BRAIN TUMOR DETECTION USING CONVOLUTIONAL NEURAL NETWORKS

Bhumika Sharma
Email: bhumikas850@gmail.com
GitHub Link: https://github.com/bhumikasharma29/Brain_Tumor_Detection

Abstract

Brain tumors are the most common and aggressive disease, leading to a very short life expectancy in their highest grade. Thus, treatment planning is a key stage to improve the quality of life of patients. MRI images are especially used to diagnose tumors in the brain. However the huge amount of data generated by MRI scan prevents manual classification of tumor vs non-tumor in a particular time Hence trusted and automatic classification schemes are essential to prevent the death rate of humans. The automatic brain tumor classification is a very challenging task in large spatial and structural variability of the surrounding region of brain tumor. Neural Networks (NN) and Support Vector Machines are the most used methods to extract information about brain tumours from MRI. In this project convolutional neural networks(CNN) are used for performing the above task.

Keywords: Convolutional Neural Networks, Brain, MRI

1 Domain Related Background

A brain tumor is a mass or growth of abnormal cells in your brain.Many different types of brain tumors exist. Some brain tumors are noncancerous (benign), and some brain tumors are cancerous (malignant). Brain tumors can begin in your brain (primary brain tumors), or cancer can begin in other parts of your body and spread to your brain (secondary, or metastatic, brain tumors). How quickly a brain tumor grows can vary greatly. The growth rate as well as location of a brain tumor determines how it will affect the function of your nervous system. Brain tumor treatment options depend on the type of brain tumor you have, as well as its size and location.

2 METHODOLOGY

2.1 THE DATASET:

A brain MRI images data set founded on Kaggle. You can find it here. The data set contains 2 folders: yes and no which contains 253 Brain MRI Images. The folder yes contains 155 Brain MRI Images that are tumorous (malignant) and the folder no contains 98 Brain MRI Images that are non-tumorous (benign). Meaning that 61% (155 images) of the data are positive examples and 39% (98 images) are negative.

2.2 DATA PREPARATION:

2.2.1 Data augmentation:

As limited data was available, we have used data augmentation, a strategy that enables practitioners to significantly increase the diversity of data available for training models, without collecting new data. Data augmentation techniques such as cropping, padding, and horizontal flipping are used and implemented using the ImageDataGenerator provided by Keras.
Before data Augmentation: It consists of 253 brain MRI image, 155 were brain tumour positive and 98 were negative.
After data augmentation: It consists of 2064 brain MRI image, 1084 were brain tumour positive and 980 were negative. I have augmented in such a way to balance the dataset

2.2.2 Data Pre-processing:

For every image, the following pre-processing steps were applied:
1. Crop the part of the image that contains only the brain (which is the most important part of the image):
The cropping technique is used to find the extreme top, bottom, left and right points of the brain using OpenCV.
2. Resize the image to have a shape of (240,240,3) because images in the dataset come in different sizes. So, all images should have the same shape to feed it as an input to the neural network.
3. Apply normalization:
to scale pixel values to the range 0 to 1

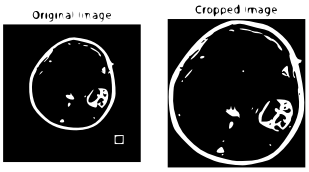
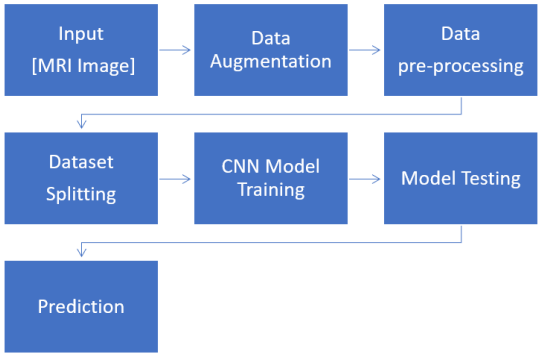


Figure 1: Cropped Image

2.2.3 Data Split:

The data was split in the following way:
70% of the data for training
15% of the data for validation (development)
15% of the data for testing

2.3 CODE FLOW BLOCK DIAGRAM



2.4 THE CNN MODEL

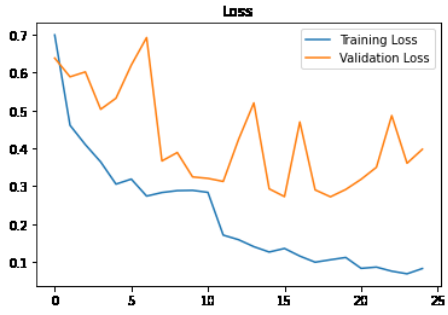
A Convolutional Neural Network (CNN) model is found to be the best suited approach for the problem statement. It is comprised of one or more convolution layers (often with a sub-sampling step) and then followed by one or more fully connected layers as in a standard multi-layer neural network. The main goal of the convolutional base is to generate features from the image. The architecture of a CNN is designed to take advantage of the 2D structure of an input image.

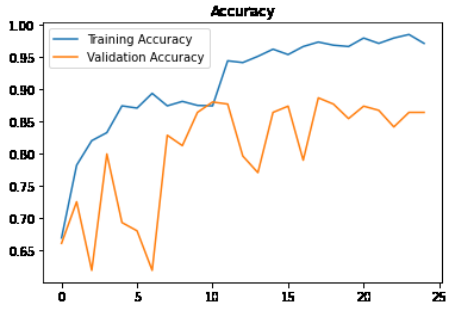
Understanding the architecture: Each input x (image) has a shape of (240, 240, 3) and is fed into the neural network. And, it goes through the following layers:
1. A Zero Padding layer
2. A convolutional layer with 32 filters, with a filter size of (7, 7) and a stride equal to 1.
3. A batch normalization layer to normalize pixel values to speed up computation.
4. A ReLU (Rectified Linear Unit) activation layer.
5. A Max Pooling layer.
6. A Max Pooling layer with f=4 and s=4, same as before.
7. A Flatten layer in order to flatten the 3-dimensional matrix into a one-dimensional vector.
8. A Dense (output unit) fully connected layer with one neuron with a sigmoid activation (since this is a binary classification task).

```
Model: "BrainDetectionModel"
_________________________________________________________________
Layer (type)                 Output Shape              Param #
=================================================================
input_1 (InputLayer)         [(None, 240, 240, 3)]     0
_________________________________________________________________
zero_padding2d (ZeroPadding2 (None, 244, 244, 3)       0
_________________________________________________________________
conv0 (Conv2D)               (None, 238, 238, 32)      4736
_________________________________________________________________
bn0 (BatchNormalization)     (None, 238, 238, 32)      128
_________________________________________________________________
activation (Activation)      (None, 238, 238, 32)      0
_________________________________________________________________
max_pool0 (MaxPooling2D)     (None, 59, 59, 32)        0
_________________________________________________________________
max_pool1 (MaxPooling2D)     (None, 14, 14, 32)        0
_________________________________________________________________
flatten (Flatten)            (None, 6272)              0
_________________________________________________________________
fc (Dense)                   (None, 1)                 6273
=================================================================
Total params: 11,137
Trainable params: 11,073
Non-trainable params: 64
_________________________________________________________________
```

2.5 Training the Model

The model was trained for 25 epochs and these are the loss accuracy plots:





3 Results

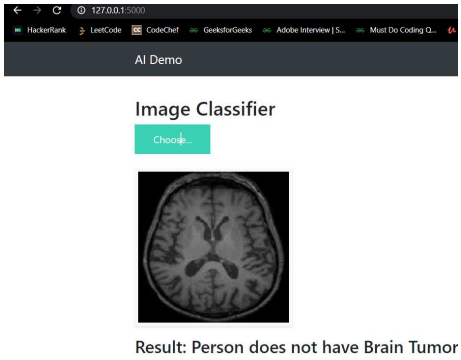
Now, the best model (the one with the best validation accuracy) detects brain tumor with:
93.5% accuracy on the test set.
0.93 F1 score on the test set.

Table 1: **Performance Table**

Measures	Validation Set	Test Set
$Accuracy$	94%	93.8%
$F1Score$	0.94	0.93

4 WEB APP

Integrating the deep learning model with a web app to run on local host using Flask.



References

[1] https://www.pyimagesearch.com/2016/04/11/finding-extreme-points-in-contours-with-opencv/

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

[3] https://medium.com/@mohamedalihabib7/brain-tumor-detection-using-convolutional-neural-networks-30ccef6612b0

[4] https://bair.berkeley.edu/blog/2019/06/07/data_aug/