In [1]:
```
# Task: Improve the speed and accuracy of detecting and localising Brain tumor based on MRI scans
# Which will drastically remove the cost of cancer diagnosis and help in early diagnoses of brain tumors
# develop a model to detect and localise Brain tumor
# You have been provided with 3900 brain MRI scans along with their brain tumor location

# Processes: Using  AI to improve diseases detection and location process
# Build and train a segmentation Res-U-Net model to localise brain tumor in images

# Talk about AI in health care
# Deep learning
# for the training data, I have two images; the brain MRI (to show if it has cancer or not) and a mask associat


# First stage we are going to train a ResNet deeplearning classifier (Residual Neural Network) model
# Second stage, if a tumor has been detected after classification, then we need to localize the tumor using Res
# Function of the image segmentation is to understand and extract information from images at the pixel-level
```

In [2]:
```
pip install opencv-python
```

```
Requirement already satisfied: opencv-python in c:\users\olayi\anaconda3\lib\site-packages (4.9.0.80)
Requirement already satisfied: numpy>=1.21.2 in c:\users\olayi\anaconda3\lib\site-packages (from opencv-python)
(1.24.3)
Note: you may need to restart the kernel to use updated packages.
```

In [3]:
```
# Importing important libraries
import pandas as pd
import numpy as np
import seaborn as sns
import matplotlib.pyplot as plt
import zipfile
import cv2
import tensorflow as tf
from skimage import io
from tensorflow.python.keras import Sequential
from tensorflow.keras import layers, optimizers
from tensorflow.keras.applications import DenseNet121
from tensorflow.keras.applications.resnet50 import ResNet50
from tensorflow.keras.layers import *
from tensorflow.keras.models import Model, load_model
from tensorflow.keras.initializers import glorot_uniform
from tensorflow.keras.utils import plot_model
from tensorflow.keras.callbacks import ReduceLROnPlateau, EarlyStopping, ModelCheckpoint, LearningRateScheduler
from IPython.display import display
from tensorflow.keras import backend as k
from sklearn.preprocessing import StandardScaler, normalize
import os
import glob
import random

%matplotlib inline

from warnings import filterwarnings
filterwarnings('ignore')
```

```
WARNING:tensorflow:From C:\Users\olayi\anaconda3\Lib\site-packages\keras\src\losses.py:2976: The name tf.losses
.sparse_softmax_cross_entropy is deprecated. Please use tf.compat.v1.losses.sparse_softmax_cross_entropy instea
d.

C:\Users\olayi\anaconda3\Lib\site-packages\paramiko\transport.py:219: CryptographyDeprecationWarning: Blowfish
has been deprecated
  "class": algorithms.Blowfish,
```

In [4]:
```
# Define the path to the downloads directory
downloads_path = os.path.join(os.path.expanduser('~'), 'Downloads')

# Check if the directory exists
if os.path.exists(r"C:\Users\olayi\Downloads\Projects\Health AI\Brain_Tumor\Healthcare+AI+Datasets\Healthcare A
    print("Downloads directory found at:", downloads_path)
else:
    print("Downloads directory not found!")

# Change directory to the downloads folder
os.chdir(r"C:\Users\olayi\Downloads\Projects\Health AI\Brain_Tumor\Healthcare+AI+Datasets\Healthcare AI Dataset
```

```
Downloads directory found at: C:\Users\olayi\Downloads
```

In [5]:
```
# Loading the dataset that contain Brain MRI and their corresponding mask
# loaded the mask path
# mask can be represented by associating pixel values with their coordinates
brain_df = pd.read_csv(r"C:\Users\olayi\Downloads\Projects\Health AI\Brain_Tumor\Healthcare+AI+Datasets\Healthc
```

In [6]:
```
brain_df.info()
```

```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3929 entries, 0 to 3928
Data columns (total 4 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   patient_id  3929 non-null   object
 1   image_path  3929 non-null   object
 2   mask_path   3929 non-null   object
 3   mask        3929 non-null   int64 
dtypes: int64(1), object(3)
memory usage: 122.9+ KB
```

In [7]:
```
brain_df.head()
```

Out[7]:

	patient_id	image_path	mask_path
0	TCGA_CS_5395_19981004	TCGA_CS_5395_19981004/TCGA_CS_5395_19981004_1.tif	TCGA_CS_5395_19981004/TCGA_CS_5395_19981004_1_...
1	TCGA_CS_5395_19981004	TCGA_CS_4944_20010208/TCGA_CS_4944_20010208_1.tif	TCGA_CS_4944_20010208/TCGA_CS_4944_20010208_1_...
2	TCGA_CS_5395_19981004	TCGA_CS_4941_19960909/TCGA_CS_4941_19960909_1.tif	TCGA_CS_4941_19960909/TCGA_CS_4941_19960909_1_...
3	TCGA_CS_5395_19981004	TCGA_CS_4943_20000902/TCGA_CS_4943_20000902_1.tif	TCGA_CS_4943_20000902/TCGA_CS_4943_20000902_1_...
4	TCGA_CS_5395_19981004	TCGA_CS_5396_20010302/TCGA_CS_5396_20010302_1.tif	TCGA_CS_5396_20010302/TCGA_CS_5396_20010302_1_...

In [8]:
```
# Percentage of healthy sampples( without tumor)
# which means we have unbalanced dataset
len(brain_df[brain_df['mask'] == 0])/len(brain_df)
```

Out[8]:
```
0.6505472130313057
```

In [9]:
```
# counts of healthy samples and unhealthy samples
brain_df['mask'].value_counts()
```

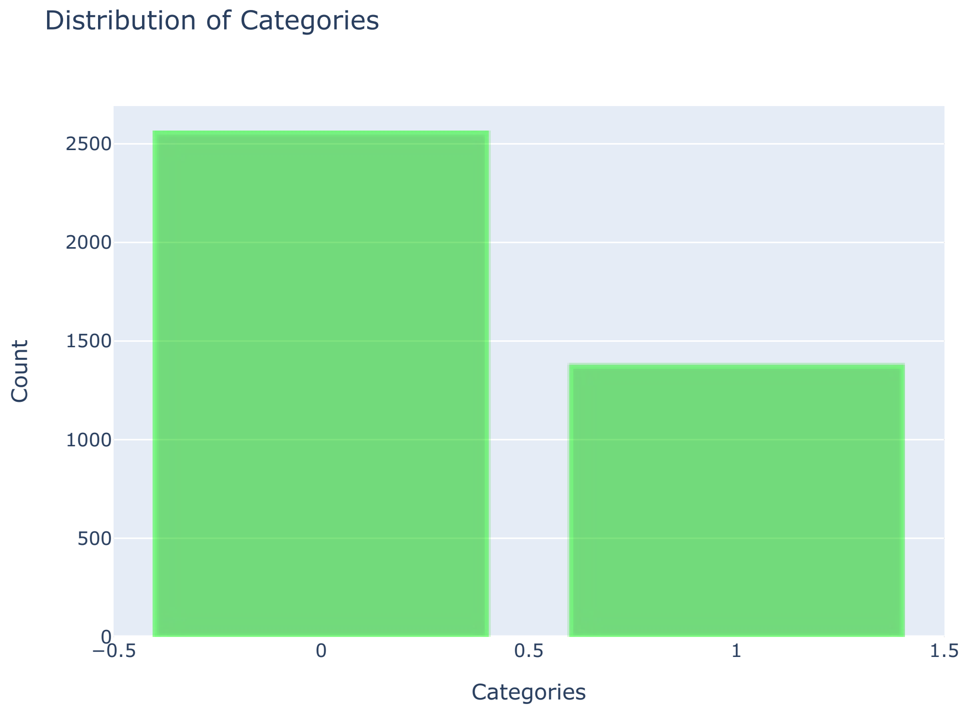
Out[9]:
```
0    2556
1    1373
Name: mask, dtype: int64
```

Performing Data Exploration

In [10]:
```
# To know the number of categories we have in the mask
brain_df['mask'].value_counts().index
```

Out[10]:
```
Int64Index([0, 1], dtype='int64')
```

In [11]:
```
# using plotly to show the categories
# plotly shows interactive bar chart
import plotly.graph_objects as go

fig = go.Figure([go.Bar(x = brain_df['mask'].value_counts().index, y = brain_df['mask'].value_counts())])
fig.update_traces(marker_color = 'rgb(0,200,0)', marker_line_color = 'rgb(0,255,0)', marker_line_width = 3, opa
fig.update_layout(title='Distribution of Categories',
                  xaxis_title='Categories',
                  yaxis_title='Count')

fig.show()
```



In [12]:
```
# mask path
brain_df['mask_path']
```

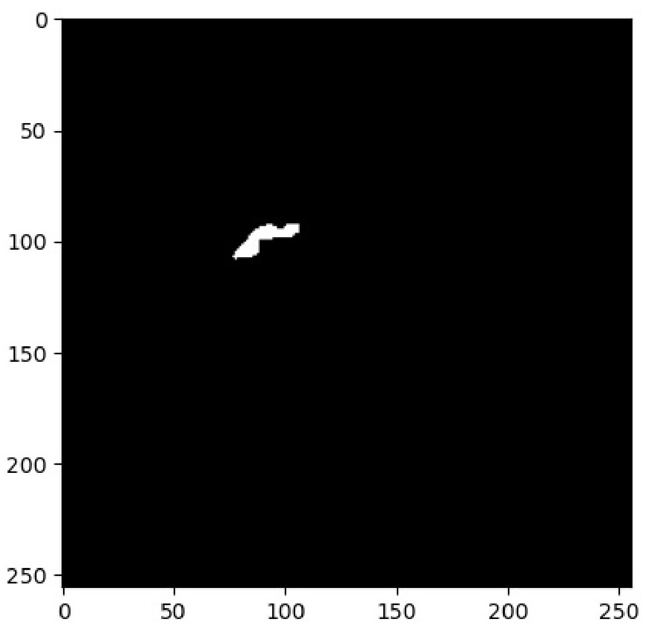
Out[12]:
```
0       TCGA_CS_5395_19981004/TCGA_CS_5395_19981004_1_...
1       TCGA_CS_4944_20010208/TCGA_CS_4944_20010208_1_...
2       TCGA_CS_4941_19960909/TCGA_CS_4941_19960909_1_...
3       TCGA_CS_4943_20000902/TCGA_CS_4943_20000902_1_...
4       TCGA_CS_5396_20010302/TCGA_CS_5396_20010302_1_...
                              ...
3924    TCGA_HT_A61B_19991127/TCGA_HT_A61B_19991127_86...
3925    TCGA_HT_A61A_20000127/TCGA_HT_A61A_20000127_87...
3926    TCGA_HT_A61B_19991127/TCGA_HT_A61B_19991127_87...
3927    TCGA_HT_A61A_20000127/TCGA_HT_A61A_20000127_88...
3928    TCGA_HT_A61B_19991127/TCGA_HT_A61B_19991127_88...
Name: mask_path, Length: 3929, dtype: object
```

In [13]:
```
# Image path
brain_df['image_path']
```

Out[13]:
```
0       TCGA_CS_5395_19981004/TCGA_CS_5395_19981004_1.tif
1       TCGA_CS_4944_20010208/TCGA_CS_4944_20010208_1.tif
2       TCGA_CS_4941_19960909/TCGA_CS_4941_19960909_1.tif
3       TCGA_CS_4943_20000902/TCGA_CS_4943_20000902_1.tif
4       TCGA_CS_5396_20010302/TCGA_CS_5396_20010302_1.tif
                              ...
3924    TCGA_HT_A61B_19991127/TCGA_HT_A61B_19991127_86...
3925    TCGA_HT_A61A_20000127/TCGA_HT_A61A_20000127_87...
3926    TCGA_HT_A61B_19991127/TCGA_HT_A61B_19991127_87...
3927    TCGA_HT_A61A_20000127/TCGA_HT_A61A_20000127_88...
3928    TCGA_HT_A61B_19991127/TCGA_HT_A61B_19991127_88...
Name: image_path, Length: 3929, dtype: object
```

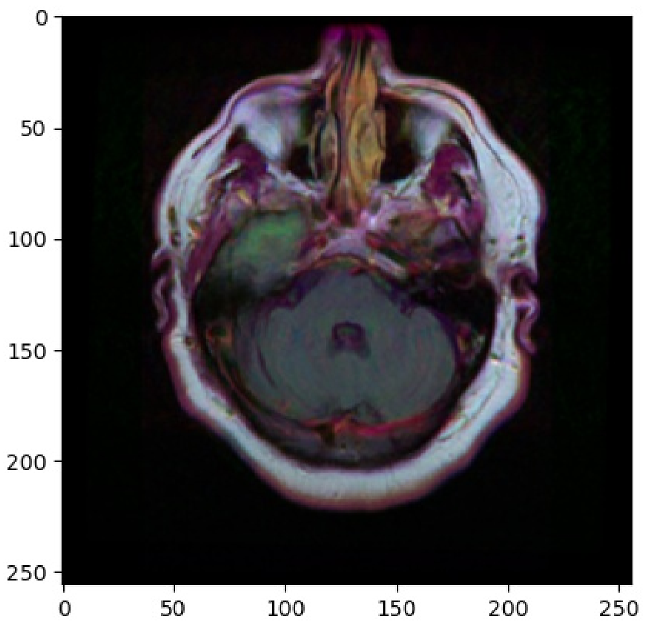
In [14]:
```
# using openCV to read the image and plt to show the image
plt.imshow(cv2.imread(brain_df.mask_path[623]))
```

Out[14]:
```
<matplotlib.image.AxesImage at 0x169bd1a2ed0>
```



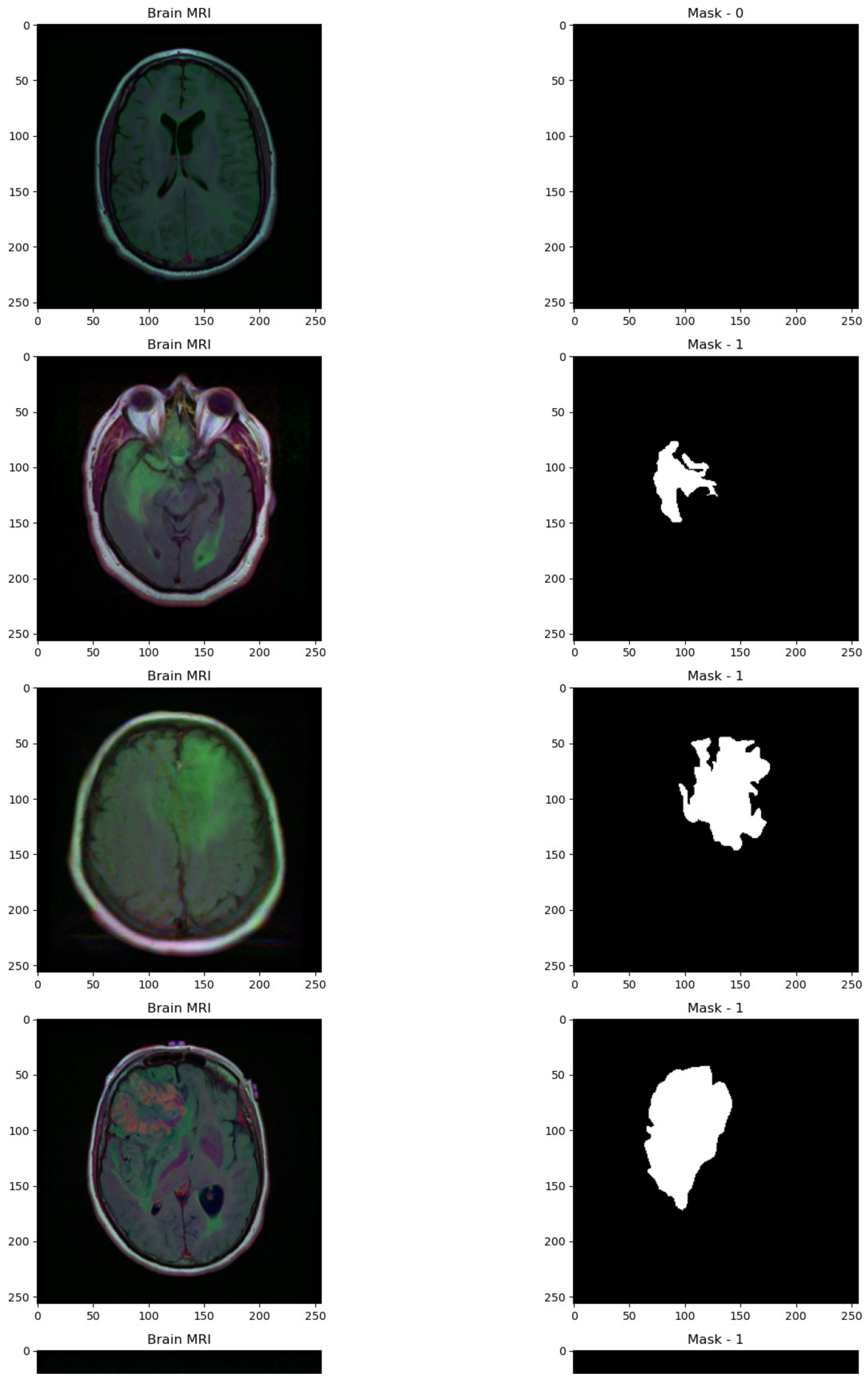
```
In [15]: plt.imshow(cv2.imread(brain_df.image_path[623]))
```

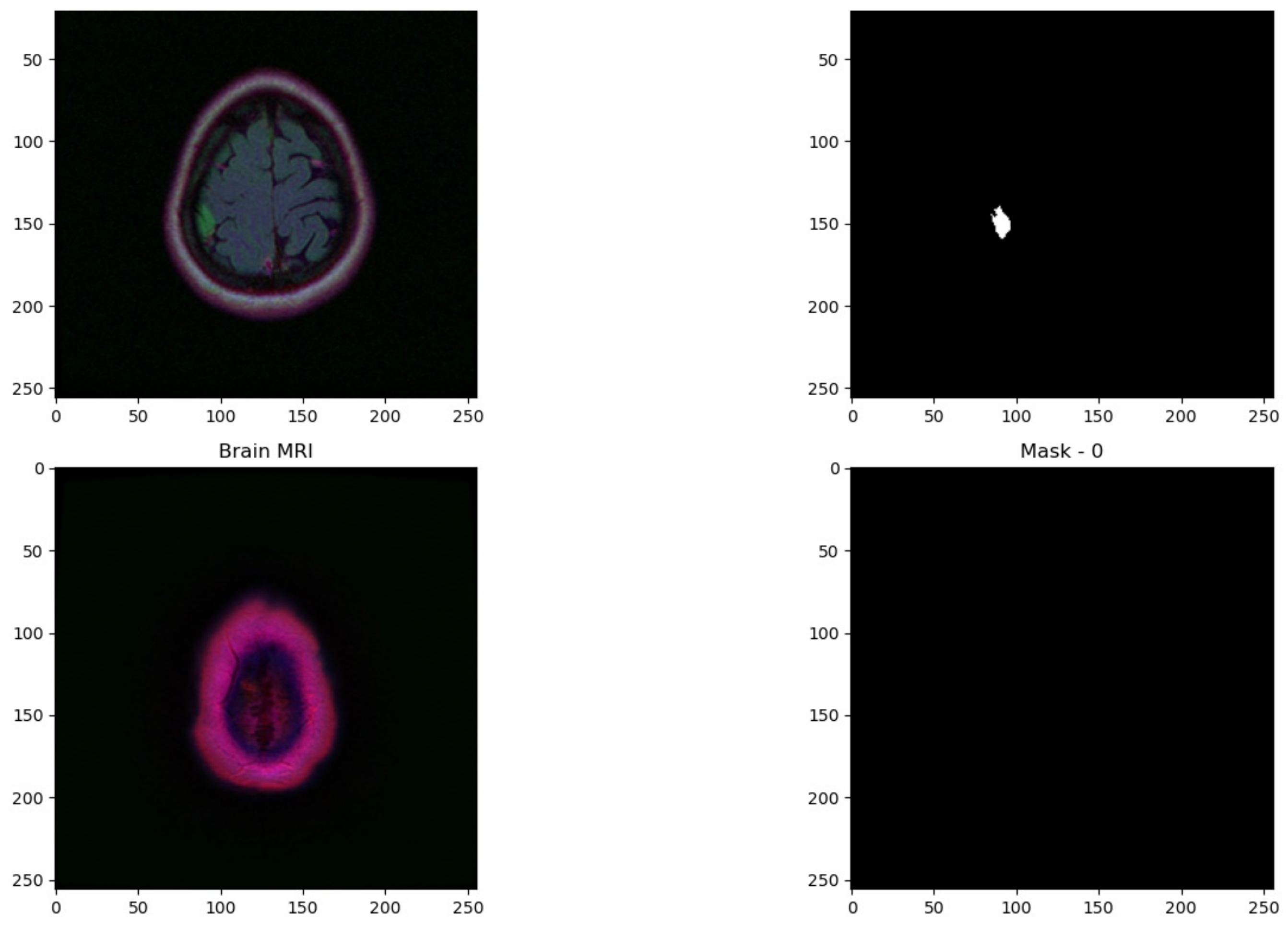
```
Out[15]: <matplotlib.image.AxesImage at 0x169bd1d3310>
```



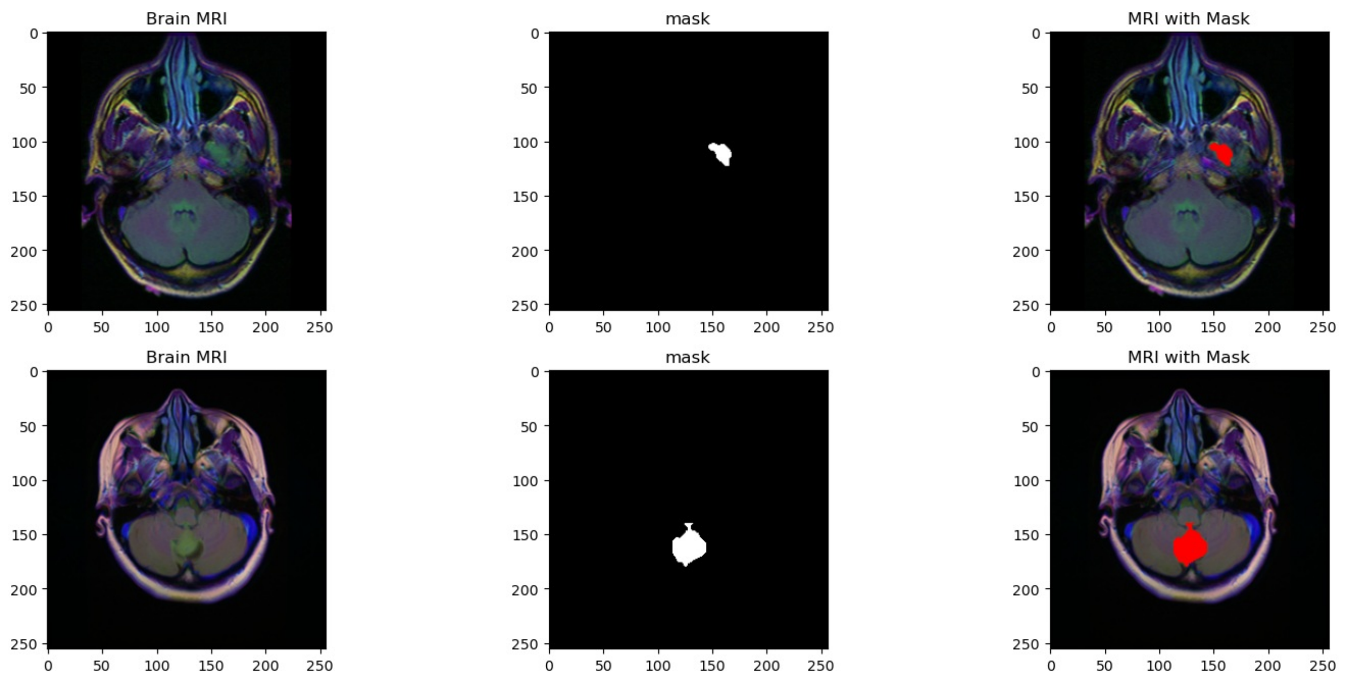
```
In [16]: # visualizing the images(MRI and mask) in the dataset

fig, axis = plt.subplots(6, 2, figsize = (16,25))
count = 0
for i in range(6):
    i = random.randint(0, len(brain_df)) # Select a random index
    axis[count][0].title.set_text('Brain MRI') # Set title
    axis[count][0].imshow(cv2.imread(brain_df.image_path[i])) # MRI
    axis[count][1].title.set_text('Mask - ' + str(brain_df['mask'][i])) # Plot title on the mask (0 or 1)
    axis[count][1].imshow(cv2.imread(brain_df.mask_path[i]))
    count += 1
```

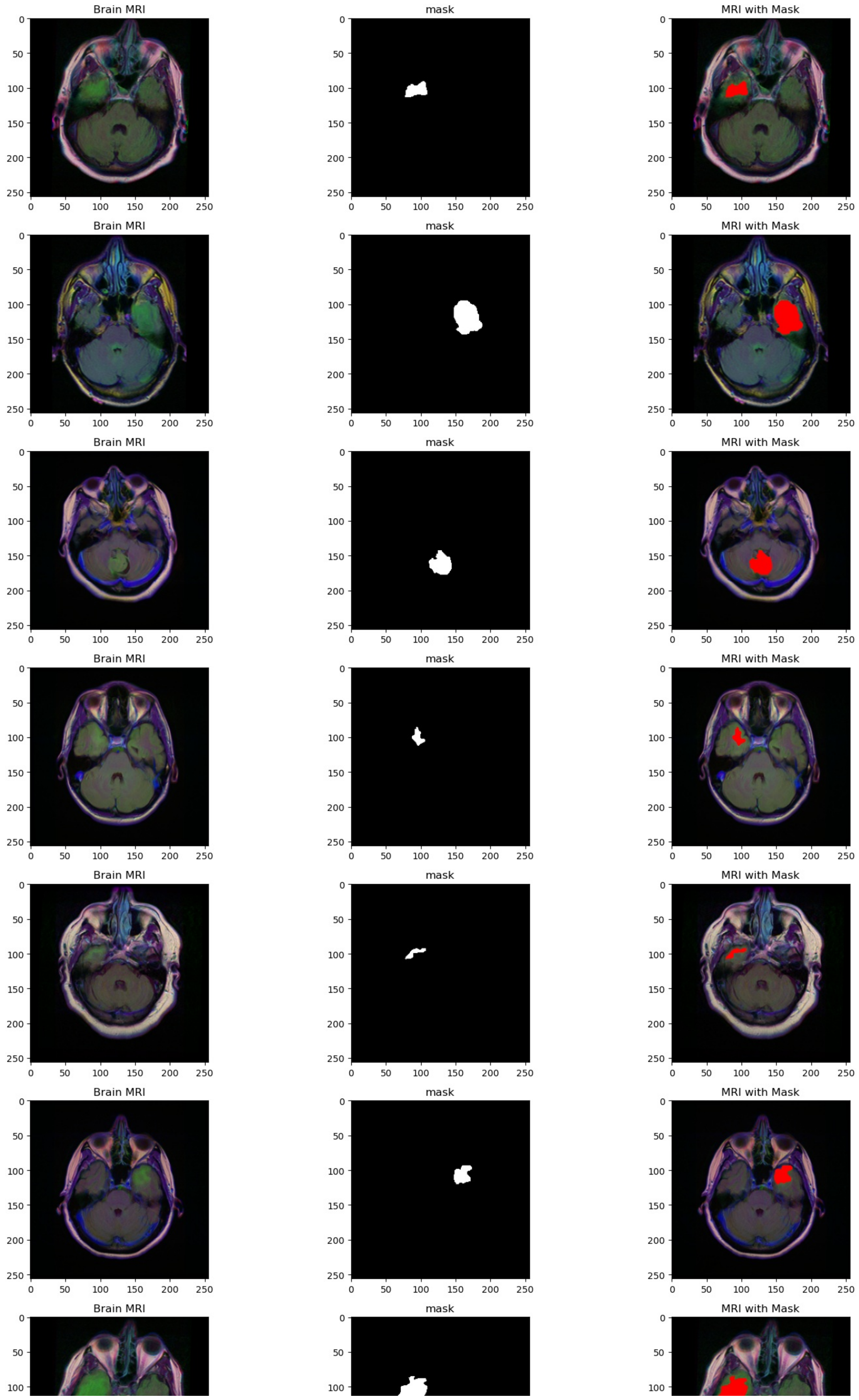
```
fig.tight_layout()
```

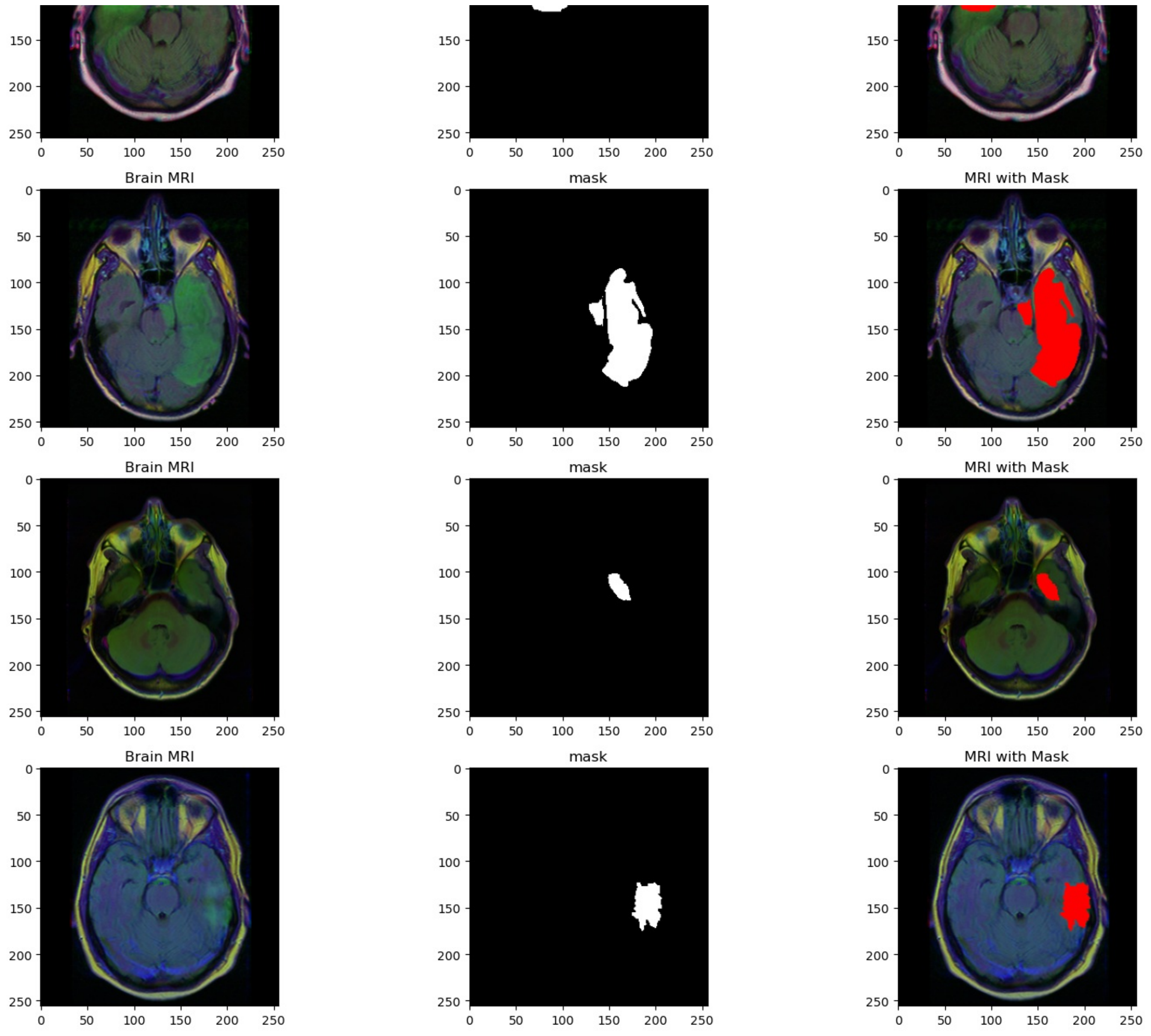




In [17]:
```
# visualizing the 12 randomly selected MRI and it correspong Mask from only unhealthy patients
# showing the MRI and the ,mask on top of each other (showing the mask in red color)

fig, axis = plt.subplots(12, 3, figsize = (16,40))
count = 0
for i in range(len(brain_df)):
    if brain_df['mask'][i] == 1 and count <12:
        img = io.imread(brain_df.image_path[i])
        axis[count][0].title.set_text('Brain MRI')
        axis[count][0].imshow(img)

        mask = io.imread(brain_df.mask_path[i])
        axis[count][1].title.set_text('mask')
        axis[count][1].imshow(mask, cmap = 'gray')

        img[mask == 255] = (255,0,0)
        axis[count][2].title.set_text('MRI with Mask')
        axis[count][2].imshow(img)
        count += 1

fig.tight_layout()
```




Brain MRI
mask
MRI with Mask
Brain MRI
mask
MRI with Mask
Brain MRI
mask
MRI with Mask
Brain MRI
mask
MRI with Mask
Brain MRI
mask
MRI with Mask
Brain MRI
mask
MRI with Mask
Brain MRI
mask
MRI with Mask
0
50
100
150
200
250



Training a Classifier Model to Detect if Tumor Exits or Not

In [18]:
```
# Drop columns not needed for the training
brain_df_train = brain_df.drop(['patient_id'], axis =1)
brain_df_train.shape
```
Out[18]:
```
(3929, 3)
```

In [19]:
```
# Convert the mask column to a categorical format.
brain_df_train['mask'] = brain_df_train['mask'].apply(lambda x : str(x))
```

In [20]:
```
brain_df_train.info()
```
```
<class 'pandas.core.frame.DataFrame'>
RangeIndex: 3929 entries, 0 to 3928
Data columns (total 3 columns):
 #   Column      Non-Null Count  Dtype 
---  ------      --------------  ----- 
 0   image_path  3929 non-null   object
 1   mask_path   3929 non-null   object
 2   mask        3929 non-null   object
dtypes: object(3)
memory usage: 92.2+ KB
```

In [21]:
```
# Split the data into train and test data
from sklearn.model_selection import train_test_split

train, test = train_test_split(brain_df_train, test_size = 0.15)
```

In [22]:
```
# Create a image generator to generate images in batch format and do some processing on the data
from keras.preprocessing.image import ImageDataGenerator

# Create a data generator which scales the data from 0 to 1 and makes validation split of 0.15
datagen = ImageDataGenerator(rescale = 1./255., validation_split = 0.15)
```

In [23]:
```
train_generator = datagen.flow_from_dataframe(
dataframe = train,
directory = './',
x_col = 'image_path',
y_col = 'mask',
subset='training',
batch_size = 16,
shuffle = True,
class_mode ='categorical',
target_size = (256, 256))

val_generator = datagen.flow_from_dataframe(
dataframe = train,
directory = './',
x_col = 'image_path',
y_col = 'mask',
subset='validation',
batch_size = 16,
shuffle = True,
class_mode ='categorical',
target_size = (256, 256))

# Create a data generator for the test images
test_datagen = ImageDataGenerator(rescale = 1./255.)

test_generator = datagen.flow_from_dataframe(
dataframe = test,
directory = './',
x_col = 'image_path',
y_col = 'mask',
batch_size = 16,
shuffle = False,
class_mode ='categorical',
target_size = (256, 256))
```

```
Found 2839 validated image filenames belonging to 2 classes.
Found 500 validated image filenames belonging to 2 classes.
Found 590 validated image filenames belonging to 2 classes.
```

In [24]:
```
# Using an already pre-trained model (Transfer learning)
# creating the ResNet50 base model

basemodel = ResNet50(weights ='imagenet', include_top = False, input_tensor= Input(shape=(256, 256, 3)))
```

```
WARNING:tensorflow:From C:\Users\olayi\anaconda3\Lib\site-packages\keras\src\backend.py:1398: The name tf.execu
ting_eagerly_outside_functions is deprecated. Please use tf.compat.v1.executing_eagerly_outside_functions inste
ad.

WARNING:tensorflow:From C:\Users\olayi\anaconda3\Lib\site-packages\keras\src\layers\normalization\batch_normali
zation.py:979: The name tf.nn.fused_batch_norm is deprecated. Please use tf.compat.v1.nn.fused_batch_norm inste
ad.
```

In [25]:
```
basemodel.summary()
```

```
Model: "resnet50"
__________________________________________________________________________________________________
 Layer (type)                Output Shape                 Param #   Connected to
==================================================================================================
 input_1 (InputLayer)        [(None, 256, 256, 3)]        0         []

 conv1_pad (ZeroPadding2D)   (None, 262, 262, 3)          0         ['input_1[0][0]']

 conv1_conv (Conv2D)         (None, 128, 128, 64)         9472      ['conv1_pad[0][0]']

 conv1_bn (BatchNormalizati  (None, 128, 128, 64)         256       ['conv1_conv[0][0]']
 on)

 conv1_relu (Activation)     (None, 128, 128, 64)         0         ['conv1_bn[0][0]']

 pool1_pad (ZeroPadding2D)   (None, 130, 130, 64)         0         ['conv1_relu[0][0]']

 pool1_pool (MaxPooling2D)   (None, 64, 64, 64)           0         ['pool1_pad[0][0]']

 conv2_block1_1_conv (Conv2  (None, 64, 64, 64)           4160      ['pool1_pool[0][0]']
 D)

 conv2_block1_1_bn (BatchNo  (None, 64, 64, 64)           256       ['conv2_block1_1_conv[0][0]']
 rmalization)

 conv2_block1_1_relu (Activ  (None, 64, 64, 64)           0         ['conv2_block1_1_bn[0][0]']
 ation)

 conv2_block1_2_conv (Conv2  (None, 64, 64, 64)           36928     ['conv2_block1_1_relu[0][0]']
 D)

 conv2_block1_2_bn (BatchNo  (None, 64, 64, 64)           256       ['conv2_block1_2_conv[0][0]']
 rmalization)

 conv2_block1_2_relu (Activ  (None, 64, 64, 64)           0         ['conv2_block1_2_bn[0][0]']
 ation)

 conv2_block1_0_conv (Conv2  (None, 64, 64, 256)          16640     ['pool1_pool[0][0]']
 D)

 conv2_block1_3_conv (Conv2  (None, 64, 64, 256)          16640     ['conv2_block1_2_relu[0][0]']
 D)

 conv2_block1_0_bn (BatchNo  (None, 64, 64, 256)          1024      ['conv2_block1_0_conv[0][0]']
 rmalization)

 conv2_block1_3_bn (BatchNo  (None, 64, 64, 256)          1024      ['conv2_block1_3_conv[0][0]']
 rmalization)

 conv2_block1_add (Add)      (None, 64, 64, 256)          0         ['conv2_block1_0_bn[0][0]',
                                                                     'conv2_block1_3_bn[0][0]']

 conv2_block1_out (Activati  (None, 64, 64, 256)          0         ['conv2_block1_add[0][0]']
 on)

 conv2_block2_1_conv (Conv2  (None, 64, 64, 64)           16448     ['conv2_block1_out[0][0]']
 D)

 conv2_block2_1_bn (BatchNo  (None, 64, 64, 64)           256       ['conv2_block2_1_conv[0][0]']
 rmalization)

 conv2_block2_1_relu (Activ  (None, 64, 64, 64)           0         ['conv2_block2_1_bn[0][0]']
 ation)

 conv2_block2_2_conv (Conv2  (None, 64, 64, 64)           36928     ['conv2_block2_1_relu[0][0]']
 D)

 conv2_block2_2_bn (BatchNo  (None, 64, 64, 64)           256       ['conv2_block2_2_conv[0][0]']
 rmalization)

 conv2_block2_2_relu (Activ  (None, 64, 64, 64)           0         ['conv2_block2_2_bn[0][0]']
 ation)

 conv2_block2_3_conv (Conv2  (None, 64, 64, 256)          16640     ['conv2_block2_2_relu[0][0]']
 D)

 conv2_block2_3_bn (BatchNo  (None, 64, 64, 256)          1024      ['conv2_block2_3_conv[0][0]']
 rmalization)

 conv2_block2_add (Add)      (None, 64, 64, 256)          0         ['conv2_block1_out[0][0]',
                                                                     'conv2_block2_3_bn[0][0]']

 conv2_block2_out (Activati  (None, 64, 64, 256)          0         ['conv2_block2_add[0][0]']
 on)

 conv2_block3_1_conv (Conv2  (None, 64, 64, 64)           16448     ['conv2_block2_out[0][0]']
 D)

 conv2_block3_1_bn (BatchNo  (None, 64, 64, 64)           256       ['conv2_block3_1_conv[0][0]']
 rmalization)

 conv2_block3_1_relu (Activ  (None, 64, 64, 64)           0         ['conv2_block3_1_bn[0][0]']
 ation)

 conv2_block3_2_conv (Conv2  (None, 64, 64, 64)           36928     ['conv2_block3_1_relu[0][0]']
 D)

 conv2_block3_2_bn (BatchNo  (None, 64, 64, 64)           256       ['conv2_block3_2_conv[0][0]']
 rmalization)

 conv2_block3_2_relu (Activ  (None, 64, 64, 64)           0         ['conv2_block3_2_bn[0][0]']
 ation)

 conv2_block3_3_conv (Conv2  (None, 64, 64, 256)          16640     ['conv2_block3_2_relu[0][0]']
 D)

 conv2_block3_3_bn (BatchNo  (None, 64, 64, 256)          1024      ['conv2_block3_3_conv[0][0]']
 rmalization)

 conv2_block3_add (Add)      (None, 64, 64, 256)          0         ['conv2_block2_out[0][0]',
                                                                     'conv2_block3_3_bn[0][0]']

 conv2_block3_out (Activati  (None, 64, 64, 256)          0         ['conv2_block3_add[0][0]']
 on)

 conv3_block1_1_conv (Conv2  (None, 32, 32, 128)          32896     ['conv2_block3_out[0][0]']
 D)

 conv3_block1_1_bn (BatchNo  (None, 32, 32, 128)          512       ['conv3_block1_1_conv[0][0]']
 rmalization)

 conv3_block1_1_relu (Activ  (None, 32, 32, 128)          0         ['conv3_block1_1_bn[0][0]']
 ation)
 conv3_block1_2_conv (Conv2  (None, 32, 32, 128)          147584    ['conv3_block1_1_relu[0][0]']
 D)

 conv3_block1_2_bn (BatchNo  (None, 32, 32, 128)          512       ['conv3_block1_2_conv[0][0]']
 rmalization)

 conv3_block1_2_relu (Activ  (None, 32, 32, 128)          0         ['conv3_block1_2_bn[0][0]']
 ation)

 conv3_block1_0_conv (Conv2  (None, 32, 32, 512)          131584    ['conv2_block3_out[0][0]']
 D)

 conv3_block1_3_conv (Conv2  (None, 32, 32, 512)          66048     ['conv3_block1_2_relu[0][0]']
 D)

 conv3_block1_0_bn (BatchNo  (None, 32, 32, 512)          2048      ['conv3_block1_0_conv[0][0]']
 rmalization)

 conv3_block1_3_bn (BatchNo  (None, 32, 32, 512)          2048      ['conv3_block1_3_conv[0][0]']
 rmalization)

 conv3_block1_add (Add)      (None, 32, 32, 512)          0         ['conv3_block1_0_bn[0][0]',
                                                                     'conv3_block1_3_bn[0][0]']

 conv3_block1_out (Activati  (None, 32, 32, 512)          0         ['conv3_block1_add[0][0]']
 on)

 conv3_block2_1_conv (Conv2  (None, 32, 32, 128)          65664     ['conv3_block1_out[0][0]']
 D)

 conv3_block2_1_bn (BatchNo  (None, 32, 32, 128)          512       ['conv3_block2_1_conv[0][0]']
 rmalization)

 conv3_block2_1_relu (Activ  (None, 32, 32, 128)          0         ['conv3_block2_1_bn[0][0]']
 ation)

 conv3_block2_2_conv (Conv2  (None, 32, 32, 128)          147584    ['conv3_block2_1_relu[0][0]']
 D)

 conv3_block2_2_bn (BatchNo  (None, 32, 32, 128)          512       ['conv3_block2_2_conv[0][0]']
 rmalization)

 conv3_block2_2_relu (Activ  (None, 32, 32, 128)          0         ['conv3_block2_2_bn[0][0]']
 ation)

 conv3_block2_3_conv (Conv2  (None, 32, 32, 512)          66048     ['conv3_block2_2_relu[0][0]']
 D)

 conv3_block2_3_bn (BatchNo  (None, 32, 32, 512)          2048      ['conv3_block2_3_conv[0][0]']
 rmalization)

 conv3_block2_add (Add)      (None, 32, 32, 512)          0         ['conv3_block1_out[0][0]',
                                                                     'conv3_block2_3_bn[0][0]']

 conv3_block2_out (Activati  (None, 32, 32, 512)          0         ['conv3_block2_add[0][0]']
 on)

 conv3_block3_1_conv (Conv2  (None, 32, 32, 128)          65664     ['conv3_block2_out[0][0]']
 D)

 conv3_block3_1_bn (BatchNo  (None, 32, 32, 128)          512       ['conv3_block3_1_conv[0][0]']
 rmalization)

 conv3_block3_1_relu (Activ  (None, 32, 32, 128)          0         ['conv3_block3_1_bn[0][0]']
 ation)

 conv3_block3_2_conv (Conv2  (None, 32, 32, 128)          147584    ['conv3_block3_1_relu[0][0]']
 D)

 conv3_block3_2_bn (BatchNo  (None, 32, 32, 128)          512       ['conv3_block3_2_conv[0][0]']
 rmalization)

 conv3_block3_2_relu (Activ  (None, 32, 32, 128)          0         ['conv3_block3_2_bn[0][0]']
 ation)

 conv3_block3_3_conv (Conv2  (None, 32, 32, 512)          66048     ['conv3_block3_2_relu[0][0]']
 D)

 conv3_block3_3_bn (BatchNo  (None, 32, 32, 512)          2048      ['conv3_block3_3_conv[0][0]']
 rmalization)

 conv3_block3_add (Add)      (None, 32, 32, 512)          0         ['conv3_block2_out[0][0]',
                                                                     'conv3_block3_3_bn[0][0]']

 conv3_block3_out (Activati  (None, 32, 32, 512)          0         ['conv3_block3_add[0][0]']
 on)

 conv3_block4_1_conv (Conv2  (None, 32, 32, 128)          65664     ['conv3_block3_out[0][0]']
 D)
 conv3_block4_1_bn (BatchNo  (None, 32, 32, 128)          512       ['conv3_block4_1_conv[0][0]']
 rmalization)

 conv3_block4_1_relu (Activ  (None, 32, 32, 128)          0         ['conv3_block4_1_bn[0][0]']
 ation)

 conv3_block4_2_conv (Conv2  (None, 32, 32, 128)          147584    ['conv3_block4_1_relu[0][0]']
 D)

 conv3_block4_2_bn (BatchNo  (None, 32, 32, 128)          512       ['conv3_block4_2_conv[0][0]']
 rmalization)

 conv3_block4_2_relu (Activ  (None, 32, 32, 128)          0         ['conv3_block4_2_bn[0][0]']
 ation)

 conv3_block4_3_conv (Conv2  (None, 32, 32, 512)          66048     ['conv3_block4_2_relu[0][0]']
 D)

 conv3_block4_3_bn (BatchNo  (None, 32, 32, 512)          2048      ['conv3_block4_3_conv[0][0]']
 rmalization)

 conv3_block4_add (Add)      (None, 32, 32, 512)          0         ['conv3_block3_out[0][0]',
                                                                     'conv3_block4_3_bn[0][0]']

 conv3_block4_out (Activati  (None, 32, 32, 512)          0         ['conv3_block4_add[0][0]']
 on)

 conv4_block1_1_conv (Conv2  (None, 16, 16, 256)          131328    ['conv3_block4_out[0][0]']
 D)

 conv4_block1_1_bn (BatchNo  (None, 16, 16, 256)          1024      ['conv4_block1_1_conv[0][0]']
 rmalization)

 conv4_block1_1_relu (Activ  (None, 16, 16, 256)          0         ['conv4_block1_1_bn[0][0]']
 ation)

 conv4_block1_2_conv (Conv2  (None, 16, 16, 256)          590080    ['conv4_block1_1_relu[0][0]']
 D)

 conv4_block1_2_bn (BatchNo  (None, 16, 16, 256)          1024      ['conv4_block1_2_conv[0][0]']
 rmalization)

 conv4_block1_2_relu (Activ  (None, 16, 16, 256)          0         ['conv4_block1_2_bn[0][0]']
 ation)

 conv4_block1_0_conv (Conv2  (None, 16, 16, 1024)         525312    ['conv3_block4_out[0][0]']
 D)

 conv4_block1_3_conv (Conv2  (None, 16, 16, 1024)         263168    ['conv4_block1_2_relu[0][0]']
 D)

 conv4_block1_0_bn (BatchNo  (None, 16, 16, 1024)         4096      ['conv4_block1_0_conv[0][0]']
 rmalization)

 conv4_block1_3_bn (BatchNo  (None, 16, 16, 1024)         4096      ['conv4_block1_3_conv[0][0]']
 rmalization)

 conv4_block1_add (Add)      (None, 16, 16, 1024)         0         ['conv4_block1_0_bn[0][0]',
                                                                     'conv4_block1_3_bn[0][0]']

 conv4_block1_out (Activati  (None, 16, 16, 1024)         0         ['conv4_block1_add[0][0]']
 on)

 conv4_block2_1_conv (Conv2  (None, 16, 16, 256)          262400    ['conv4_block1_out[0][0]']
 D)

 conv4_block2_1_bn (BatchNo  (None, 16, 16, 256)          1024      ['conv4_block2_1_conv[0][0]']
 rmalization)

 conv4_block2_1_relu (Activ  (None, 16, 16, 256)          0         ['conv4_block2_1_bn[0][0]']
 ation)

 conv4_block2_2_conv (Conv2  (None, 16, 16, 256)          590080    ['conv4_block2_1_relu[0][0]']
 D)

 conv4_block2_2_bn (BatchNo  (None, 16, 16, 256)          1024      ['conv4_block2_2_conv[0][0]']
 rmalization)

 conv4_block2_2_relu (Activ  (None, 16, 16, 256)          0         ['conv4_block2_2_bn[0][0]']
 ation)

 conv4_block2_3_conv (Conv2  (None, 16, 16, 1024)         263168    ['conv4_block2_2_relu[0][0]']
 D)

 conv4_block2_3_bn (BatchNo  (None, 16, 16, 1024)         4096      ['conv4_block2_3_conv[0][0]']
 rmalization)

 conv4_block2_add (Add)      (None, 16, 16, 1024)         0         ['conv4_block1_out[0][0]',
                                                                  'conv4_block2_3_bn[0][0]']

 conv4_block2_out (Activati  (None, 16, 16, 1024)        0         ['conv4_block2_add[0][0]']
 on)

 conv4_block3_1_conv (Conv2  (None, 16, 16, 256)         262400    ['conv4_block2_out[0][0]']
 D)

 conv4_block3_1_bn (BatchNo  (None, 16, 16, 256)         1024      ['conv4_block3_1_conv[0][0]']
 rmalization)

 conv4_block3_1_relu (Activ  (None, 16, 16, 256)         0         ['conv4_block3_1_bn[0][0]']
 ation)

 conv4_block3_2_conv (Conv2  (None, 16, 16, 256)         590080    ['conv4_block3_1_relu[0][0]']
 D)

 conv4_block3_2_bn (BatchNo  (None, 16, 16, 256)         1024      ['conv4_block3_2_conv[0][0]']
 rmalization)

 conv4_block3_2_relu (Activ  (None, 16, 16, 256)         0         ['conv4_block3_2_bn[0][0]']
 ation)

 conv4_block3_3_conv (Conv2  (None, 16, 16, 1024)        263168    ['conv4_block3_2_relu[0][0]']
 D)

 conv4_block3_3_bn (BatchNo  (None, 16, 16, 1024)        4096      ['conv4_block3_3_conv[0][0]']
 rmalization)

 conv4_block3_add (Add)      (None, 16, 16, 1024)        0         ['conv4_block2_out[0][0]',
                                                                    'conv4_block3_3_bn[0][0]']

 conv4_block3_out (Activati  (None, 16, 16, 1024)        0         ['conv4_block3_add[0][0]']
 on)

 conv4_block4_1_conv (Conv2  (None, 16, 16, 256)         262400    ['conv4_block3_out[0][0]']
 D)

 conv4_block4_1_bn (BatchNo  (None, 16, 16, 256)         1024      ['conv4_block4_1_conv[0][0]']
 rmalization)

 conv4_block4_1_relu (Activ  (None, 16, 16, 256)         0         ['conv4_block4_1_bn[0][0]']
 ation)

 conv4_block4_2_conv (Conv2  (None, 16, 16, 256)         590080    ['conv4_block4_1_relu[0][0]']
 D)

 conv4_block4_2_bn (BatchNo  (None, 16, 16, 256)         1024      ['conv4_block4_2_conv[0][0]']
 rmalization)

 conv4_block4_2_relu (Activ  (None, 16, 16, 256)         0         ['conv4_block4_2_bn[0][0]']
 ation)

 conv4_block4_3_conv (Conv2  (None, 16, 16, 1024)        263168    ['conv4_block4_2_relu[0][0]']
 D)

 conv4_block4_3_bn (BatchNo  (None, 16, 16, 1024)        4096      ['conv4_block4_3_conv[0][0]']
 rmalization)

 conv4_block4_add (Add)      (None, 16, 16, 1024)        0         ['conv4_block3_out[0][0]',
                                                                    'conv4_block4_3_bn[0][0]']

 conv4_block4_out (Activati  (None, 16, 16, 1024)        0         ['conv4_block4_add[0][0]']
 on)

 conv4_block5_1_conv (Conv2  (None, 16, 16, 256)         262400    ['conv4_block4_out[0][0]']
 D)

 conv4_block5_1_bn (BatchNo  (None, 16, 16, 256)         1024      ['conv4_block5_1_conv[0][0]']
 rmalization)

 conv4_block5_1_relu (Activ  (None, 16, 16, 256)         0         ['conv4_block5_1_bn[0][0]']
 ation)

 conv4_block5_2_conv (Conv2  (None, 16, 16, 256)         590080    ['conv4_block5_1_relu[0][0]']
 D)

 conv4_block5_2_bn (BatchNo  (None, 16, 16, 256)         1024      ['conv4_block5_2_conv[0][0]']
 rmalization)

 conv4_block5_2_relu (Activ  (None, 16, 16, 256)         0         ['conv4_block5_2_bn[0][0]']
 ation)

 conv4_block5_3_conv (Conv2  (None, 16, 16, 1024)        263168    ['conv4_block5_2_relu[0][0]']
 D)

 conv4_block5_3_bn (BatchNo  (None, 16, 16, 1024)        4096      ['conv4_block5_3_conv[0][0]']
 rmalization)
 conv4_block5_add (Add)      (None, 16, 16, 1024)         0         ['conv4_block4_out[0][0]',
                                                                     'conv4_block5_3_bn[0][0]']

 conv4_block5_out (Activati  (None, 16, 16, 1024)         0         ['conv4_block5_add[0][0]']
 on)

 conv4_block6_1_conv (Conv2  (None, 16, 16, 256)          262400    ['conv4_block5_out[0][0]']
 D)

 conv4_block6_1_bn (BatchNo  (None, 16, 16, 256)          1024      ['conv4_block6_1_conv[0][0]']
 rmalization)

 conv4_block6_1_relu (Activ  (None, 16, 16, 256)          0         ['conv4_block6_1_bn[0][0]']
 ation)

 conv4_block6_2_conv (Conv2  (None, 16, 16, 256)          590080    ['conv4_block6_1_relu[0][0]']
 D)

 conv4_block6_2_bn (BatchNo  (None, 16, 16, 256)          1024      ['conv4_block6_2_conv[0][0]']
 rmalization)

 conv4_block6_2_relu (Activ  (None, 16, 16, 256)          0         ['conv4_block6_2_bn[0][0]']
 ation)

 conv4_block6_3_conv (Conv2  (None, 16, 16, 1024)         263168    ['conv4_block6_2_relu[0][0]']
 D)

 conv4_block6_3_bn (BatchNo  (None, 16, 16, 1024)         4096      ['conv4_block6_3_conv[0][0]']
 rmalization)

 conv4_block6_add (Add)      (None, 16, 16, 1024)         0         ['conv4_block5_out[0][0]',
                                                                     'conv4_block6_3_bn[0][0]']

 conv4_block6_out (Activati  (None, 16, 16, 1024)         0         ['conv4_block6_add[0][0]']
 on)

 conv5_block1_1_conv (Conv2  (None, 8, 8, 512)            524800    ['conv4_block6_out[0][0]']
 D)

 conv5_block1_1_bn (BatchNo  (None, 8, 8, 512)            2048      ['conv5_block1_1_conv[0][0]']
 rmalization)

 conv5_block1_1_relu (Activ  (None, 8, 8, 512)            0         ['conv5_block1_1_bn[0][0]']
 ation)

 conv5_block1_2_conv (Conv2  (None, 8, 8, 512)            2359808   ['conv5_block1_1_relu[0][0]']
 D)

 conv5_block1_2_bn (BatchNo  (None, 8, 8, 512)            2048      ['conv5_block1_2_conv[0][0]']
 rmalization)

 conv5_block1_2_relu (Activ  (None, 8, 8, 512)            0         ['conv5_block1_2_bn[0][0]']
 ation)

 conv5_block1_0_conv (Conv2  (None, 8, 8, 2048)           2099200   ['conv4_block6_out[0][0]']
 D)

 conv5_block1_3_conv (Conv2  (None, 8, 8, 2048)           1050624   ['conv5_block1_2_relu[0][0]']
 D)

 conv5_block1_0_bn (BatchNo  (None, 8, 8, 2048)           8192      ['conv5_block1_0_conv[0][0]']
 rmalization)

 conv5_block1_3_bn (BatchNo  (None, 8, 8, 2048)           8192      ['conv5_block1_3_conv[0][0]']
 rmalization)

 conv5_block1_add (Add)      (None, 8, 8, 2048)           0         ['conv5_block1_0_bn[0][0]',
                                                                     'conv5_block1_3_bn[0][0]']

 conv5_block1_out (Activati  (None, 8, 8, 2048)           0         ['conv5_block1_add[0][0]']
 on)

 conv5_block2_1_conv (Conv2  (None, 8, 8, 512)            1049088   ['conv5_block1_out[0][0]']
 D)

 conv5_block2_1_bn (BatchNo  (None, 8, 8, 512)            2048      ['conv5_block2_1_conv[0][0]']
 rmalization)

 conv5_block2_1_relu (Activ  (None, 8, 8, 512)            0         ['conv5_block2_1_bn[0][0]']
 ation)

 conv5_block2_2_conv (Conv2  (None, 8, 8, 512)            2359808   ['conv5_block2_1_relu[0][0]']
 D)

 conv5_block2_2_bn (BatchNo  (None, 8, 8, 512)            2048      ['conv5_block2_2_conv[0][0]']
 rmalization)

 conv5_block2_2_relu (Activ  (None, 8, 8, 512)            0         ['conv5_block2_2_bn[0][0]']
 ation)
 conv5_block2_3_conv (Conv2  (None, 8, 8, 2048)           1050624   ['conv5_block2_2_relu[0][0]']
 D)

 conv5_block2_3_bn (BatchNo  (None, 8, 8, 2048)           8192      ['conv5_block2_3_conv[0][0]']
 rmalization)

 conv5_block2_add (Add)      (None, 8, 8, 2048)           0         ['conv5_block1_out[0][0]',
                                                                     'conv5_block2_3_bn[0][0]']

 conv5_block2_out (Activati  (None, 8, 8, 2048)           0         ['conv5_block2_add[0][0]']
 on)

 conv5_block3_1_conv (Conv2  (None, 8, 8, 512)            1049088   ['conv5_block2_out[0][0]']
 D)

 conv5_block3_1_bn (BatchNo  (None, 8, 8, 512)            2048      ['conv5_block3_1_conv[0][0]']
 rmalization)

 conv5_block3_1_relu (Activ  (None, 8, 8, 512)            0         ['conv5_block3_1_bn[0][0]']
 ation)

 conv5_block3_2_conv (Conv2  (None, 8, 8, 512)            2359808   ['conv5_block3_1_relu[0][0]']
 D)

 conv5_block3_2_bn (BatchNo  (None, 8, 8, 512)            2048      ['conv5_block3_2_conv[0][0]']
 rmalization)

 conv5_block3_2_relu (Activ  (None, 8, 8, 512)            0         ['conv5_block3_2_bn[0][0]']
 ation)

 conv5_block3_3_conv (Conv2  (None, 8, 8, 2048)           1050624   ['conv5_block3_2_relu[0][0]']
 D)

 conv5_block3_3_bn (BatchNo  (None, 8, 8, 2048)           8192      ['conv5_block3_3_conv[0][0]']
 rmalization)

 conv5_block3_add (Add)      (None, 8, 8, 2048)           0         ['conv5_block2_out[0][0]',
                                                                     'conv5_block3_3_bn[0][0]']

 conv5_block3_out (Activati  (None, 8, 8, 2048)           0         ['conv5_block3_add[0][0]']
 on)

==================================================================================================
Total params: 23587712 (89.98 MB)
Trainable params: 23534592 (89.78 MB)
Non-trainable params: 53120 (207.50 KB)
__________________________________________________________________________________________________
```

In [26]:
```python
# freeze the model
for layer in basemodel.layers:
    layers.trainable = False
```

In [27]:
```python
# add a classification head to the model

headmodel = basemodel.output
headmodel = AveragePooling2D(pool_size= (4,4))(headmodel)
headmodel = Flatten(name = 'flatten')(headmodel)
headmodel = Dense(256, activation= 'relu')(headmodel)
headmodel = Dropout(0.3)(headmodel)
headmodel = Dense(256, activation= 'relu')(headmodel)
headmodel = Dropout(0.3)(headmodel)
headmodel = Dense(2, activation= 'softmax')(headmodel)

model = Model(inputs = basemodel.input, outputs = headmodel)
```

In [28]:
```python
model.summary()
```

```
Model: "model"
__________________________________________________________________________________________________
 Layer (type)                Output Shape                 Param #   Connected to
==================================================================================================
 input_1 (InputLayer)        [(None, 256, 256, 3)]        0         []

 conv1_pad (ZeroPadding2D)   (None, 262, 262, 3)          0         ['input_1[0][0]']

 conv1_conv (Conv2D)         (None, 128, 128, 64)         9472      ['conv1_pad[0][0]']

 conv1_bn (BatchNormalizati  (None, 128, 128, 64)         256       ['conv1_conv[0][0]']
 on)

 conv1_relu (Activation)     (None, 128, 128, 64)         0         ['conv1_bn[0][0]']

 pool1_pad (ZeroPadding2D)   (None, 130, 130, 64)         0         ['conv1_relu[0][0]']

 pool1_pool (MaxPooling2D)   (None, 64, 64, 64)           0         ['pool1_pad[0][0]']

 conv2_block1_1_conv (Conv2  (None, 64, 64, 64)           4160      ['pool1_pool[0][0]']
 D)

 conv2_block1_1_bn (BatchNo  (None, 64, 64, 64)          256       ['conv2_block1_1_conv[0][0]']
 rmalization)

 conv2_block1_1_relu (Activ  (None, 64, 64, 64)          0         ['conv2_block1_1_bn[0][0]']
 ation)

 conv2_block1_2_conv (Conv2  (None, 64, 64, 64)          36928     ['conv2_block1_1_relu[0][0]']
 D)

 conv2_block1_2_bn (BatchNo  (None, 64, 64, 64)          256       ['conv2_block1_2_conv[0][0]']
 rmalization)

 conv2_block1_2_relu (Activ  (None, 64, 64, 64)          0         ['conv2_block1_2_bn[0][0]']
 ation)

 conv2_block1_0_conv (Conv2  (None, 64, 64, 256)         16640     ['pool1_pool[0][0]']
 D)

 conv2_block1_3_conv (Conv2  (None, 64, 64, 256)         16640     ['conv2_block1_2_relu[0][0]']
 D)

 conv2_block1_0_bn (BatchNo  (None, 64, 64, 256)         1024      ['conv2_block1_0_conv[0][0]']
 rmalization)

 conv2_block1_3_bn (BatchNo  (None, 64, 64, 256)         1024      ['conv2_block1_3_conv[0][0]']
 rmalization)

 conv2_block1_add (Add)      (None, 64, 64, 256)         0         ['conv2_block1_0_bn[0][0]',
                                                                    'conv2_block1_3_bn[0][0]']

 conv2_block1_out (Activati  (None, 64, 64, 256)         0         ['conv2_block1_add[0][0]']
 on)

 conv2_block2_1_conv (Conv2  (None, 64, 64, 64)          16448     ['conv2_block1_out[0][0]']
 D)

 conv2_block2_1_bn (BatchNo  (None, 64, 64, 64)          256       ['conv2_block2_1_conv[0][0]']
 rmalization)

 conv2_block2_1_relu (Activ  (None, 64, 64, 64)          0         ['conv2_block2_1_bn[0][0]']
 ation)

 conv2_block2_2_conv (Conv2  (None, 64, 64, 64)          36928     ['conv2_block2_1_relu[0][0]']
 D)

 conv2_block2_2_bn (BatchNo  (None, 64, 64, 64)          256       ['conv2_block2_2_conv[0][0]']
 rmalization)

 conv2_block2_2_relu (Activ  (None, 64, 64, 64)          0         ['conv2_block2_2_bn[0][0]']
 ation)

 conv2_block2_3_conv (Conv2  (None, 64, 64, 256)         16640     ['conv2_block2_2_relu[0][0]']
 D)

 conv2_block2_3_bn (BatchNo  (None, 64, 64, 256)         1024      ['conv2_block2_3_conv[0][0]']
 rmalization)

 conv2_block2_add (Add)      (None, 64, 64, 256)         0         ['conv2_block1_out[0][0]',
                                                                    'conv2_block2_3_bn[0][0]']

 conv2_block2_out (Activati  (None, 64, 64, 256)         0         ['conv2_block2_add[0][0]']
 on)

 conv2_block3_1_conv (Conv2  (None, 64, 64, 64)          16448     ['conv2_block2_out[0][0]']
 D)

 conv2_block3_1_bn (BatchNo  (None, 64, 64, 64)          256       ['conv2_block3_1_conv[0][0]']
 rmalization)

 conv2_block3_1_relu (Activ  (None, 64, 64, 64)          0         ['conv2_block3_1_bn[0][0]']
 ation)

 conv2_block3_2_conv (Conv2  (None, 64, 64, 64)          36928     ['conv2_block3_1_relu[0][0]']
 D)

 conv2_block3_2_bn (BatchNo  (None, 64, 64, 64)          256       ['conv2_block3_2_conv[0][0]']
 rmalization)

 conv2_block3_2_relu (Activ  (None, 64, 64, 64)          0         ['conv2_block3_2_bn[0][0]']
 ation)

 conv2_block3_3_conv (Conv2  (None, 64, 64, 256)         16640     ['conv2_block3_2_relu[0][0]']
 D)

 conv2_block3_3_bn (BatchNo  (None, 64, 64, 256)         1024      ['conv2_block3_3_conv[0][0]']
 rmalization)
 conv2_block3_add (Add)      (None, 64, 64, 256)          0         ['conv2_block2_out[0][0]',
                                                                     'conv2_block3_3_bn[0][0]']

 conv2_block3_out (Activati  (None, 64, 64, 256)          0         ['conv2_block3_add[0][0]']
 on)

 conv3_block1_1_conv (Conv2  (None, 32, 32, 128)          32896     ['conv2_block3_out[0][0]']
 D)

 conv3_block1_1_bn (BatchNo  (None, 32, 32, 128)          512       ['conv3_block1_1_conv[0][0]']
 rmalization)

 conv3_block1_1_relu (Activ  (None, 32, 32, 128)          0         ['conv3_block1_1_bn[0][0]']
 ation)

 conv3_block1_2_conv (Conv2  (None, 32, 32, 128)          147584    ['conv3_block1_1_relu[0][0]']
 D)

 conv3_block1_2_bn (BatchNo  (None, 32, 32, 128)          512       ['conv3_block1_2_conv[0][0]']
 rmalization)

 conv3_block1_2_relu (Activ  (None, 32, 32, 128)          0         ['conv3_block1_2_bn[0][0]']
 ation)

 conv3_block1_0_conv (Conv2  (None, 32, 32, 512)          131584    ['conv2_block3_out[0][0]']
 D)

 conv3_block1_3_conv (Conv2  (None, 32, 32, 512)          66048     ['conv3_block1_2_relu[0][0]']
 D)

 conv3_block1_0_bn (BatchNo  (None, 32, 32, 512)          2048      ['conv3_block1_0_conv[0][0]']
 rmalization)

 conv3_block1_3_bn (BatchNo  (None, 32, 32, 512)          2048      ['conv3_block1_3_conv[0][0]']
 rmalization)

 conv3_block1_add (Add)      (None, 32, 32, 512)          0         ['conv3_block1_0_bn[0][0]',
                                                                     'conv3_block1_3_bn[0][0]']

 conv3_block1_out (Activati  (None, 32, 32, 512)          0         ['conv3_block1_add[0][0]']
 on)

 conv3_block2_1_conv (Conv2  (None, 32, 32, 128)          65664     ['conv3_block1_out[0][0]']
 D)

 conv3_block2_1_bn (BatchNo  (None, 32, 32, 128)          512       ['conv3_block2_1_conv[0][0]']
 rmalization)

 conv3_block2_1_relu (Activ  (None, 32, 32, 128)          0         ['conv3_block2_1_bn[0][0]']
 ation)

 conv3_block2_2_conv (Conv2  (None, 32, 32, 128)          147584    ['conv3_block2_1_relu[0][0]']
 D)

 conv3_block2_2_bn (BatchNo  (None, 32, 32, 128)          512       ['conv3_block2_2_conv[0][0]']
 rmalization)

 conv3_block2_2_relu (Activ  (None, 32, 32, 128)          0         ['conv3_block2_2_bn[0][0]']
 ation)

 conv3_block2_3_conv (Conv2  (None, 32, 32, 512)          66048     ['conv3_block2_2_relu[0][0]']
 D)

 conv3_block2_3_bn (BatchNo  (None, 32, 32, 512)          2048      ['conv3_block2_3_conv[0][0]']
 rmalization)

 conv3_block2_add (Add)      (None, 32, 32, 512)          0         ['conv3_block1_out[0][0]',
                                                                     'conv3_block2_3_bn[0][0]']

 conv3_block2_out (Activati  (None, 32, 32, 512)          0         ['conv3_block2_add[0][0]']
 on)

 conv3_block3_1_conv (Conv2  (None, 32, 32, 128)          65664     ['conv3_block2_out[0][0]']
 D)

 conv3_block3_1_bn (BatchNo  (None, 32, 32, 128)          512       ['conv3_block3_1_conv[0][0]']
 rmalization)

 conv3_block3_1_relu (Activ  (None, 32, 32, 128)          0         ['conv3_block3_1_bn[0][0]']
 ation)

 conv3_block3_2_conv (Conv2  (None, 32, 32, 128)          147584    ['conv3_block3_1_relu[0][0]']
 D)

 conv3_block3_2_bn (BatchNo  (None, 32, 32, 128)          512       ['conv3_block3_2_conv[0][0]']
 rmalization)

 conv3_block3_2_relu (Activ  (None, 32, 32, 128)          0         ['conv3_block3_2_bn[0][0]']
 ation)
 conv3_block3_3_conv (Conv2  (None, 32, 32, 512)          66048     ['conv3_block3_2_relu[0][0]']
 D)

 conv3_block3_3_bn (BatchNo  (None, 32, 32, 512)          2048      ['conv3_block3_3_conv[0][0]']
 rmalization)

 conv3_block3_add (Add)      (None, 32, 32, 512)          0         ['conv3_block2_out[0][0]',
                                                                     'conv3_block3_3_bn[0][0]']

 conv3_block3_out (Activati  (None, 32, 32, 512)          0         ['conv3_block3_add[0][0]']
 on)

 conv3_block4_1_conv (Conv2  (None, 32, 32, 128)          65664     ['conv3_block3_out[0][0]']
 D)

 conv3_block4_1_bn (BatchNo  (None, 32, 32, 128)          512       ['conv3_block4_1_conv[0][0]']
 rmalization)

 conv3_block4_1_relu (Activ  (None, 32, 32, 128)          0         ['conv3_block4_1_bn[0][0]']
 ation)

 conv3_block4_2_conv (Conv2  (None, 32, 32, 128)          147584    ['conv3_block4_1_relu[0][0]']
 D)

 conv3_block4_2_bn (BatchNo  (None, 32, 32, 128)          512       ['conv3_block4_2_conv[0][0]']
 rmalization)

 conv3_block4_2_relu (Activ  (None, 32, 32, 128)          0         ['conv3_block4_2_bn[0][0]']
 ation)

 conv3_block4_3_conv (Conv2  (None, 32, 32, 512)          66048     ['conv3_block4_2_relu[0][0]']
 D)

 conv3_block4_3_bn (BatchNo  (None, 32, 32, 512)          2048      ['conv3_block4_3_conv[0][0]']
 rmalization)

 conv3_block4_add (Add)      (None, 32, 32, 512)          0         ['conv3_block3_out[0][0]',
                                                                     'conv3_block4_3_bn[0][0]']

 conv3_block4_out (Activati  (None, 32, 32, 512)          0         ['conv3_block4_add[0][0]']
 on)

 conv4_block1_1_conv (Conv2  (None, 16, 16, 256)          131328    ['conv3_block4_out[0][0]']
 D)

 conv4_block1_1_bn (BatchNo  (None, 16, 16, 256)          1024      ['conv4_block1_1_conv[0][0]']
 rmalization)

 conv4_block1_1_relu (Activ  (None, 16, 16, 256)          0         ['conv4_block1_1_bn[0][0]']
 ation)

 conv4_block1_2_conv (Conv2  (None, 16, 16, 256)          590080    ['conv4_block1_1_relu[0][0]']
 D)

 conv4_block1_2_bn (BatchNo  (None, 16, 16, 256)          1024      ['conv4_block1_2_conv[0][0]']
 rmalization)

 conv4_block1_2_relu (Activ  (None, 16, 16, 256)          0         ['conv4_block1_2_bn[0][0]']
 ation)

 conv4_block1_0_conv (Conv2  (None, 16, 16, 1024)         525312    ['conv3_block4_out[0][0]']
 D)

 conv4_block1_3_conv (Conv2  (None, 16, 16, 1024)         263168    ['conv4_block1_2_relu[0][0]']
 D)

 conv4_block1_0_bn (BatchNo  (None, 16, 16, 1024)         4096      ['conv4_block1_0_conv[0][0]']
 rmalization)

 conv4_block1_3_bn (BatchNo  (None, 16, 16, 1024)         4096      ['conv4_block1_3_conv[0][0]']
 rmalization)

 conv4_block1_add (Add)      (None, 16, 16, 1024)         0         ['conv4_block1_0_bn[0][0]',
                                                                     'conv4_block1_3_bn[0][0]']

 conv4_block1_out (Activati  (None, 16, 16, 1024)         0         ['conv4_block1_add[0][0]']
 on)

 conv4_block2_1_conv (Conv2  (None, 16, 16, 256)          262400    ['conv4_block1_out[0][0]']
 D)

 conv4_block2_1_bn (BatchNo  (None, 16, 16, 256)          1024      ['conv4_block2_1_conv[0][0]']
 rmalization)

 conv4_block2_1_relu (Activ  (None, 16, 16, 256)          0         ['conv4_block2_1_bn[0][0]']
 ation)

 conv4_block2_2_conv (Conv2  (None, 16, 16, 256)          590080    ['conv4_block2_1_relu[0][0]']
 D)

 conv4_block2_2_bn (BatchNo  (None, 16, 16, 256)          1024      ['conv4_block2_2_conv[0][0]']
 rmalization)

 conv4_block2_2_relu (Activ  (None, 16, 16, 256)          0         ['conv4_block2_2_bn[0][0]']
 ation)

 conv4_block2_3_conv (Conv2  (None, 16, 16, 1024)         263168    ['conv4_block2_2_relu[0][0]']
 D)

 conv4_block2_3_bn (BatchNo  (None, 16, 16, 1024)         4096      ['conv4_block2_3_conv[0][0]']
 rmalization)

 conv4_block2_add (Add)      (None, 16, 16, 1024)         0         ['conv4_block1_out[0][0]',
                                                                      'conv4_block2_3_bn[0][0]']

 conv4_block2_out (Activati  (None, 16, 16, 1024)         0         ['conv4_block2_add[0][0]']
 on)

 conv4_block3_1_conv (Conv2  (None, 16, 16, 256)          262400    ['conv4_block2_out[0][0]']
 D)

 conv4_block3_1_bn (BatchNo  (None, 16, 16, 256)          1024      ['conv4_block3_1_conv[0][0]']
 rmalization)

 conv4_block3_1_relu (Activ  (None, 16, 16, 256)          0         ['conv4_block3_1_bn[0][0]']
 ation)

 conv4_block3_2_conv (Conv2  (None, 16, 16, 256)          590080    ['conv4_block3_1_relu[0][0]']
 D)

 conv4_block3_2_bn (BatchNo  (None, 16, 16, 256)          1024      ['conv4_block3_2_conv[0][0]']
 rmalization)

 conv4_block3_2_relu (Activ  (None, 16, 16, 256)          0         ['conv4_block3_2_bn[0][0]']
 ation)

 conv4_block3_3_conv (Conv2  (None, 16, 16, 1024)         263168    ['conv4_block3_2_relu[0][0]']
 D)

 conv4_block3_3_bn (BatchNo  (None, 16, 16, 1024)         4096      ['conv4_block3_3_conv[0][0]']
 rmalization)

 conv4_block3_add (Add)      (None, 16, 16, 1024)         0         ['conv4_block2_out[0][0]',
                                                                      'conv4_block3_3_bn[0][0]']

 conv4_block3_out (Activati  (None, 16, 16, 1024)         0         ['conv4_block3_add[0][0]']
 on)

 conv4_block4_1_conv (Conv2  (None, 16, 16, 256)          262400    ['conv4_block3_out[0][0]']
 D)

 conv4_block4_1_bn (BatchNo  (None, 16, 16, 256)          1024      ['conv4_block4_1_conv[0][0]']
 rmalization)

 conv4_block4_1_relu (Activ  (None, 16, 16, 256)          0         ['conv4_block4_1_bn[0][0]']
 ation)

 conv4_block4_2_conv (Conv2  (None, 16, 16, 256)          590080    ['conv4_block4_1_relu[0][0]']
 D)

 conv4_block4_2_bn (BatchNo  (None, 16, 16, 256)          1024      ['conv4_block4_2_conv[0][0]']
 rmalization)

 conv4_block4_2_relu (Activ  (None, 16, 16, 256)          0         ['conv4_block4_2_bn[0][0]']
 ation)

 conv4_block4_3_conv (Conv2  (None, 16, 16, 1024)         263168    ['conv4_block4_2_relu[0][0]']
 D)

 conv4_block4_3_bn (BatchNo  (None, 16, 16, 1024)         4096      ['conv4_block4_3_conv[0][0]']
 rmalization)

 conv4_block4_add (Add)      (None, 16, 16, 1024)         0         ['conv4_block3_out[0][0]',
                                                                      'conv4_block4_3_bn[0][0]']

 conv4_block4_out (Activati  (None, 16, 16, 1024)         0         ['conv4_block4_add[0][0]']
 on)

 conv4_block5_1_conv (Conv2  (None, 16, 16, 256)          262400    ['conv4_block4_out[0][0]']
 D)

 conv4_block5_1_bn (BatchNo  (None, 16, 16, 256)          1024      ['conv4_block5_1_conv[0][0]']
 rmalization)

 conv4_block5_1_relu (Activ  (None, 16, 16, 256)          0         ['conv4_block5_1_bn[0][0]']
 ation)
```

```
 conv4_block5_2_conv (Conv2  (None, 16, 16, 256)          590080    ['conv4_block5_1_relu[0][0]']
 D)

 conv4_block5_2_bn (BatchNo  (None, 16, 16, 256)          1024      ['conv4_block5_2_conv[0][0]']
 rmalization)

 conv4_block5_2_relu (Activ  (None, 16, 16, 256)          0         ['conv4_block5_2_bn[0][0]']
 ation)

 conv4_block5_3_conv (Conv2  (None, 16, 16, 1024)         263168    ['conv4_block5_2_relu[0][0]']
 D)

 conv4_block5_3_bn (BatchNo  (None, 16, 16, 1024)         4096      ['conv4_block5_3_conv[0][0]']
 rmalization)

 conv4_block5_add (Add)      (None, 16, 16, 1024)         0         ['conv4_block4_out[0][0]',
                                                                     'conv4_block5_3_bn[0][0]']

 conv4_block5_out (Activati  (None, 16, 16, 1024)         0         ['conv4_block5_add[0][0]']
 on)

 conv4_block6_1_conv (Conv2  (None, 16, 16, 256)          262400    ['conv4_block5_out[0][0]']
 D)

 conv4_block6_1_bn (BatchNo  (None, 16, 16, 256)          1024      ['conv4_block6_1_conv[0][0]']
 rmalization)

 conv4_block6_1_relu (Activ  (None, 16, 16, 256)          0         ['conv4_block6_1_bn[0][0]']
 ation)

 conv4_block6_2_conv (Conv2  (None, 16, 16, 256)          590080    ['conv4_block6_1_relu[0][0]']
 D)

 conv4_block6_2_bn (BatchNo  (None, 16, 16, 256)          1024      ['conv4_block6_2_conv[0][0]']
 rmalization)

 conv4_block6_2_relu (Activ  (None, 16, 16, 256)          0         ['conv4_block6_2_bn[0][0]']
 ation)

 conv4_block6_3_conv (Conv2  (None, 16, 16, 1024)         263168    ['conv4_block6_2_relu[0][0]']
 D)

 conv4_block6_3_bn (BatchNo  (None, 16, 16, 1024)         4096      ['conv4_block6_3_conv[0][0]']
 rmalization)

 conv4_block6_add (Add)      (None, 16, 16, 1024)         0         ['conv4_block5_out[0][0]',
                                                                     'conv4_block6_3_bn[0][0]']

 conv4_block6_out (Activati  (None, 16, 16, 1024)         0         ['conv4_block6_add[0][0]']
 on)

 conv5_block1_1_conv (Conv2  (None, 8, 8, 512)            524800    ['conv4_block6_out[0][0]']
 D)

 conv5_block1_1_bn (BatchNo  (None, 8, 8, 512)            2048      ['conv5_block1_1_conv[0][0]']
 rmalization)

 conv5_block1_1_relu (Activ  (None, 8, 8, 512)            0         ['conv5_block1_1_bn[0][0]']
 ation)

 conv5_block1_2_conv (Conv2  (None, 8, 8, 512)            2359808   ['conv5_block1_1_relu[0][0]']
 D)

 conv5_block1_2_bn (BatchNo  (None, 8, 8, 512)            2048      ['conv5_block1_2_conv[0][0]']
 rmalization)

 conv5_block1_2_relu (Activ  (None, 8, 8, 512)            0         ['conv5_block1_2_bn[0][0]']
 ation)

 conv5_block1_0_conv (Conv2  (None, 8, 8, 2048)           2099200   ['conv4_block6_out[0][0]']
 D)

 conv5_block1_3_conv (Conv2  (None, 8, 8, 2048)           1050624   ['conv5_block1_2_relu[0][0]']
 D)

 conv5_block1_0_bn (BatchNo  (None, 8, 8, 2048)           8192      ['conv5_block1_0_conv[0][0]']
 rmalization)

 conv5_block1_3_bn (BatchNo  (None, 8, 8, 2048)           8192      ['conv5_block1_3_conv[0][0]']
 rmalization)

 conv5_block1_add (Add)      (None, 8, 8, 2048)           0         ['conv5_block1_0_bn[0][0]',
                                                                     'conv5_block1_3_bn[0][0]']

 conv5_block1_out (Activati  (None, 8, 8, 2048)           0         ['conv5_block1_add[0][0]']
 on)

 conv5_block2_1_conv (Conv2  (None, 8, 8, 512)            1049088   ['conv5_block1_out[0][0]']
 D)
 conv5_block2_1_bn (BatchNo  (None, 8, 8, 512)            2048      ['conv5_block2_1_conv[0][0]']
 rmalization)

 conv5_block2_1_relu (Activ  (None, 8, 8, 512)            0         ['conv5_block2_1_bn[0][0]']
 ation)

 conv5_block2_2_conv (Conv2  (None, 8, 8, 512)            2359808   ['conv5_block2_1_relu[0][0]']
 D)

 conv5_block2_2_bn (BatchNo  (None, 8, 8, 512)            2048      ['conv5_block2_2_conv[0][0]']
 rmalization)

 conv5_block2_2_relu (Activ  (None, 8, 8, 512)            0         ['conv5_block2_2_bn[0][0]']
 ation)

 conv5_block2_3_conv (Conv2  (None, 8, 8, 2048)           1050624   ['conv5_block2_2_relu[0][0]']
 D)

 conv5_block2_3_bn (BatchNo  (None, 8, 8, 2048)           8192      ['conv5_block2_3_conv[0][0]']
 rmalization)

 conv5_block2_add (Add)      (None, 8, 8, 2048)           0         ['conv5_block1_out[0][0]',
                                                                     'conv5_block2_3_bn[0][0]']

 conv5_block2_out (Activati  (None, 8, 8, 2048)           0         ['conv5_block2_add[0][0]']
 on)

 conv5_block3_1_conv (Conv2  (None, 8, 8, 512)            1049088   ['conv5_block2_out[0][0]']
 D)

 conv5_block3_1_bn (BatchNo  (None, 8, 8, 512)            2048      ['conv5_block3_1_conv[0][0]']
 rmalization)

 conv5_block3_1_relu (Activ  (None, 8, 8, 512)            0         ['conv5_block3_1_bn[0][0]']
 ation)

 conv5_block3_2_conv (Conv2  (None, 8, 8, 512)            2359808   ['conv5_block3_1_relu[0][0]']
 D)

 conv5_block3_2_bn (BatchNo  (None, 8, 8, 512)            2048      ['conv5_block3_2_conv[0][0]']
 rmalization)

 conv5_block3_2_relu (Activ  (None, 8, 8, 512)            0         ['conv5_block3_2_bn[0][0]']
 ation)

 conv5_block3_3_conv (Conv2  (None, 8, 8, 2048)           1050624   ['conv5_block3_2_relu[0][0]']
 D)

 conv5_block3_3_bn (BatchNo  (None, 8, 8, 2048)           8192      ['conv5_block3_3_conv[0][0]']
 rmalization)

 conv5_block3_add (Add)      (None, 8, 8, 2048)           0         ['conv5_block2_out[0][0]',
                                                                     'conv5_block3_3_bn[0][0]']

 conv5_block3_out (Activati  (None, 8, 8, 2048)           0         ['conv5_block3_add[0][0]']
 on)

 average_pooling2d (Average  (None, 2, 2, 2048)           0         ['conv5_block3_out[0][0]']
 Pooling2D)

 flatten (Flatten)           (None, 8192)                 0         ['average_pooling2d[0][0]']

 dense (Dense)               (None, 256)                  2097408   ['flatten[0][0]']

 dropout (Dropout)           (None, 256)                  0         ['dense[0][0]']

 dense_1 (Dense)             (None, 256)                  65792     ['dropout[0][0]']

 dropout_1 (Dropout)         (None, 256)                  0         ['dense_1[0][0]']

 dense_2 (Dense)             (None, 2)                    514       ['dropout_1[0][0]']

==================================================================================================
Total params: 25751426 (98.23 MB)
Trainable params: 25698306 (98.03 MB)
Non-trainable params: 53120 (207.50 KB)
__________________________________________________________________________________________________
```

In [29]:
```python
# compile the model
model.compile(loss= 'categorical_crossentropy', optimizer = 'adam', metrics = ['accuracy'])
```

```
WARNING:tensorflow:From C:\Users\olayi\anaconda3\Lib\site-packages\keras\src\optimizers\__init__.py:309: The na
me tf.train.Optimizer is deprecated. Please use tf.compat.v1.train.Optimizer instead.
```

In [30]:
```python
# use early stopping to exit training if validation loss is not decreasing even after certain epochs (patience)
earlystopping = EarlyStopping(monitor = 'val_loss', mode = 'min', verbose = 1, patience = 20)
```

```
# save the best model with the least validation loss
checkpointer = ModelCheckpoint(filepath='classifier-resnet-weights.hdf5', verbose = 1, save_best_only = True)
```

In [31]:
```
history = model.fit(train_generator, steps_per_epoch=train_generator.n // 16, epochs = 10, validation_data= val
```

```
Epoch 1/10
WARNING:tensorflow:From C:\Users\olayi\anaconda3\Lib\site-packages\keras\src\utils\tf_utils.py:492: The name tf
.ragged.RaggedTensorValue is deprecated. Please use tf.compat.v1.ragged.RaggedTensorValue instead.

WARNING:tensorflow:From C:\Users\olayi\anaconda3\Lib\site-packages\keras\src\engine\base_layer_utils.py:384: Th
e name tf.executing_eagerly_outside_functions is deprecated. Please use tf.compat.v1.executing_eagerly_outside_
functions instead.

177/177 [==============================] - ETA: 0s - loss: 0.9508 - accuracy: 0.6532
Epoch 1: val_loss improved from inf to 0.65904, saving model to classifier-resnet-weights.hdf5
177/177 [==============================] - 755s 4s/step - loss: 0.9508 - accuracy: 0.6532 - val_loss: 0.6590 -
val_accuracy: 0.6331
Epoch 2/10
177/177 [==============================] - ETA: 0s - loss: 0.5868 - accuracy: 0.7535
Epoch 2: val_loss did not improve from 0.65904
177/177 [==============================] - 693s 4s/step - loss: 0.5868 - accuracy: 0.7535 - val_loss: 0.6681 -
val_accuracy: 0.6331
Epoch 3/10
177/177 [==============================] - ETA: 0s - loss: 0.4516 - accuracy: 0.7977
Epoch 3: val_loss did not improve from 0.65904
177/177 [==============================] - 663s 4s/step - loss: 0.4516 - accuracy: 0.7977 - val_loss: 0.6843 -
val_accuracy: 0.6351
Epoch 4/10
177/177 [==============================] - ETA: 0s - loss: 0.3710 - accuracy: 0.8353
Epoch 4: val_loss did not improve from 0.65904
177/177 [==============================] - 705s 4s/step - loss: 0.3710 - accuracy: 0.8353 - val_loss: 0.7307 -
val_accuracy: 0.6310
Epoch 5/10
177/177 [==============================] - ETA: 0s - loss: 0.2947 - accuracy: 0.8700
Epoch 5: val_loss did not improve from 0.65904
177/177 [==============================] - 5530s 31s/step - loss: 0.2947 - accuracy: 0.8700 - val_loss: 0.6770
- val_accuracy: 0.6694
Epoch 6/10
177/177 [==============================] - ETA: 0s - loss: 0.2807 - accuracy: 0.8874
Epoch 6: val_loss improved from 0.65904 to 0.51411, saving model to classifier-resnet-weights.hdf5
177/177 [==============================] - 1010s 6s/step - loss: 0.2807 - accuracy: 0.8874 - val_loss: 0.5141 -
val_accuracy: 0.7722
Epoch 7/10
177/177 [==============================] - ETA: 0s - loss: 0.3441 - accuracy: 0.8753
Epoch 7: val_loss did not improve from 0.51411
177/177 [==============================] - 141604s 805s/step - loss: 0.3441 - accuracy: 0.8753 - val_loss: 0.77
13 - val_accuracy: 0.6976
Epoch 8/10
177/177 [==============================] - ETA: 0s - loss: 0.2707 - accuracy: 0.8916
Epoch 8: val_loss improved from 0.51411 to 0.31863, saving model to classifier-resnet-weights.hdf5
177/177 [==============================] - 701s 4s/step - loss: 0.2707 - accuracy: 0.8916 - val_loss: 0.3186 -
val_accuracy: 0.8609
Epoch 9/10
177/177 [==============================] - ETA: 0s - loss: 0.2450 - accuracy: 0.9065
Epoch 9: val_loss did not improve from 0.31863
177/177 [==============================] - 11596s 66s/step - loss: 0.2450 - accuracy: 0.9065 - val_loss: 0.3296
- val_accuracy: 0.8589
Epoch 10/10
177/177 [==============================] - ETA: 0s - loss: 0.2187 - accuracy: 0.9175
Epoch 10: val_loss did not improve from 0.31863
177/177 [==============================] - 8441s 48s/step - loss: 0.2187 - accuracy: 0.9175 - val_loss: 0.3216
- val_accuracy: 0.8891
```

In [32]:
```
# save model architecture to json file for future use
model_json = model.to_json()
with open('classifier_resnet_model.json', 'w') as json_file:
    json_file.write(model_json)
```

Assess Trained Model Performance

In [33]:
```
# load json_file

with open('classifier_resnet_model.json', 'r') as json_file:
    json_savedModel = json_file.read()

model = tf.keras.models.model_from_json(json_savedModel)
model.load_weights('classifier-resnet-weights.hdf5')
model.compile(loss = 'categorical_crossentropy', optimizer= 'adam', metrics = ['accuracy'])
```

In [34]:
```
# make prediction
test_pred = model.predict(test_generator, steps = test_generator.n // 16, verbose = 1)
```

```
36/36 [==============================] - 38s 956ms/step
```

In [35]:
```
test_pred.shape
```

```
Out[35]: (576, 2)
```

```
In [36]: test_pred
```

```
Out[36]: array([[9.9453026e-01, 5.4697674e-03],
                [9.7976160e-01, 2.0238433e-02],
                [4.4421285e-01, 5.5578715e-01],
                ...,
                [9.9669302e-01, 3.3069481e-03],
                [9.2080944e-12, 1.0000000e+00],
                [9.5426548e-01, 4.5734555e-02]], dtype=float32)
```

```
In [37]: # obtain the predicted class from the model prediction
         predict = []
         for i in test_pred:
             predict.append(str(np.argmax(i)))

         predict = np.asarray(predict)
```

```
In [38]: predict
```

```
Out[38]: array(['0', '0', '1', '1', '1', '0', '0', '1', '1', '0', '0', '0', '0',
                '0', '0', '1', '1', '1', '0', '1', '1', '1', '0', '0', '0', '1',
                '0', '1', '0', '1', '1', '0', '0', '0', '1', '0', '0', '0', '0',
                '0', '0', '1', '1', '1', '0', '1', '1', '0', '1', '0', '1', '0',
                '1', '0', '0', '0', '1', '1', '0', '1', '0', '0', '0', '0', '1',
                '0', '1', '0', '0', '1', '1', '0', '0', '1', '0', '1', '0', '1',
                '1', '0', '0', '0', '0', '0', '0', '0', '0', '0', '1', '0', '1',
                '0', '0', '0', '0', '0', '0', '1', '0', '0', '1', '1', '0', '0',
                '0', '1', '0', '1', '0', '0', '0', '0', '1', '0', '1', '1', '1',
                '1', '1', '1', '0', '0', '0', '0', '1', '1', '0', '1', '1', '1',
                '0', '0', '1', '1', '1', '0', '0', '0', '1', '1', '1', '1', '0',
                '0', '0', '0', '1', '0', '1', '0', '0', '0', '1', '1', '1', '0',
                '1', '0', '0', '0', '1', '1', '0', '0', '1', '1', '0', '0', '0',
                '0', '0', '1', '1', '1', '0', '0', '1', '0', '1', '0', '0', '1',
                '0', '1', '1', '0', '1', '1', '1', '1', '0', '1', '1', '1', '0',
                '0', '0', '1', '1', '0', '0', '0', '1', '1', '1', '1', '0', '1',
                '1', '1', '0', '0', '0', '1', '0', '0', '0', '1', '0', '0', '1',
                '1', '0', '0', '1', '0', '1', '1', '1', '0', '1', '1', '0', '0',
                '0', '0', '0', '0', '0', '0', '1', '0', '0', '0', '0', '0', '1',
                '1', '1', '1', '0', '0', '0', '0', '0', '0', '0', '1', '1', '1',
                '0', '1', '0', '0', '0', '1', '0', '0', '1', '1', '1', '1', '0',
                '0', '1', '0', '0', '0', '0', '0', '0', '0', '1', '0', '1', '0',
                '0', '1', '1', '1', '1', '1', '0', '0', '0', '0', '0', '1', '1',
                '0', '0', '1', '0', '0', '1', '0', '1', '1', '0', '0', '0', '0',
                '1', '0', '0', '1', '0', '0', '1', '0', '0', '0', '0', '1', '1',
                '0', '1', '1', '0', '0', '0', '1', '0', '1', '0', '0', '0', '1',
                '1', '0', '0', '0', '1', '0', '1', '0', '1', '0', '0', '1', '1',
                '0', '1', '1', '0', '1', '1', '0', '1', '0', '0', '1', '0', '0',
                '0', '1', '1', '0', '1', '1', '0', '1', '0', '0', '1', '0', '1',
                '0', '1', '0', '1', '1', '0', '0', '0', '1', '0', '1', '0', '0',
                '0', '1', '0', '1', '0', '0', '1', '0', '1', '0', '0', '1', '0',
                '0', '1', '0', '0', '0', '1', '0', '0', '1', '0', '0', '0', '1',
                '1', '1', '0', '0', '0', '0', '1', '0', '0', '0', '1', '0', '0',
                '0', '1', '0', '1', '0', '1', '1', '0', '0', '0', '1', '0', '0',
                '0', '1', '0', '1', '1', '1', '0', '0', '0', '1', '1', '1', '1',
                '0', '1', '1', '0', '1', '0', '0', '0', '0', '0', '1', '0', '0',
                '0', '0', '0', '1', '0', '0', '0', '0', '0', '0', '0', '1', '0',
                '0', '0', '0', '0', '0', '1', '1', '1', '0', '0', '0', '0', '1',
                '1', '1', '0', '0', '0', '1', '1', '0', '1', '0', '0', '0', '0',
                '0', '1', '0', '0', '1', '0', '0', '0', '1', '0', '1', '1', '0',
                '1', '0', '0', '1', '1', '1', '0', '1', '1', '1', '0', '0', '1',
                '0', '0', '0', '0', '0', '0', '0', '0', '1', '0', '0', '0', '0',
                '0', '1', '1', '0', '0', '1', '1', '0', '0', '0', '0', '0', '0',
                '0', '0', '1', '0', '0', '1', '0', '0', '0', '1', '1', '1', '0',
                '1', '0', '1', '0'], dtype='<U1')
```

```
In [39]: # compare the predictions to the original test data
         # since we have used test generator, it limited the images to len(predict), due to batch size
         original_test = np.asarray(test['mask'])[:len(predict)]
         len(original_test)
```

```
Out[39]: 576
```

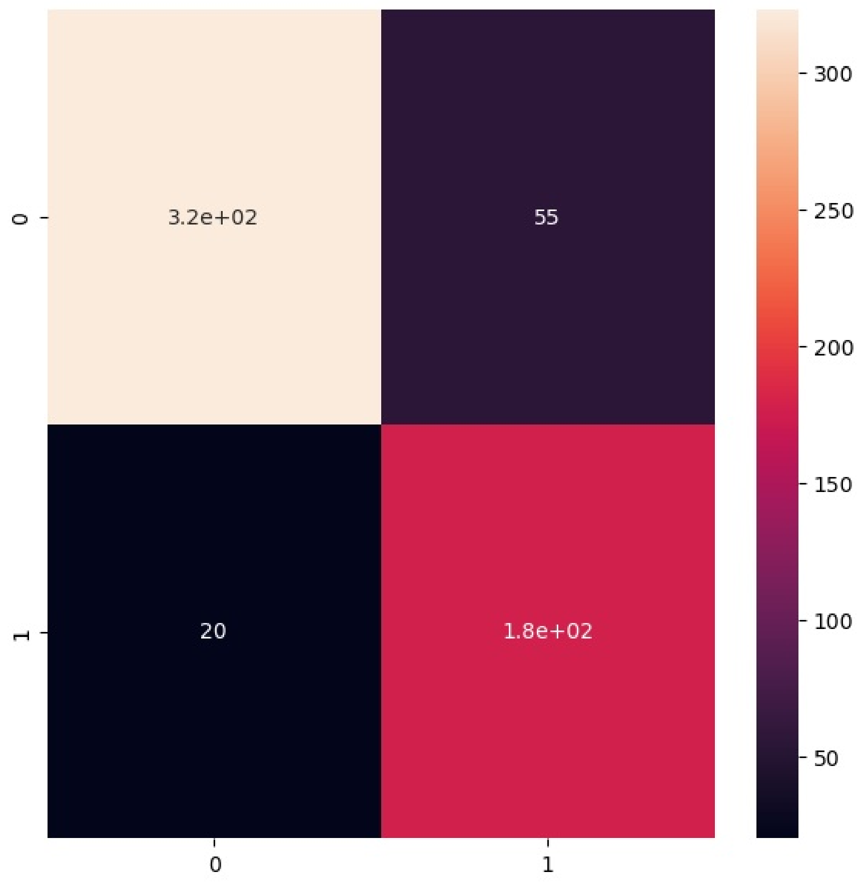
```
In [40]: # obtain the accuracy of the model
         from sklearn.metrics import accuracy_score

         accuracy = accuracy_score(original_test, predict)
         accuracy
```

```
Out[40]: 0.8697916666666666
```

```
In [41]: # plot the confusion matrix
         from sklearn.metrics import confusion_matrix

         cm = confusion_matrix(original_test, predict)
         plt.figure(figsize = (7,7))
```

```
sns.heatmap(cm, annot = True)
```

Out[41]:
```
<Axes: >
```



In [42]:
```
# Obatin the classification report
from sklearn.metrics import classification_report

report = classification_report(original_test, predict, labels = [0,1])
print(report)
```

```
              precision    recall  f1-score   support

           0       0.94      0.85      0.90       378
           1       0.76      0.90      0.83       198

   micro avg       0.87      0.87      0.87       576
   macro avg       0.85      0.88      0.86       576
weighted avg       0.88      0.87      0.87       576
```

Build A segmentation Model to Locate the Tumor

In [43]:
```
# Get the dataframe containing MRIs which have masks associated with them
brain_df_mask = brain_df[brain_df['mask']==1]
brain_df_mask.shape
```

Out[43]:
```
(1373, 4)
```

In [44]:
```
# split the data into train the test set
from sklearn.model_selection import train_test_split
X_train, X_val = train_test_split(brain_df_mask, test_size = 0.15)
X_test, X_val = train_test_split(X_val, test_size = 0.5)

print('X_train ', X_train)
print('X_val ', X_val)
print('X_test ', X_test)
```

```
X_train                   patient_id  \
2744  TCGA_HT_7855_19951020
3358  TCGA_HT_A5RC_19990831
1773  TCGA_DU_A5TT_19980318
3016  TCGA_HT_7879_19981009
2691  TCGA_HT_7856_19950831
...                     ...
1684  TCGA_DU_8165_19970205
2746  TCGA_HT_7855_19951020
2986  TCGA_HT_7882_19970125
3480  TCGA_DU_8164_19970111
3532  TCGA_DU_8168_19970503
```

```
                                             image_path  \
2744  TCGA_FG_6690_20020226/TCGA_FG_6690_20020226_27...
3358  TCGA_DU_6401_19831001/TCGA_DU_6401_19831001_37...
1773  TCGA_CS_6665_20010817/TCGA_CS_6665_20010817_17...
3016  TCGA_FG_6690_20020226/TCGA_FG_6690_20020226_31...
2691  TCGA_HT_A61A_20000127/TCGA_HT_A61A_20000127_26...
...                                                 ...
1684  TCGA_DU_7008_19830723/TCGA_DU_7008_19830723_16...
2746  TCGA_FG_7643_20021104/TCGA_FG_7643_20021104_27...
2986  TCGA_DU_6399_19830416/TCGA_DU_6399_19830416_31...
3480  TCGA_DU_6405_19851005/TCGA_DU_6405_19851005_42...
3532  TCGA_DU_7014_19860618/TCGA_DU_7014_19860618_44...

                                              mask_path  mask
2744  TCGA_FG_6690_20020226/TCGA_FG_6690_20020226_27...     1
3358  TCGA_DU_6401_19831001/TCGA_DU_6401_19831001_37...     1
1773  TCGA_CS_6665_20010817/TCGA_CS_6665_20010817_17...     1
3016  TCGA_FG_6690_20020226/TCGA_FG_6690_20020226_31...     1
2691  TCGA_HT_A61A_20000127/TCGA_HT_A61A_20000127_26...     1
...                                                 ...   ...
1684  TCGA_DU_7008_19830723/TCGA_DU_7008_19830723_16...     1
2746  TCGA_FG_7643_20021104/TCGA_FG_7643_20021104_27...     1
2986  TCGA_DU_6399_19830416/TCGA_DU_6399_19830416_31...     1
3480  TCGA_DU_6405_19851005/TCGA_DU_6405_19851005_42...     1
3532  TCGA_DU_7014_19860618/TCGA_DU_7014_19860618_44...     1

[1167 rows x 4 columns]
X_val                   patient_id  \
2667  TCGA_HT_7616_19940813
1601  TCGA_DU_A5TP_19970614
2150  TCGA_FG_6691_20020405
2009  TCGA_FG_6689_20020326
1519  TCGA_DU_7306_19930512
...                        ...
3042  TCGA_HT_7877_19980917
3205  TCGA_HT_8563_19981209
2239  TCGA_FG_7634_20000128
3390  TCGA_HT_A61B_19991127
1835  TCGA_FG_5962_20000626

                                             image_path  \
2667  TCGA_DU_A5TY_19970709/TCGA_DU_A5TY_19970709_26...
1601  TCGA_FG_7637_20000922/TCGA_FG_7637_20000922_15...
2150  TCGA_FG_7634_20000128/TCGA_FG_7634_20000128_20...
2009  TCGA_DU_7018_19911220/TCGA_DU_7018_19911220_19...
1519  TCGA_HT_8113_19930809/TCGA_HT_8113_19930809_14...
...                                                 ...
3042  TCGA_DU_6401_19831001/TCGA_DU_6401_19831001_31...
3205  TCGA_HT_A61A_20000127/TCGA_HT_A61A_20000127_34...
2239  TCGA_DU_8166_19970322/TCGA_DU_8166_19970322_21...
3390  TCGA_DU_7010_19860307/TCGA_DU_7010_19860307_38...
1835  TCGA_HT_7692_19960724/TCGA_HT_7692_19960724_17...

                                              mask_path  mask
2667  TCGA_DU_A5TY_19970709/TCGA_DU_A5TY_19970709_26...     1
1601  TCGA_FG_7637_20000922/TCGA_FG_7637_20000922_15...     1
2150  TCGA_FG_7634_20000128/TCGA_FG_7634_20000128_20...     1
2009  TCGA_DU_7018_19911220/TCGA_DU_7018_19911220_19...     1
1519  TCGA_HT_8113_19930809/TCGA_HT_8113_19930809_14...     1
...                                                 ...   ...
3042  TCGA_DU_6401_19831001/TCGA_DU_6401_19831001_31...     1
3205  TCGA_HT_A61A_20000127/TCGA_HT_A61A_20000127_34...     1
2239  TCGA_DU_8166_19970322/TCGA_DU_8166_19970322_21...     1
3390  TCGA_DU_7010_19860307/TCGA_DU_7010_19860307_38...     1
1835  TCGA_HT_7692_19960724/TCGA_HT_7692_19960724_17...     1

[103 rows x 4 columns]
X_test                   patient_id  \
1816  TCGA_DU_A5TY_19970709
3783  TCGA_DU_7304_19930325
1484  TCGA_DU_7306_19930512
1842  TCGA_FG_5962_20000626
995   TCGA_DU_6405_19851005
...                        ...
1794  TCGA_DU_A5TT_19980318
1639  TCGA_DU_8166_19970322
2771  TCGA_HT_7692_19960724
2154  TCGA_FG_6691_20020405
1286  TCGA_DU_8163_19961119

                                             image_path  \
1816  TCGA_FG_6692_20020606/TCGA_FG_6692_20020606_17...
3783  TCGA_FG_A60K_20040224/TCGA_FG_A60K_20040224_56...
1484  TCGA_DU_A5TU_19980312/TCGA_DU_A5TU_19980312_14...
1842  TCGA_HT_7882_19970125/TCGA_HT_7882_19970125_17...
995   TCGA_CS_5393_19990606/TCGA_CS_5393_19990606_10...
...                                                 ...
1794  TCGA_DU_7008_19830723/TCGA_DU_7008_19830723_17...
1639  TCGA_DU_8168_19970503/TCGA_DU_8168_19970503_15...
```

```
2771  TCGA_FG_6688_20020215/TCGA_FG_6688_20020215_27...
2154  TCGA_FG_A4MU_20030903/TCGA_FG_A4MU_20030903_20...
1286  TCGA_HT_7874_19950902/TCGA_HT_7874_19950902_12...

                                              mask_path  mask
1816  TCGA_FG_6692_20020606/TCGA_FG_6692_20020606_17...     1
3783  TCGA_FG_A60K_20040224/TCGA_FG_A60K_20040224_56...     1
1484  TCGA_DU_A5TU_19980312/TCGA_DU_A5TU_19980312_14...     1
1842  TCGA_HT_7882_19970125/TCGA_HT_7882_19970125_17...     1
995   TCGA_CS_5393_19990606/TCGA_CS_5393_19990606_10...     1
...                                                 ...   ...
1794  TCGA_DU_7008_19830723/TCGA_DU_7008_19830723_17...     1
1639  TCGA_DU_8168_19970503/TCGA_DU_8168_19970503_15...     1
2771  TCGA_FG_6688_20020215/TCGA_FG_6688_20020215_27...     1
2154  TCGA_FG_A4MU_20030903/TCGA_FG_A4MU_20030903_20...     1
1286  TCGA_HT_7874_19950902/TCGA_HT_7874_19950902_12...     1

[103 rows x 4 columns]
```

In [45]:
```python
# Create a seperate list for the imageId, classId to pass into the generator

train_ids = list(X_train.image_path)
train_mask = list(X_train.mask_path)

val_ids = list(X_val.image_path)
val_mask = list(X_val.mask_path)
```

In [46]:
```python
# utilities file contains the code for custom loass function and custom data generator
from utilities import DataGenerator

training_gen = DataGenerator(train_ids, train_mask)
validation_gen = DataGenerator(val_ids, val_mask)
```

In [47]:
```python
# building ResUNet model
# Using the res block

def resblock(X, f):

    # make a copy of the input
    X_copy = X

    # main path
    X = Conv2D(f, kernel_size = (1,1), strides = (1,1), kernel_initializer= 'he_normal')(X)
    X = BatchNormalization()(X)
    X = Activation('relu')(X)

    X = Conv2D(f, kernel_size = (3,3), strides = (1,1), padding = 'same', kernel_initializer= 'he_normal')(X)
    X = BatchNormalization()(X)

    # Short path
    X_copy = Conv2D(f, kernel_size = (1,1), strides = (1,1), kernel_initializer= 'he_normal')(X_copy)
    X_copy = BatchNormalization()(X_copy)


    # Adding the output from the main path and the short path together
    X = add([X,X_copy])
    X = Activation('relu')(X)

    return X
```

In [48]:
```python
# function to upscale and concetenate the values passed
def upsample_concat(x, skip):
    x = UpSampling2D((2,2))(x)
    merge = Concatenate()([x, skip])

    return merge
```

In [49]:
```python
# build the network
input_shape = (256, 256, 3)

# Input tensor shape
X_input = Input(input_shape)

# Stage 1
conv1_in = Conv2D(16,3, activation ='relu', padding  = 'same', kernel_initializer = 'he_normal')(X_input)
conv1_in = BatchNormalization()(conv1_in)
conv1_in = Conv2D(16,3, activation ='relu', padding  = 'same', kernel_initializer = 'he_normal')(conv1_in)
conv1_in = BatchNormalization()(conv1_in)
pool_1 =  MaxPool2D(pool_size= (2,2))(conv1_in)

# Stage 2
conv2_in = resblock(pool_1,32)
pool_2 = MaxPool2D(pool_size= (2,2))(conv2_in)

# Stage 3
conv3_in = resblock(pool_2,64)
pool_3 = MaxPool2D(pool_size= (2,2))(conv3_in)
```

```
# Stage 4
conv4_in = resblock(pool_3,128)
pool_4 = MaxPool2D(pool_size= (2,2))(conv4_in)

# Stage 5 (Bottle Neck)
conv5_in = resblock(pool_4,256)

# Upsale stage 1
up_1 = upsample_concat(conv5_in, conv4_in)
up_1 = resblock(up_1, 128)

# Upsale stage 2
up_2 = upsample_concat(up_1, conv3_in)
up_2 = resblock(up_2, 64)

# Upsale stage 3
up_3 = upsample_concat(up_2, conv2_in)
up_3 = resblock(up_3, 32)

# Upsale stage 4
up_4 = upsample_concat(up_3, conv1_in)
up_4 = resblock(up_4, 16)


# Final Output
output = Conv2D(1, (1,1), padding = 'same', activation = 'sigmoid')(up_4)

model_seg = Model(inputs = X_input, outputs = output)
```

In [50]:
```
model_seg.summary()
```

```
Model: "model_1"
__________________________________________________________________________________________________
 Layer (type)                   Output Shape         Param #     Connected to                     
==================================================================================================
 input_2 (InputLayer)           [(None, 256, 256, 3)]  0         []                               
                                                                                                  
 conv2d (Conv2D)                (None, 256, 256, 16)  448        ['input_2[0][0]']                
                                                                                                  
 batch_normalization (Batch     (None, 256, 256, 16)  64         ['conv2d[0][0]']                 
 Normalization)                                                                                   
                                                                                                  
 conv2d_1 (Conv2D)              (None, 256, 256, 16)  2320       ['batch_normalization[0][0]']    
                                                                                                  
 batch_normalization_1 (Bat     (None, 256, 256, 16)  64         ['conv2d_1[0][0]']               
 chNormalization)                                                                                 
                                                                                                  
 max_pooling2d (MaxPooling2     (None, 128, 128, 16)  0          ['batch_normalization_1[0][0]'
 D)                                                              ]                                
                                                                                                  
 conv2d_2 (Conv2D)              (None, 128, 128, 32)  544        ['max_pooling2d[0][0]']          
                                                                                                  
 batch_normalization_2 (Bat     (None, 128, 128, 32)  128        ['conv2d_2[0][0]']               
 chNormalization)                                                                                 
                                                                                                  
 activation (Activation)        (None, 128, 128, 32)  0          ['batch_normalization_2[0][0]'
                                                                 ]                                
                                                                                                  
 conv2d_3 (Conv2D)              (None, 128, 128, 32)  9248       ['activation[0][0]']             
                                                                                                  
 conv2d_4 (Conv2D)              (None, 128, 128, 32)  544        ['max_pooling2d[0][0]']          
                                                                                                  
 batch_normalization_3 (Bat     (None, 128, 128, 32)  128        ['conv2d_3[0][0]']               
 chNormalization)                                                                                 
                                                                                                  
 batch_normalization_4 (Bat     (None, 128, 128, 32)  128        ['conv2d_4[0][0]']               
 chNormalization)                                                                                 
                                                                                                  
 add (Add)                      (None, 128, 128, 32)  0          ['batch_normalization_3[0][0]'
                                                                 , 'batch_normalization_4[0][0]
                                                                 ']                               
                                                                                                  
 activation_1 (Activation)      (None, 128, 128, 32)  0          ['add[0][0]']                    
                                                                                                  
 max_pooling2d_1 (MaxPoolin     (None, 64, 64, 32)   0           ['activation_1[0][0]']           
 g2D)                                                                                             
                                                                                                  
 conv2d_5 (Conv2D)              (None, 64, 64, 64)   2112        ['max_pooling2d_1[0][0]']        
                                                                                                  
 batch_normalization_5 (Bat     (None, 64, 64, 64)   256         ['conv2d_5[0][0]']               
 chNormalization)                                                                                 
                                                                                                  
 activation_2 (Activation)      (None, 64, 64, 64)   0           ['batch_normalization_5[0][0]'
                                                                 ]                                
                                                                                                  
 conv2d_6 (Conv2D)              (None, 64, 64, 64)   36928       ['activation_2[0][0]']           
```

```
conv2d_7 (Conv2D)           (None, 64, 64, 64)           2112       ['max_pooling2d_1[0][0]']

batch_normalization_6 (Bat  (None, 64, 64, 64)           256        ['conv2d_6[0][0]']
chNormalization)

batch_normalization_7 (Bat  (None, 64, 64, 64)           256        ['conv2d_7[0][0]']
chNormalization)

add_1 (Add)                 (None, 64, 64, 64)           0          ['batch_normalization_6[0][0]'
                                                                    , 'batch_normalization_7[0][0]
                                                                    ']

activation_3 (Activation)   (None, 64, 64, 64)           0          ['add_1[0][0]']

max_pooling2d_2 (MaxPoolin  (None, 32, 32, 64)           0          ['activation_3[0][0]']
g2D)

conv2d_8 (Conv2D)           (None, 32, 32, 128)          8320       ['max_pooling2d_2[0][0]']

batch_normalization_8 (Bat  (None, 32, 32, 128)          512        ['conv2d_8[0][0]']
chNormalization)

activation_4 (Activation)   (None, 32, 32, 128)          0          ['batch_normalization_8[0][0]'
                                                                    ]

conv2d_9 (Conv2D)           (None, 32, 32, 128)          147584     ['activation_4[0][0]']

conv2d_10 (Conv2D)          (None, 32, 32, 128)          8320       ['max_pooling2d_2[0][0]']

batch_normalization_9 (Bat  (None, 32, 32, 128)          512        ['conv2d_9[0][0]']
chNormalization)

batch_normalization_10 (Ba  (None, 32, 32, 128)          512        ['conv2d_10[0][0]']
tchNormalization)

add_2 (Add)                 (None, 32, 32, 128)          0          ['batch_normalization_9[0][0]'
                                                                    , 'batch_normalization_10[0][0
                                                                    ]']

activation_5 (Activation)   (None, 32, 32, 128)          0          ['add_2[0][0]']

max_pooling2d_3 (MaxPoolin  (None, 16, 16, 128)          0          ['activation_5[0][0]']
g2D)

conv2d_11 (Conv2D)          (None, 16, 16, 256)          33024      ['max_pooling2d_3[0][0]']

batch_normalization_11 (Ba  (None, 16, 16, 256)          1024       ['conv2d_11[0][0]']
tchNormalization)

activation_6 (Activation)   (None, 16, 16, 256)          0          ['batch_normalization_11[0][0]
                                                                    ']

conv2d_12 (Conv2D)          (None, 16, 16, 256)          590080     ['activation_6[0][0]']

conv2d_13 (Conv2D)          (None, 16, 16, 256)          33024      ['max_pooling2d_3[0][0]']

batch_normalization_12 (Ba  (None, 16, 16, 256)          1024       ['conv2d_12[0][0]']
tchNormalization)

batch_normalization_13 (Ba  (None, 16, 16, 256)          1024       ['conv2d_13[0][0]']
tchNormalization)

add_3 (Add)                 (None, 16, 16, 256)          0          ['batch_normalization_12[0][0]
                                                                    ',
                                                                     'batch_normalization_13[0][0]
                                                                    ']

activation_7 (Activation)   (None, 16, 16, 256)          0          ['add_3[0][0]']

up_sampling2d (UpSampling2  (None, 32, 32, 256)          0          ['activation_7[0][0]']
D)

concatenate (Concatenate)   (None, 32, 32, 384)          0          ['up_sampling2d[0][0]',
                                                                     'activation_5[0][0]']

conv2d_14 (Conv2D)          (None, 32, 32, 128)          49280      ['concatenate[0][0]']

batch_normalization_14 (Ba  (None, 32, 32, 128)          512        ['conv2d_14[0][0]']
tchNormalization)

activation_8 (Activation)   (None, 32, 32, 128)          0          ['batch_normalization_14[0][0]
                                                                    ']

conv2d_15 (Conv2D)          (None, 32, 32, 128)          147584     ['activation_8[0][0]']

conv2d_16 (Conv2D)          (None, 32, 32, 128)          49280      ['concatenate[0][0]']

batch_normalization_15 (Ba  (None, 32, 32, 128)          512        ['conv2d_15[0][0]']
tchNormalization)
```

```
 batch_normalization_16 (Ba  (None, 32, 32, 128)          512       ['conv2d_16[0][0]']
 tchNormalization)

 add_4 (Add)                 (None, 32, 32, 128)          0         ['batch_normalization_15[0][0]
                                                                    ',
                                                                     'batch_normalization_16[0][0]
                                                                    ']

 activation_9 (Activation)   (None, 32, 32, 128)          0         ['add_4[0][0]']

 up_sampling2d_1 (UpSamplin  (None, 64, 64, 128)          0         ['activation_9[0][0]']
 g2D)

 concatenate_1 (Concatenate  (None, 64, 64, 192)          0         ['up_sampling2d_1[0][0]',
 )                                                                   'activation_3[0][0]']

 conv2d_17 (Conv2D)          (None, 64, 64, 64)           12352     ['concatenate_1[0][0]']

 batch_normalization_17 (Ba  (None, 64, 64, 64)           256       ['conv2d_17[0][0]']
 tchNormalization)

 activation_10 (Activation)  (None, 64, 64, 64)           0         ['batch_normalization_17[0][0]
                                                                    ']

 conv2d_18 (Conv2D)          (None, 64, 64, 64)           36928     ['activation_10[0][0]']

 conv2d_19 (Conv2D)          (None, 64, 64, 64)           12352     ['concatenate_1[0][0]']

 batch_normalization_18 (Ba  (None, 64, 64, 64)           256       ['conv2d_18[0][0]']
 tchNormalization)

 batch_normalization_19 (Ba  (None, 64, 64, 64)           256       ['conv2d_19[0][0]']
 tchNormalization)

 add_5 (Add)                 (None, 64, 64, 64)           0         ['batch_normalization_18[0][0]
                                                                    ',
                                                                     'batch_normalization_19[0][0]
                                                                    ']

 activation_11 (Activation)  (None, 64, 64, 64)           0         ['add_5[0][0]']

 up_sampling2d_2 (UpSamplin  (None, 128, 128, 64)         0         ['activation_11[0][0]']
 g2D)

 concatenate_2 (Concatenate  (None, 128, 128, 96)         0         ['up_sampling2d_2[0][0]',
 )                                                                   'activation_1[0][0]']

 conv2d_20 (Conv2D)          (None, 128, 128, 32)         3104      ['concatenate_2[0][0]']

 batch_normalization_20 (Ba  (None, 128, 128, 32)         128       ['conv2d_20[0][0]']
 tchNormalization)

 activation_12 (Activation)  (None, 128, 128, 32)         0         ['batch_normalization_20[0][0]
                                                                    ']

 conv2d_21 (Conv2D)          (None, 128, 128, 32)         9248      ['activation_12[0][0]']

 conv2d_22 (Conv2D)          (None, 128, 128, 32)         3104      ['concatenate_2[0][0]']

 batch_normalization_21 (Ba  (None, 128, 128, 32)         128       ['conv2d_21[0][0]']
 tchNormalization)

 batch_normalization_22 (Ba  (None, 128, 128, 32)         128       ['conv2d_22[0][0]']
 tchNormalization)

 add_6 (Add)                 (None, 128, 128, 32)         0         ['batch_normalization_21[0][0]
                                                                    ',
                                                                     'batch_normalization_22[0][0]
                                                                    ']

 activation_13 (Activation)  (None, 128, 128, 32)         0         ['add_6[0][0]']

 up_sampling2d_3 (UpSamplin  (None, 256, 256, 32)         0         ['activation_13[0][0]']
 g2D)

 concatenate_3 (Concatenate  (None, 256, 256, 48)         0         ['up_sampling2d_3[0][0]',
 )                                                                   'batch_normalization_1[0][0]'
                                                                    ]

 conv2d_23 (Conv2D)          (None, 256, 256, 16)         784       ['concatenate_3[0][0]']

 batch_normalization_23 (Ba  (None, 256, 256, 16)         64        ['conv2d_23[0][0]']
 tchNormalization)

 activation_14 (Activation)  (None, 256, 256, 16)         0         ['batch_normalization_23[0][0]
                                                                    ']

 conv2d_24 (Conv2D)          (None, 256, 256, 16)         2320      ['activation_14[0][0]']
 conv2d_25 (Conv2D)          (None, 256, 256, 16)         784       ['concatenate_3[0][0]']

 batch_normalization_24 (Ba  (None, 256, 256, 16)         64        ['conv2d_24[0][0]']
 tchNormalization)

 batch_normalization_25 (Ba  (None, 256, 256, 16)         64        ['conv2d_25[0][0]']
 tchNormalization)

 add_7 (Add)                 (None, 256, 256, 16)         0         ['batch_normalization_24[0][0]
                                                                    ',
                                                                     'batch_normalization_25[0][0]
                                                                    ']

 activation_15 (Activation)  (None, 256, 256, 16)         0         ['add_7[0][0]']

 conv2d_26 (Conv2D)          (None, 256, 256, 1)          17        ['activation_15[0][0]']

==================================================================================================
Total params: 1210513 (4.62 MB)
Trainable params: 1206129 (4.60 MB)
Non-trainable params: 4384 (17.12 KB)
__________________________________________________________________________________________________
```

Train A Segmentation ResUNet Model To Localize Tumor

In [51]:
```python
# using the custom loss function Focal_tversky, tversky_loss, tversky
from utilities import focal_tversky, tversky_loss, tversky
```

In [52]:
```python
# code for focal_tversky
from keras.losses import binary_crossentropy
import keras.backend as K
import tensorflow as tf



epsilon = 1e-5
smooth = 1

def dsc(y_true, y_pred):
    smooth = 1.
    y_true_f = K.flatten(y_true)
    y_pred_f = K.flatten(y_pred)
    intersection = K.sum(y_true_f * y_pred_f)
    score = (2. * intersection + smooth) / (K.sum(y_true_f) + K.sum(y_pred_f) + smooth)
    return score

def dice_loss(y_true, y_pred):
    loss = 1 - dsc(y_true, y_pred)
    return loss

def bce_dice_loss(y_true, y_pred):
    loss = binary_crossentropy(y_true, y_pred) + dice_loss(y_true, y_pred)
    return loss

def confusion(y_true, y_pred):
    smooth=1
    y_pred_pos = K.clip(y_pred, 0, 1)
    y_pred_neg = 1 - y_pred_pos
    y_pos = K.clip(y_true, 0, 1)
    y_neg = 1 - y_pos
    tp = K.sum(y_pos * y_pred_pos)
    fp = K.sum(y_neg * y_pred_pos)
    fn = K.sum(y_pos * y_pred_neg)
    prec = (tp + smooth)/(tp+fp+smooth)
    recall = (tp+smooth)/(tp+fn+smooth)
    return prec, recall

def tp(y_true, y_pred):
    smooth = 1
    y_pred_pos = K.round(K.clip(y_pred, 0, 1))
    y_pos = K.round(K.clip(y_true, 0, 1))
    tp = (K.sum(y_pos * y_pred_pos) + smooth)/ (K.sum(y_pos) + smooth)
    return tp

def tn(y_true, y_pred):
    smooth = 1
    y_pred_pos = K.round(K.clip(y_pred, 0, 1))
    y_pred_neg = 1 - y_pred_pos
    y_pos = K.round(K.clip(y_true, 0, 1))
    y_neg = 1 - y_pos
    tn = (K.sum(y_neg * y_pred_neg) + smooth) / (K.sum(y_neg) + smooth )
    return tn

def tversky(y_true, y_pred):
    y_true_pos = K.flatten(y_true)
    y_pred_pos = K.flatten(y_pred)
    true_pos = K.sum(y_true_pos * y_pred_pos)
    false_neg = K.sum(y_true_pos * (1-y_pred_pos))
    false_pos = K.sum((1-y_true_pos)*y_pred_pos)
    alpha = 0.7
    return (true_pos + smooth)/(true_pos + alpha*false_neg + (1-alpha)*false_pos + smooth)

def tversky_loss(y_true, y_pred):
    return 1 - tversky(y_true,y_pred)

def focal_tversky(y_true,y_pred):
    y_true = tf.cast(y_true, tf.float32)  # Cast y_true to float32
    y_pred = tf.cast(y_pred, tf.float32)
    pt_1 = tversky(y_true, y_pred)
    gamma = 0.75
    return K.pow((1-pt_1), gamma)
```

In [53]:
```
# compile model

adam = tf.keras.optimizers.legacy.Adam(lr = 0.05, epsilon = 0.1)
model_seg.compile(optimizer= adam, loss = focal_tversky, metrics = [tversky])
```

In [54]:
```
# use early stopping to exit training if validation loss is not decreasing even after certain epochs (patience)
earlystopping = EarlyStopping(monitor = 'val_loss', mode = 'min', verbose = 1, patience = 20)

# save the best model with the least validation loss
checkpointer = ModelCheckpoint(filepath='ResUNet-weights.hdf5', verbose = 1, save_best_only = True)
```

In [55]:
```
history = model_seg.fit(training_gen, epochs = 40, validation_data= validation_gen, callbacks= [checkpointer, e
```

```
Epoch 1/40
72/72 [==============================] - ETA: 0s - loss: 0.8447 - tversky: 0.2007
Epoch 1: val_loss improved from inf to 0.75451, saving model to ResUNet-weights.hdf5
72/72 [==============================] - 102s 1s/step - loss: 0.8447 - tversky: 0.2007 - val_loss: 0.7545 - val
_tversky: 0.3127
Epoch 2/40
72/72 [==============================] - ETA: 0s - loss: 0.5082 - tversky: 0.5898
Epoch 2: val_loss improved from 0.75451 to 0.47842, saving model to ResUNet-weights.hdf5
72/72 [==============================] - 96s 1s/step - loss: 0.5082 - tversky: 0.5898 - val_loss: 0.4784 - val_
tversky: 0.6244
Epoch 3/40
72/72 [==============================] - ETA: 0s - loss: 0.3789 - tversky: 0.7236
Epoch 3: val_loss did not improve from 0.47842
72/72 [==============================] - 106s 1s/step - loss: 0.3789 - tversky: 0.7236 - val_loss: 0.5118 - val
_tversky: 0.5892
Epoch 4/40
72/72 [==============================] - ETA: 0s - loss: 0.3349 - tversky: 0.7658
Epoch 4: val_loss did not improve from 0.47842
72/72 [==============================] - 108s 2s/step - loss: 0.3349 - tversky: 0.7658 - val_loss: 0.5491 - val
_tversky: 0.5476
Epoch 5/40
72/72 [==============================] - ETA: 0s - loss: 0.3042 - tversky: 0.7933
Epoch 5: val_loss improved from 0.47842 to 0.34035, saving model to ResUNet-weights.hdf5
72/72 [==============================] - 149s 2s/step - loss: 0.3042 - tversky: 0.7933 - val_loss: 0.3403 - val
_tversky: 0.7612
Epoch 6/40
72/72 [==============================] - ETA: 0s - loss: 0.2718 - tversky: 0.8224
Epoch 6: val_loss improved from 0.34035 to 0.28135, saving model to ResUNet-weights.hdf5
72/72 [==============================] - 7448s 105s/step - loss: 0.2718 - tversky: 0.8224 - val_loss: 0.2813 -
val_tversky: 0.8152
Epoch 7/40
72/72 [==============================] - ETA: 0s - loss: 0.2590 - tversky: 0.8336
Epoch 7: val_loss did not improve from 0.28135
72/72 [==============================] - 146s 2s/step - loss: 0.2590 - tversky: 0.8336 - val_loss: 0.4022 - val
_tversky: 0.7023
Epoch 8/40
72/72 [==============================] - ETA: 0s - loss: 0.2463 - tversky: 0.8442
Epoch 8: val_loss did not improve from 0.28135
72/72 [==============================] - 141s 2s/step - loss: 0.2463 - tversky: 0.8442 - val_loss: 0.3328 - val
_tversky: 0.7686
Epoch 9/40
72/72 [==============================] - ETA: 0s - loss: 0.2289 - tversky: 0.8588
Epoch 9: val_loss did not improve from 0.28135
72/72 [==============================] - 138s 2s/step - loss: 0.2289 - tversky: 0.8588 - val_loss: 0.3500 - val
_tversky: 0.7532
Epoch 10/40
72/72 [==============================] - ETA: 0s - loss: 0.2254 - tversky: 0.8606
Epoch 10: val_loss improved from 0.28135 to 0.26285, saving model to ResUNet-weights.hdf5
72/72 [==============================] - 139s 2s/step - loss: 0.2254 - tversky: 0.8606 - val_loss: 0.2629 - val
_tversky: 0.8311
Epoch 11/40
72/72 [==============================] - ETA: 0s - loss: 0.2121 - tversky: 0.8720
Epoch 11: val_loss improved from 0.26285 to 0.24320, saving model to ResUNet-weights.hdf5
72/72 [==============================] - 141s 2s/step - loss: 0.2121 - tversky: 0.8720 - val_loss: 0.2432 - val
_tversky: 0.8473
Epoch 12/40
72/72 [==============================] - ETA: 0s - loss: 0.1986 - tversky: 0.8831
Epoch 12: val_loss did not improve from 0.24320
72/72 [==============================] - 134s 2s/step - loss: 0.1986 - tversky: 0.8831 - val_loss: 0.2750 - val
```

```
_tversky: 0.8198
Epoch 13/40
72/72 [==============================] - ETA: 0s - loss: 0.2041 - tversky: 0.8785
Epoch 13: val_loss did not improve from 0.24320
72/72 [==============================] - 139s 2s/step - loss: 0.2041 - tversky: 0.8785 - val_loss: 0.2935 - val
_tversky: 0.8044
Epoch 14/40
72/72 [==============================] - ETA: 0s - loss: 0.1905 - tversky: 0.8891
Epoch 14: val_loss did not improve from 0.24320
72/72 [==============================] - 131s 2s/step - loss: 0.1905 - tversky: 0.8891 - val_loss: 0.2896 - val
_tversky: 0.8071
Epoch 15/40
72/72 [==============================] - ETA: 0s - loss: 0.1786 - tversky: 0.8986
Epoch 15: val_loss improved from 0.24320 to 0.21935, saving model to ResUNet-weights.hdf5
72/72 [==============================] - 133s 2s/step - loss: 0.1786 - tversky: 0.8986 - val_loss: 0.2194 - val
_tversky: 0.8659
Epoch 16/40
72/72 [==============================] - ETA: 0s - loss: 0.1713 - tversky: 0.9041
Epoch 16: val_loss did not improve from 0.21935
72/72 [==============================] - 134s 2s/step - loss: 0.1713 - tversky: 0.9041 - val_loss: 0.2678 - val
_tversky: 0.8269
Epoch 17/40
72/72 [==============================] - ETA: 0s - loss: 0.1593 - tversky: 0.9130
Epoch 17: val_loss improved from 0.21935 to 0.21474, saving model to ResUNet-weights.hdf5
72/72 [==============================] - 130s 2s/step - loss: 0.1593 - tversky: 0.9130 - val_loss: 0.2147 - val
_tversky: 0.8710
Epoch 18/40
72/72 [==============================] - ETA: 0s - loss: 0.1494 - tversky: 0.9202
Epoch 18: val_loss improved from 0.21474 to 0.20514, saving model to ResUNet-weights.hdf5
72/72 [==============================] - 134s 2s/step - loss: 0.1494 - tversky: 0.9202 - val_loss: 0.2051 - val
_tversky: 0.8777
Epoch 19/40
72/72 [==============================] - ETA: 0s - loss: 0.1488 - tversky: 0.9205
Epoch 19: val_loss did not improve from 0.20514
72/72 [==============================] - 133s 2s/step - loss: 0.1488 - tversky: 0.9205 - val_loss: 0.2670 - val
_tversky: 0.8268
Epoch 20/40
72/72 [==============================] - ETA: 0s - loss: 0.1505 - tversky: 0.9193
Epoch 20: val_loss improved from 0.20514 to 0.19853, saving model to ResUNet-weights.hdf5
72/72 [==============================] - 133s 2s/step - loss: 0.1505 - tversky: 0.9193 - val_loss: 0.1985 - val
_tversky: 0.8839
Epoch 21/40
72/72 [==============================] - ETA: 0s - loss: 0.1416 - tversky: 0.9258
Epoch 21: val_loss improved from 0.19853 to 0.18324, saving model to ResUNet-weights.hdf5
72/72 [==============================] - 133s 2s/step - loss: 0.1416 - tversky: 0.9258 - val_loss: 0.1832 - val
_tversky: 0.8954
Epoch 22/40
72/72 [==============================] - ETA: 0s - loss: 0.1495 - tversky: 0.9200
Epoch 22: val_loss did not improve from 0.18324
72/72 [==============================] - 133s 2s/step - loss: 0.1495 - tversky: 0.9200 - val_loss: 0.1991 - val
_tversky: 0.8829
Epoch 23/40
72/72 [==============================] - ETA: 0s - loss: 0.1456 - tversky: 0.9228
Epoch 23: val_loss did not improve from 0.18324
72/72 [==============================] - 132s 2s/step - loss: 0.1456 - tversky: 0.9228 - val_loss: 0.2055 - val
_tversky: 0.8777
Epoch 24/40
72/72 [==============================] - ETA: 0s - loss: 0.1454 - tversky: 0.9231
Epoch 24: val_loss did not improve from 0.18324
72/72 [==============================] - 132s 2s/step - loss: 0.1454 - tversky: 0.9231 - val_loss: 0.2337 - val
_tversky: 0.8553
Epoch 25/40
72/72 [==============================] - ETA: 0s - loss: 0.1339 - tversky: 0.9312
Epoch 25: val_loss did not improve from 0.18324
72/72 [==============================] - 133s 2s/step - loss: 0.1339 - tversky: 0.9312 - val_loss: 0.1893 - val
_tversky: 0.8908
Epoch 26/40
72/72 [==============================] - ETA: 0s - loss: 0.1268 - tversky: 0.9361
Epoch 26: val_loss improved from 0.18324 to 0.17267, saving model to ResUNet-weights.hdf5
72/72 [==============================] - 670s 9s/step - loss: 0.1268 - tversky: 0.9361 - val_loss: 0.1727 - val
_tversky: 0.9037
Epoch 27/40
72/72 [==============================] - ETA: 0s - loss: 0.1204 - tversky: 0.9403
Epoch 27: val_loss did not improve from 0.17267
72/72 [==============================] - 532s 7s/step - loss: 0.1204 - tversky: 0.9403 - val_loss: 0.1820 - val
_tversky: 0.8968
Epoch 28/40
72/72 [==============================] - ETA: 0s - loss: 0.1201 - tversky: 0.9405
Epoch 28: val_loss did not improve from 0.17267
72/72 [==============================] - 134s 2s/step - loss: 0.1201 - tversky: 0.9405 - val_loss: 0.1813 - val
_tversky: 0.8972
Epoch 29/40
72/72 [==============================] - ETA: 0s - loss: 0.1212 - tversky: 0.9397
Epoch 29: val_loss did not improve from 0.17267
72/72 [==============================] - 133s 2s/step - loss: 0.1212 - tversky: 0.9397 - val_loss: 0.1850 - val
_tversky: 0.8944
Epoch 30/40
72/72 [==============================] - ETA: 0s - loss: 0.1179 - tversky: 0.9419
Epoch 30: val_loss did not improve from 0.17267
```

```
72/72 [==============================] - 134s 2s/step - loss: 0.1179 - tversky: 0.9419 - val_loss: 0.1781 - val
_tversky: 0.8995
Epoch 31/40
72/72 [==============================] - ETA: 0s - loss: 0.1125 - tversky: 0.9455
Epoch 31: val_loss did not improve from 0.17267
72/72 [==============================] - 134s 2s/step - loss: 0.1125 - tversky: 0.9455 - val_loss: 0.1937 - val
_tversky: 0.8875
Epoch 32/40
72/72 [==============================] - ETA: 0s - loss: 0.1082 - tversky: 0.9482
Epoch 32: val_loss improved from 0.17267 to 0.17189, saving model to ResUNet-weights.hdf5
72/72 [==============================] - 132s 2s/step - loss: 0.1082 - tversky: 0.9482 - val_loss: 0.1719 - val
_tversky: 0.9043
Epoch 33/40
72/72 [==============================] - ETA: 0s - loss: 0.1077 - tversky: 0.9485
Epoch 33: val_loss did not improve from 0.17189
72/72 [==============================] - 132s 2s/step - loss: 0.1077 - tversky: 0.9485 - val_loss: 0.1899 - val
_tversky: 0.8907
Epoch 34/40
72/72 [==============================] - ETA: 0s - loss: 0.1118 - tversky: 0.9459
Epoch 34: val_loss did not improve from 0.17189
72/72 [==============================] - 131s 2s/step - loss: 0.1118 - tversky: 0.9459 - val_loss: 0.1929 - val
_tversky: 0.8874
Epoch 35/40
72/72 [==============================] - ETA: 0s - loss: 0.1065 - tversky: 0.9493
Epoch 35: val_loss did not improve from 0.17189
72/72 [==============================] - 132s 2s/step - loss: 0.1065 - tversky: 0.9493 - val_loss: 0.1810 - val
_tversky: 0.8973
Epoch 36/40
72/72 [==============================] - ETA: 0s - loss: 0.1038 - tversky: 0.9510
Epoch 36: val_loss did not improve from 0.17189
72/72 [==============================] - 131s 2s/step - loss: 0.1038 - tversky: 0.9510 - val_loss: 0.2112 - val
_tversky: 0.8725
Epoch 37/40
72/72 [==============================] - ETA: 0s - loss: 0.1012 - tversky: 0.9527
Epoch 37: val_loss did not improve from 0.17189
72/72 [==============================] - 131s 2s/step - loss: 0.1012 - tversky: 0.9527 - val_loss: 0.1852 - val
_tversky: 0.8936
Epoch 38/40
72/72 [==============================] - ETA: 0s - loss: 0.0993 - tversky: 0.9539
Epoch 38: val_loss did not improve from 0.17189
72/72 [==============================] - 131s 2s/step - loss: 0.0993 - tversky: 0.9539 - val_loss: 0.1868 - val
_tversky: 0.8928
Epoch 39/40
72/72 [==============================] - ETA: 0s - loss: 0.0980 - tversky: 0.9547
Epoch 39: val_loss improved from 0.17189 to 0.16629, saving model to ResUNet-weights.hdf5
72/72 [==============================] - 131s 2s/step - loss: 0.0980 - tversky: 0.9547 - val_loss: 0.1663 - val
_tversky: 0.9081
Epoch 40/40
72/72 [==============================] - ETA: 0s - loss: 0.0967 - tversky: 0.9554
Epoch 40: val_loss did not improve from 0.16629
72/72 [==============================] - 131s 2s/step - loss: 0.0967 - tversky: 0.9554 - val_loss: 0.1769 - val
_tversky: 0.9004
```

In [56]:
```python
# save model architecture to json file for future use
model_json = model_seg.to_json()
with open('ResUNet_model.json', 'w') as json_file:
    json_file.write(model_json)
```

Assess Trained Segmentation ResUNet

In [58]:
```python
from utilities import focal_tversky, tversky_loss, tversky

with open('ResUNet_model.json', 'r') as json_file:
    json_savedModel= json_file.read()

# load the model architecture
model_seg = tf.keras.models.model_from_json(json_savedModel)
model_seg.load_weights('ResUNet-weights.hdf5')
adam = tf.keras.optimizers.Adam(lr = 0.05, epsilon = 0.1)
model_seg.compile(optimizer = adam, loss = focal_tversky, metrics = [tversky])
```

```
WARNING:absl:`lr` is deprecated in Keras optimizer, please use `learning_rate` or use the legacy optimizer, e.g
.,tf.keras.optimizers.legacy.Adam.
```

In [62]:
```python
# Utilities file contains the code for custom loss function and custom data generator
from utilities import prediction

# making prediction
image_id, mask, has_mask = prediction(test, model, model_seg)
```

```
1/1 [==============================] - 0s 211ms/step
1/1 [==============================] - 0s 230ms/step
1/1 [==============================] - 0s 204ms/step
1/1 [==============================] - 0s 105ms/step
1/1 [==============================] - 0s 217ms/step
1/1 [==============================] - 0s 109ms/step
1/1 [==============================] - 0s 191ms/step
```

```
1/1 [==============================] - 0s 99ms/step
1/1 [==============================] - 0s 224ms/step
1/1 [==============================] - 0s 213ms/step
1/1 [==============================] - 0s 221ms/step
1/1 [==============================] - 0s 116ms/step
1/1 [==============================] - 0s 212ms/step
1/1 [==============================] - 0s 100ms/step
1/1 [==============================] - 0s 210ms/step
1/1 [==============================] - 0s 224ms/step
1/1 [==============================] - 0s 200ms/step
1/1 [==============================] - 0s 211ms/step
1/1 [==============================] - 0s 234ms/step
1/1 [==============================] - 0s 181ms/step
1/1 [==============================] - 0s 166ms/step
1/1 [==============================] - 0s 105ms/step
1/1 [==============================] - 0s 205ms/step
1/1 [==============================] - 0s 109ms/step
1/1 [==============================] - 0s 198ms/step
1/1 [==============================] - 0s 101ms/step
1/1 [==============================] - 0s 201ms/step
1/1 [==============================] - 0s 185ms/step
1/1 [==============================] - 0s 100ms/step
1/1 [==============================] - 0s 221ms/step
1/1 [==============================] - 0s 101ms/step
1/1 [==============================] - 0s 191ms/step
1/1 [==============================] - 0s 117ms/step
1/1 [==============================] - 0s 223ms/step
1/1 [==============================] - 0s 166ms/step
1/1 [==============================] - 0s 204ms/step
1/1 [==============================] - 0s 213ms/step
1/1 [==============================] - 0s 118ms/step
1/1 [==============================] - 0s 209ms/step
1/1 [==============================] - 0s 198ms/step
1/1 [==============================] - 0s 99ms/step
1/1 [==============================] - 0s 175ms/step
1/1 [==============================] - 0s 220ms/step
1/1 [==============================] - 0s 112ms/step
1/1 [==============================] - 0s 221ms/step
1/1 [==============================] - 0s 97ms/step
1/1 [==============================] - 0s 236ms/step
1/1 [==============================] - 0s 226ms/step
1/1 [==============================] - 0s 223ms/step
1/1 [==============================] - 0s 227ms/step
1/1 [==============================] - 0s 92ms/step
1/1 [==============================] - 0s 202ms/step
1/1 [==============================] - 0s 207ms/step
1/1 [==============================] - 0s 231ms/step
1/1 [==============================] - 0s 214ms/step
1/1 [==============================] - 0s 197ms/step
1/1 [==============================] - 0s 189ms/step
1/1 [==============================] - 0s 199ms/step
1/1 [==============================] - 0s 114ms/step
1/1 [==============================] - 0s 233ms/step
1/1 [==============================] - 0s 107ms/step
1/1 [==============================] - 0s 234ms/step
1/1 [==============================] - 0s 115ms/step
1/1 [==============================] - 0s 226ms/step
1/1 [==============================] - 0s 200ms/step
1/1 [==============================] - 0s 122ms/step
1/1 [==============================] - 0s 217ms/step
1/1 [==============================] - 0s 102ms/step
1/1 [==============================] - 0s 187ms/step
1/1 [==============================] - 0s 201ms/step
1/1 [==============================] - 0s 103ms/step
1/1 [==============================] - 0s 214ms/step
1/1 [==============================] - 0s 192ms/step
1/1 [==============================] - 0s 91ms/step
1/1 [==============================] - 0s 214ms/step
1/1 [==============================] - 0s 229ms/step
1/1 [==============================] - 0s 109ms/step
1/1 [==============================] - 0s 243ms/step
1/1 [==============================] - 0s 217ms/step
1/1 [==============================] - 0s 229ms/step
1/1 [==============================] - 0s 189ms/step
1/1 [==============================] - 0s 104ms/step
1/1 [==============================] - 0s 218ms/step
1/1 [==============================] - 0s 107ms/step
1/1 [==============================] - 0s 180ms/step
1/1 [==============================] - 0s 194ms/step
1/1 [==============================] - 0s 105ms/step
1/1 [==============================] - 0s 168ms/step
1/1 [==============================] - 0s 202ms/step
1/1 [==============================] - 0s 219ms/step
1/1 [==============================] - 0s 214ms/step
1/1 [==============================] - 0s 205ms/step
1/1 [==============================] - 0s 105ms/step
1/1 [==============================] - 0s 240ms/step
1/1 [==============================] - 0s 219ms/step
1/1 [==============================] - 0s 115ms/step
```

```
1/1 [==============================] - 0s 231ms/step
1/1 [==============================] - 0s 219ms/step
1/1 [==============================] - 0s 226ms/step
1/1 [==============================] - 0s 114ms/step
1/1 [==============================] - 0s 200ms/step
1/1 [==============================] - 0s 98ms/step
1/1 [==============================] - 0s 183ms/step
1/1 [==============================] - 0s 191ms/step
1/1 [==============================] - 0s 223ms/step
1/1 [==============================] - 0s 116ms/step
1/1 [==============================] - 0s 213ms/step
1/1 [==============================] - 0s 234ms/step
1/1 [==============================] - 0s 99ms/step
1/1 [==============================] - 0s 209ms/step
1/1 [==============================] - 0s 200ms/step
1/1 [==============================] - 0s 112ms/step
1/1 [==============================] - 0s 225ms/step
1/1 [==============================] - 0s 109ms/step
1/1 [==============================] - 0s 185ms/step
1/1 [==============================] - 0s 219ms/step
1/1 [==============================] - 0s 225ms/step
1/1 [==============================] - 0s 224ms/step
1/1 [==============================] - 0s 201ms/step
1/1 [==============================] - 0s 223ms/step
1/1 [==============================] - 0s 201ms/step
1/1 [==============================] - 0s 188ms/step
1/1 [==============================] - 0s 204ms/step
1/1 [==============================] - 0s 233ms/step
1/1 [==============================] - 0s 112ms/step
1/1 [==============================] - 0s 207ms/step
1/1 [==============================] - 0s 209ms/step
1/1 [==============================] - 0s 115ms/step
1/1 [==============================] - 0s 209ms/step
1/1 [==============================] - 0s 208ms/step
1/1 [==============================] - 0s 211ms/step
1/1 [==============================] - 0s 207ms/step
1/1 [==============================] - 0s 210ms/step
1/1 [==============================] - 0s 213ms/step
1/1 [==============================] - 0s 217ms/step
1/1 [==============================] - 0s 114ms/step
1/1 [==============================] - 0s 229ms/step
1/1 [==============================] - 0s 219ms/step
1/1 [==============================] - 0s 231ms/step
1/1 [==============================] - 0s 117ms/step
1/1 [==============================] - 0s 232ms/step
1/1 [==============================] - 0s 113ms/step
1/1 [==============================] - 0s 223ms/step
1/1 [==============================] - 0s 232ms/step
1/1 [==============================] - 0s 204ms/step
1/1 [==============================] - 0s 210ms/step
1/1 [==============================] - 0s 114ms/step
1/1 [==============================] - 0s 221ms/step
1/1 [==============================] - 0s 217ms/step
1/1 [==============================] - 0s 113ms/step
1/1 [==============================] - 0s 234ms/step
1/1 [==============================] - 0s 226ms/step
1/1 [==============================] - 0s 218ms/step
1/1 [==============================] - 0s 194ms/step
1/1 [==============================] - 0s 144ms/step
1/1 [==============================] - 0s 115ms/step
1/1 [==============================] - 0s 216ms/step
1/1 [==============================] - 0s 212ms/step
1/1 [==============================] - 0s 110ms/step
1/1 [==============================] - 0s 203ms/step
1/1 [==============================] - 0s 90ms/step
1/1 [==============================] - 0s 200ms/step
1/1 [==============================] - 0s 114ms/step
1/1 [==============================] - 0s 209ms/step
1/1 [==============================] - 0s 101ms/step
1/1 [==============================] - 0s 200ms/step
1/1 [==============================] - 0s 108ms/step
1/1 [==============================] - 0s 222ms/step
1/1 [==============================] - 0s 111ms/step
1/1 [==============================] - 0s 209ms/step
1/1 [==============================] - 0s 223ms/step
1/1 [==============================] - 0s 209ms/step
1/1 [==============================] - 0s 200ms/step
1/1 [==============================] - 0s 230ms/step
1/1 [==============================] - 0s 113ms/step
1/1 [==============================] - 0s 206ms/step
1/1 [==============================] - 0s 113ms/step
1/1 [==============================] - 0s 222ms/step
1/1 [==============================] - 0s 221ms/step
1/1 [==============================] - 0s 107ms/step
1/1 [==============================] - 0s 215ms/step
1/1 [==============================] - 0s 105ms/step
1/1 [==============================] - 0s 208ms/step
1/1 [==============================] - 0s 105ms/step
1/1 [==============================] - 0s 221ms/step
```

```
1/1 [==============================] - 0s 205ms/step
1/1 [==============================] - 0s 198ms/step
1/1 [==============================] - 0s 114ms/step
1/1 [==============================] - 0s 213ms/step
1/1 [==============================] - 0s 102ms/step
1/1 [==============================] - 0s 214ms/step
1/1 [==============================] - 0s 112ms/step
1/1 [==============================] - 0s 210ms/step
1/1 [==============================] - 0s 188ms/step
1/1 [==============================] - 0s 215ms/step
1/1 [==============================] - 0s 226ms/step
1/1 [==============================] - 0s 116ms/step
1/1 [==============================] - 0s 214ms/step
1/1 [==============================] - 0s 108ms/step
1/1 [==============================] - 0s 197ms/step
1/1 [==============================] - 0s 95ms/step
1/1 [==============================] - 0s 208ms/step
1/1 [==============================] - 0s 114ms/step
1/1 [==============================] - 0s 217ms/step
1/1 [==============================] - 0s 219ms/step
1/1 [==============================] - 0s 213ms/step
1/1 [==============================] - 0s 215ms/step
1/1 [==============================] - 0s 219ms/step
1/1 [==============================] - 0s 116ms/step
1/1 [==============================] - 0s 226ms/step
1/1 [==============================] - 0s 199ms/step
1/1 [==============================] - 0s 117ms/step
1/1 [==============================] - 0s 221ms/step
1/1 [==============================] - 0s 220ms/step
1/1 [==============================] - 0s 210ms/step
1/1 [==============================] - 0s 236ms/step
1/1 [==============================] - 0s 96ms/step
1/1 [==============================] - 0s 199ms/step
1/1 [==============================] - 0s 98ms/step
1/1 [==============================] - 0s 189ms/step
1/1 [==============================] - 0s 113ms/step
1/1 [==============================] - 0s 187ms/step
1/1 [==============================] - 0s 213ms/step
1/1 [==============================] - 0s 105ms/step
1/1 [==============================] - 0s 209ms/step
1/1 [==============================] - 0s 204ms/step
1/1 [==============================] - 0s 225ms/step
1/1 [==============================] - 0s 219ms/step
1/1 [==============================] - 0s 115ms/step
1/1 [==============================] - 0s 218ms/step
1/1 [==============================] - 0s 102ms/step
1/1 [==============================] - 0s 205ms/step
1/1 [==============================] - 0s 207ms/step
1/1 [==============================] - 0s 199ms/step
1/1 [==============================] - 0s 96ms/step
1/1 [==============================] - 0s 194ms/step
1/1 [==============================] - 0s 113ms/step
1/1 [==============================] - 0s 215ms/step
1/1 [==============================] - 0s 195ms/step
1/1 [==============================] - 0s 208ms/step
1/1 [==============================] - 0s 218ms/step
1/1 [==============================] - 0s 213ms/step
1/1 [==============================] - 0s 214ms/step
1/1 [==============================] - 0s 109ms/step
1/1 [==============================] - 0s 205ms/step
1/1 [==============================] - 0s 98ms/step
1/1 [==============================] - 0s 213ms/step
1/1 [==============================] - 0s 112ms/step
1/1 [==============================] - 0s 203ms/step
1/1 [==============================] - 0s 176ms/step
1/1 [==============================] - 0s 196ms/step
1/1 [==============================] - 0s 91ms/step
1/1 [==============================] - 0s 209ms/step
1/1 [==============================] - 0s 192ms/step
1/1 [==============================] - 0s 100ms/step
1/1 [==============================] - 0s 209ms/step
1/1 [==============================] - 0s 197ms/step
1/1 [==============================] - 0s 217ms/step
1/1 [==============================] - 0s 109ms/step
1/1 [==============================] - 0s 208ms/step
1/1 [==============================] - 0s 212ms/step
1/1 [==============================] - 0s 113ms/step
1/1 [==============================] - 0s 214ms/step
1/1 [==============================] - 0s 86ms/step
1/1 [==============================] - 0s 210ms/step
1/1 [==============================] - 0s 211ms/step
1/1 [==============================] - 0s 110ms/step
1/1 [==============================] - 0s 200ms/step
1/1 [==============================] - 0s 106ms/step
1/1 [==============================] - 0s 208ms/step
1/1 [==============================] - 0s 117ms/step
1/1 [==============================] - 0s 211ms/step
1/1 [==============================] - 0s 108ms/step
1/1 [==============================] - 0s 219ms/step
```

```
1/1 [==============================] - 0s 194ms/step
1/1 [==============================] - 0s 113ms/step
1/1 [==============================] - 0s 226ms/step
1/1 [==============================] - 0s 108ms/step
1/1 [==============================] - 0s 222ms/step
1/1 [==============================] - 0s 110ms/step
1/1 [==============================] - 0s 209ms/step
1/1 [==============================] - 0s 210ms/step
1/1 [==============================] - 0s 215ms/step
1/1 [==============================] - 0s 201ms/step
1/1 [==============================] - 0s 94ms/step
1/1 [==============================] - 0s 208ms/step
1/1 [==============================] - 0s 106ms/step
1/1 [==============================] - 0s 212ms/step
1/1 [==============================] - 0s 192ms/step
1/1 [==============================] - 0s 221ms/step
1/1 [==============================] - 0s 208ms/step
1/1 [==============================] - 0s 93ms/step
1/1 [==============================] - 0s 213ms/step
1/1 [==============================] - 0s 111ms/step
1/1 [==============================] - 0s 214ms/step
1/1 [==============================] - 0s 104ms/step
1/1 [==============================] - 0s 204ms/step
1/1 [==============================] - 0s 97ms/step
1/1 [==============================] - 0s 214ms/step
1/1 [==============================] - 0s 208ms/step
1/1 [==============================] - 0s 104ms/step
1/1 [==============================] - 0s 212ms/step
1/1 [==============================] - 0s 108ms/step
1/1 [==============================] - 0s 207ms/step
1/1 [==============================] - 0s 112ms/step
1/1 [==============================] - 0s 207ms/step
1/1 [==============================] - 0s 200ms/step
1/1 [==============================] - 0s 211ms/step
1/1 [==============================] - 0s 206ms/step
1/1 [==============================] - 0s 109ms/step
1/1 [==============================] - 0s 205ms/step
1/1 [==============================] - 0s 216ms/step
1/1 [==============================] - 0s 209ms/step
1/1 [==============================] - 0s 219ms/step
1/1 [==============================] - 0s 116ms/step
1/1 [==============================] - 0s 220ms/step
1/1 [==============================] - 0s 207ms/step
1/1 [==============================] - 0s 195ms/step
1/1 [==============================] - 0s 118ms/step
1/1 [==============================] - 0s 215ms/step
1/1 [==============================] - 0s 99ms/step
1/1 [==============================] - 0s 204ms/step
1/1 [==============================] - 0s 198ms/step
1/1 [==============================] - 0s 202ms/step
1/1 [==============================] - 0s 99ms/step
1/1 [==============================] - 0s 207ms/step
1/1 [==============================] - 0s 221ms/step
1/1 [==============================] - 0s 108ms/step
1/1 [==============================] - 0s 230ms/step
1/1 [==============================] - 0s 108ms/step
1/1 [==============================] - 0s 207ms/step
1/1 [==============================] - 0s 107ms/step
1/1 [==============================] - 0s 206ms/step
1/1 [==============================] - 0s 216ms/step
1/1 [==============================] - 0s 88ms/step
1/1 [==============================] - 0s 202ms/step
1/1 [==============================] - 0s 111ms/step
1/1 [==============================] - 0s 202ms/step
1/1 [==============================] - 0s 206ms/step
1/1 [==============================] - 0s 215ms/step
1/1 [==============================] - 0s 218ms/step
1/1 [==============================] - 0s 182ms/step
1/1 [==============================] - 0s 202ms/step
1/1 [==============================] - 0s 203ms/step
1/1 [==============================] - 0s 196ms/step
1/1 [==============================] - 0s 216ms/step
1/1 [==============================] - 0s 109ms/step
1/1 [==============================] - 0s 195ms/step
1/1 [==============================] - 0s 220ms/step
1/1 [==============================] - 0s 226ms/step
1/1 [==============================] - 0s 210ms/step
1/1 [==============================] - 0s 246ms/step
1/1 [==============================] - 0s 204ms/step
1/1 [==============================] - 0s 90ms/step
1/1 [==============================] - 0s 204ms/step
1/1 [==============================] - 0s 104ms/step
1/1 [==============================] - 0s 231ms/step
1/1 [==============================] - 0s 133ms/step
1/1 [==============================] - 0s 240ms/step
1/1 [==============================] - 0s 120ms/step
1/1 [==============================] - 0s 223ms/step
1/1 [==============================] - 0s 244ms/step
1/1 [==============================] - 0s 241ms/step
```

```
1/1 [==============================] - 0s 207ms/step
1/1 [==============================] - 0s 219ms/step
1/1 [==============================] - 0s 251ms/step
1/1 [==============================] - 0s 235ms/step
1/1 [==============================] - 0s 227ms/step
1/1 [==============================] - 0s 107ms/step
1/1 [==============================] - 0s 238ms/step
1/1 [==============================] - 0s 90ms/step
1/1 [==============================] - 0s 235ms/step
1/1 [==============================] - 0s 118ms/step
1/1 [==============================] - 0s 247ms/step
1/1 [==============================] - 0s 229ms/step
1/1 [==============================] - 0s 116ms/step
1/1 [==============================] - 0s 196ms/step
1/1 [==============================] - 0s 203ms/step
1/1 [==============================] - 0s 226ms/step
1/1 [==============================] - 0s 213ms/step
1/1 [==============================] - 0s 105ms/step
1/1 [==============================] - 0s 225ms/step
1/1 [==============================] - 0s 227ms/step
1/1 [==============================] - 0s 207ms/step
1/1 [==============================] - 0s 106ms/step
1/1 [==============================] - 0s 221ms/step
1/1 [==============================] - 0s 90ms/step
1/1 [==============================] - 0s 195ms/step
1/1 [==============================] - 0s 110ms/step
1/1 [==============================] - 0s 207ms/step
1/1 [==============================] - 0s 113ms/step
1/1 [==============================] - 0s 202ms/step
1/1 [==============================] - 0s 209ms/step
1/1 [==============================] - 0s 208ms/step
1/1 [==============================] - 0s 104ms/step
1/1 [==============================] - 0s 223ms/step
1/1 [==============================] - 0s 218ms/step
1/1 [==============================] - 0s 208ms/step
1/1 [==============================] - 0s 216ms/step
1/1 [==============================] - 0s 219ms/step
1/1 [==============================] - 0s 210ms/step
1/1 [==============================] - 0s 211ms/step
1/1 [==============================] - 0s 209ms/step
1/1 [==============================] - 0s 112ms/step
1/1 [==============================] - 0s 198ms/step
1/1 [==============================] - 0s 218ms/step
1/1 [==============================] - 0s 109ms/step
1/1 [==============================] - 0s 211ms/step
1/1 [==============================] - 0s 209ms/step
1/1 [==============================] - 0s 218ms/step
1/1 [==============================] - 0s 106ms/step
1/1 [==============================] - 0s 214ms/step
1/1 [==============================] - 0s 107ms/step
1/1 [==============================] - 0s 213ms/step
1/1 [==============================] - 0s 102ms/step
1/1 [==============================] - 0s 205ms/step
1/1 [==============================] - 0s 115ms/step
1/1 [==============================] - 0s 225ms/step
1/1 [==============================] - 0s 113ms/step
1/1 [==============================] - 0s 199ms/step
1/1 [==============================] - 0s 204ms/step
1/1 [==============================] - 0s 226ms/step
1/1 [==============================] - 0s 223ms/step
1/1 [==============================] - 0s 218ms/step
1/1 [==============================] - 0s 211ms/step
1/1 [==============================] - 0s 103ms/step
1/1 [==============================] - 0s 217ms/step
1/1 [==============================] - 0s 111ms/step
1/1 [==============================] - 0s 192ms/step
1/1 [==============================] - 0s 211ms/step
1/1 [==============================] - 0s 212ms/step
1/1 [==============================] - 0s 107ms/step
1/1 [==============================] - 0s 197ms/step
1/1 [==============================] - 0s 206ms/step
1/1 [==============================] - 0s 220ms/step
1/1 [==============================] - 0s 118ms/step
1/1 [==============================] - 0s 224ms/step
1/1 [==============================] - 0s 221ms/step
1/1 [==============================] - 0s 106ms/step
1/1 [==============================] - 0s 204ms/step
1/1 [==============================] - 0s 112ms/step
1/1 [==============================] - 0s 198ms/step
1/1 [==============================] - 0s 198ms/step
1/1 [==============================] - 0s 204ms/step
1/1 [==============================] - 0s 198ms/step
1/1 [==============================] - 0s 175ms/step
1/1 [==============================] - 0s 105ms/step
1/1 [==============================] - 0s 213ms/step
1/1 [==============================] - 0s 189ms/step
1/1 [==============================] - 0s 223ms/step
1/1 [==============================] - 0s 111ms/step
1/1 [==============================] - 0s 227ms/step
```

```
1/1 [==============================] - 0s 180ms/step
1/1 [==============================] - 0s 229ms/step
1/1 [==============================] - 0s 99ms/step
1/1 [==============================] - 0s 195ms/step
1/1 [==============================] - 0s 195ms/step
1/1 [==============================] - 0s 233ms/step
1/1 [==============================] - 0s 198ms/step
1/1 [==============================] - 0s 166ms/step
1/1 [==============================] - 0s 82ms/step
1/1 [==============================] - 0s 178ms/step
1/1 [==============================] - 0s 102ms/step
1/1 [==============================] - 0s 182ms/step
1/1 [==============================] - 0s 191ms/step
1/1 [==============================] - 0s 100ms/step
1/1 [==============================] - 0s 185ms/step
1/1 [==============================] - 0s 101ms/step
1/1 [==============================] - 0s 221ms/step
1/1 [==============================] - 0s 206ms/step
1/1 [==============================] - 0s 203ms/step
1/1 [==============================] - 0s 213ms/step
1/1 [==============================] - 0s 105ms/step
1/1 [==============================] - 0s 202ms/step
1/1 [==============================] - 0s 229ms/step
1/1 [==============================] - 0s 106ms/step
1/1 [==============================] - 0s 216ms/step
1/1 [==============================] - 0s 222ms/step
1/1 [==============================] - 0s 215ms/step
1/1 [==============================] - 0s 222ms/step
1/1 [==============================] - 0s 109ms/step
1/1 [==============================] - 0s 205ms/step
1/1 [==============================] - 0s 118ms/step
1/1 [==============================] - 0s 191ms/step
1/1 [==============================] - 0s 229ms/step
1/1 [==============================] - 0s 202ms/step
1/1 [==============================] - 0s 223ms/step
1/1 [==============================] - 0s 108ms/step
1/1 [==============================] - 0s 201ms/step
1/1 [==============================] - 0s 202ms/step
1/1 [==============================] - 0s 119ms/step
1/1 [==============================] - 0s 189ms/step
1/1 [==============================] - 0s 231ms/step
1/1 [==============================] - 0s 103ms/step
1/1 [==============================] - 0s 211ms/step
1/1 [==============================] - 0s 192ms/step
1/1 [==============================] - 0s 201ms/step
1/1 [==============================] - 0s 103ms/step
1/1 [==============================] - 0s 216ms/step
1/1 [==============================] - 0s 105ms/step
1/1 [==============================] - 0s 191ms/step
1/1 [==============================] - 0s 195ms/step
1/1 [==============================] - 0s 107ms/step
1/1 [==============================] - 0s 192ms/step
1/1 [==============================] - 0s 98ms/step
1/1 [==============================] - 0s 215ms/step
1/1 [==============================] - 0s 188ms/step
1/1 [==============================] - 0s 121ms/step
1/1 [==============================] - 0s 225ms/step
1/1 [==============================] - 0s 118ms/step
1/1 [==============================] - 0s 205ms/step
1/1 [==============================] - 0s 200ms/step
1/1 [==============================] - 0s 110ms/step
1/1 [==============================] - 0s 227ms/step
1/1 [==============================] - 0s 190ms/step
1/1 [==============================] - 0s 148ms/step
1/1 [==============================] - 0s 110ms/step
1/1 [==============================] - 0s 207ms/step
1/1 [==============================] - 0s 214ms/step
1/1 [==============================] - 0s 231ms/step
1/1 [==============================] - 0s 200ms/step
1/1 [==============================] - 0s 113ms/step
1/1 [==============================] - 0s 222ms/step
1/1 [==============================] - 0s 114ms/step
1/1 [==============================] - 0s 212ms/step
1/1 [==============================] - 0s 213ms/step
1/1 [==============================] - 0s 109ms/step
1/1 [==============================] - 0s 210ms/step
1/1 [==============================] - 0s 114ms/step
1/1 [==============================] - 0s 217ms/step
1/1 [==============================] - 0s 217ms/step
1/1 [==============================] - 0s 115ms/step
1/1 [==============================] - 0s 221ms/step
1/1 [==============================] - 0s 208ms/step
1/1 [==============================] - 0s 232ms/step
1/1 [==============================] - 0s 112ms/step
1/1 [==============================] - 0s 223ms/step
1/1 [==============================] - 0s 216ms/step
1/1 [==============================] - 0s 105ms/step
1/1 [==============================] - 0s 213ms/step
1/1 [==============================] - 0s 228ms/step
```

```
1/1 [==============================] - 0s 114ms/step
1/1 [==============================] - 0s 221ms/step
1/1 [==============================] - 0s 231ms/step
1/1 [==============================] - 0s 115ms/step
1/1 [==============================] - 0s 215ms/step
1/1 [==============================] - 0s 95ms/step
1/1 [==============================] - 0s 213ms/step
1/1 [==============================] - 0s 207ms/step
1/1 [==============================] - 0s 206ms/step
1/1 [==============================] - 0s 214ms/step
1/1 [==============================] - 0s 114ms/step
1/1 [==============================] - 0s 209ms/step
1/1 [==============================] - 0s 179ms/step
1/1 [==============================] - 0s 87ms/step
1/1 [==============================] - 0s 174ms/step
1/1 [==============================] - 0s 188ms/step
1/1 [==============================] - 0s 182ms/step
1/1 [==============================] - 0s 188ms/step
1/1 [==============================] - 0s 105ms/step
1/1 [==============================] - 0s 217ms/step
1/1 [==============================] - 0s 228ms/step
1/1 [==============================] - 0s 114ms/step
1/1 [==============================] - 0s 197ms/step
1/1 [==============================] - 0s 211ms/step
1/1 [==============================] - 0s 199ms/step
1/1 [==============================] - 0s 107ms/step
1/1 [==============================] - 0s 205ms/step
1/1 [==============================] - 0s 235ms/step
1/1 [==============================] - 0s 107ms/step
1/1 [==============================] - 0s 211ms/step
1/1 [==============================] - 0s 194ms/step
1/1 [==============================] - 0s 210ms/step
1/1 [==============================] - 0s 106ms/step
1/1 [==============================] - 0s 213ms/step
1/1 [==============================] - 0s 226ms/step
1/1 [==============================] - 0s 222ms/step
1/1 [==============================] - 0s 108ms/step
1/1 [==============================] - 0s 218ms/step
1/1 [==============================] - 0s 223ms/step
1/1 [==============================] - 0s 226ms/step
1/1 [==============================] - 0s 209ms/step
1/1 [==============================] - 0s 112ms/step
1/1 [==============================] - 0s 206ms/step
1/1 [==============================] - 0s 226ms/step
1/1 [==============================] - 0s 248ms/step
1/1 [==============================] - 0s 118ms/step
1/1 [==============================] - 0s 209ms/step
1/1 [==============================] - 0s 217ms/step
1/1 [==============================] - 0s 215ms/step
1/1 [==============================] - 0s 208ms/step
1/1 [==============================] - 0s 106ms/step
1/1 [==============================] - 0s 222ms/step
1/1 [==============================] - 0s 109ms/step
1/1 [==============================] - 0s 207ms/step
1/1 [==============================] - 0s 135ms/step
1/1 [==============================] - 0s 212ms/step
1/1 [==============================] - 0s 215ms/step
1/1 [==============================] - 0s 213ms/step
1/1 [==============================] - 0s 221ms/step
1/1 [==============================] - 0s 209ms/step
1/1 [==============================] - 0s 92ms/step
1/1 [==============================] - 0s 231ms/step
1/1 [==============================] - 0s 211ms/step
1/1 [==============================] - 0s 225ms/step
1/1 [==============================] - 0s 221ms/step
1/1 [==============================] - 0s 107ms/step
1/1 [==============================] - 0s 210ms/step
1/1 [==============================] - 0s 203ms/step
1/1 [==============================] - 0s 235ms/step
1/1 [==============================] - 0s 224ms/step
1/1 [==============================] - 0s 90ms/step
1/1 [==============================] - 0s 226ms/step
1/1 [==============================] - 0s 207ms/step
1/1 [==============================] - 0s 108ms/step
1/1 [==============================] - 0s 224ms/step
1/1 [==============================] - 0s 208ms/step
1/1 [==============================] - 0s 101ms/step
1/1 [==============================] - 0s 199ms/step
1/1 [==============================] - 0s 112ms/step
1/1 [==============================] - 0s 221ms/step
1/1 [==============================] - 0s 214ms/step
1/1 [==============================] - 0s 243ms/step
1/1 [==============================] - 0s 228ms/step
1/1 [==============================] - 0s 113ms/step
1/1 [==============================] - 0s 199ms/step
1/1 [==============================] - 0s 218ms/step
1/1 [==============================] - 0s 215ms/step
1/1 [==============================] - 0s 199ms/step
1/1 [==============================] - 0s 115ms/step
```

```
1/1 [==============================] - 0s 221ms/step
1/1 [==============================] - 0s 214ms/step
1/1 [==============================] - 0s 119ms/step
1/1 [==============================] - 0s 208ms/step
1/1 [==============================] - 0s 113ms/step
1/1 [==============================] - 0s 218ms/step
1/1 [==============================] - 0s 106ms/step
1/1 [==============================] - 0s 221ms/step
1/1 [==============================] - 0s 229ms/step
1/1 [==============================] - 0s 209ms/step
1/1 [==============================] - 0s 203ms/step
1/1 [==============================] - 0s 112ms/step
1/1 [==============================] - 0s 209ms/step
1/1 [==============================] - 0s 109ms/step
1/1 [==============================] - 0s 238ms/step
1/1 [==============================] - 0s 113ms/step
1/1 [==============================] - 0s 205ms/step
1/1 [==============================] - 0s 106ms/step
1/1 [==============================] - 0s 210ms/step
1/1 [==============================] - 0s 203ms/step
1/1 [==============================] - 0s 108ms/step
1/1 [==============================] - 0s 217ms/step
1/1 [==============================] - 0s 101ms/step
1/1 [==============================] - 0s 200ms/step
1/1 [==============================] - 0s 221ms/step
1/1 [==============================] - 0s 108ms/step
1/1 [==============================] - 0s 214ms/step
1/1 [==============================] - 0s 225ms/step
1/1 [==============================] - 0s 226ms/step
1/1 [==============================] - 0s 201ms/step
1/1 [==============================] - 0s 209ms/step
1/1 [==============================] - 0s 205ms/step
1/1 [==============================] - 0s 113ms/step
1/1 [==============================] - 0s 173ms/step
1/1 [==============================] - 0s 173ms/step
1/1 [==============================] - 0s 200ms/step
1/1 [==============================] - 0s 219ms/step
1/1 [==============================] - 0s 209ms/step
1/1 [==============================] - 0s 201ms/step
1/1 [==============================] - 0s 115ms/step
1/1 [==============================] - 0s 193ms/step
1/1 [==============================] - 0s 208ms/step
1/1 [==============================] - 0s 216ms/step
1/1 [==============================] - 0s 204ms/step
1/1 [==============================] - 0s 206ms/step
1/1 [==============================] - 0s 220ms/step
1/1 [==============================] - 0s 189ms/step
1/1 [==============================] - 0s 195ms/step
1/1 [==============================] - 0s 109ms/step
1/1 [==============================] - 0s 218ms/step
1/1 [==============================] - 0s 198ms/step
1/1 [==============================] - 0s 196ms/step
1/1 [==============================] - 0s 207ms/step
1/1 [==============================] - 0s 196ms/step
1/1 [==============================] - 0s 206ms/step
1/1 [==============================] - 0s 215ms/step
1/1 [==============================] - 0s 111ms/step
1/1 [==============================] - 0s 223ms/step
1/1 [==============================] - 0s 103ms/step
1/1 [==============================] - 0s 219ms/step
1/1 [==============================] - 0s 97ms/step
1/1 [==============================] - 0s 207ms/step
1/1 [==============================] - 0s 213ms/step
1/1 [==============================] - 0s 207ms/step
1/1 [==============================] - 0s 224ms/step
1/1 [==============================] - 0s 227ms/step
1/1 [==============================] - 0s 112ms/step
1/1 [==============================] - 0s 209ms/step
1/1 [==============================] - 0s 104ms/step
1/1 [==============================] - 0s 201ms/step
1/1 [==============================] - 0s 107ms/step
1/1 [==============================] - 0s 200ms/step
1/1 [==============================] - 0s 207ms/step
1/1 [==============================] - 0s 195ms/step
1/1 [==============================] - 0s 201ms/step
1/1 [==============================] - 0s 103ms/step
1/1 [==============================] - 0s 206ms/step
1/1 [==============================] - 0s 107ms/step
1/1 [==============================] - 0s 207ms/step
1/1 [==============================] - 0s 204ms/step
1/1 [==============================] - 0s 117ms/step
1/1 [==============================] - 0s 208ms/step
1/1 [==============================] - 0s 209ms/step
1/1 [==============================] - 0s 217ms/step
1/1 [==============================] - 0s 206ms/step
1/1 [==============================] - 0s 229ms/step
1/1 [==============================] - 0s 230ms/step
1/1 [==============================] - 0s 104ms/step
1/1 [==============================] - 0s 206ms/step
```

```
1/1 [==============================] - 0s 228ms/step
1/1 [==============================] - 0s 225ms/step
1/1 [==============================] - 0s 107ms/step
1/1 [==============================] - 0s 238ms/step
1/1 [==============================] - 0s 212ms/step
1/1 [==============================] - 0s 212ms/step
1/1 [==============================] - 0s 215ms/step
1/1 [==============================] - 0s 102ms/step
1/1 [==============================] - 0s 206ms/step
1/1 [==============================] - 0s 212ms/step
1/1 [==============================] - 0s 114ms/step
1/1 [==============================] - 0s 199ms/step
1/1 [==============================] - 0s 103ms/step
1/1 [==============================] - 0s 208ms/step
1/1 [==============================] - 0s 209ms/step
1/1 [==============================] - 0s 104ms/step
1/1 [==============================] - 0s 193ms/step
1/1 [==============================] - 0s 224ms/step
1/1 [==============================] - 0s 215ms/step
1/1 [==============================] - 0s 109ms/step
1/1 [==============================] - 0s 214ms/step
1/1 [==============================] - 0s 112ms/step
1/1 [==============================] - 0s 229ms/step
1/1 [==============================] - 0s 99ms/step
1/1 [==============================] - 0s 213ms/step
1/1 [==============================] - 0s 231ms/step
1/1 [==============================] - 0s 109ms/step
1/1 [==============================] - 0s 208ms/step
1/1 [==============================] - 0s 101ms/step
1/1 [==============================] - 0s 206ms/step
1/1 [==============================] - 0s 110ms/step
1/1 [==============================] - 0s 238ms/step
1/1 [==============================] - 0s 227ms/step
1/1 [==============================] - 0s 238ms/step
1/1 [==============================] - 0s 116ms/step
1/1 [==============================] - 0s 245ms/step
1/1 [==============================] - 0s 232ms/step
1/1 [==============================] - 0s 231ms/step
1/1 [==============================] - 0s 227ms/step
1/1 [==============================] - 0s 216ms/step
1/1 [==============================] - 0s 222ms/step
1/1 [==============================] - 0s 224ms/step
1/1 [==============================] - 0s 230ms/step
1/1 [==============================] - 0s 208ms/step
1/1 [==============================] - 0s 104ms/step
1/1 [==============================] - 0s 208ms/step
1/1 [==============================] - 0s 196ms/step
1/1 [==============================] - 0s 212ms/step
1/1 [==============================] - 0s 211ms/step
1/1 [==============================] - 0s 199ms/step
1/1 [==============================] - 0s 209ms/step
1/1 [==============================] - 0s 106ms/step
1/1 [==============================] - 0s 213ms/step
1/1 [==============================] - 0s 95ms/step
1/1 [==============================] - 0s 203ms/step
1/1 [==============================] - 0s 201ms/step
1/1 [==============================] - 0s 207ms/step
1/1 [==============================] - 0s 115ms/step
1/1 [==============================] - 0s 214ms/step
1/1 [==============================] - 0s 117ms/step
1/1 [==============================] - 0s 208ms/step
1/1 [==============================] - 0s 215ms/step
1/1 [==============================] - 0s 215ms/step
1/1 [==============================] - 0s 205ms/step
1/1 [==============================] - 0s 207ms/step
1/1 [==============================] - 0s 201ms/step
1/1 [==============================] - 0s 201ms/step
1/1 [==============================] - 0s 214ms/step
1/1 [==============================] - 0s 229ms/step
1/1 [==============================] - 0s 116ms/step
1/1 [==============================] - 0s 204ms/step
1/1 [==============================] - 0s 233ms/step
1/1 [==============================] - 0s 229ms/step
1/1 [==============================] - 0s 105ms/step
1/1 [==============================] - 0s 216ms/step
1/1 [==============================] - 0s 207ms/step
1/1 [==============================] - 0s 208ms/step
1/1 [==============================] - 0s 227ms/step
1/1 [==============================] - 0s 113ms/step
1/1 [==============================] - 0s 208ms/step
1/1 [==============================] - 0s 82ms/step
1/1 [==============================] - 0s 220ms/step
1/1 [==============================] - 0s 99ms/step
1/1 [==============================] - 0s 184ms/step
1/1 [==============================] - 0s 200ms/step
1/1 [==============================] - 0s 113ms/step
1/1 [==============================] - 0s 220ms/step
1/1 [==============================] - 0s 190ms/step
1/1 [==============================] - 0s 112ms/step
```

```
1/1 [==============================] - 0s 213ms/step
1/1 [==============================] - 0s 208ms/step
1/1 [==============================] - 0s 106ms/step
1/1 [==============================] - 0s 209ms/step
1/1 [==============================] - 0s 216ms/step
1/1 [==============================] - 0s 110ms/step
1/1 [==============================] - 0s 210ms/step
1/1 [==============================] - 0s 211ms/step
1/1 [==============================] - 0s 98ms/step
1/1 [==============================] - 0s 209ms/step
1/1 [==============================] - 0s 102ms/step
1/1 [==============================] - 0s 214ms/step
1/1 [==============================] - 0s 98ms/step
1/1 [==============================] - 0s 207ms/step
1/1 [==============================] - 0s 218ms/step
1/1 [==============================] - 0s 208ms/step
1/1 [==============================] - 0s 115ms/step
1/1 [==============================] - 0s 208ms/step
1/1 [==============================] - 0s 205ms/step
1/1 [==============================] - 0s 190ms/step
1/1 [==============================] - 0s 198ms/step
```

In [63]:
```
# creating a dataframe for the result
df_pred = pd.DataFrame({'image_path': image_id,'predicted_mask': mask,'has_mask': has_mask})
df_pred
```

Out[63]:

	image_path	predicted_mask	has_mask
0	TCGA_DU_5872_19950223/TCGA_DU_5872_19950223_67...	No mask	0
1	TCGA_FG_A4MT_20020212/TCGA_FG_A4MT_20020212_3.tif	No mask	0
2	TCGA_DU_A5TW_19980228/TCGA_DU_A5TW_19980228_29...	[[[6.579732e-06], [5.0200333e-07], [1.8209814...	1
3	TCGA_FG_7637_20000922/TCGA_FG_7637_20000922_17...	[[[1.3748944e-06], [5.6489725e-08], [3.797600...	1
4	TCGA_CS_6666_20011109/TCGA_CS_6666_20011109_25...	[[[4.1414055e-06], [1.4924562e-06], [7.196988...	1
...	...	...	...
585	TCGA_HT_A61A_20000127/TCGA_HT_A61A_20000127_23...	[[[3.941212e-06], [1.2909313e-06], [7.090385e...	1
586	TCGA_DU_5872_19950223/TCGA_DU_5872_19950223_69...	No mask	0
587	TCGA_DU_6404_19850629/TCGA_DU_6404_19850629_24...	No mask	0
588	TCGA_DU_8165_19970205/TCGA_DU_8165_19970205_7.tif	No mask	0
589	TCGA_DU_6404_19850629/TCGA_DU_6404_19850629_53...	No mask	0

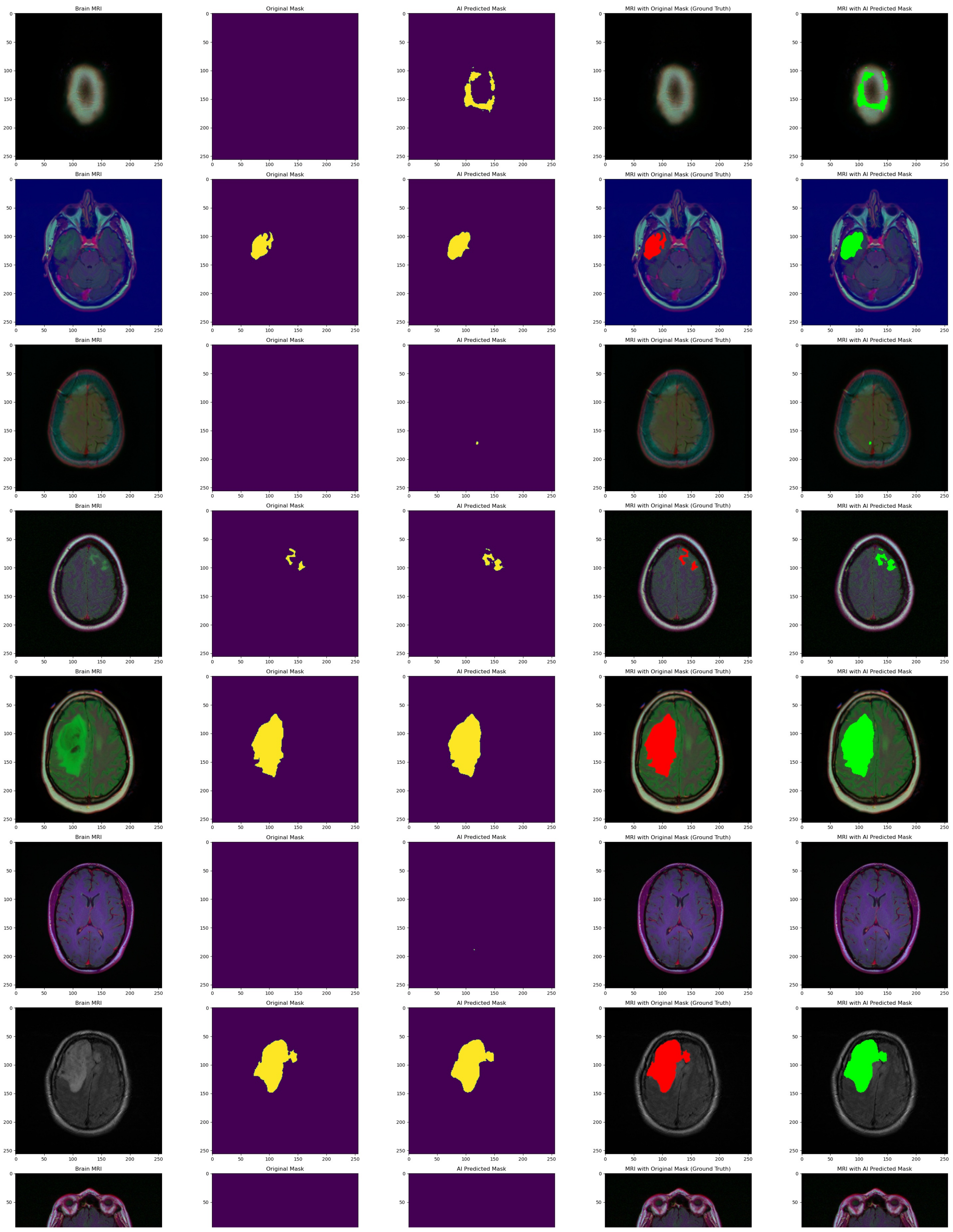
590 rows × 3 columns

In [64]:
```
# Merge the dataframe containing predicted results with the original test data.
df_pred = test.merge(df_pred, on = 'image_path')
df_pred.head()
```

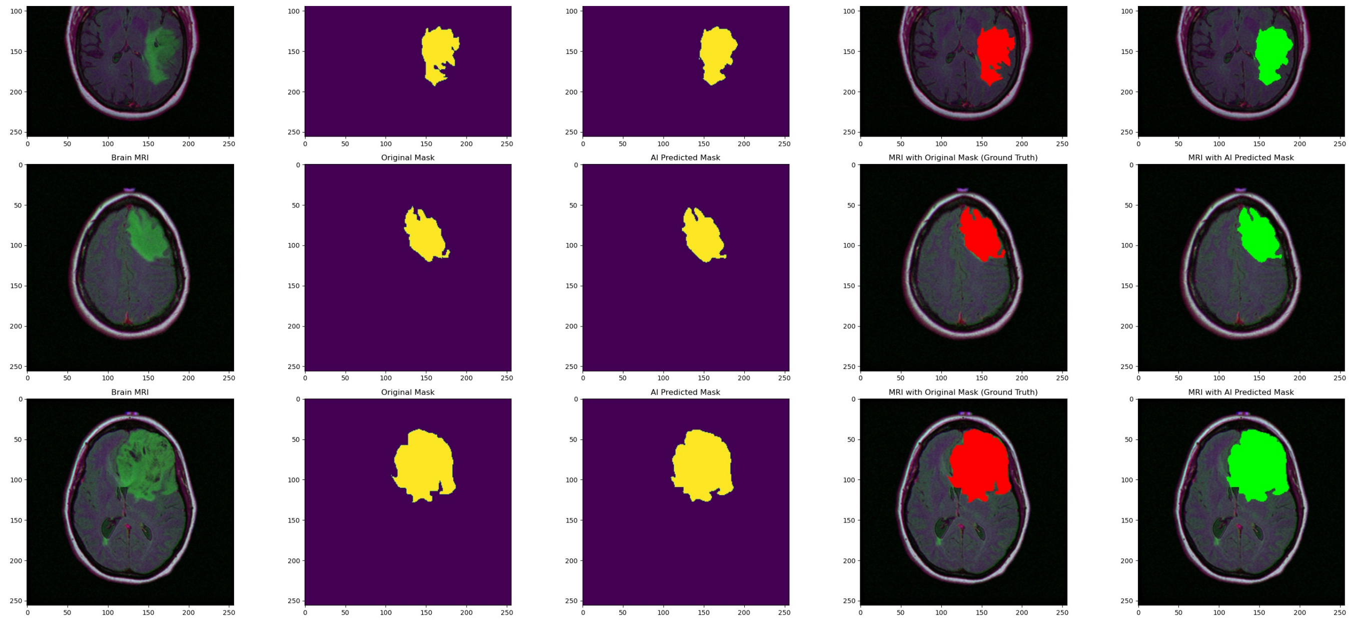
Out[64]:

	image_path	mask_path	mask	predicted_mask
0	TCGA_DU_5872_19950223/TCGA_DU_5872_19950223_67...	TCGA_DU_5872_19950223/TCGA_DU_5872_19950223_67...	0	No mask
1	TCGA_FG_A4MT_20020212/TCGA_FG_A4MT_20020212_3.tif	TCGA_FG_A4MT_20020212/TCGA_FG_A4MT_20020212_3_...	0	No mask
2	TCGA_DU_A5TW_19980228/TCGA_DU_A5TW_19980228_29...	TCGA_DU_A5TW_19980228/TCGA_DU_A5TW_19980228_29...	0	[[[6.579732e-06] [5.0200333e-07] [1.8209814..
3	TCGA_FG_7637_20000922/TCGA_FG_7637_20000922_17...	TCGA_FG_7637_20000922/TCGA_FG_7637_20000922_17...	1	[[[1.3748944e-06] [5.6489725e-08] [3.797600..
4	TCGA_CS_6666_20011109/TCGA_CS_6666_20011109_25...	TCGA_CS_6666_20011109/TCGA_CS_6666_20011109_25...	0	[[[4.1414055e-06] [1.4924562e-06] [7.196988..

In [65]:
```
count = 0
fig, axs = plt.subplots(10, 5, figsize=(30, 50))
for i in range(len(df_pred)):
  if df_pred['has_mask'][i] == 1 and count < 10:
    # read the images and convert them to RGB format
    img = io.imread(df_pred.image_path[i])
    img = cv2.cvtColor(img, cv2.COLOR_BGR2RGB)
    axs[count][0].title.set_text("Brain MRI")
    axs[count][0].imshow(img)

    # Obtain the mask for the image
    mask = io.imread(df_pred.mask_path[i])
    axs[count][1].title.set_text("Original Mask")
    axs[count][1].imshow(mask)
    # Obtain the predicted mask for the image
    predicted_mask = np.asarray(df_pred.predicted_mask[i])[0].squeeze().round()
    axs[count][2].title.set_text("AI Predicted Mask")
    axs[count][2].imshow(predicted_mask)

    # Apply the mask to the image 'mask==255'
    img[mask == 255] = (255, 0, 0)
    axs[count][3].title.set_text("MRI with Original Mask (Ground Truth)")
    axs[count][3].imshow(img)

    img_ = io.imread(df_pred.image_path[i])
    img_ = cv2.cvtColor(img_, cv2.COLOR_BGR2RGB)
    img_[predicted_mask == 1] = (0, 255, 0)
    axs[count][4].title.set_text("MRI with AI Predicted Mask")
    axs[count][4].imshow(img_)
    count += 1

fig.tight_layout()
```



Brain MRI
Original Mask
AI Predicted Mask
MRI with Original Mask (Ground Truth)
MRI with AI Predicted Mask


Brain MRI
Original Mask
AI Predicted Mask
MRI with Original Mask (Ground Truth)
MRI with AI Predicted Mask
Brain MRI
Original Mask
AI Predicted Mask
MRI with Original Mask (Ground Truth)
MRI with AI Predicted Mask

```
In [ ]:
```

Loading [MathJax]/jax/output/CommonHTML/fonts/TeX/fontdata.js