**Introduction:**

In the project we are going to diagnose the brain MR images and suggesting the user that whether the given MR image has the tumor or not and classifying the image as Benign or malignant.

For implementing this we have used Convolutional Neural Network (CNN) for classification of the images in the 2 main classes i.e. Benign and malignant. We will give the input image in the Nifty format (.nii) our software will convert that file into a simple image (.png) and perform classification on that image using CNN. We have trained our CNN classifier on our training dataset and saved that classifier using .h5 file so that it will not require retraining the classifier repeatedly for each image, this will reduce the time of execution of the algorithm.

CNN(convolutional Neural Network)

CNNs, like neural networks, are made up of neurons with learnable weights and biases. Each neuron receives several inputs, takes a weighted sum over them, pass it through an activation function and responds with an output. The whole network has a loss function and all the tips and tricks that we developed for neural networks still apply on CNNs.

CNNs derive their name from the “convolution” operator. The primary purpose of Convolution in case of CNNs is to extract features from the input image. Convolution preserves the spatial relationship between pixels by learning image features using small squares of input data.

Brief intro of Brain Tumors:

Brain tumor is any mass that results from an abnormal and an uncontrolled growth of cells in the brain. Its threat level depends on a combination of factors like the type of tumor, its location, its size and its state of development. Brain Tumors can be

1. cancerous (malignant)
2. non-cancerous (benign)

Benign brain tumors are low grade, non-cancerous brain tumors, which, grow slowly and push aside normal tissue but do not invade the surrounding normal tissue. They are homogeneous, well defined and are known as non- metastatic tumors, because they do not form any secondary tumor. Whereas, malignant brain tumors are cancerous brain tumors, which grow rapidly and invade the surrounding normal tissue. Malignant brain tumors or cancerous brain tumors counted among the most deadly diseases.

Literature Review:

**Paper-1**

**Title**:-**Classification using deep learning neural networks for brain tumors**

**Methodology**:-

The methodology for classification of brain tumor using DNN includes following four main steps.

**step-1 Brain MRIs Dataset acquisition**

There are more than 120 types of brain tumors which differ in origin, location, size, characteristics of the tumor tissues. In this paper three types of malignant brain tumour types are considered :

1. Glioblastoma

2. Sarcoma

3. Metastatic bronchogenic carcinoma

**step-2 Image segmentation using Fuzzy C-means**

Image segmentation is used to separate different normal brain tissue from brain tumor tissue. Fuzzy C-means is used to segment the brain MRI into 5 sections.

**step-3 Feature extraction using discrete wavelet transform (DWT) and reduction using Principle component**

**analysis (PCA) technique**

After segmentation features of the segmented tumor is extracted using discrete wavelet transform (DWT). Methodology utilizes a 3-levels decomposition of Haar wavelet to extract 32\*32 i.e 1024 features for each brain MRI. But this number is not so big compared as to the number of feature maps resulted by the convolution filters of Convolutional Neural Network. Thus the principal components analysis (PCA) is used to approximate the original extracted features with lower dimensional feature vectors.

**Step-4** **Classification using DNN**

After the features are extracted and selected, the classification step using DNN is performed on the resulted feature vector. Classification is performed by using 7-fold cross validation technique for building and training the DNN of 7 hidden layers structure.

**Disadvantages**:-

1.Require more hardware specification and take more time for processing for large size images like(256\*256)

2.Require separate method for segmenation and feature extraction.

**Paper-2**

**Title:-Methods for interpreting and understanding deep neural networks**

In this paper we have studied the problem of interpreting a deep neural network model and explaining its predictions.

Machine learning techniques such as deep neural networkshave become an indispensable tool for a wide range of applica-tions such as image classification, speech recognition, or naturallanguage processing.Techniques for interpreting and understandingwhat the model has learned have therefore become a key ingredient of a robust validation procedure. So interpretability has become more important in applications such as medicine or self-driving car, where the reliance of the model on the correct features mustbe guaranteed.

**1. Interpreting DNN model:-**

This section focuses on the problem of interpreting a conceptlearned by a deep neural network (DNN). A deep neural network (DNN) is an artificial neural network (ANN) with multiple layers between the input and output layers. The DNN finds the correct mathematical manipulation to turn the input into the output, whether it be a linear relationship or a non-linear relationship.As well as DNN is a collection ofneurons organized in a sequence of multiple layers, where neuronsreceive as input the neuron activations from the previous layer, andperform a simple computation.The neurons of the networkjointly implement a complex nonlinear mapping from the input tothe output. This mapping is learned from the data by adapting theweights of each neuron using a technique called error backpropagation.

* 1. Activation Maximization

Activation maximization is an analysis framework that searches for an input pattern that produces a maximum model response for a quantity of interest .

**2. Explaining DNN decisions**

In this section, we take input for a given data point x, which is used to find representative of a certain concept ω c encoded at the output of the deep neural network (DNN). The output neuron that encodes this concept can be described as a function f ( x ) of the input. Data point x can be viewed as a collection of features ( x i ) di = 1 , and to assign to each of these, a score R i determining how relevant the feature x i is for explaining f ( x ) .

**3.Layer-wise relevance propagation (LRP)**

LRP is a backward propagation technique designed for explanation. The LRP technique is based on conservation principle, where each neuron receives a share of the network output, and redistributes it to its predecessors in equal amount, until the input variables are reached.

**4.Evaluating explanation quality**

Some strategies to systematically and objectively assess the quality of explanations.

4.1. Transfer with a simple task

4.2. Explanation continuity

4.3 Explanation selectivity

4.1 discusses how a simple related task can serve as a proxy for that purpose. 4.2 and 4.3 discuss how to perform such quality assesse-ment by looking analytically at the explanation function and its relation to the prediction.

**6.Applications**

Domains as extraction of domain knowledge, computer-assisted decisions, data filtering, or compliance

1)Model validation

2)Analysis of scientific data

**Paper-3**

**Title:-Visualizing Higher-Layer Feature of a Deep Network**

Only model definitions and the quantitative analyses is not enough, there is also need for qualitative comparisons of the solutions learned by various DNN architectures. In this paper we have studied good qualitative interpretations of high level features represented by such models.

**1.Models**

In this paper we have studied two models. The first model is a Deep Belief Net (DBN).This model is obtained by training and stacking three layers as Restricted Boltzmann Machines (RBM) in a greedy manner. The second model, by Vincent et al. (2008), is the so-called Stacked Denoising Auto-Encoder (SDAE).

**2.Maximizing the activation**

We find the input patterns of bounded norm which maximize the activation of a given hidden unit.Asthe activation function of a unit in the first layer is a linear function of the input, in the case of the first layer, this input pattern is

proportional to the filter itself. The reason behind this idea is that a pattern to which the unit is responding

maximally could be a good first-order representation of what a unit is doing. We can find this for a given unit, the input sample(s) (from either the training or the test set) that give rise to the highest activation of the unit. Activation Maximization is applicable to any network in which we can compute the above gradients.

In this we have studied Sampling from a unit of a Deep Belief Network.

**3. Sampling a Unit**

The activation maximization method produces features and it decides which examples would “fit” these features; the sampling method produces examples and it lets us decide which features these examples have in common.

**Limitations**

one cannot find a simple representation of a higher layer unit as we scale the datasets to larger and larger images i.e DNN cannot process larger size iamges efficiently.

**Methodology:**

Our proposed methodology is based on the CNN architecture for the classification where the classifier identifies the Brain Tumor in Brain MR Images.

The proposed methodology for classifying the brain tumors in Brain MR Images is as follows:

1. Data acquisition
2. Data preprocessing
3. Splitting of data
4. Implementation of CNN

1. Data acquisition:

According to World Health Organization, there are total 120 types of brain tumors out of which we are focusing on the two main types of tumors Low Grade Glioma (LGG) and High Grade Glioma (HGG) which is also known as Benign and Malignant respectively.

For this project, data has collected from BraTS2018 challenge. We have downloaded the data with the ground truth-values from [https://www.cbica.upenn.edu/sbia/Spyridon.Bakas/MICCAI\_BraTS/2018/MICCAI\_BraTS\_2018\_Data\_Validation.zip [1](https://www.cbica.upenn.edu/sbia/Spyridon.Bakas/MICCAI_BraTS/2018/MICCAI_BraTS_2018_Data_Validation.zip%20%5b1)].

We have downloaded 285 patients real brain MR Images out of which 210 were HGG and 75 were LGG. All the images were in the sequence flair, t1, t2 and t1ce. These are the types of MR images in medical field depending on the various weighed conditions of imaging.

In total, we have downloaded 1421 Brain magnetic resonance images (MRI) out which 1046 are HGG and 345 are LGG images.

2. Data pre-processing:

Formation of dataset:

As dataset is of real Brain MR Images, it was arranged in separate folder for each patient. As the downloaded dataset was in the subfolder format, which is not suitable for the classification so it was necessary to arrange that data in the proper format.

We arranged that data in two main folders namely LGG and HGG.

Preprocessing of Images:

The images in the dataset was in nifty format, which is a well-known format for medical imaging. These images are 3 Dimensional images which are not supported by CNN as it works on 2D images only so we converted that all images into a simple 2D format (.png format) manually.

For conversion, we load the MR image first using nilearn library and converted that image into a 2D format using library function. We done this same process on all the images.

3. Splitting of dataset:

After conversion of that all .nii files into .png format we split data into training and testing sets. As the standard split ratio is 75-80% for training, we split our data into 80% for training and remaining for testing.

While splitting our dataset we avoided the overlapping of the data i.e. we avoided same image in training as well as testing set as it may cause error in accuracy of the classifier.

4. Implementation of CNN:

After splitting dataset into training and testing sets, we implemented the CNN.

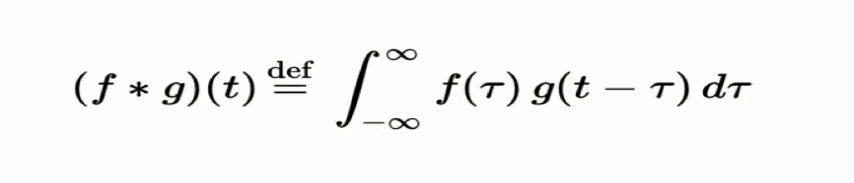
We built CNN in python using Keras and Tensorflow at the backend. Each image is given to the classifier with some transformations in the original image for training and testing and the accuracy of the system is observed after each epoch.

CNN can be implemented using 4 main steps:

1. Convolution Layer
2. Maxpooling Layer
3. Flattening Layer
4. Total connection

Now we will discuss all these stages in brief

1. Convolution Level

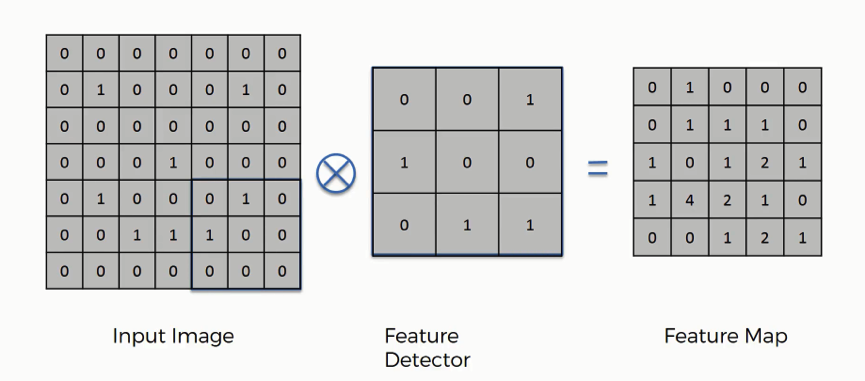
[](http://www.superdatascience.com/wp-content/uploads/2018/08/Convolutional_Neural_Networks_CNN_Step1_Img1.png)Convolution is a function derived from two given functions by integration, which expresses how the other modifies the shape of one. That can sound baffling as it is, but to make matters worse, we can look at the convolution formula:

Convolution operation works using these 3 elements

* Input image
* Feature Detector
* Feature map

Input image: As humans, computers cannot see the image as it is, images are nothing but a pixel value for the computer. Consider the following image as an example

We can see this as a smiling face but computer will see this image as a pixel matrix. Computer will see this image as



With 1 representing value is present and 0 representing absence of the value.

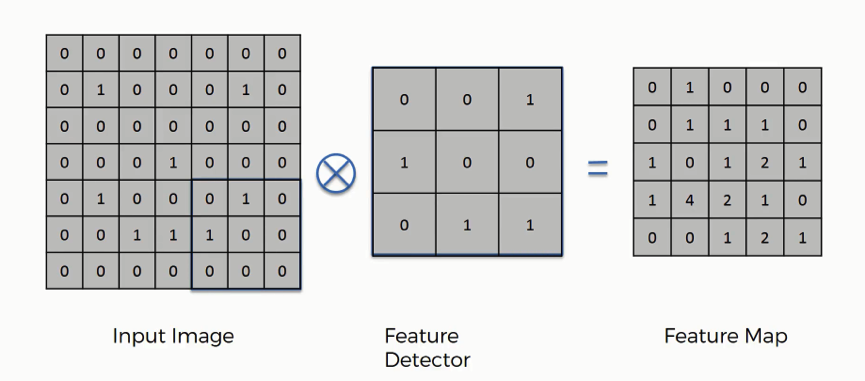
Feature Detector: It is simply a feature matrix, which compare for a specific feature in the image matrix depending on the image size and task we can create the feature detector for this example we will consider 3\*3 feature detector which is more conventional, it is often referred as filter. We can show the process as

Figure – Convolutional Operation

Feature map:

It is the product of image matrix and feature detector matrix. As shown in the above figure.

2. MAX POOLING

In this step, we enable the convolutional neural network to detect the required pattern from the image when presented in any manner like different viewing angle, lightening condition, etc. In order to perform pooling, we must finish convolution operation first, so that we will have feature map ready.

In this we pool out the max values from the feature map using some fixed pixel strides (2\*2 pixel strides are shown in the example below) and creates new pooled feature map.

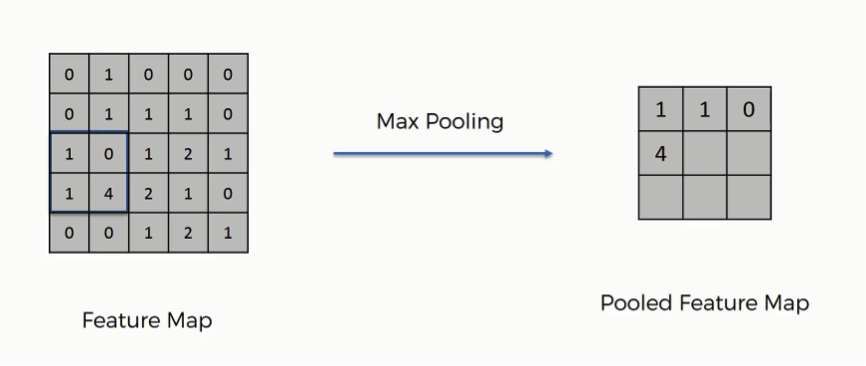
[](http://www.superdatascience.com/wp-content/uploads/2018/08/Convolutional_Neural_Networks_CNN_Step2_Img3.png)1.2.1Pooled Feature Map:

Figure – Max Pooling

Similar to convolution step, pooling disposes the unnecessary information or features from the original input and gives the optimized patterns.

1.3 Flattening

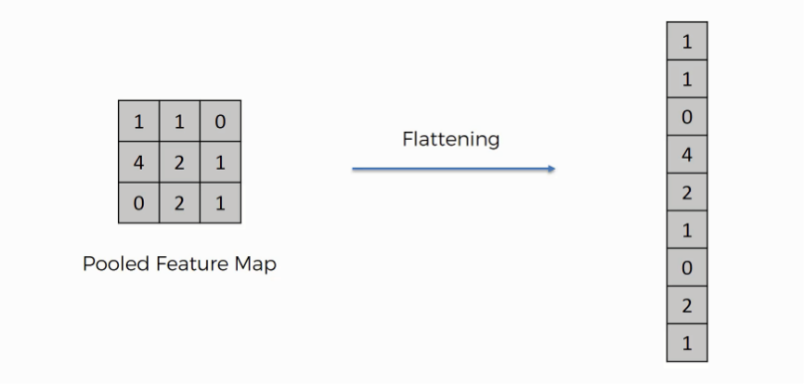
[](http://www.superdatascience.com/wp-content/uploads/2018/08/Convolutional_Neural_Networks_CNN_Step3_Img1.png)As the name suggest, we literally flatten our pooled feature map into a vector as shown in the image below.

Figure- Flattening of pooled feature map

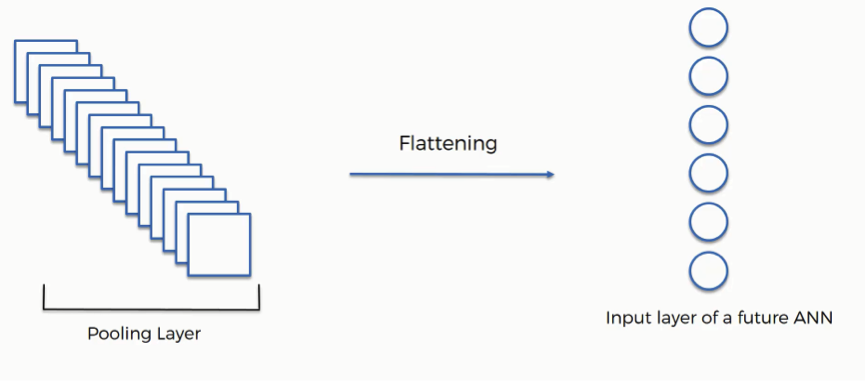
[](http://www.superdatascience.com/wp-content/uploads/2018/08/Convolutional_Neural_Networks_CNN_Step3_Img2.png)We perform flattening because we have to insert this data into an artificial neural network in the next step. As we have many pooled layers we flatten all these in into a vector and pass this vector as a input to the neural network for further processing.

Figure – Inserting data into NN

* 1. FULL CONNECTION

This is the step where we connect our convolution layer to our artificial neural network and create the Convolutional Neural Network. Neural network takes this data as input, learns the various features, patterns, attributes and combine them to wider attributes to classify the images.

References

[1] https://www.med.upenn.edu/sbia/brats2018/data.html