JETIR.ORG ISSN: 2349-5162 | ESTD Year : 2014 | Monthly Issue
JOURNAL OF EMERGING TECHNOLOGIES AND INNOVATIVE RESEARCH (JETIR)
An International Scholarly Open Access, Peer-reviewed, Refereed Journal

Multi Classification of Brain Tumor Using Deep Learning(Convolutional Neural Network Architectures)

[1]Dr. S. Anuradha, [2]R V V Sarath, [3]M. Thanush, [4]D. Kavya, [5]K Sai Pranav
[1]Assistant Professor, [2]Student, [3]Student, [4]Student, [5]Student
[1]Computer Science Engineering,
[1]GITAM(Deemed to be University), Visakhapatnam, India

Abstract : During the last several years, we've come across a number of diseases that have had a significant impact on thousands of people. For the new diseases, there were few facilitations and prediction tools, and one of the primary causing issues is a brain tumor. According to current studies, histological examination is utilised to diagnose brain tumors, as opposed to more recent hospital approaches such as magnetic resonance imaging (MRI). This method is not accurate by any machine; it simply specifies brain imaging, and the clinician must determine whether or not a person has a brain tumor. We compared the prediction of brain tumors with EfficientNet b-0 and VGG 16.0 in the Convolutional Neural Network(CNN) Algorithm utilizing deep learning in this project. Glioma, pituitary, meningioma, and no tumor classifications are among the tumors classified by this algorithm. EffiecientNet stores millions of images and retrieves them in an algorithm which was very useful for this project. CNN algorithm, as well as deep learning, are critical in the development of brain tumor prediction tools in our daily lives.

***Keywords* - Brain tumor detection, Convolutional Neural Network, Magnetic Resonance imaging, EfficientNet b-0, VGG 16.**

I.INTRODUCTION

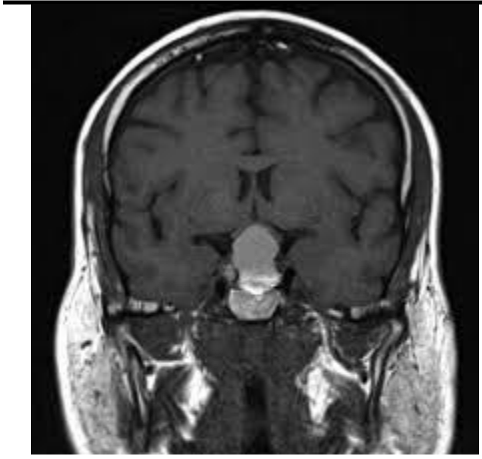
A brain tumor is the growth of abnormal cells in your brain. They can be caused due to heavy exposure to radiation, X-rays, Obesity, Family history, Past cancers, Exposure to chemicals. Brain tumors can begin in your brain as primary brain tumors or spread from your body to brain (metastatic) tumors. Some are benign and some are malignant.

There are several types of brain tumors

i) pituitary tumor
ii) Glioma tumor
iii) meningioma tumor.

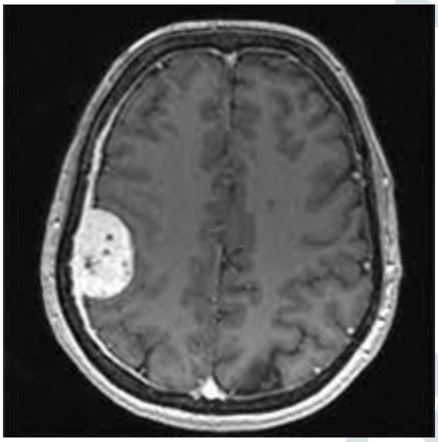
Pituitary Tumor:

An abnormal growth in the pituitary gland is known as a pituitary tumor. The pituitary gland is a tiny gland near the base of the brain that produces hormones.



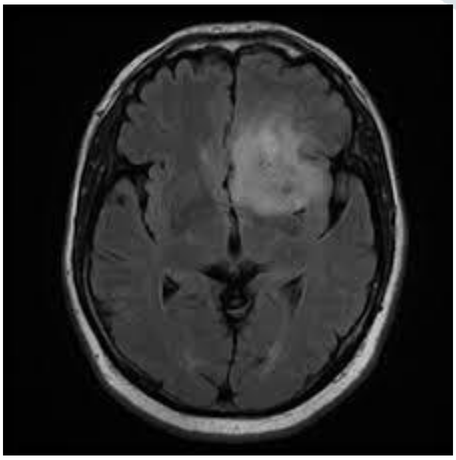
Meningioma Tumor:

A tumor that grows on the membranes that cover the brain and spinal cord immediately within the skull. On the three layers of membranes that make up the meninges, the tumour grows. These tumours normally take a long time to develop. Up to 90% of them are completely safe.



Glioma Tumor:

Gliomas develop when glial cells expand out of control. These cells normally support neurons and aid the operation of your central nervous system. Gliomas most commonly develop in the brain, although they can also develop in the spinal cord. Gliomas are carcinogenic however, some can grow slowly. They're primary brain tumors, which means they come from the brain tissue. Gliomas seldom extend outside the brain or spine, although they can be fatal.



II. CURRENT SCENARIO

In the year of 2022-2023 so far over 25,000 adults in which over 14,000 men and 10,000 women Only in The United States were been diagnosed with primary brain tumors, comparatively the occurrence of brain tumors is found to be extremely rare in India, only 2-3 out of 1,00,000 people are affected/diagnosed with a brain tumor.

Diagnosing a brain tumor requires a neurological exam, scans (MRI, PET, CT) and sometimes biopsy.

In cancer diagnosis, treatment planning, and assessment, early detection and categorization of brain tumors are critical. The present medical technique for detecting, classifying, and grading brain tumors is the same as before. This method is more intrusive and time-consuming, and it can lead to sample mistakes. It is feasible to assist physicians and radiologists in reducing the time necessary for right diagnosis by using computer assisted automated detection and diagnostic systems that strive to produce quicker and more accurate conclusions.

III. EXISTING SYSTEMS

Brain tumors are now identified with the help of a doctor, who will propose many steps to determine whether or not a person has a brain tumor. Several steps are involved in this procedure, including magnetic resonance imaging, a neurological examination, and the collection of a sample of aberrant tissue. These steps include the following: The procedure of analyzing your biopsy sample and establishing which sort of brain tumor you have is time-consuming. Consider getting a second opinion at a medical center where many brain biopsies are analyzed each year if you're unsure about your diagnosis.

IV. ADVANTAGES

1. The use of different types of models may proceed for further research.
2. Various machine learning models can assess time complexity and accuracy, allowing us to measure different things.
3. Various machine learning techniques produce high-quality results.

V. DISADVANTAGES

- Time-Consuming process may take a long time.
- Even while magnetic resonance imaging is performed there needs to be a doctor, to specify it.
- Some approaches are not adoptable for real-time collection of the database implementation.

VI. PROPOSED MODEL

In this project, We compared Convolutional neural network architectures(VGG-16 and EfficiemtNet B0) to detect brain tumors. The input is an image, and the machine predicts whether a person has a brain tumor or not, and if so, what type of tumor they have.

3.1Literature Review

R Lokesh Kumar_2021[1], proposed a ResNet50 architecture in CNN which uses five types of convolution blocks with varying convolution layers and filter sizes, residual network and global average pooling (RNGAP) uses transfer learning.

Sarkar et al.[2] presented a model for MRI images to determine the kind of brain tumour. For classification, a 2D Convolutional Neural Network (CNN) was used, with a "overall accuracy of 91.3 percent for identifying meningioma, glioma, and pituitary tumours."

Abdul Hannah Khan[3], proposed a brain tumor identification system using “IoMT and cnn for data acquisition, pre-processing and application via Hierarchical Deep Learning-Based Brain Tumor Classifier (HDL2B TUMOR-CLASSIFIER)”.

Naresh Ghorpade[4], Report on various types of methods developed for classifying brain tumor images over the last few decades.

Emrah Irmak[5] used CNN and grid search for hyper parameterization to create three models for categorizing brain tumor images, with results of 99, 92, and 98 percent.

A.Jasm[6] presented a survey study on image processing, image mining, transformation, and feature extraction in categorizing brain tumor images using deep convolutional neural networks with a 98 percent accuracy rate.

3.2 Problem Statement and Objective

- In this project we propose to build an EfficientNet CNN model and VGG 16 for the classification of types of Brain tumor and to classify whether the Particular MRI image is having a tumor or not and also what type of tumor.
- The main objective of this project is to compare the two CNN models and choose the best by comparing their accuracy on testing data.

3.3 System Methodology

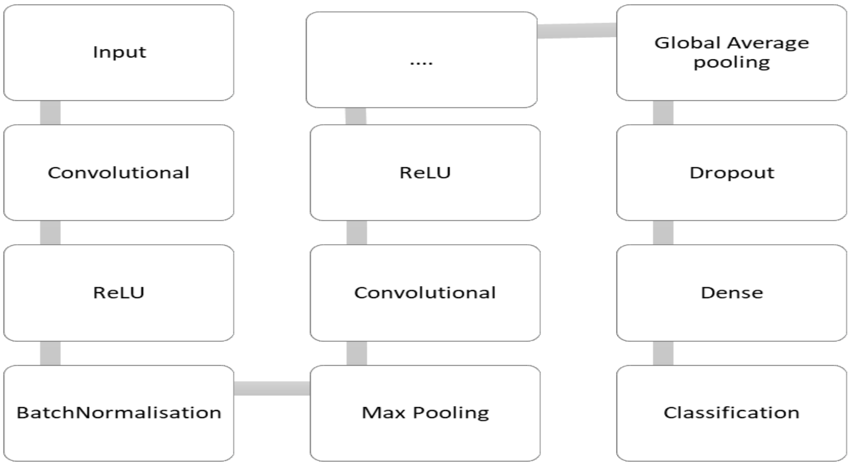
Convolutional Neural Network in use of VGC 16 and EfficientNet b-0

In this project, we have used Convolutional Neural Network with the assistance of EfficientNet architecture through (tensorflow. keras.applications) for classification of brain tumor images. So, basically EfficientNetB0 returns an image classification model, loaded with weights pre-trained on ImageNet.

Adding a global average pooling layer post the efficient net model and then we use dropouts to avoid overfitting and finally add a dense layer with 4 output neurons for classification

Resizing the image shape to 150 x 150 and appending all the images into a single list and converting it into an array. Checking the shape of training images and the number of images in the training data. Converting the labels to one hot encoded labels.

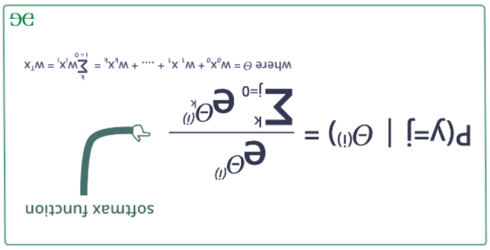
Adding a global average pooling layer post the efficient net model and then we use dropouts to avoid overfitting and finally add a dense layer with 4 output neurons for classification.



CLASSIFICATION USING VGG-16:

VGG 16 architecture is a simple CNN architecture used for ImageNet and large visual database. This architecture in Convolutional neural networks helps to recognize direct visuals of pixel images.

->On the ImageNet dataset, which comprises 14 million images belonging to 1000 classes, this model achieves 77 percent top-5 test accuracy.
->We use softmax function to consider a probability of various attributes.
it is represented as,



CLASSIFICATION USING EFFICIENTNET B-0:

Convolutional Neural Networks (ConvNets) are often built with a fixed resource budget and then scaled up for higher accuracy as more resources become available. We investigate model scaling in depth in this research and find that properly balancing network depth, breadth, and resolution can improve performance. Based on this finding, we present a new scaling approach that uses a simple yet very effective compound coefficient to equally scale all depth/width/resolution dimensions. This strategy is shown to be effective in scaling up MobileNets and ResNets.

I. Overview of Technologies

Python is used to code the project. CNN-style classification systems are employed. The models will be trained and thoroughly checked that they are as precise as possible when classifying a brain tumor. Numpy, pandas, sklearn, keras, and Tensorflow will all be used. Python is used to code the project. CNN layers (VGG-16 and EfficientNet B0) are utilized as classification techniques. The model will be trained and tested to achieve the highest level of accuracy when it comes to classifying a brain tumor. Numpy, pandas, sklearn, keras, Tensorflow, and seaborn will all be used

Tensorflow:
TensorFlow is a Google-developed open-source library aimed towards deep learning applications.

Sklearn:
The sklearn package includes several useful machine learning and statistical modeling methods, including classification, regression, and clustering.

Keras:

Keras is a strong and easy-to-use open source Python framework for building and analyzing deep learning algorithms. It covers the Theano and TensorFlow rapid numerical computing frameworks, which allow you to create and train neural network models with just a few lines of code.

Seaborn:

Seaborn is a matplotlib native library for data display and analysis.

Numpys:

NumPy is used to execute a wide range of array-based mathematical operations.

Pandas:

Pandas is mostly used for tabular data analysis and manipulation in Data Frames.

3.1Data and Sources of Data

DATA:

There are various types of Brain MRI Images such as:

- fMRI
- T1-weighted
- T2-weighted
- Diffusion-weighted MRI
- Fluid Attenuated Inversion recovery MRI
- Gradient Record(GRE)

T1-weighted MRI Images of Brain with Glioma, Meningioma, Pituitary, and No Tumor are the type of MRI images that we have selected to do in this project.

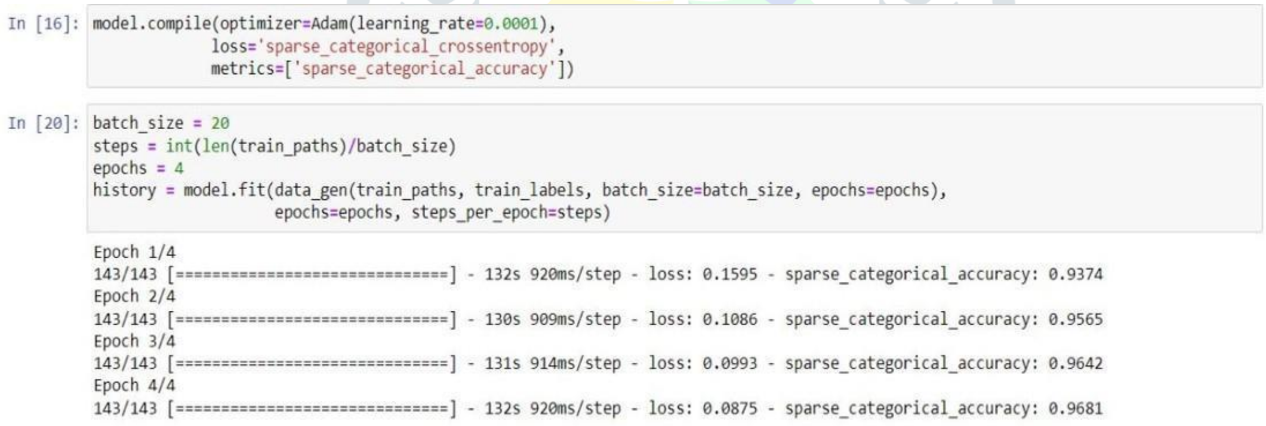
The cerebral spinal fluid around the brain is darker in T1-weighted pictures than it is in T2-weighted images.

Kaggle was used to acquire the data set. The dataset comprises multiple pictures as input for various tumor occurrences to assess whether the provided input image meets all of the reviewed to ensure whether or not the person has a brain tumor.

There are around 2400 and more views on training data, as well as a series of 500 images for testing data in the data.

3.2 Implementation of VGG-16 and Result

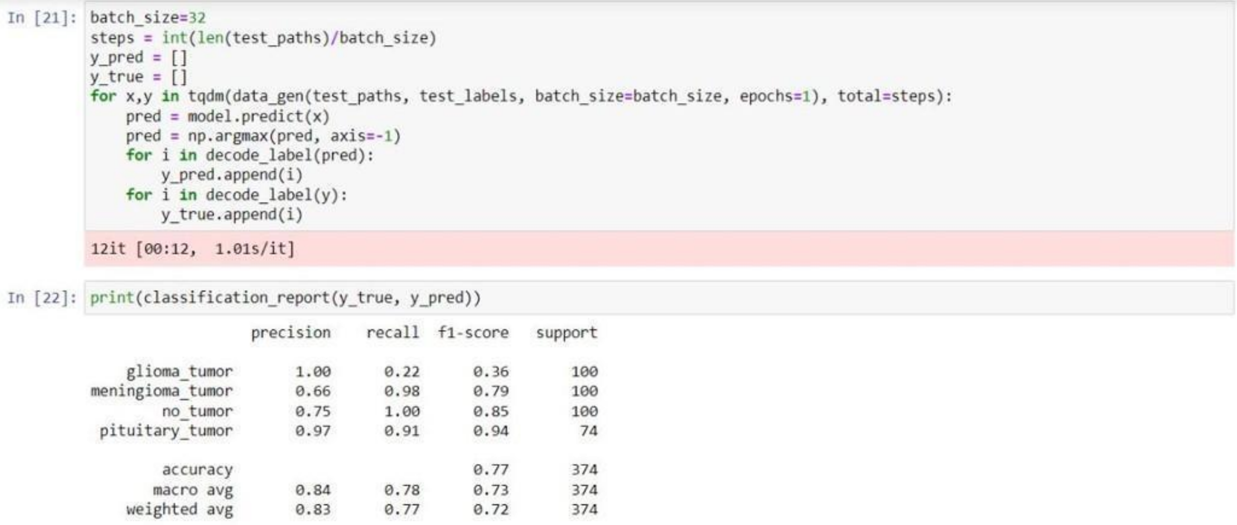
Compiling the model on training data and fitting it on testing data(shown in the results)



```
In [16]: model.compile(optimizer=Adam(learning_rate=0.0001),
                       loss='sparse_categorical_crossentropy',
                       metrics=['sparse_categorical_accuracy'])
```

```
In [20]: batch_size = 20
         steps = int(len(train_paths)/batch_size)
         epochs = 4
         history = model.fit(data_gen(train_paths, train_labels, batch_size=batch_size, epochs=epochs),
                             epochs=epochs, steps_per_epoch=steps)
```

```
Epoch 1/4
143/143 [==============================] - 132s 920ms/step - loss: 0.1595 - sparse_categorical_accuracy: 0.9374
Epoch 2/4
143/143 [==============================] - 130s 909ms/step - loss: 0.1086 - sparse_categorical_accuracy: 0.9565
Epoch 3/4
143/143 [==============================] - 131s 914ms/step - loss: 0.0993 - sparse_categorical_accuracy: 0.9642
Epoch 4/4
143/143 [==============================] - 132s 920ms/step - loss: 0.0875 - sparse_categorical_accuracy: 0.9681
```

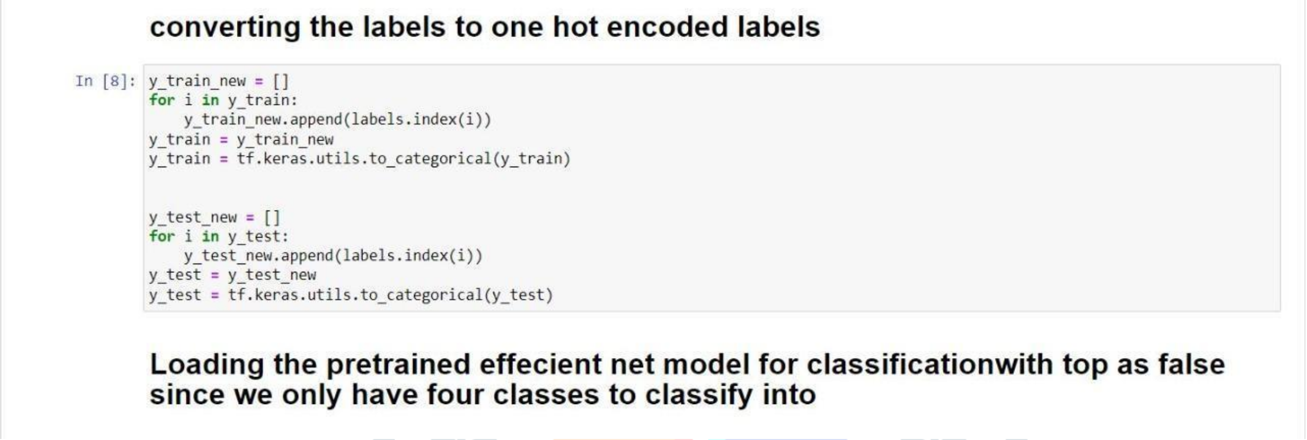


```
In [21]: batch_size=32
         steps = int(len(test_paths)/batch_size)
         y_pred = []
         y_true = []
         for x,y in tqdm(data_gen(test_paths, test_labels, batch_size=batch_size, epochs=1), total=steps):
             pred = model.predict(x)
             pred = np.argmax(pred, axis=-1)
             for i in decode_label(pred):
                 y_pred.append(i)
             for i in decode_label(y):
                 y_true.append(i)

         12it [00:12,  1.01s/it]
```

```
In [22]: print(classification_report(y_true, y_pred))

                   precision    recall  f1-score   support

      glioma_tumor       1.00      0.22      0.36       100
  meningioma_tumor       0.66      0.98      0.79       100
          no_tumor       0.75      1.00      0.85       100
   pituitary_tumor       0.97      0.91      0.94        74

          accuracy                           0.77       374
         macro avg       0.84      0.78      0.73       374
      weighted avg       0.83      0.77      0.72       374
```

Achieved an accuracy score of 77 percent on applying VGG 16 on testing data(shuffled).

3.3 Implementation of EfficientNet B0 Architecture and Result

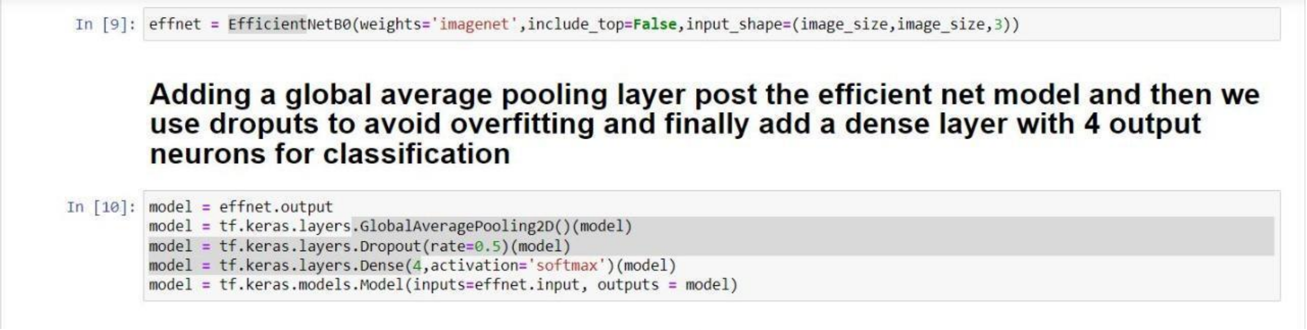


converting the labels to one hot encoded labels

```
In [8]: y_train_new = []
        for i in y_train:
            y_train_new.append(labels.index(i))
        y_train = y_train_new
        y_train = tf.keras.utils.to_categorical(y_train)


        y_test_new = []
        for i in y_test:
            y_test_new.append(labels.index(i))
        y_test = y_test_new
        y_test = tf.keras.utils.to_categorical(y_test)
```

Loading the pretrained effecient net model for classificationwith top as false since we only have four classes to classify into

Loading the efficient net model.

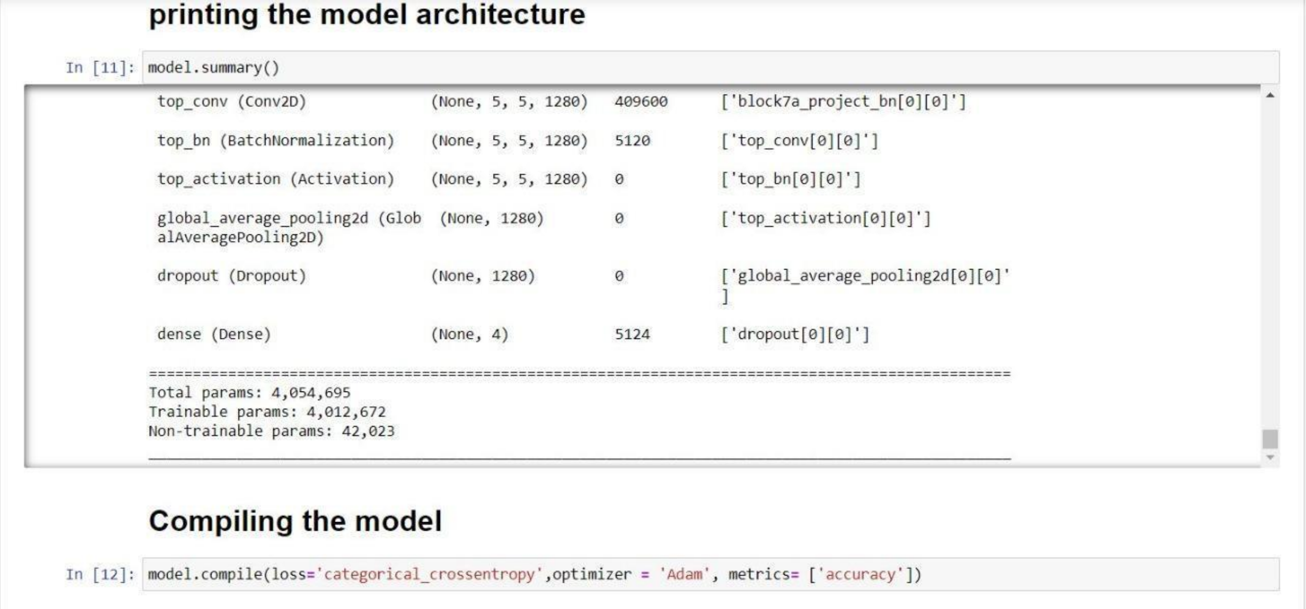


```
In [9]: effnet = EfficientNetB0(weights='imagenet',include_top=False,input_shape=(image_size,image_size,3))
```

Adding a global average pooling layer post the efficient net model and then we use droputs to avoid overfitting and finally add a dense layer with 4 output neurons for classification

```
In [10]: model = effnet.output
         model = tf.keras.layers.GlobalAveragePooling2D()(model)
         model = tf.keras.layers.Dropout(rate=0.5)(model)
         model = tf.keras.layers.Dense(4,activation='softmax')(model)
         model = tf.keras.models.Model(inputs=effnet.input, outputs = model)
```

Adding a global avg pooling layer post the implementation of the model and then use dropout to avoid overfitting of data and finally add a dense layer with 4 output neurons.



printing the model architecture

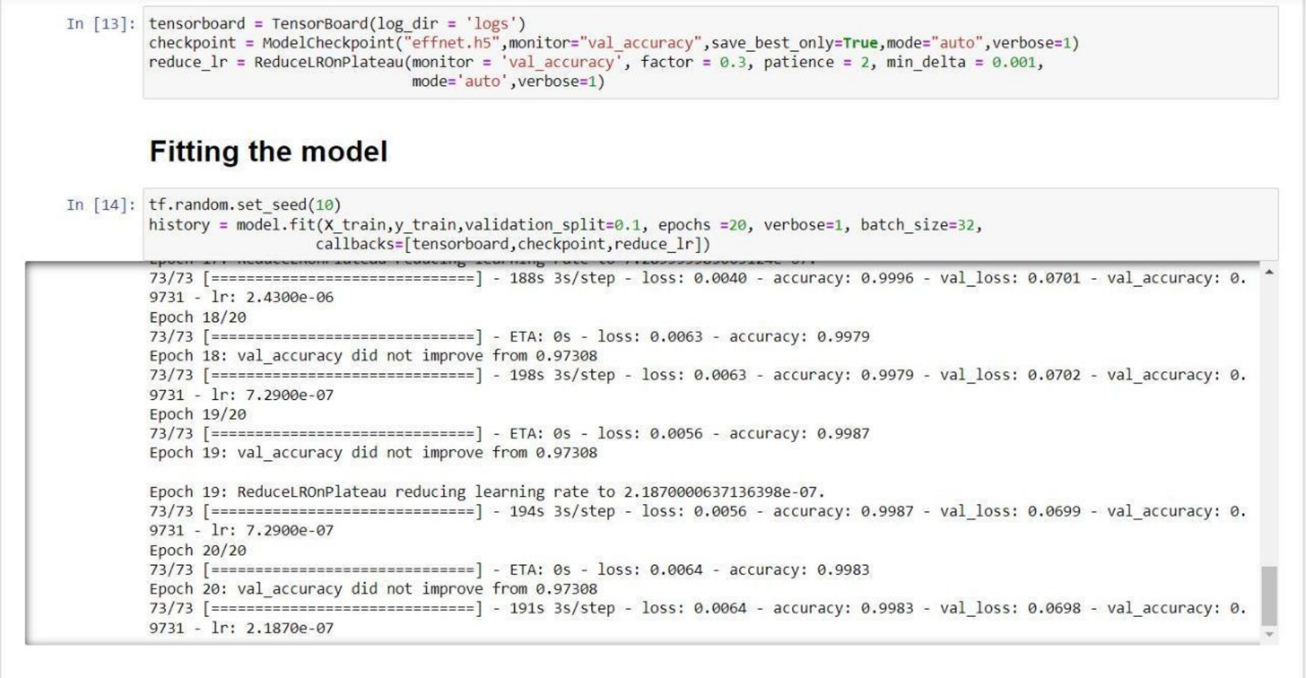
```
In [11]: model.summary()
```

```
 top_conv (Conv2D)              (None, 5, 5, 1280)   409600      ['block7a_project_bn[0][0]']

 top_bn (BatchNormalization)    (None, 5, 5, 1280)   5120        ['top_conv[0][0]']

 top_activation (Activation)    (None, 5, 5, 1280)   0           ['top_bn[0][0]']

 global_average_pooling2d (Glob  (None, 1280)        0           ['top_activation[0][0]']
 alAveragePooling2D)

 dropout (Dropout)              (None, 1280)         0           ['global_average_pooling2d[0][0]'
                                                                  ]

 dense (Dense)                  (None, 4)            5124        ['dropout[0][0]']

==================================================================================================
Total params: 4,054,695
Trainable params: 4,012,672
Non-trainable params: 42,023
__________________________________________________________________________________________________
```

Compiling the model

```
In [12]: model.compile(loss='categorical_crossentropy',optimizer = 'Adam', metrics= ['accuracy'])
```

Fitting of the model



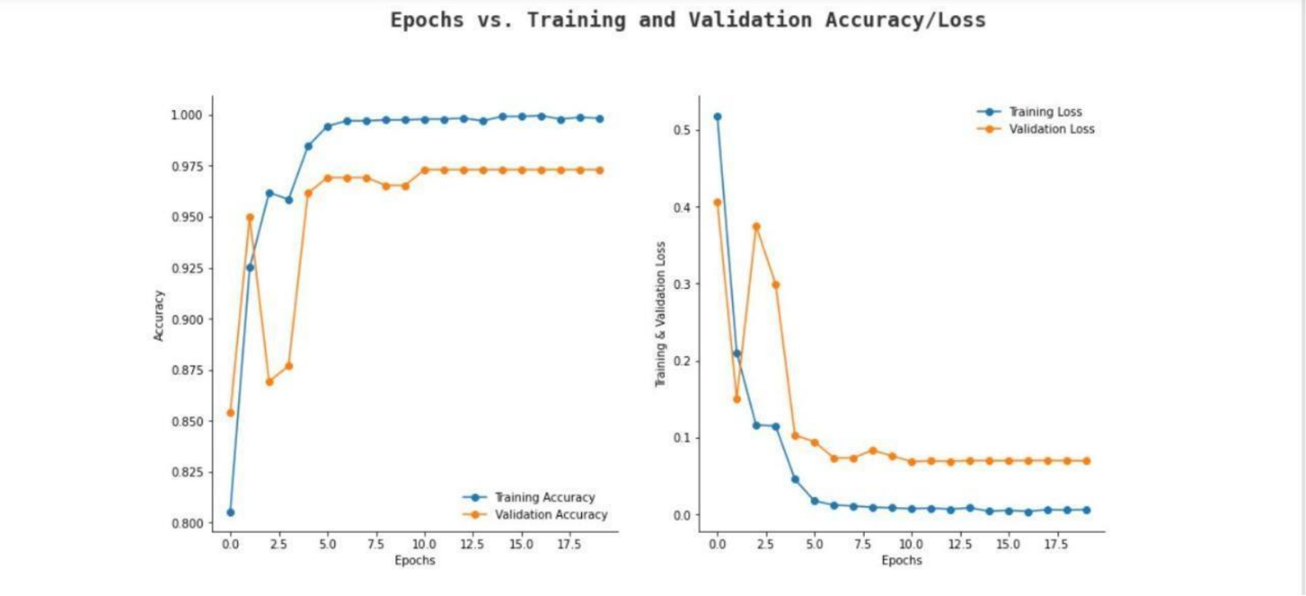
```
In [13]: tensorboard = TensorBoard(log_dir = 'logs')
         checkpoint = ModelCheckpoint("effnet.h5",monitor="val_accuracy",save_best_only=True,mode="auto",verbose=1)
         reduce_lr = ReduceLROnPlateau(monitor = 'val_accuracy', factor = 0.3, patience = 2, min_delta = 0.001,
                                       mode='auto',verbose=1)
```

Fitting the model

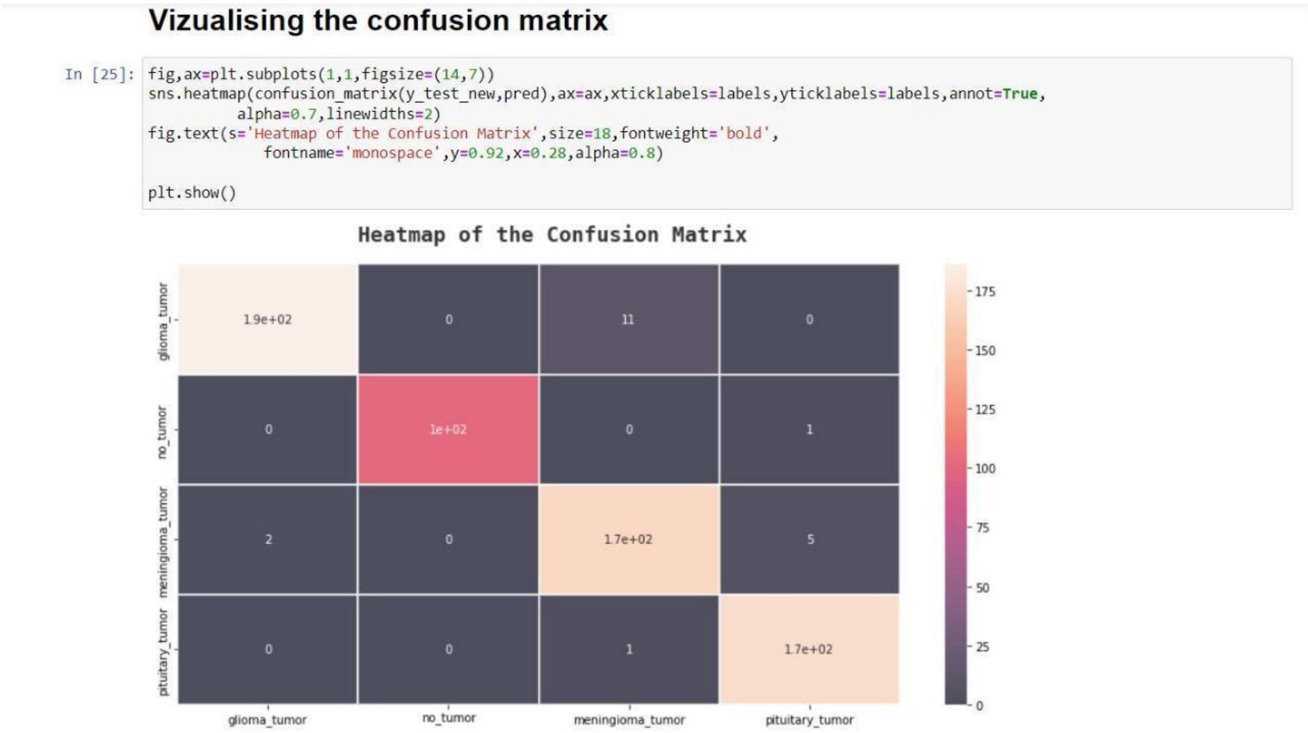
```
In [14]: tf.random.set_seed(10)
         history = model.fit(X_train,y_train,validation_split=0.1, epochs =20, verbose=1, batch_size=32,
                            callbacks=[tensorboard,checkpoint,reduce_lr])
```

```
73/73 [==============================] - 188s 3s/step - loss: 0.0040 - accuracy: 0.9996 - val_loss: 0.0701 - val_accuracy: 0.
9731 - lr: 2.4300e-06
Epoch 18/20
73/73 [==============================] - ETA: 0s - loss: 0.0063 - accuracy: 0.9979
Epoch 18: val_accuracy did not improve from 0.97308
73/73 [==============================] - 198s 3s/step - loss: 0.0063 - accuracy: 0.9979 - val_loss: 0.0702 - val_accuracy: 0.
9731 - lr: 7.2900e-07
Epoch 19/20
73/73 [==============================] - ETA: 0s - loss: 0.0056 - accuracy: 0.9987
Epoch 19: val_accuracy did not improve from 0.97308

Epoch 19: ReduceLROnPlateau reducing learning rate to 2.1870000637136398e-07.
73/73 [==============================] - 194s 3s/step - loss: 0.0056 - accuracy: 0.9987 - val_loss: 0.0699 - val_accuracy: 0.
9731 - lr: 7.2900e-07
Epoch 20/20
73/73 [==============================] - ETA: 0s - loss: 0.0064 - accuracy: 0.9983
Epoch 20: val_accuracy did not improve from 0.97308
73/73 [==============================] - 191s 3s/step - loss: 0.0064 - accuracy: 0.9983 - val_loss: 0.0698 - val_accuracy: 0.
9731 - lr: 2.1870e-07
```

Plotting the accuracy loss graph



Visualization of Confusion Matrix



Vizualising the confusion matrix

```
In [25]: fig,ax=plt.subplots(1,1,figsize=(14,7))
         sns.heatmap(confusion_matrix(y_test_new,pred),ax=ax,xticklabels=labels,yticklabels=labels,annot=True,
                    alpha=0.7,linewidths=2)
         fig.text(s='Heatmap of the Confusion Matrix',size=18,fontweight='bold',
                       fontname='monospace',y=0.92,x=0.28,alpha=0.8)

         plt.show()
```

Heatmap of the Confusion Matrix

	glioma_tumor	no_tumor	meningioma_tumor	pituitary_tumor
glioma_tumor	1.9e+02	0	11	0
no_tumor	0	1e+02	0	1
meningioma_tumor	2	0	1.7e+02	5
pituitary_tumor	0	0	1	1.7e+02

Achieved an accuracy score of 97 percent on applying EfficientNet B0 on Testing Data(Shuffled).

Button-based widget for uploading the image and predicting:

```
In [20]: uploader = widgets.FileUpload()
         display(uploader)
```

```
In [21]: button = widgets.Button(description='Predict')
         out = widgets.Output()
         def on_button_clicked(_):
             with out:
                 clear_output()
                 try:
                     img_pred(uploader)

                 except:
                     print('No Image Uploaded/Invalid Image File')
         button.on_click(on_button_clicked)
         widgets.VBox([button,out])
```

```
The Model predicts that it is a Meningioma Tumor
```

3.4Conclusion and Future Scope

This project multi-classification of Brain tumor images for the purpose of early diagnosis of the tumor. The application of EfficientNet CNN model on classifying the brain tumor images(2800 training and 500 testing) has succeeded by attaining an accuracy of 97-98(due to random shuffling of images every time it trains) percent on testing images in comparison to the VGG-16 CNN model which resulted in 77 percent on testing images. This project can be assisted for medical diagnosis.

The proposed approach needs a lot of training data set for better accuracy. In medical diagnosis gathering of medical data takes long time and in few cases the data sets might not be available. In order to overcome this CNN algorithm is best for recognising of tumors from MRI images and gives high accuracy. It also shortens the time required for correct diagnosis for physicians and radiologists.

REFERENCES

[1] R Lokesh Kumar, Multi-class brain tumor classification using residual network and global average pooling.(April 2021)Kluwer Academic Publishers.

[2] Sarkar al Joel S Joy, A review of brain tumor segmentation of MRI images using machine learning algorithm(2021), Journal of Imaging.

[3] Abdul Hannah Khan, Intelligent Model for Brain Tumor Identification Using Deep Learning(January,2022), Hindawi Publications.

[4] Naresh Ghorpude, Brain MRI Segmentation and Tumor Detection: Challenges, Techniques and Applications(IEEE Xplore),(May 2021), IEEE Publishers.

[5] Multi-Classification of Brain Tumor MRI Images Using Deep Convolutional Neural Network with Fully Optimized Framework Emrah Irmak1(SpringerLink, April 2021).

[6] A Survey Paper on Image Mining Techniques and Classification Brain Tumor Dhamea A. Jasm, Murtadha M Hamad, Azmi Tawfek Hussein Alraw.(Journals of Physics Conference Series,2021).