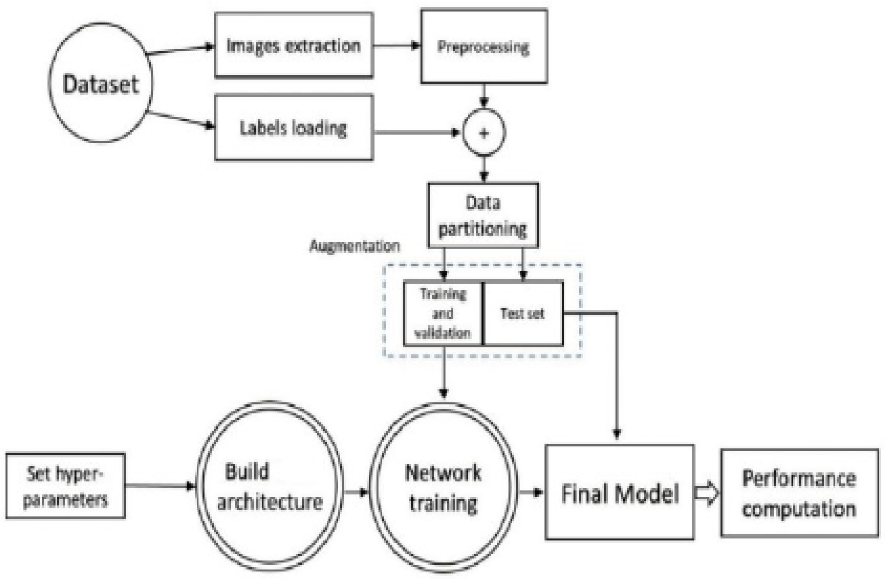


Figure 2: Brain tumour image classification system

4.2 Dataset

A publicly available brain tumour dataset was used in this investigation. This collection includes images of brain X-rays from people who have had brain tumours. This collection contains 2,513 images of brain tumours and 2,087 images of healthy people [11]. Figure 3 shows X-ray scans of a brain tumour and a healthy person.

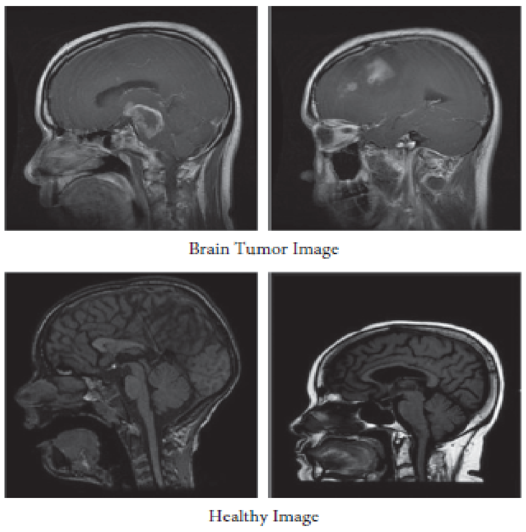


Figure 3: Brain tumor and healthy image

4.3 Tools

Python is an effective data processing tool, particularly when dealing with deep learning techniques. Several Python-based packages are studied in this study to implement our techniques. Here, listed some of the python packages are imported there are numpy, matplotlib, shutil and keras. The entire project setup is done in Google Colaboratory. The best environment for python projects is Google colab.

4.4 Block Diagram of the System

Figure 4 displays a block diagram of a dataset divided into two sections: patients with brain tumours and healthy persons, with input as an X-ray image.

Preprocessing tasks included gathering images, splitting the dataset, and evaluating augmentation approaches before training the model. The findings were improved once the model was fitted and fine-tuned. The confusion matrix, model loss, and model accuracy have been plotted to show how loss and accuracy fluctuate with epoch. Finally, if a user inputs a picture into the model, the output portion may assess if the image depicts a patient with a brain tumour or not. The block diagram presents the entire system in the simplest way possible. Making decisions is an important part of this system, and it plays an important role in the research.

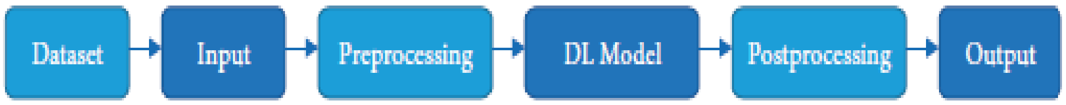


Figure 4: System block diagram

4.5 Preprocessing of Data

There is a preparation phase before the data is trained and evaluated. With a smaller image, it works better. In this study, the scaled picture was 256 x256 pixels. The The next step is to create an array from all of the photographs in the collection. For use in the loop function, the image is converted to an array. The image is used as a preprocessed input by MobileNetV2. The coding is the final stage. Data that has been tagged is converted into a numerical label that can be evaluated and analysed. Following that, the dataset is divided into three sections: 70% for training, 15% for validation, and the last 15% for testing.

4.6 Background of Proposed CNN Architecture

CNNs introduce the idea of hidden layers by using neural networks. When a single vector gets an input picture, the neural network's hidden layers execute a range of neural transformations. Each hidden layer has a huge number of neurons, and the previous layer of each neuron is linked to the subsequent layer of neurons. However, neurons within the same layer are not connected. Each neuron has a distinct function and an input component that is weighted. After functions and weights are applied, each neuron's output is skewed toward a positive or negative value. This method traverses many hidden layers in order to arrive at a conclusion. The final layer is a fully connected layer that mixes all the hidden layers to generate the final result. Scaling is a significant disadvantage in a typical neural network. The proposed architecture model shown in verbatim as Model : sequential.

Deep transfer learning's base layer is the convolutional layer. This is the group that is responsible for deciding the design characteristics. The original image is filtered by this layer. A convolution process multiplies weight ranges with the input. A filter is created by multiplying an array of input data by a 2D collection of weights. A dot product produces a single value when applied to a filter-sized area of the source and filter. This component acts as a buffer between both the input's filter-sized patches and the filter. It is lower than the source and is applied here to multiply several inputs using the same filter. Because it covers the whole frame systematically, the filter is designed as a one-of-a-kind technique for detecting certain types of features.

The pooling layer is used to summarize the characteristics by permitting featured down sampling. Average pooling and max pooling are two extensively utilized pooling approaches that characterize the average existence of a function and its maximum active existence, respectively [12]. Indeed, the pooling layer eliminates superfluous characteristics from the pictures and renders them readable. The layer averages the value of its current view each time it utilizes average pooling. When max pooling is used, the layer picks the largest value from the current view of the filter each time. The max pooling approach picks only the highest value using the matrix size set in each feature map, leading to fewer output neurons. As a result, the picture gets very tiny but the situation stays the same.

The flatten layer converts data from the matrix to a onedimensional array that may be used in the fully linked layer. Vectors may be flattened. In the last step, the classifier in [13] is applied. When considering CNN, last two stages are flattening and fully connected layers. It is converted to a 1D array in preparation for the next fully connected layer of image classification. Fully connected layers are demonstrated to be particularly helpful for computer vision applications and are largely used in CNNs. The CNN technique's first stages are convolution and pooling, which divide the image down into its constituent features and analyze them separately [14]. Each input is linked to all the neurons in a fully connected layer. In this study, both Sigmoid and ReLu activation functions are applied to predict forecast the output. That concludes the CNN's last few layers and most critical layers.

A well-known neural network for image recognition and classification is the Convolutional Neural Network (ConvNet or CNN). CNN is particularly good at extracting complicated information for classification purposes. [15]CNN employs consecutive convolution layers and a nonlinear ReLU function. The feature map is downsized using the Maxpooling layer. Each neuron in the fully connected layer is coupled to every other neuron in the previous dense layer. The function uses a probability distribution to keep all output values in a class between 0 and 1.

The below model is the proposed architecture of CNN Model.

```
    Model: "sequential"
_________________________________________________________________
 Layer (type)                Output Shape              Param #   
=================================================================
 conv2d (Conv2D)             (None, 222, 222, 16)      448       
                                                                 
 conv2d_1 (Conv2D)           (None, 220, 220, 32)      4640      
                                                                 
 max_pooling2d (MaxPooling2D  (None, 110, 110, 32)     0         
 )                                                               
                                                                 
 conv2d_2 (Conv2D)           (None, 108, 108, 64)      18496     
                                                                 
 max_pooling2d_1 (MaxPooling  (None, 54, 54, 64)       0         
 2D)                                                             
                                                                 
 conv2d_3 (Conv2D)           (None, 52, 52, 128)       73856     
                                                                 
 max_pooling2d_2 (MaxPooling  (None, 26, 26, 128)      0         
 2D)                                                             
                                                                 
 dropout (Dropout)           (None, 26, 26, 128)       0         
                                                                 
 flatten (Flatten)           (None, 86528)             0         
                                                                 
 dense (Dense)               (None, 64)                5537856   
                                                                 
 dropout_1 (Dropout)         (None, 64)                0         
                                                                 
 dense_1 (Dense)             (None, 1)                 65        
                                                                 
=================================================================
Total params: 5,635,361
Trainable params: 5,635,361
Non-trainable params: 0
_________________________________________________________________
```

4.7 Transfer Learning with MobileNetV2

Analyzing and categorizing big data are expensive and time-consuming processes. To tackle this issue, it is possible to investigate the well-known transfer learning approach which does not necessitate a large dataset to be applied. Calculations become easier and less costly. Transfer learning is a technique that involves using a model that has been trained on a large dataset to transfer its knowledge to a new model that needs to be trained with much less data than required.

4.8 MobileNetV2 Architecture

MobileNetV2 is a fully convolutional architecture[16] suited for mobile devices. The mobilenetv2 is a model imported from the basic model for that model is modified to get better efficiency. Here, in this model adding the last layer as a dense layer with unit = 1, means becomes flatten the 2D matrix into 1D. Later applied activation function as sigmoid for dense layer. The Sigmoid Function curve looks like an S-shape. The main reason why we use the sigmoid function is that it exists between (0 to 1). Therefore, it is especially used for models where we have to predict the probabiity as an output.

The main goal of this approach was to keep the model from utilizing too many weights and from gaining a wide knowledge of the input. For this dataset, a batch size of 32 images was utilized. As a consequence, 32 images were learnt in a single cycle. In general, the model would grow bigger as the batch size increased. However, this reduces the model's ability to classify certain unusual classes. As a result, there is a tradeoff between generality and specificity when calculating this number. Over a wide range of model sizes, MobileNetV2 enhances the performance. Every line of MobileNetV2 is made up of n times as many repetitive layers [17]. In MobileNet, depth-wise separable is used to factorize the regular state into depth-wise convolution. This entails 13 depth, commonly known as point-wise convolution [18].

The following model is the built architecture of MobileNet Model and modifications to the original model like adding dropouts, BatchNormalization, and adding Activator to dense layer.

```
 Model: "model"
_________________________________________________________________
 Layer (type)                Output Shape              Param #
=================================================================
 input_1 (InputLayer)        [(None, 224, 224, 3)]     0

 conv1 (Conv2D)              (None, 112, 112, 32)      864

 conv1_bn (BatchNormalizatio  (None, 112, 112, 32)     128
 n)

 conv1_relu (ReLU)           (None, 112, 112, 32)      0

 conv_dw_1 (DepthwiseConv2D)  (None, 112, 112, 32)     288

 conv_dw_1_bn (BatchNormaliz  (None, 112, 112, 32)     128
 ation)
```

```
conv_dw_1_relu (ReLU)       (None, 112, 112, 32)      0

conv_pw_1 (Conv2D)          (None, 112, 112, 64)      2048

conv_pw_1_bn (BatchNormaliz  (None, 112, 112, 64)     256
ation)

conv_pw_1_relu (ReLU)       (None, 112, 112, 64)      0

conv_pad_2 (ZeroPadding2D)  (None, 113, 113, 64)      0

conv_dw_2 (DepthwiseConv2D)  (None, 56, 56, 64)       576

conv_dw_2_bn (BatchNormaliz  (None, 56, 56, 64)       256
ation)

conv_dw_2_relu (ReLU)       (None, 56, 56, 64)        0

conv_pw_2 (Conv2D)          (None, 56, 56, 128)       8192

conv_pw_2_bn (BatchNormaliz  (None, 56, 56, 128)      512
ation)

conv_pw_2_relu (ReLU)       (None, 56, 56, 128)       0

conv_dw_3 (DepthwiseConv2D)  (None, 56, 56, 128)      1152

conv_dw_3_bn (BatchNormaliz  (None, 56, 56, 128)      512
ation)

conv_dw_3_relu (ReLU)       (None, 56, 56, 128)       0

conv_pw_3 (Conv2D)          (None, 56, 56, 128)       16384

conv_pw_3_bn (BatchNormaliz  (None, 56, 56, 128)      512
ation)

conv_pw_3_relu (ReLU)       (None, 56, 56, 128)       0

conv_pad_4 (ZeroPadding2D)  (None, 57, 57, 128)       0

conv_dw_4 (DepthwiseConv2D)  (None, 28, 28, 128)      1152

conv_dw_4_bn (BatchNormaliz  (None, 28, 28, 128)      512
ation)
```

```
conv_dw_4_relu (ReLU)       (None, 28, 28, 128)       0

conv_pw_4 (Conv2D)          (None, 28, 28, 256)       32768

conv_pw_4_bn (BatchNormaliz  (None, 28, 28, 256)      1024
ation)

conv_pw_4_relu (ReLU)       (None, 28, 28, 256)       0

conv_dw_5 (DepthwiseConv2D)  (None, 28, 28, 256)      2304

conv_dw_5_bn (BatchNormaliz  (None, 28, 28, 256)      1024
ation)

conv_dw_5_relu (ReLU)       (None, 28, 28, 256)       0

conv_pw_5 (Conv2D)          (None, 28, 28, 256)       65536

conv_pw_5_bn (BatchNormaliz  (None, 28, 28, 256)      1024
ation)

conv_pw_5_relu (ReLU)       (None, 28, 28, 256)       0

conv_pad_6 (ZeroPadding2D)  (None, 29, 29, 256)       0

conv_dw_6 (DepthwiseConv2D)  (None, 14, 14, 256)      2304

conv_dw_6_bn (BatchNormaliz  (None, 14, 14, 256)      1024
ation)

conv_dw_6_relu (ReLU)       (None, 14, 14, 256)       0

conv_pw_6 (Conv2D)          (None, 14, 14, 512)       131072

conv_pw_6_bn (BatchNormaliz  (None, 14, 14, 512)      2048
ation)

conv_pw_6_relu (ReLU)       (None, 14, 14, 512)       0

conv_dw_7 (DepthwiseConv2D)  (None, 14, 14, 512)      4608

conv_dw_7_bn (BatchNormaliz  (None, 14, 14, 512)      2048
ation)

conv_dw_7_relu (ReLU)       (None, 14, 14, 512)       0
```

```
conv_pw_7 (Conv2D)          (None, 14, 14, 512)       262144

conv_pw_7_bn (BatchNormaliz  (None, 14, 14, 512)      2048
ation)

conv_pw_7_relu (ReLU)       (None, 14, 14, 512)       0

conv_dw_8 (DepthwiseConv2D)  (None, 14, 14, 512)      4608

conv_dw_8_bn (BatchNormaliz  (None, 14, 14, 512)      2048
ation)

conv_dw_8_relu (ReLU)       (None, 14, 14, 512)       0

conv_pw_8 (Conv2D)          (None, 14, 14, 512)       262144

conv_pw_8_bn (BatchNormaliz  (None, 14, 14, 512)      2048
ation)

conv_pw_8_relu (ReLU)       (None, 14, 14, 512)       0

conv_dw_9 (DepthwiseConv2D)  (None, 14, 14, 512)      4608

conv_dw_9_bn (BatchNormaliz  (None, 14, 14, 512)      2048
ation)

conv_dw_9_relu (ReLU)       (None, 14, 14, 512)       0

conv_pw_9 (Conv2D)          (None, 14, 14, 512)       262144

conv_pw_9_bn (BatchNormaliz  (None, 14, 14, 512)      2048
ation)

conv_pw_9_relu (ReLU)       (None, 14, 14, 512)       0

conv_dw_10 (DepthwiseConv2D  (None, 14, 14, 512)      4608
)

conv_dw_10_bn (BatchNormali  (None, 14, 14, 512)      2048
zation)

conv_dw_10_relu (ReLU)      (None, 14, 14, 512)       0

conv_pw_10 (Conv2D)         (None, 14, 14, 512)       262144

conv_pw_10_bn (BatchNormali  (None, 14, 14, 512)      2048
zation)

conv_pw_10_relu (ReLU)      (None, 14, 14, 512)       0

conv_dw_11 (DepthwiseConv2D  (None, 14, 14, 512)      4608
)

conv_dw_11_bn (BatchNormali  (None, 14, 14, 512)      2048
zation)

conv_dw_11_relu (ReLU)      (None, 14, 14, 512)       0

conv_pw_11 (Conv2D)         (None, 14, 14, 512)       262144

conv_pw_11_bn (BatchNormali  (None, 14, 14, 512)      2048
zation)

conv_pw_11_relu (ReLU)      (None, 14, 14, 512)       0

conv_pad_12 (ZeroPadding2D)  (None, 15, 15, 512)      0

conv_dw_12 (DepthwiseConv2D  (None, 7, 7, 512)        4608
)

conv_dw_12_bn (BatchNormali  (None, 7, 7, 512)        2048
zation)

conv_dw_12_relu (ReLU)      (None, 7, 7, 512)         0

conv_pw_12 (Conv2D)         (None, 7, 7, 1024)        524288

conv_pw_12_bn (BatchNormali  (None, 7, 7, 1024)       4096
zation)

conv_pw_12_relu (ReLU)      (None, 7, 7, 1024)        0

conv_dw_13 (DepthwiseConv2D  (None, 7, 7, 1024)       9216
)

conv_dw_13_bn (BatchNormali  (None, 7, 7, 1024)       4096
zation)

conv_dw_13_relu (ReLU)      (None, 7, 7, 1024)        0

conv_pw_13 (Conv2D)         (None, 7, 7, 1024)        1048576
```

```
 conv_pw_13_bn (BatchNormali  (None, 7, 7, 1024)       4096
 zation)

 conv_pw_13_relu (ReLU)      (None, 7, 7, 1024)       0

 flatten_1 (Flatten)         (None, 50176)            0

 dense_2 (Dense)             (None, 1)                50177

=================================================================
Total params: 3,279,041
Trainable params: 50,177
Non-trainable params: 3,228,864
_________________________________________________________________
```

5 Materials and methods

The two techniques Mobile-NetV2 and CNN are applied on the brain tumor dataset and their performance on classifying the image is analyzed. The model is compiled with the adam optimization technique and binary crossentropy loss function. The model is generated and trained by providing the training images and the validation images. Once the model is trained, it is tested using the test image set. The dataset is given to the CNN technique. Steps followed in applying CNN on the brain tumor dataset are

1. Import the needed packages
2. Import the data folder(Brain Tumor and Healthy)
3. Set the class labels for images(0 for Brain Tumor and 1 for No Brain Tumor or Healthy)
4. Convert the images into shape(256X256)
5. Normalize the Image
6. Split the images into the train, validation and test set images.
7. Create the sequential model.
8. Compile the model.
9. Apply it on the train dataset(use validation set to evaluate the training performance).
10. Evaluate the model using the test images.
11. Plot the graph comparing the training and validation accuracy.
12. We have to predict the model by using given a image from the available dataset .

The CNN sequential model is generated by implementing different layers. The input image is reshaped into 256x256. The convolve layer is applied on the input image with the relu as activation function, padding as same which means the output images looks like the input image and the number of filters are 3,279,041 for different convolve layers. The max pooling applied with the 2x2 window size and droupout function is called with 20% of droupout. Flatten method is applied to convert the features into one dimensional array. The fully connected layer is done by calling the dense method with the number of units as 256 and relu as the activation function. The output layer has 1 unit to represent the two classes and the sigmoid as activation function. The implementation is done using Python language and are executed in google colab. The model is applied for 100 epoches with the training and the validation dataset.

6 Result and Analysis

Two classifiers are created in the proposed paper: a CNN classifier and an MobileNet classifier. Data set have total 4600 images both of the Classifier was trained using 70%(3220 images) of the data set, validation using 15%(690 images)of data set and tested using the remaining 15% (690 images) of the data set. Only the 690 images from the test image data set are used in the study. It is ensured that all classifiers use the same dataset for the analysis.

The image data are added to the variable named data which is of ndarray datatype. The class labels of the images are also generated and stored in the variable data_target which is also an ndarray. The images are then added inside the dataframe. The image dataset is divided into training, validation and testing dataset. Figure 5 & 6 represents the accuracy and loss obtained when the CNN model is applied on the training and validation dataset When CNN model is applied on the training data for hundred epochs training accuracy obtained is 97.83% and a validation accuracy of 95.89%. The same when applied on the testing data gives 96.31% accuracy.

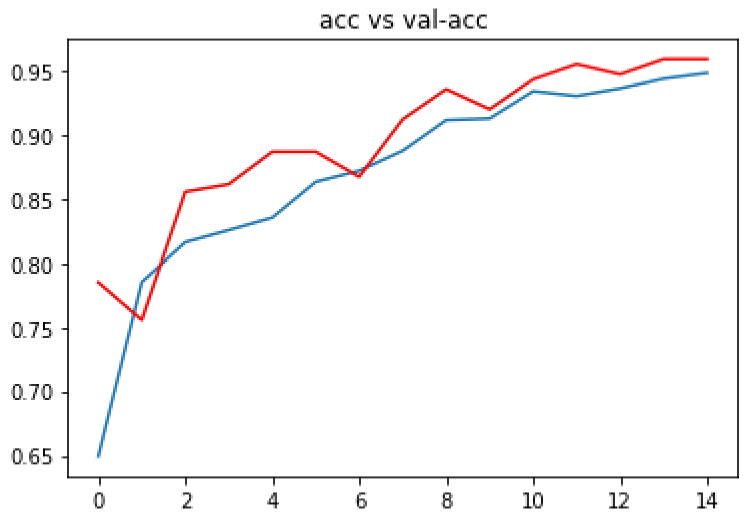


Figure 5: Training/Validation accuracy of CNN model

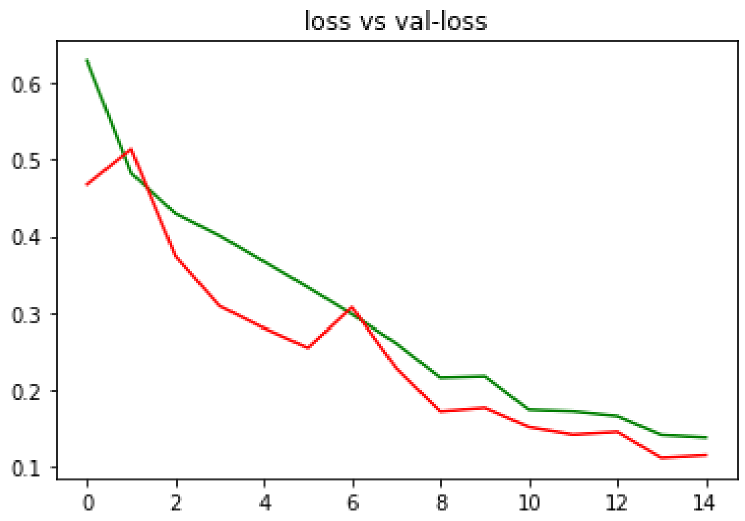


Figure 6: loss of CNN model

The same analysis is applied on MobileNet for compare the two mobiles which is giving the better accuracy. The Figure 7 & 8 represents the accuracy and loss obtained when the MobileNet model is applied on the training and validation dataset When MobileNet model is applied on the training data for hundred epochs training accuracy obtained is 99.18% and a validation accuracy of 99.02%. The same when applied on the testing data gives 98.98% accuracy.

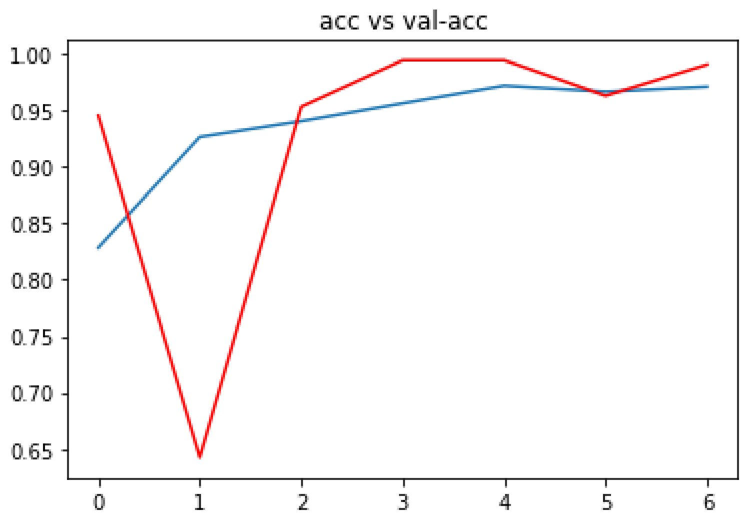


Figure 7: Training/Validation accuracy of MobileNet model

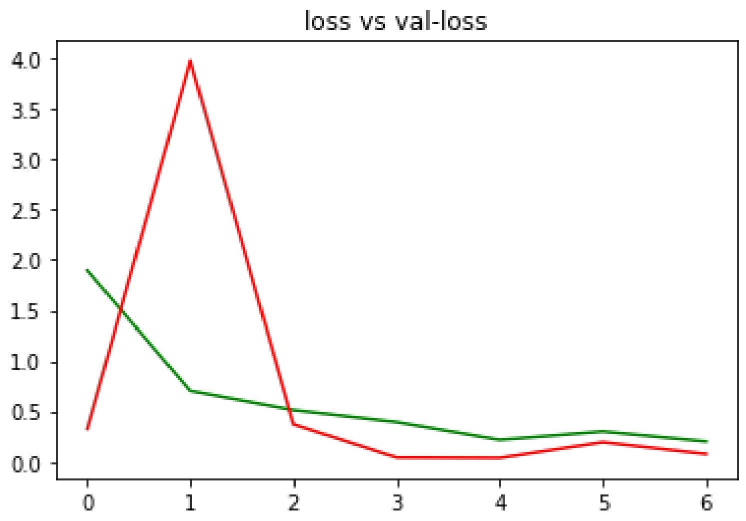


Figure 8: loss of MobileNet model

Model Test: This study also involved real-world assessment, which fed the model data in the form of X-ray scans of the brain. The real-time predictions are depicted in Figures 9 and 10. Figure 9 depicts the output of a brain tumor. The model correctly predicted the input image of a brain tumor. On the other hand, Figure 10 shows a healthy brain.

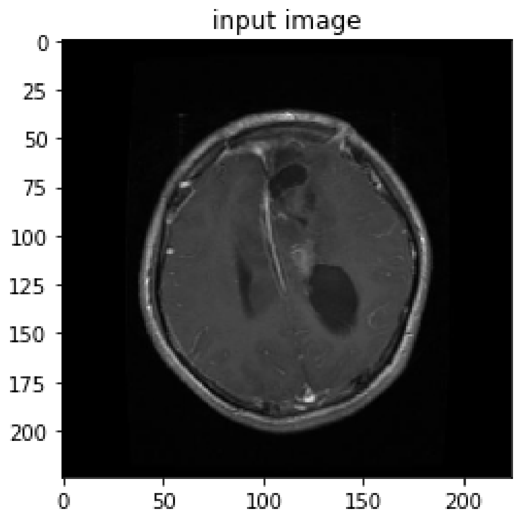


Figure 9: Prediction-brain tumor.

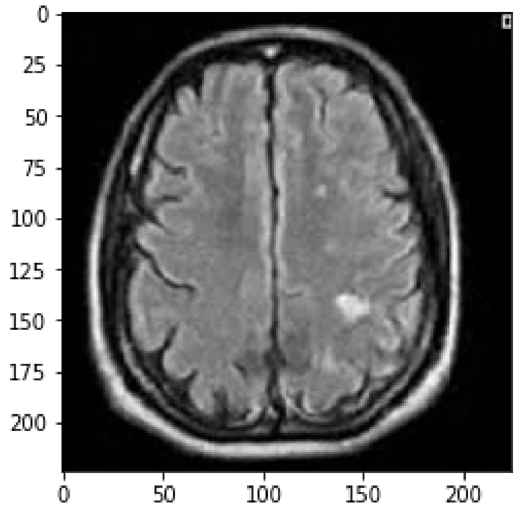
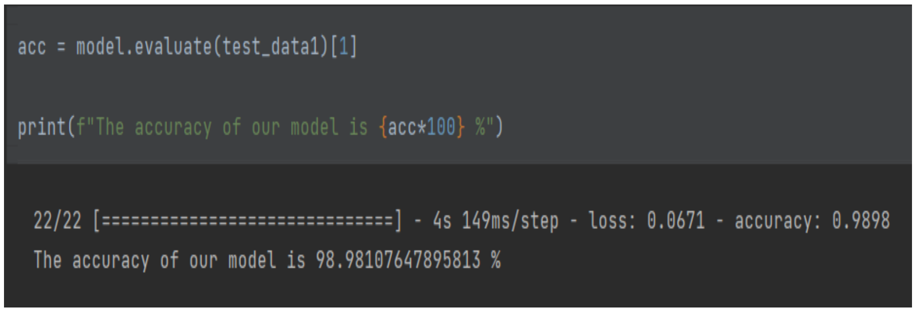


Figure 10: Prediction-healthy.

7 Conclusion

Experimental results from the two classifiers namely, MobilNetV2 and CNN for the classification of Brain Tumours using MR images. Here, in the comparison of the two models, MobileNetv2 has achieved the highest accuracy is 98.98% as shown in the Figure 11. In future optimization techniques can be applied so as to decide the number of layers and filters that can used in a model. As of now for the given dataset the MobileNetV2 proves to be the better technique in predicting the presence of brain tumor.



```
acc = model.evaluate(test_data1)[1]

print(f"The accuracy of our model is {acc*100} %")

 22/22 [==============================] - 4s 149ms/step - loss: 0.0671 - accuracy: 0.9898
 The accuracy of our model is 98.98107647895813 %
```

Figure 11: Accuracy of MobileNet

References

[1] Raheleh Hashemzehi Seyyed Javad Seyyed Mahdavi Maryam Kheirabadi Seyed Reza Kamel 2020 Detection of brain tumors from MRI images base on deep learning using hybrid model CNN and NADE Elsevier B.V. on behalf of Nalecz Institute of Biocybernetics and Biomedical Engineering of the Polish Academy of Sciences Online Publication.

[2] Surveillance Research Program. SEER*Explorer: an interactive website for SEER cancer statistics. National Cancer Institute; 2021. Accessed April 02, 2022. seer.cancer.gov/explorer/

[3] Rajeshwar Nalbalwar Umakant Majhi Raj Patil Prof.Sudhanshu Gonge 2014 Detection of Brain Tumor by using ANN International Journal of Research in Advent Technology.

[4] M. Matsugu, K. Mori, Y. Mitari, and Y. Kaneda, "Subject independent facial expression recognition with robust face detection using a convolutional neural network," Neural Networks, vol. 16, no. 5-6, pp. 555–559, 2003.

[5] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," Nature, vol. 521, no. 7553, pp. 436–444, 2015.

[6] A. Sadiq, I.F. Nizami, S.M. Anwar, M. Majid, Blind image quality assessment using natural scene statistics of stationary wavelet transform, Optik 205 (2020) 164189, https://doi.org/10.1016/j.ijleo.2020.164189.

[7] Johnson DR, Guerin JB, Giannini C, Morris JM, Eckel LJ, Kaufmann TJ (2017) 2016 updates to the WHO brain tumor classification system: what the radiologist needs to know. Radiographics 37:2164–2180.

[8] Wright E, Amankwah EK, Winesett SP, Tuite GF, Jallo G, Carey C et al (2019) Incidentally found brain tumors in the pediatric population: a case series and proposed treatment algorithm. J Neurooncol 141:355–361.

[9] Garrick R, Rotundo E, Chugh SS, Brevik TA (2021) Acute kidney injury in the elderly surgical patient. Emergency general surgery in geriatrics. Springer, New York, pp 205–227.

[10] Lehmann ALCF, Alfieri DF, de Araújo MCM, Trevisani ER, Nagao MR, Pesente FS, Gelinski JR, de Freitas LB, Flauzino T, Lehmann MF, Lozovoy MAB (2021) Carotid intima media thickness measurements coupled with stroke severity strongly predict short-term outcome in patients with acute ischemic stroke: a machine learning study. Metab Brain Dis 36:1747–1761.

[11] Brian Tumor Dataset, https://www.kaggle.com/preetviradiya/brian-tumor-dataset.

[12] J. Brownlee, "A gentle introduction to pooling layers for convolutional neural networks," Machine Learning Mastery, 2020, https://machinelearningmastery.com/pooling-layers-forconvolutionalneural- networks/.

[13] J. Jeong, The Most Intuitive and Easiest Guide for CNN, Medium, New York, NY, USA, 2021, https://towardsdatascience. com/the-most-intuitive-and-easiest-guide-for-convolutional-neural-network3607be47480.

[14] S. Saha, A Comprehensive Guide to Convolutional Neural Networks—theELI5 Way, Medium, New York, NY, USA, 2021, https://towardsdatascience.com/a-comprehensive-guide-to-convolutional-neural-networks-the-eli5-way-3bd2b1164a53.

[15] G.P. Zhang, ”Neural networks for classification: a survey,” IEEE Transact ions on Systems, Man, and Cybernetics, vol. 30, no. 4, pp. 451-462, November 2000.

[16] MobileNetV2, https://paperswithcode.com/method/mobile netv2.

[17] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and C. L. Chieh, “MobileNetV2: inverted residuals and linear bottlenecks,” in Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4510–4520, Salt Lake City, UT, USA, June 2018.

[18] A. G. Howard, M. Zhu, B. Chen et al., “MobileNets: efficient convolutional neural networks for mobile vision applications,” 2017, https://arxiv.org/abs/1704.04861.