Deep Learning Project

on

Brain Tumour Classification

Submitted by

Ranjana Chakraverty

Registration No.: 12014011

Program Name: B.Tech. (CSE - Data Science (ML and AI))

Under the guidance of

Abhinaya A

School of Computer Science and Engineering

Lovely Professional University, Phagwara

(August-October, 2023)

DECLARATION

I hereby declare that I have completed my machine learning project from 24th August 2023 to 24th October 2023 under the guidance of Abhinaya A. I have worked with full dedication during these eight weeks of training and my learning outcomes fulfil the requirements of training for the award of degree of B.Tech. (CSE - Data Science (ML and AI)), Lovely Professional University, Phagwara.

Ranjana Chakraverty

Registration Number: 12014011

Date: 24th October 2023

ACKNOWLEDGEMENT

I would like to thank my professors at LPU of the School of Computer Science and Engineering for their support. I would like to thank the instructors at Upgrad, especially Abhinaya A and B N Bhargav, who were very helpful and always available to give clear my doubts.

TABLE OF CONTENTS

OBJECTIVE AND SCOPE

One of the most pressing issues of the present times is the ever-present danger of various types of cancers. Cancer is a chronic disease, and while it is treatable by a variety of methods, most medical authorities agree that most of the time, cancer isn't completely curable. It is, as a result, one of the most feared diseases of the present times.

Cancer can affect any part of the body, and this project deals with a prominent subclassification of cancer which is cancer of the brain. Cancer shows up in most parts of the body, in the form of tumours. Trained medical professional can look at brain scans and tell whether there is any cancer in the brain, which subtype it belongs to, which stage it is at and whether the tumour is benign or malignant. Until perhaps a decade ago, this was the norm. Only recently, has there been an effort by both the medical and technical domain to try to mechanize or automate some portions of healthcare to improve efficiency and reduce redundancy. Also, if things like detection are done with the help of a machine, the doctor can concentrate on other aspects like diagnosis and treatment. There might have been a concern at one time regarding whether machines can replicate human expertise and whether using machines can impact medical treatments negatively. Ironically, however, many machines, happen to have accuracy that parallels that of humans, some our even better than us. This is of course because machines to not suffer from human problems like fatigue and boredom.

Machine learning, which is a subset of computer science has been used extensively in the medical field with limited success. Many algorithms made errors and took a significant amount of human effort with unsatisfactory results. A subset of machine learning, known as deep learning, however, changed this perception completely. Deep learning aimed to mimic the functioning of the human brain in its endeavour to achieve human level intelligence and accuracy. A subset, known as convolutional neural networks has become particularly effective in detection and recognition of images. These were extrapolated for use in the medical field and were very successful. Today, many healthcare providers use machines superimposed with such deep learning constructs for use in disease detection. This project uses CNNs to develop a model which can detect the type of brain tumour in the human brain with high accuracy. Such a construct, might be easily found within a detection system already in use.

INTRODUCTION

Brain cancer is the result of cancerous cell growth in the brain. The cancer cells form tumours which can be of different types. Treatment for brain cancer focuses on removing the tumour and then destroying any remaining cancer cells.

Primary brain cancer, also known simply as brain cancer, is an overgrowth of cells in the brain that forms masses called brain tumours. This is different than cancer which starts in another part of the body and spreads to the brain which is called secondary or metastasized brain cancer.

A brain tumour is a growth of cells in the brain or near it. Brain tumours can happen in the brain tissue, brain nerves, the pituitary gland, the pineal gland, and the membranes that cover the surface of the brain.

Two different types of primary brain tumours exist. Some brain tumours are not cancerous. These are called noncancerous brain tumours or benign brain tumours. Other brain tumours are brain cancers, also called malignant brain tumours. Brain cancers may grow quickly. The cancer cells can invade and destroy the brain tissue. Malignant tumours can disrupt the way the body works. They can be life threatening and need to be treated as soon as they are detected.

Symptoms

- headaches that are usually worse in the morning
- nausea
- vomiting
- a lack of coordination
- a lack of balance
- difficulty walking
- memory lapses
- difficulty thinking
- speech problems
- vision problems
- personality changes

- abnormal eye movements
- muscle jerking
- muscle twitching
- unexplained passing out, or syncope
- drowsiness
- numbness or tingling in the arms or legs
- seizures

It is pertinent to remember that most of these symptoms are quite common, so it is unlikely that just facing one of them implies the disease. But if one experiences these symptoms for more than a week, if they have come on suddenly, if they're not relieved by over-the-counter pain medications, or if they cause any alarm, it's a good idea to get checked by a doctor.

Causes

The exact cause of primary brain cancer is unknown. But studies have shown a link between exposure to high doses of ionizing radiation and increased risk of brain cancer. Most common sources of ionizing radiation come from frequent medical imaging tests (CT scans, X-rays), radiation therapy treatments, and possible workplace exposure.

Other risk factors that might be related to developing brain cancer include:

- increased age
- a family history of brain cancer
- long-term smoking
- exposure to pesticides, herbicides, and fertilizer
- working with elements that can cause cancer, like lead, plastic, rubber, petroleum, and some textiles
- having an Epstein-Barr virus infection, or mononucleosis

Types

Common types of brain tumours include:

- Gliomas tumours
- Choroid plexus tumours
- Embryonal tumours
- Germ cell tumours
- Pineal tumours
- Meningiomas
- Nerve tumours
- Pituitary tumours

Diagnosis

- Neurological exam
- Computer tomography (CT) scan
- Magnetic Resonance Imaging (MRI)
- Positron emission tomography (PET) scan
- Collecting a sample of tissue and lab testing it

Stages

A brain tumour's stage is assigned when the tumour cells are tested in a lab. The stage shows how quickly the cells are growing and multiplying. The stage is based on how the cells look under a microscope. The grades range from 1 to 4.

A grade 1 brain tumour grows slowly. The cells are not very different from the healthy cells nearby. As the grade gets higher, the cells undergo changes so that they start to look very different. A grade 4 brain tumour grows very fast. The cells don't look anything like nearby healthy cells.

Treatment

Treatment for a brain tumor depends on whether the tumour is benign or malignant. Treatment options also depend on the type, size, stage and location of the brain tumour. Options might include surgery, radiation therapy, radiosurgery, chemotherapy, and targeted therapy.

PROFILE & ANALYSIS OF THE PROBLEM

For this project, the data was sourced from Kaggle and its link is provided below.

https://www.kaggle.com/datasets/sartajbhuvaji/brain-tumor-classification-mri

This dataset contains MRIs of various patients with suspected tumours. The data has already been split into training and test sets in two separate folders. Four classes of images are present in the dataset.

- Glioma tumours
- Meningioma tumour
- Pituitary tumour
- No tumour

Brain tumours are complex. There are a lot of variations in the sizes and location of the brain tumours. This makes it difficult to locate, recognize and identify a tumour. Also, a professional neurologist is required for MRI analysis. Often in developing countries, the lack of skilful doctors and lack of knowledge about tumours makes it challenging and time-consuming to generate reports from MRI. So, a CNN based system to take over this problem can contribute positively to the field.

Since deep learning constructs must be used, convolutional neural networks (CNNs), which are highly effective for analysing image data will be utilised in this project. The goal is to correctly classify images into one of the four classes and achieve high accuracy. In cases, such as brain tumours, CNNs can achieve accuracy even better than human abilities. With the construction of this model is the hope that in the future many domains within the medical profession shall start using CNNs for detection purposes.

METHOD

A detailed set of steps has been employed to accomplish this multiclass classification task.

- Importing all the relevant libraries, especially the ones required for computer vision through CNNs
- Data preparation and visualisation
- Performing one hot encoding for all the images, this is particularly useful for multiclass classification problems
- Using transfer learning, which makes use of a pretrained model. This is done to reduce training time as deep neural networks can take days to train
- Compiling the model
- Training the model
- Inspecting the training and validation accuracy & loss
- Evaluating the results with the help of various evaluation metrics used for classification
- I used Kaggle's coding facility to code for this deep learning project. This is because Kaggle allows training easily with GPUs, otherwise training take multiple hours. Since the GPU was used, the training fortunately didn't take long but still yielded acceptable results

<u>Specifics</u>

- The numpy function argmax was used for picking out the maximum probability values from among the different entries of the one hot vector, this is done for making predictions
- Various callbacks have been used in order to in one sense, automate the model learning perceptively.
    - ReduceLROnPlateau: reduce learning rate when a metric has stopped improving.
    - ModelCheckpoint: Callback to save the Keras model or model weights at some frequency.
    - TensorBoard: Enable visualizations for TensorBoard.
- Categorical crossentropy was the loss used and adam optimiser

RESULTS

- Accuracy: 98%

	precision	recall
0	0.98	0.96
1	0.96	1.00
2	0.98	0.98
3	1.00	1.00

Where

- 0 means glioma tumour
- 1 means no tumour
- 2 means meningioma tumour
- 3 means pituitary tumour

Full ipynb notebook available at the end of this file.

BIBLIOGRAPHY

healthline.com

mayoclinic.org

brain-tumor-classification

October 23, 2023

0.1 Importing libraries

[1]:
```
import pandas as pd
import numpy as np
import matplotlib.pyplot as plt
import seaborn as sns
from sklearn.utils import shuffle
from sklearn.model_selection import train_test_split
from tensorflow.keras.applications import EfficientNetB0
import cv2
import tensorflow as tf
from tqdm import tqdm
import os
from tensorflow.keras.callbacks import EarlyStopping, ReduceLROnPlateau,␣
 ↪TensorBoard, ModelCheckpoint
from sklearn.metrics import classification_report,confusion_matrix
from warnings import filterwarnings

for dirname, _, filenames in os.walk('/kaggle/input'):
    for filename in filenames:
        print(os.path.join(dirname, filename))
```

```
/kaggle/input/brain-tumor-classification-mri/Training/no_tumor/image(84).jpg
/kaggle/input/brain-tumor-classification-mri/Training/no_tumor/image(44).jpg
/kaggle/input/brain-tumor-classification-mri/Training/no_tumor/image(245).jpg
/kaggle/input/brain-tumor-classification-mri/Training/no_tumor/6.jpg
/kaggle/input/brain-tumor-classification-mri/Training/no_tumor/image(238).jpg
/kaggle/input/brain-tumor-classification-mri/Training/no_tumor/image(196).jpg
/kaggle/input/brain-tumor-classification-mri/Training/no_tumor/image(108).jpg
/kaggle/input/brain-tumor-classification-mri/Training/no_tumor/image(310).jpg
/kaggle/input/brain-tumor-classification-mri/Training/no_tumor/image (5).jpg
/kaggle/input/brain-tumor-classification-mri/Training/no_tumor/image(186).jpg
/kaggle/input/brain-tumor-classification-mri/Training/no_tumor/image(29).jpg
/kaggle/input/brain-tumor-classification-mri/Training/no_tumor/image(140).jpg
/kaggle/input/brain-tumor-classification-mri/Training/no_tumor/image(224).jpg
/kaggle/input/brain-tumor-classification-mri/Training/no_tumor/image (61).jpg
/kaggle/input/brain-tumor-classification-mri/Training/no_tumor/image(173).jpg
/kaggle/input/brain-tumor-classification-mri/Training/no_tumor/image(52).jpg
```

```
/kaggle/input/brain-tumor-classification-mri/Training/meningioma_tumor/m
(194).jpg
/kaggle/input/brain-tumor-classification-
mri/Training/meningioma_tumor/m1(171).jpg
/kaggle/input/brain-tumor-classification-mri/Training/meningioma_tumor/m2
(32).jpg
/kaggle/input/brain-tumor-classification-mri/Training/meningioma_tumor/m2
(64).jpg
/kaggle/input/brain-tumor-classification-mri/Training/meningioma_tumor/m2
(119).jpg
/kaggle/input/brain-tumor-classification-mri/Training/meningioma_tumor/m
(72).jpg
/kaggle/input/brain-tumor-classification-mri/Training/meningioma_tumor/m3
(94).jpg
/kaggle/input/brain-tumor-classification-mri/Training/meningioma_tumor/m3
(225).jpg
/kaggle/input/brain-tumor-classification-mri/Training/meningioma_tumor/m2
(101).jpg
/kaggle/input/brain-tumor-classification-mri/Training/meningioma_tumor/m2
(146).jpg
/kaggle/input/brain-tumor-classification-
mri/Training/meningioma_tumor/m1(66).jpg
/kaggle/input/brain-tumor-classification-
mri/Training/meningioma_tumor/m1(19).jpg
/kaggle/input/brain-tumor-classification-mri/Training/meningioma_tumor/m2
(9).jpg
/kaggle/input/brain-tumor-classification-mri/Training/meningioma_tumor/m
(63).jpg
/kaggle/input/brain-tumor-classification-mri/Training/meningioma_tumor/m2
(100).jpg
/kaggle/input/brain-tumor-classification-mri/Training/meningioma_tumor/m
(20).jpg
/kaggle/input/brain-tumor-classification-mri/Training/meningioma_tumor/m
(161).jpg
/kaggle/input/brain-tumor-classification-mri/Training/meningioma_tumor/m3
(164).jpg
/kaggle/input/brain-tumor-classification-mri/Training/meningioma_tumor/m3
(21).jpg
/kaggle/input/brain-tumor-classification-
mri/Training/meningioma_tumor/m1(143).jpg
/kaggle/input/brain-tumor-classification-mri/Training/meningioma_tumor/m
(167).jpg
/kaggle/input/brain-tumor-classification-mri/Training/meningioma_tumor/m
(28).jpg
/kaggle/input/brain-tumor-classification-mri/Training/meningioma_tumor/m3
(14).jpg
/kaggle/input/brain-tumor-classification-mri/Training/meningioma_tumor/m
(24).jpg
```

/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(66).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(73).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(82).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(11).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(85).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(10).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(28).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(80).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(20).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(22).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(100).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(17).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(37).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(18).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(39).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(38).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(16).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(97).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(30).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(34).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(12).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(3).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(59).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(47).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(77).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(78).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(63).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(76).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(61).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(19).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(45).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(31).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(87).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(60).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(54).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(57).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(46).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(92).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(24).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(70).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(72).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(55).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(26).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(23).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(2).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(43).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(56).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(49).jpg

```
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(62).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(99).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(67).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(50).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(83).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(1).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(48).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(51).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(36).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(91).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(65).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(86).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(41).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(64).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(7).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(42).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(98).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(95).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(93).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(71).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(8).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(40).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(89).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(58).jpg
/kaggle/input/brain-tumor-classification-mri/Testing/glioma_tumor/image(4).jpg
```

[2]:
```
labels = ['glioma_tumor','no_tumor','meningioma_tumor','pituitary_tumor']
```

The images are in simple folders. So, they first need to brought into a format which can be input into a neural network. To do this, I will add all the images to arrays and then preprocess them. Although the data is divided into training and testing sets, I will add it all to the same list, and then shuffle it well to increase accuracy.

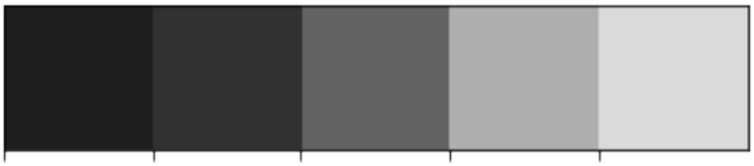
0.2 Importing images

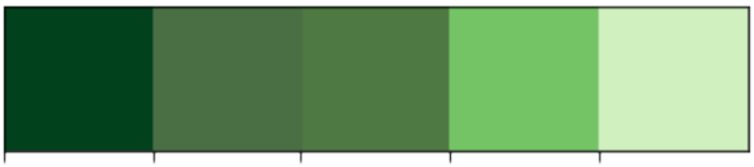
[3]:
```
X_train = []
y_train = []
image_size = 150
for i in labels:
    folderPath = os.path.join('../input/
 ↪brain-tumor-classification-mri','Training',i)
    for j in tqdm(os.listdir(folderPath)):
        img = cv2.imread(os.path.join(folderPath,j))
        img = cv2.resize(img,(image_size, image_size))
        X_train.append(img)
        y_train.append(i)
```

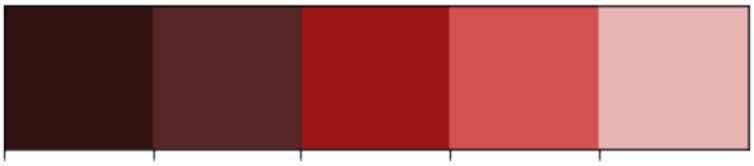
```
for i in labels:
    folderPath = os.path.join('../input/
 ↪brain-tumor-classification-mri','Testing',i)
    for j in tqdm(os.listdir(folderPath)):
        img = cv2.imread(os.path.join(folderPath,j))
        img = cv2.resize(img,(image_size,image_size))
        X_train.append(img)
        y_train.append(i)

X_train = np.array(X_train)
y_train = np.array(y_train)
```

```
100%|      | 826/826 [00:09<00:00, 85.60it/s]
100%|      | 395/395 [00:04<00:00, 97.76it/s]
100%|      | 822/822 [00:09<00:00, 88.65it/s]
100%|      | 827/827 [00:09<00:00, 87.09it/s]
100%|      | 100/100 [00:01<00:00, 97.01it/s]
100%|      | 105/105 [00:00<00:00, 123.44it/s]
100%|      | 115/115 [00:01<00:00, 101.50it/s]
100%|      | 74/74 [00:01<00:00, 67.88it/s]
```

[4]:
```
colors_dark = ["#1F1F1F", "#313131", '#636363', '#AEAEAE', '#DADADA']
colors_red = ["#331313", "#582626", '#9E1717', '#D35151', '#E9B4B4']
colors_green = ['#01411C','#4B6F44','#4F7942','#74C365','#D0F0C0']

sns.palplot(colors_dark)
sns.palplot(colors_green)
sns.palplot(colors_red)
```

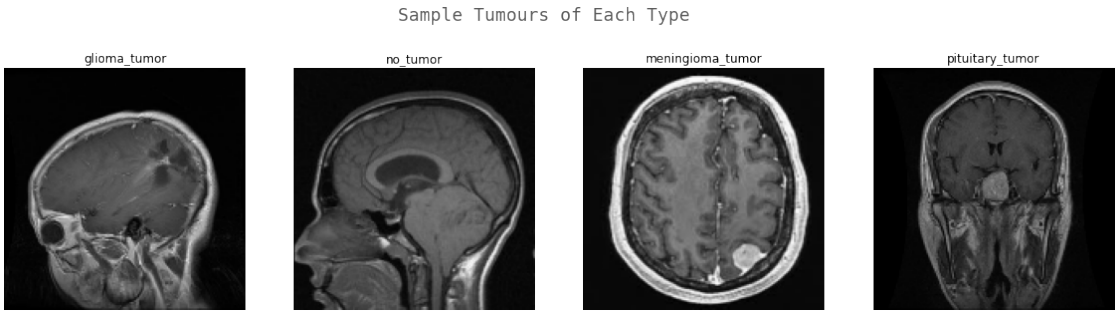






Visualising a few sample images is important for better understanding.

[5]:
```
k = 0
fig, ax = plt.subplots(1,4,figsize = (20,20))
fig.text(s = 'Sample Tumours of Each Type',size = 18,fontname='monospace',color
 ↪= colors_dark[1],y = 0.62,x = 0.4,alpha = 0.8)
for i in labels:
    j = 0
    while True :
        if y_train[j] == i:
            ax[k].imshow(X_train[j])
            ax[k].set_title(y_train[j])
            ax[k].axis('off')
            k += 1
            break
        j += 1
```



[6]:
```
X_train, y_train = shuffle(X_train, y_train, random_state = 101)
```

0.3 Train test split

[7]:
```
X_train, X_test, y_train, y_test = train_test_split(X_train,y_train, test_size
 ↪= 0.2, random_state = 101)
```

This is a multiclass classification problem and therefore one hot encoding is appropriate.

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

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

I am going to use a pretrained model to reduce training effort and time.

So, all the CNNs layers, except the output layer will be from another popular pretrained model used for multiclass classification.

I'll be using the EfficientNetB0 model which will use the weights from the ImageNet dataset. EfficientNetB0 is a convolutional neural network that is trained on more than a million images from the ImageNet database.

The include_top parameter is set to False so that the network doesn't include the output layer from the pre-built model. My model has its own customised output layer with one hot encoding.

0.4 CNN

[10]:
```
pretrained = EfficientNetB0(weights = 'imagenet',include_top =␣
↪False,input_shape = (image_size, image_size, 3))
```

```
Downloading data from https://storage.googleapis.com/keras-
applications/efficientnetb0_notop.h5
16711680/16705208 [==============================] - 1s 0us/step
```

[11]:
```
model = pretrained.output
model = tf.keras.layers.GlobalAveragePooling2D()(model)
model = tf.keras.layers.Dropout(rate = 0.5)(model)
model = tf.keras.layers.Dense(4, activation='softmax')(model)
model = tf.keras.models.Model(inputs = pretrained.input, outputs = model)
```

[12]:
```
model.summary()
```

```
Model: "model"
__________________________________________________________________________
__________________
Layer (type)                    Output Shape         Param #     Connected to
==========================================================================
==================
input_1 (InputLayer)            [(None, 150, 150, 3) 0
__________________________________________________________________________
```

```
------------------
rescaling (Rescaling)           (None, 150, 150, 3)  0           input_1[0][0]
------------------------------------------------------------------------------
------------------
normalization (Normalization)   (None, 150, 150, 3)  7           rescaling[0][0]
------------------------------------------------------------------------------
------------------
stem_conv_pad (ZeroPadding2D)   (None, 151, 151, 3)  0
normalization[0][0]
------------------------------------------------------------------------------
------------------
stem_conv (Conv2D)              (None, 75, 75, 32)   864
stem_conv_pad[0][0]
------------------------------------------------------------------------------
------------------
stem_bn (BatchNormalization)    (None, 75, 75, 32)   128         stem_conv[0][0]
------------------------------------------------------------------------------
------------------
stem_activation (Activation)    (None, 75, 75, 32)   0           stem_bn[0][0]
------------------------------------------------------------------------------
------------------
block1a_dwconv (DepthwiseConv2D (None, 75, 75, 32)   288
stem_activation[0][0]
------------------------------------------------------------------------------
------------------
block1a_bn (BatchNormalization) (None, 75, 75, 32)   128
block1a_dwconv[0][0]
------------------------------------------------------------------------------
------------------
block1a_activation (Activation) (None, 75, 75, 32)   0
block1a_bn[0][0]
------------------------------------------------------------------------------
------------------
block1a_se_squeeze (GlobalAvera (None, 32)           0
block1a_activation[0][0]
------------------------------------------------------------------------------
------------------
block1a_se_reshape (Reshape)    (None, 1, 1, 32)     0
block1a_se_squeeze[0][0]
------------------------------------------------------------------------------
------------------
block1a_se_reduce (Conv2D)      (None, 1, 1, 8)      264
block1a_se_reshape[0][0]
------------------------------------------------------------------------------
------------------
block1a_se_expand (Conv2D)      (None, 1, 1, 32)     288
block1a_se_reduce[0][0]
------------------------------------------------------------------------------
```

```
------------------
block7a_dwconv (DepthwiseConv2D (None, 5, 5, 1152)   10368
block7a_expand_activation[0][0]
--------------------------------------------------------------------------------
------------------
block7a_bn (BatchNormalization) (None, 5, 5, 1152)   4608
block7a_dwconv[0][0]
--------------------------------------------------------------------------------
------------------
block7a_activation (Activation) (None, 5, 5, 1152)   0
block7a_bn[0][0]
--------------------------------------------------------------------------------
------------------
block7a_se_squeeze (GlobalAvera (None, 1152)         0
block7a_activation[0][0]
--------------------------------------------------------------------------------
------------------
block7a_se_reshape (Reshape)    (None, 1, 1, 1152)   0
block7a_se_squeeze[0][0]
--------------------------------------------------------------------------------
------------------
block7a_se_reduce (Conv2D)      (None, 1, 1, 48)     55344
block7a_se_reshape[0][0]
--------------------------------------------------------------------------------
------------------
block7a_se_expand (Conv2D)      (None, 1, 1, 1152)   56448
block7a_se_reduce[0][0]
--------------------------------------------------------------------------------
------------------
block7a_se_excite (Multiply)    (None, 5, 5, 1152)   0
block7a_activation[0][0]
block7a_se_expand[0][0]
--------------------------------------------------------------------------------
------------------
block7a_project_conv (Conv2D)   (None, 5, 5, 320)    368640
block7a_se_excite[0][0]
--------------------------------------------------------------------------------
------------------
block7a_project_bn (BatchNormal (None, 5, 5, 320)    1280
block7a_project_conv[0][0]
--------------------------------------------------------------------------------
------------------
top_conv (Conv2D)               (None, 5, 5, 1280)   409600
block7a_project_bn[0][0]
--------------------------------------------------------------------------------
------------------
top_bn (BatchNormalization)     (None, 5, 5, 1280)   5120        top_conv[0][0]
--------------------------------------------------------------------------------
```

```
------------------
top_activation (Activation)     (None, 5, 5, 1280)   0           top_bn[0][0]
------------------------------------------------------------------------------
------------------
global_average_pooling2d (Globa (None, 1280)         0
top_activation[0][0]
------------------------------------------------------------------------------
------------------
dropout (Dropout)               (None, 1280)         0
global_average_pooling2d[0][0]
------------------------------------------------------------------------------
------------------
dense (Dense)                   (None, 4)            5124        dropout[0][0]
==============================================================================
==================
Total params: 4,054,695
Trainable params: 4,012,672
Non-trainable params: 42,023
------------------------------------------------------------------------------
------------------
```

[13]:
```
model.compile(loss = 'categorical_crossentropy',optimizer = 'adam', metrics =␣
↪['accuracy'])
```

[14]:
```
tensorboard = TensorBoard(log_dir = 'logs')
checkpoint = ModelCheckpoint("effnet.h5", monitor =␣
↪"val_accuracy",save_best_only = True, mode = "auto", verbose = 1)
reduce_lr = ReduceLROnPlateau(monitor = 'val_accuracy', factor = 0.3, patience␣
↪= 2, min_delta = 0.001, mode = 'auto',verbose = 1)
```

0.5 Model training

[15]:
```
history = model.fit(X_train, y_train, validation_split = 0.1, epochs = 12,␣
↪verbose = 1, batch_size = 32, callbacks=[tensorboard,checkpoint,reduce_lr])
```

```
Epoch 1/12
74/74 [==============================] - 24s 153ms/step - loss: 0.6764 -
accuracy: 0.7359 - val_loss: 0.6981 - val_accuracy: 0.7863

Epoch 00001: val_accuracy improved from -inf to 0.78626, saving model to
effnet.h5
Epoch 2/12
74/74 [==============================] - 8s 112ms/step - loss: 0.1792 -
accuracy: 0.9367 - val_loss: 0.6355 - val_accuracy: 0.8244

Epoch 00002: val_accuracy improved from 0.78626 to 0.82443, saving model to
effnet.h5
```

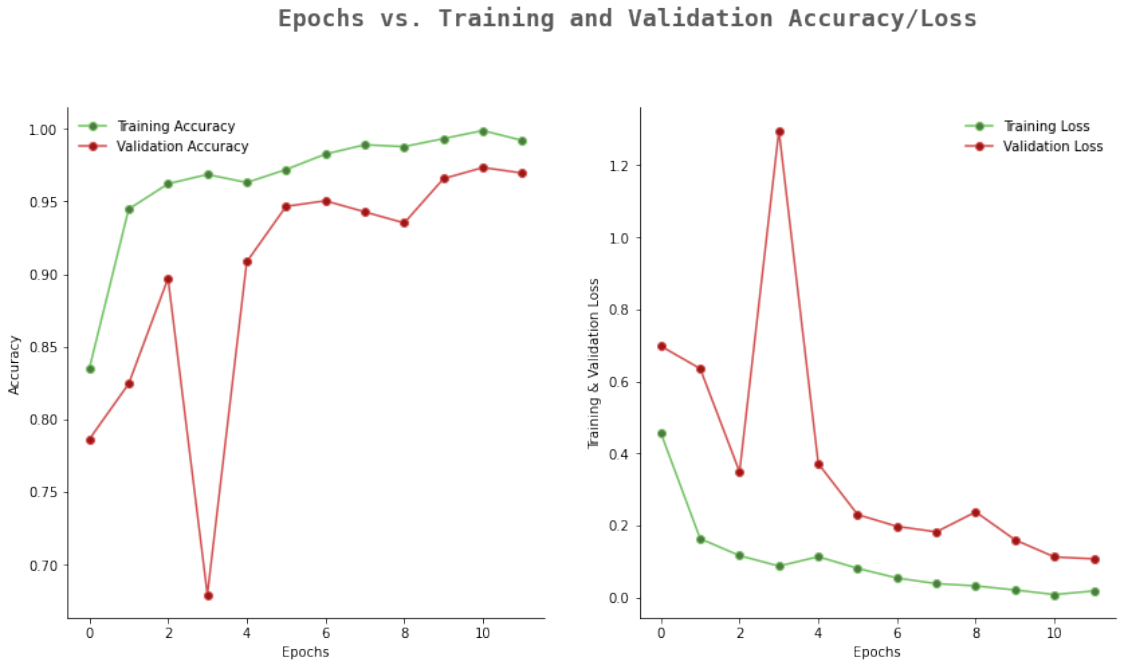
```
Epoch 3/12
74/74 [==============================] - 8s 112ms/step - loss: 0.1096 -
accuracy: 0.9623 - val_loss: 0.3490 - val_accuracy: 0.8969

Epoch 00003: val_accuracy improved from 0.82443 to 0.89695, saving model to
effnet.h5
Epoch 4/12
74/74 [==============================] - 8s 111ms/step - loss: 0.0879 -
accuracy: 0.9693 - val_loss: 1.2965 - val_accuracy: 0.6794

Epoch 00004: val_accuracy did not improve from 0.89695
Epoch 5/12
74/74 [==============================] - 8s 112ms/step - loss: 0.0872 -
accuracy: 0.9709 - val_loss: 0.3718 - val_accuracy: 0.9084

Epoch 00005: val_accuracy improved from 0.89695 to 0.90840, saving model to
effnet.h5
Epoch 6/12
74/74 [==============================] - 8s 112ms/step - loss: 0.0861 -
accuracy: 0.9692 - val_loss: 0.2301 - val_accuracy: 0.9466

Epoch 00006: val_accuracy improved from 0.90840 to 0.94656, saving model to
effnet.h5
Epoch 7/12
74/74 [==============================] - 8s 111ms/step - loss: 0.0633 -
accuracy: 0.9820 - val_loss: 0.1974 - val_accuracy: 0.9504

Epoch 00007: val_accuracy improved from 0.94656 to 0.95038, saving model to
effnet.h5
Epoch 8/12
74/74 [==============================] - 8s 112ms/step - loss: 0.0431 -
accuracy: 0.9890 - val_loss: 0.1822 - val_accuracy: 0.9427

Epoch 00008: val_accuracy did not improve from 0.95038
Epoch 9/12
74/74 [==============================] - 8s 111ms/step - loss: 0.0248 -
accuracy: 0.9917 - val_loss: 0.2376 - val_accuracy: 0.9351

Epoch 00009: val_accuracy did not improve from 0.95038

Epoch 00009: ReduceLROnPlateau reducing learning rate to 0.0003000000142492354.
Epoch 10/12
74/74 [==============================] - 8s 112ms/step - loss: 0.0238 -
accuracy: 0.9918 - val_loss: 0.1600 - val_accuracy: 0.9656

Epoch 00010: val_accuracy improved from 0.95038 to 0.96565, saving model to
effnet.h5
Epoch 11/12
```

```
74/74 [==============================] - 8s 112ms/step - loss: 0.0090 -
accuracy: 0.9992 - val_loss: 0.1125 - val_accuracy: 0.9733

Epoch 00011: val_accuracy improved from 0.96565 to 0.97328, saving model to
effnet.h5
Epoch 12/12
74/74 [==============================] - 8s 111ms/step - loss: 0.0277 -
accuracy: 0.9881 - val_loss: 0.1073 - val_accuracy: 0.9695

Epoch 00012: val_accuracy did not improve from 0.97328
```

0.6 Evaluating model performance

[16]:
```
filterwarnings('ignore')

epochs = [i for i in range(12)]
fig, ax = plt.subplots(1,2,figsize=(14,7))
train_acc = history.history['accuracy']
train_loss = history.history['loss']
val_acc = history.history['val_accuracy']
val_loss = history.history['val_loss']

fig.text(s='Epochs vs. Training and Validation Accuracy/
 ↪Loss',size=18,fontweight='bold',
             fontname='monospace',color=colors_dark[1],y=1,x=0.28,alpha=0.8)

sns.despine()
ax[0].plot(epochs, train_acc,␣
 ↪marker='o',markerfacecolor=colors_green[2],color=colors_green[3],
           label = 'Training Accuracy')
ax[0].plot(epochs, val_acc,␣
 ↪marker='o',markerfacecolor=colors_red[2],color=colors_red[3],
           label = 'Validation Accuracy')
ax[0].legend(frameon=False)
ax[0].set_xlabel('Epochs')
ax[0].set_ylabel('Accuracy')

sns.despine()
ax[1].plot(epochs, train_loss,␣
 ↪marker='o',markerfacecolor=colors_green[2],color=colors_green[3],
           label ='Training Loss')
ax[1].plot(epochs, val_loss,␣
 ↪marker='o',markerfacecolor=colors_red[2],color=colors_red[3],
           label = 'Validation Loss')
ax[1].legend(frameon=False)
ax[1].set_xlabel('Epochs')
ax[1].set_ylabel('Training & Validation Loss')
```

```
fig.show()
```



[17]:
```
pred = model.predict(X_test)
pred = np.argmax(pred, axis = 1)
y_test_new = np.argmax(y_test,axis = 1)
```

0.7 Evaluation metrics

[18]:
```
print(classification_report(y_test_new, pred))
```

```
              precision    recall  f1-score   support

           0       0.98      0.96      0.97       168
           1       0.98      0.98      0.98       108
           2       0.97      0.98      0.97       201
           3       0.98      0.99      0.99       176

    accuracy                           0.98       653
   macro avg       0.98      0.98      0.98       653
weighted avg       0.98      0.98      0.98       653
```

[20]:
```
fig,ax=plt.subplots(1,1,figsize=(14,7))
```

```
sns.
  ↪heatmap(confusion_matrix(y_test_new,pred),ax=ax,xticklabels=labels,yticklabels=labels,annot
           cmap=colors_green[::-1],alpha=0.
  ↪7,linewidths=2,linecolor=colors_dark[3])
fig.text(s='Heatmap of the Confusion Matrix',size=18,fontweight='bold',
             fontname='monospace',color=colors_dark[1],y=0.92,x=0.28,alpha=0.8)

plt.show()
```