BRAIN TUMOR DETECTION USING DEEP LEARNING

Systems & Biomedical Engineering Department
Special Functions & Partial Differential Equations (MTH2245)

πrates

Dr. Samah El-Tantawy
Faculty of Engineering - Cairo University

Contents

Table of Figures

Abstract

Convolutional Neural Network is a class of artificial neural networks, analyze visual imagery. The backpropagation of the model will use a Partial Differential Equation to calculate the gradient of the loss function with respect to parameters. Calculating the gradient helps us to optimize and update the neural network weight and achieve the best weights that will affect our model accuracy. This finally helps us in predicting and detecting the brain tumor using Deep Learning.

Introduction

Brain tumors are diseases caused by the growth of abnormal cells in the brain. There are two main categories of brain tumors: Benign brain tumors and Malignant brain tumors. Magnetic Imaging Resonance (MRI) can detect these tumors effectively.

Because brain tumors are rare and of different types, it is difficult to predict the survival rate of patients who are prone to develop tumors. About 15 out of 100 people with brain tumors can survive for more than ten years after being diagnosed. Treatment of brain tumors depends on several factors, including the type of tumor, the degree of cell abnormalities, and where it is in the brain.

Recently, brain tumor diagnosis has been made using machine learning, specifically Deep Learning (DL) models and algorithms, these models and algorithms make brain tumor prediction and detection very quickly and with greater accuracy that helps medical staff make quick decisions which can lead to an increase in the rate of recovery from these tumors. [1]

Artificial Neural Network (ANN) and Convolution Neural Network (CNN) are used to make this real. They are inspired by the idea of the biological neural networks to act like the human brain, they have different connected layers of neurons. It depends on giving the model training data, and it detects the features, then applies the concept of "try and catch" to see the accuracy of the model and then makes a backpropagation to improve the accuracy by PDE. [2]

Backpropagation is a technique for swiftly calculating derivatives, and it is a learning algorithm used by ANN to compute a gradient descent for weights. The weights are updated backward, from output to input, which gives the algorithm its name. Backpropagation is based on the partial derivative $\partial C/\partial w$ of the cost function 'C' (used for calculating how much wrong the model was in its prediction) for any weight 'w' (bias 'b') in the network. When we update the weights and biases, the expression shows us how rapidly the cost changes. [3]

Literature review

Steps of Brain Tumor Detection Process:

- In the last few years, there has been an increase in the use of DL Methods, especially CNN.
- Rather than using hand-crafted features, DL models learn
  - Complex
  - Task Adaptive
  - High Level Features directly from the Data.
- Due to DL benefits, it's models are used for
  - Classification
  - Segmentation
  - Brain Tumor Detection.
- The common DL models used is in detection of brain tumors are
  - Convolutional Neural Network (CNN), which we will use.
  - Stacked Denoising Auto-Encoder (SDAE)
  - Recurrent Neural Network. (RNN)
- The next figure show the standard steps which lead to brain tumor detection. (Figure 1) [4]

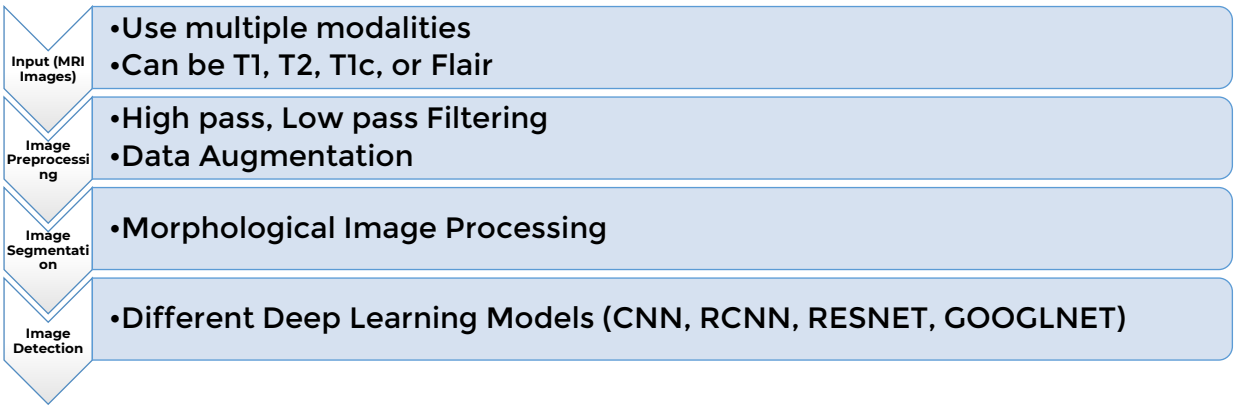


Figure 1 - The standard steps which lead to bain tumor detection

Contributors Work (Figure 2)

Peter D. Chang

- Proposed a fully convolutional neural network with hyper-local features for brain tumor segmentation. The network was composed of only 130,400 parameters and could complete segmentation for an entire brain volume in less than one second.

Konstantinos Kamnitsas

- Employed DeepMedic, a 3D CNN architecture for lesion segmentation, extended with residual connections.

Tseng Kuan Lun

- Presented a fully-automatic segmentation method by utilizing a CNN.

Balaji Pandian

- Proposed a fully automated approach based on a 3D CNN with subvolumetraining procedures for brain tumor segmentation in multi-modal MRIs.

Adria Casamitjana

- Proposed two fully convolutional 3D CNN architectures which are a variant of the two-pathway DeepMedic net.

Bi Song

- Proposed anatomy-guided brain tumor segmentation and classification to delineate tumor structures into the active tumorous core, necrosis, and edema.

Abdelrahman Ellwaa

- Introduced an iterative random forest approach which tried to improve its accuracy by iteratively choosing the best patients.

Figure 2 - Contributors work during last decades

Mathematical Modelling

Deep Learning model layers:

Our Model consists of conv2D with padding, max-pooling and FNN network which consist of forward and backward propagations. [5]

1. CNN

Convolutional neural network (CNN) is a class of artificial neural network, most applied to analyze visual imagery. The shared-weight architecture of the convolution are kernels or filters that slide along input features and provide translation equivariant responses known as feature maps. A kernel is a grid of weights "overlaid" on image, centered on one pixel. Each weight multiplied with pixel underneath it.

- Output = $\sum p\ Wp * (Pixel)p$

Padding technique: Using Kernels directly, there will be an "edge effect" because of the output matrix dimensions will be:

- Width = Old width – filter width +1
- Heigh = Old heigh – filter heigh +1

Padding adds extra pixels of value zero around the frame, so every pixel of the original image will be a center pixel as the kernel moves across the image.

Example without padding:

1	2	0	3	1
1	0	0	2	2
2	1	2	1	1
0	0	1	0	0
1	2	1	1	1

Input

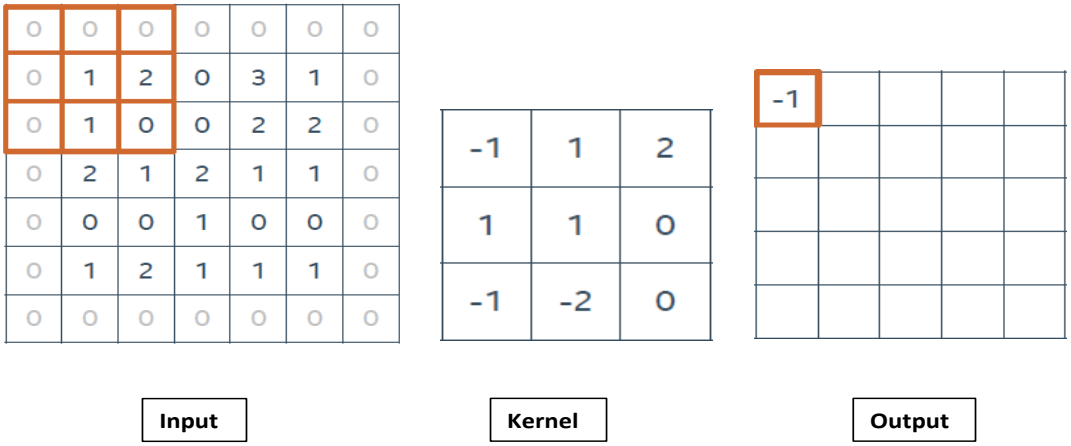
-1	1	2
1	1	0
-1	-2	0

kernel

-2		

kernel

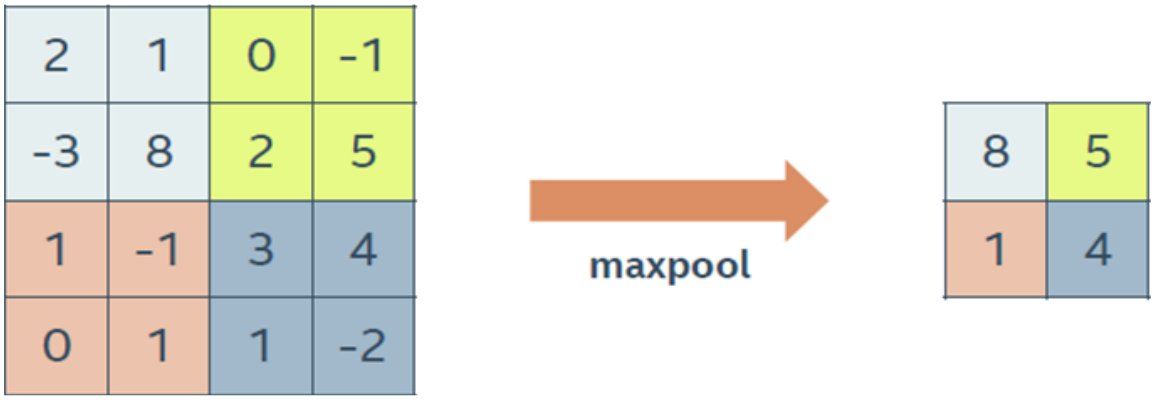
Example with padding:



2. Pooling

Pooling is reducing the image size by mapping a patch of pixels to a single value. There are types of pooling operations such as max pooling and avg pooling. We will use max pooling.

Ex. 2x2 max pooling.



3. FNN

3.1 Forward Propagation:

Forward propagation refers to calculate the intermediate variables and output for neural network from input layer to output layer, we now will assume that we have one-hidden layer for simplicity,

Let us assume that $x \in \mathbb{R}d$, so the intermediate variable is:

$$a^{(1)} = x$$

$$z^{(2)} = W^{(1)}x + b$$

Where $W^{(1)} \in \mathbb{R}^{h*d}$ is the weighted parameter of the hidden layer, and h is the number of neurons in each layer, after calculating the intermediate variable z, we will get the activation vector a using the activation function $ReLU$,

$$a^{(2)} = ReLU\left(z^{(2)}\right)$$

$$z^{(3)} = W^{(2)}a^{(2)} + b$$

$$\hat{y} = \sigma\left(z^{(3)}\right)$$

Now, we will assume that the loss function is l and the label of the example is y, so we can calculate the loss of this single data example,

$$L = l(\hat{y}y)$$

And the regulation term s can be:

$$s = \frac{\lambda}{2}\left(\left\|W^{(1)}\right\|^2_F + \left\|W^{(2)}\right\|^2_F\right)$$

$$J = L + s$$

So, in this way we can calculate the output of the neural network and the cost function of it.

For the first time we will put weighted matrix W randomly, and calculate the output and loss function, but we need to update these values to get better performance, using the label of the data, this process is called Backward Propagation.

3.2 Backward Propagation:

Backpropagation is the inverse of forward propagation, it works from output to input, trying to update the weighted matrix, using chain rule on it, so if we have y = $f(x)$ and z = $g(y)$ so we can get $\partial z/\partial x$ equals:

$$\frac{\partial z}{\partial x} = \frac{\partial z}{\partial y}\frac{\partial y}{\partial x}$$

So, from forward propagation equations, we need to update $W(1)$ and $W(2)$, and to do that we need to get $\partial J/\partial W(1)$ and $\partial J/\partial W(2)$,

$$\frac{\partial J}{\partial y} = \frac{\partial J}{\partial L}\frac{\partial L}{\partial y}$$

$$J = L + s, \therefore \frac{\partial J}{\partial L} = 1 \; and \; \frac{\partial J}{\partial s} = 1$$

$$\frac{\partial J}{\partial y} = \frac{\partial L}{\partial y}$$

Next, we need to calculate the gradient of the regularization term to both weighted parameters

$$\frac{\partial s}{\partial W^{(1)}} = \lambda W^{(1)}, \qquad \frac{\partial s}{\partial W^{(2)}} = \lambda W^{(2)}$$

Now, we can calculate the gradient $\partial J/\partial W^{(2)}$, using chain rule,

$$\frac{\partial J}{\partial W^{(2)}} = \frac{\partial J}{\partial y}\frac{\partial y}{\partial W^{(2)}} + \frac{\partial y}{\partial x}\frac{\partial y}{\partial x} = \frac{\partial J}{\partial y}a^{(2)} + \lambda W^{(2)}$$

calculating $W(1)$, by continuing backpropagation along the output layer to hidden layer,

$$\frac{\partial J}{\partial a^{(2)}} = \frac{\partial J}{\partial y}\frac{\partial y}{\partial a^{(2)}} = \frac{\partial J}{\partial y} * W^{(2)}$$

Assuming that the activation function is $\emptyset$, so we can calculate $\partial J/\partial z^{(2)}$

$$\frac{\partial J}{\partial W^{(1)}} = \frac{\partial J}{\partial z^{(2)}}\frac{\partial z^{(2)}}{\partial W^{(1)}} + \frac{\partial J}{\partial s}\frac{\partial s}{\partial W^{(1)}} = \frac{\partial J}{\partial z^{(2)}}a^{(1)} + \lambda W^{(1)}$$

And we can continue in this way to calculate all values in this way, in our application we will use more than 1 hidden layer, but we will use activation function for all hidden layers as ReLU and for output layer we will use Sigmoid function to get an output as 0 or 1.

So, we need to calculate $\sigma'(x)$:

$$\sigma(x) = \frac{1}{1+e^{-x}}$$

$$\grave{\sigma}(x) = \frac{0-(-e^{-x})}{(1+e^{-x})^2} = \frac{-e^{-x}}{(1+e^{-x})^2} = \frac{1}{1+e^{-x}} - \frac{1}{(1+e^{-x})^2} = \frac{1}{1+e^{-x}}\left(1 - \frac{1}{1+e^{-x}}\right)$$

And for ReLU(x):

$$ReLU(x) = \begin{cases} 0, x < 0 \\ x, x \geq 0 \end{cases}$$

$$So, \quad ReLU(x) = \begin{cases} 0, x < 0 \\ 1, x \geq 0 \end{cases}$$

We will use (categorical cross entropy) as a loss function with sigmoid as shown before or Soft-max:

$$C.E. = -\sum_{n=1}^{n} yi * \log(\hat{y}i)$$

Nice property with soft-max:

$$\frac{\partial C.E.}{\partial Softmax} * \frac{\partial Softmax}{\partial zi} = \hat{v}i - vi$$

Methods

1. Exploring data:

- We use Brain Tumor Detection 2020 Kaggle dataset.
- It consists of 3065 MRI images separated to two classes (Tumor – Non-Tumor) brain.
- Both classes have almost the same number of images which is a positive point because unbalanced classes lead to underfitting and poor generalization to the test set.

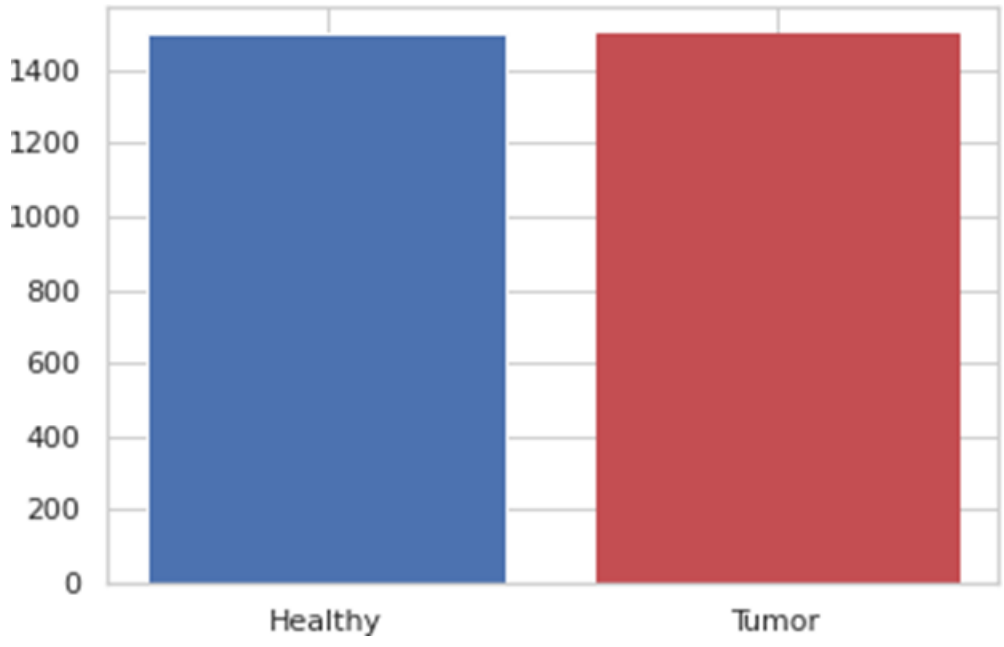


Figure 3 - Data Distribution

Brain Tumors are complex. There are a lot of abnormalities in the sizes and location of the brain tumor(s).

- This makes it difficult to completely understand the nature of the tumor.
- Train sample shows the variety of the tumor in different patients.

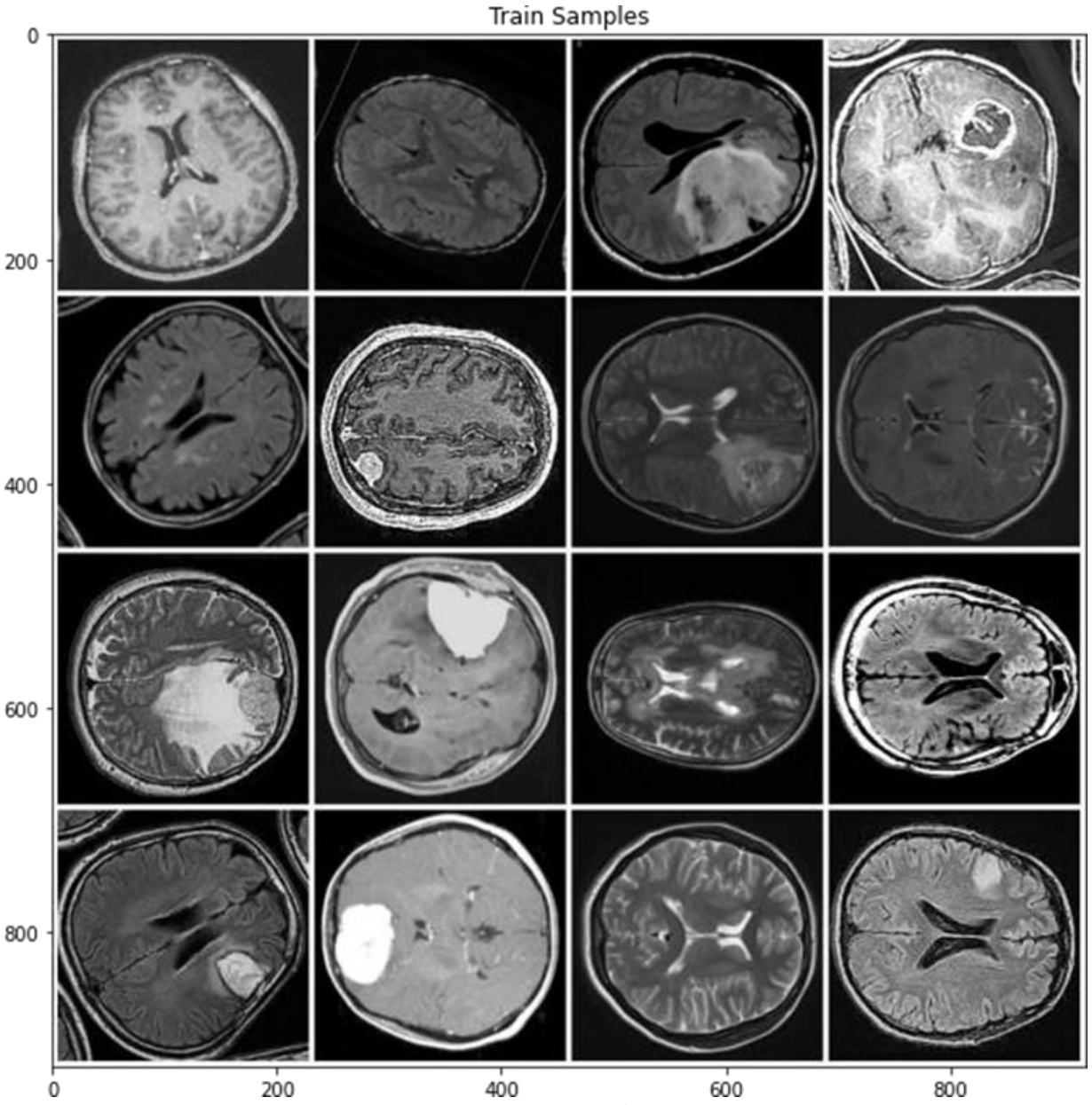


Figure 4 - Train Samples

2. Data analyzing:

- Image histogram:
  - Graphical representation of tonal distribution of an image.
  - Represents the number of pixels in each tonal value by plotting it.
- Equalization histogram:
  - Equalization implies mapping one distribution to another, so the Intensity value are spread over the whole range.
  - To accomplish this effect, the remapping should be the cumulative distribution function (CDF). For the histogram H(i), its CDF H (i) is:

$$\grave{H}(i) = \sum_{j=0}^{i} H(j)$$

To use this mapping function, we must normalize H (i) between 0 and 255. [1]

- Figure 5 shows the difference between image and equalization plots.

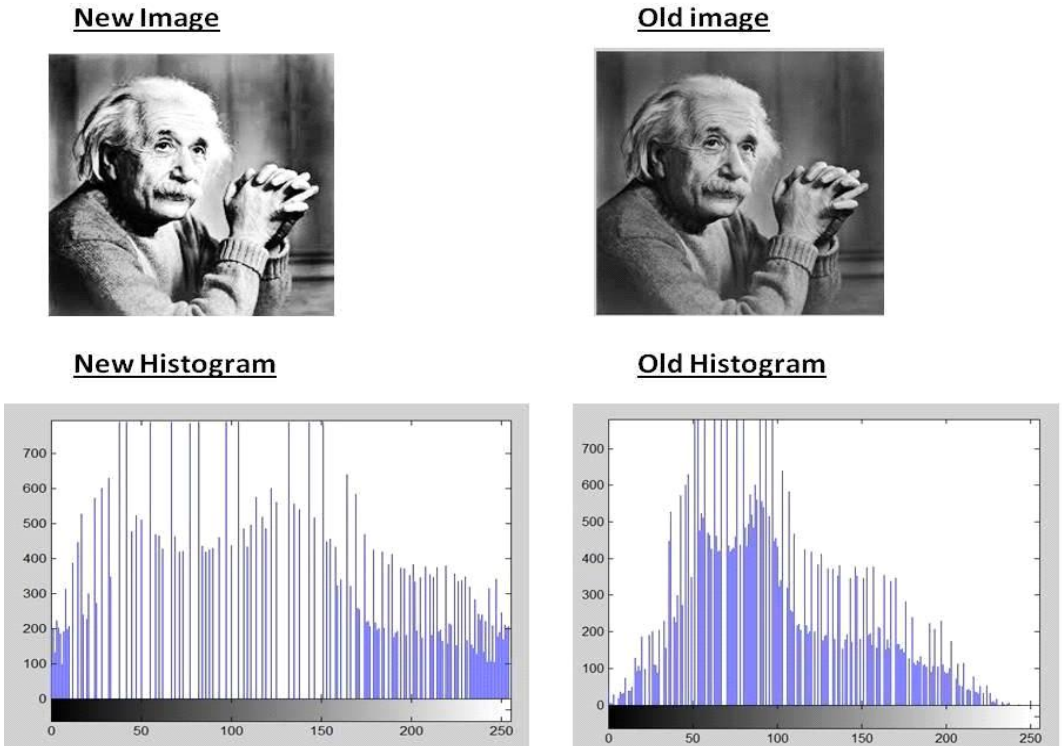


Figure 5 – Equalization histogram VS Image histogram [2]

- Equalization Histogram in our data:
  - We used cv2.COLOR_RGB2LAB to apply equalization histogram to perceptual lightness.
  - As illustrated in Figure 6, the first row shows the equalization histogram for healthy brain and the second one is for a brain with tumor.
  - Analyzing the histograms, we can deduce that the affected brains have much more perceptual lightness intensity.

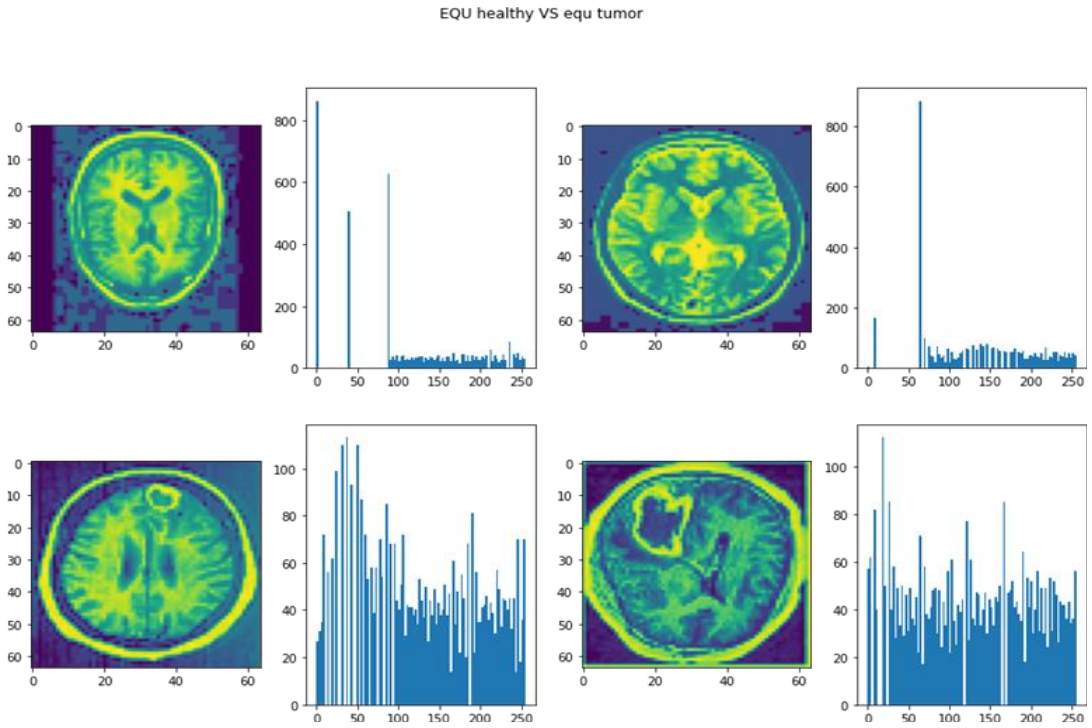


Figure 6 - EQU healthy VS tumor

3- CNN Model

- Input shape resized to be (64,64,3) (64,64) is the size of each image, 3 is the number of layers (RGB). [6]

Model consists of:

- **Three Conv2D Layers:**
  - This layer creates a convolution kernel that is convolved with the layer input to produce a tensor of outputs.
  - Each one has:
    - Activation layer with Relu.
    - Maxpoling2D has pool size (2,2).
- **Flatten layer:**
  - Flattens the multi-dimensional input tensors into a single dimension, so you can model your input layer and build your neural network model, then pass those data into every single neuron of the model effectively.
- **Dense (NN) layer:**
  - Neural network layer that is connected deeply. basically, used for changing the dimensions of the vector.
- **Activation.**
- **Dropout:**
  - Randomly sets input units to 0 with a frequency of rate at each step during training time, which helps prevent overfitting.
- **Dense.**
- **Activation**.

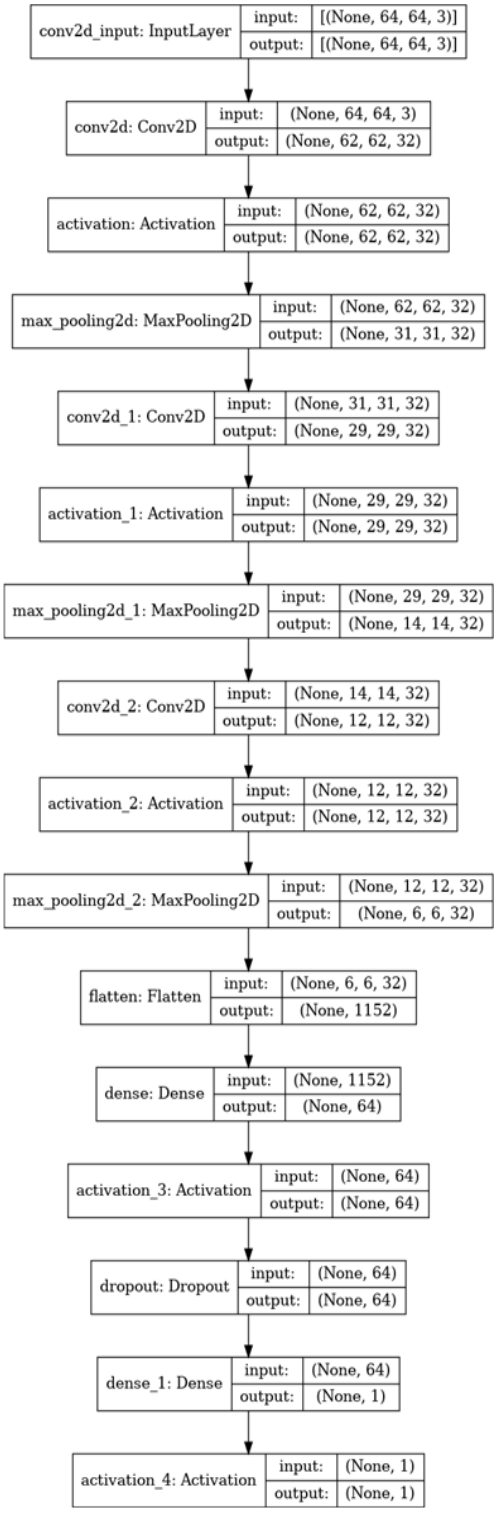


Figure 7 - Model Summary

4. Training the model:

- Compile parameters:

- **Loss:**
  - Loss function is the function we use for backpropagation.
  - binary_crossentropy computes the cross-entropy loss between true labels and predicted labels.
  - Cross-entropy loss is used for binary (0 or 1) classification, therefore is used in our model.
- **Optimizer:**
  - optimizer is a function or an algorithm that modifies the attributes of the neural network, such as weights and learning rate.
  - Thus, it helps in reducing the overall loss and improve the accuracy.
  - Adam is a replacement optimization algorithm for stochastic gradient descent for training deep learning models that can handle sparse gradients on noisy problems.
- **Metrics:**
  - Metric functions are similar to loss functions, except that the results from evaluating a metric are not used when training the model. Note that you may use any loss function as a metric.

- Fit parameters:

- **x_train:**
  - Input processed images arrays.
- **y_train:**
  - Output labels.
- **Batch size:**
  - Number of training examples utilized in one iteration.
- **Epochs:**
  - Number of epochs means how many times you go through your training set.
  - The model is updated each time a batch is processed, which means that it can be updated multiple times during one epoch.
  - If batch_size is set equal to the length of x, then the model will be updated once per epoch.
- **Validation data:**
  - It is the data used to update the model.
  - Data is trained and model is tune with the results of metrics (accuracy, loss etc.) that get from the validation set.
- **Shuffle:**
  - Boolean parameter determines whether to shuffle the training data before each epoch.

Results

5. Result:

- Validation accuracy 97%
- Best epoch 17

```
150/150 [==============================] - 4s 30ms/step - loss: 0.0254 - accuracy: 0.
9921 - val_loss: 0.0998 - val_accuracy: 0.9750
Epoch 17/17
150/150 [==============================] - 4s 25ms/step - loss: 0.0105 - accuracy: 0.
9962 - val_loss: 0.0732 - val_accuracy: 0.9700
```

Figure 8 - Training Result

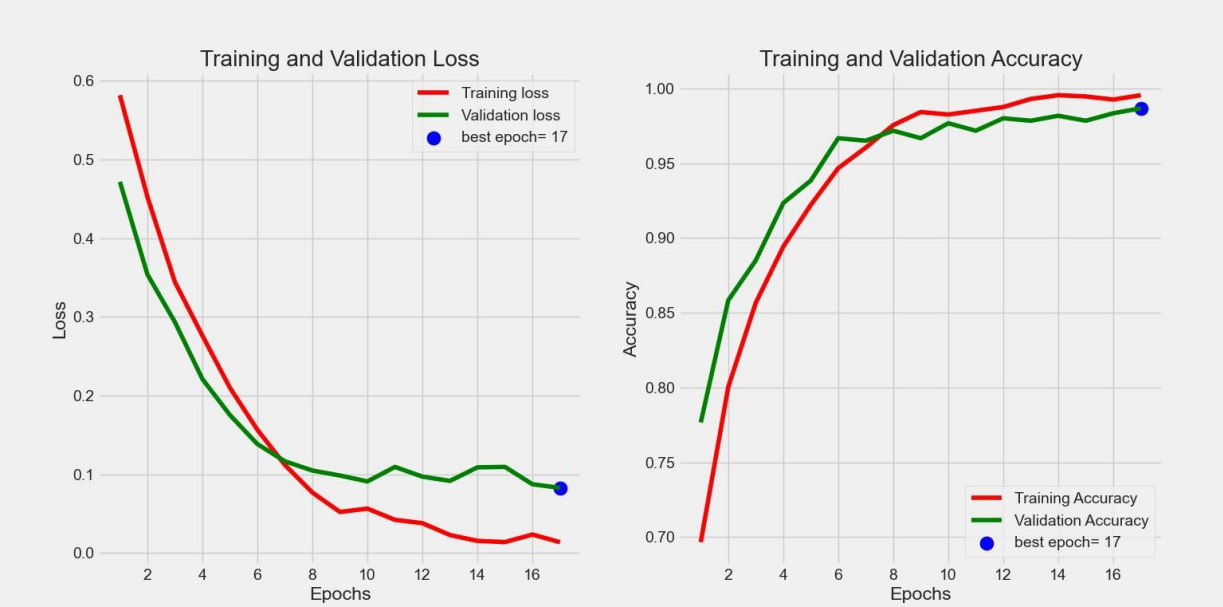


Figure 9 - Loss and Accuracy

Conclusion

From the above-mentioned detailed review, it can be concluded that there is a strong need of fully automatic unified framework that can efficiently detect and classify the brain tumor into multiple classes with less complexity, Using CNN most commonly applied to analyze visual imagery then pooling which reducing the image size by mapping a patch of pixels to a single value and finally the Forward and Backward Propagation as it is clear that Deep Learning techniques and algorithms have great power and ability.

References

[1]- S. S. Kunapuli and P. C. Bhallamudi, “Tumor detection,” Tumor Detection - an overview | ScienceDirect Topics. [Online]. Available: https://www.sciencedirect.com/topics/computer-science/tumor-detection

[2]- S. J; “Deep learning in neural networks: An overview,” Neural networks: the official journal of the International Neural Network Society. [Online]. Available: https://pubmed.ncbi.nlm.nih.gov/25462637/.

[3]- “Brain tumor detection using convolutional neural network,” IEEE Xplore. [Online]. Available: https://ieeexplore.ieee.org/document/8934561

[4]- M. Nazir, S. Shakil, and K. Khurshid, “(PDF) role of deep learning in brain tumor detection and ...” [Online]. Available: https://www.researchgate.net/publication/351609962_Role_of_Deep_Learning_in_Brain_Tumor_Detection_and_Classification_2015_to_2020_A_Review

[5]- “Deep learning course,” Intel. [Online]. Available: https://www.intel.com/content/www/us/en/developer/learn/course-deep-learning.html

[6]- S. Ravichandiran, Hands-on deep learning algorithms with python: Master deep learning algorithms with extensive math by implementing them using tensorflow. Birmingham: Packt Publishing, 2019.

πrates Crew

Name	Sec.	B.N.
Ibrahim Mohamed Ibrahim	1	2
Esraa Ali Esmail	1	12
Dina Khalid Mohammad	1	31
Kamel Mohamed Zaghloul	2	10
Mahmoud Yaser Mahmoud	2	29
Neveen Mohamed Ayman	2	49
Youssef Shaban Mohamed	2	57
Youssef Kadry Anwar	2	59