Brain Tumor Detection using Machine Learning Models

B.Sc. Semester VI Project
Department of Computer Science, Gurudas College

Bhargav Basu
Soptorshi Bhattacharjee
Brahmajit Das
Rajarshi Sardar

Gurudas College, Calcutta University

Overview

Motivation

The motivation is to develop a software with better segmentation capability for use in medical imaging to detect diseases like brain tumor. Image segmentation has been identified as the key problem of medical image analysis and remains a popular and challenging area of research. Image segmentation is increasingly used in many clinical and research applications to analyze medical imaging datasets; which motivated us to present a snapshot of dynamically changing field of medical image segmentation.

The motivation of this work is to increase patient safety by providing better and more precise data for medical decision.

Domain Description

- **Neurological Examination**: It is a series of test to measures the function of the patients nervous system and also his/her physical and mental alertness.

Domain Description

- **Neurological Examination**: It is a series of test to measures the function of the patients nervous system and also his/her physical and mental alertness.
- **Machine Learning**: Machine learning approaches address these problems by mainly using hand-crafted features (or pre-defined features).

Domain Description

- **Neurological Examination**: It is a series of test to measures the function of the patients nervous system and also his/her physical and mental alertness.
- **Machine Learning**: Machine learning approaches address these problems by mainly using hand-crafted features (or pre-defined features).
- **Brain Scans**: Brain scan is a picture of the internal structure of the brain. A specialized machine takes a scan in the same way as a digital camera takes a photograph.

Background

We propose the use of ML algorithms to overcome the drawbacks of traditional classifiers. We investigate and compare the performance of various machine learning models, namely **CNN**, **VGG 16** and **ResNet 50** ; implemented using the frameworks Tensorflow and fast.ai.

Methodology

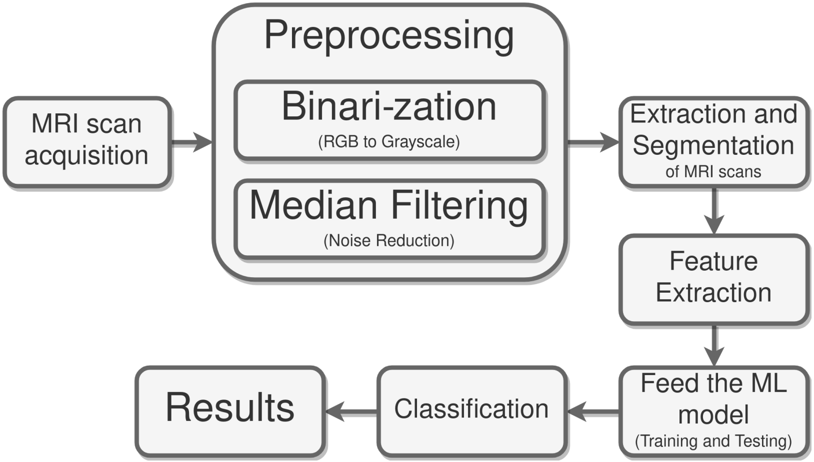
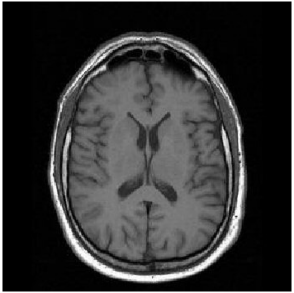
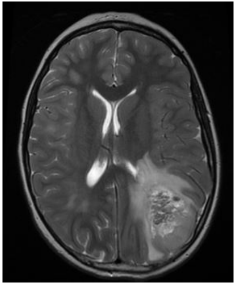


Figure: Proposed Methodology

The MRI brain images are acquired and are given as input to pre-processing stage.



(a) MRI scan shown no presence of tumor



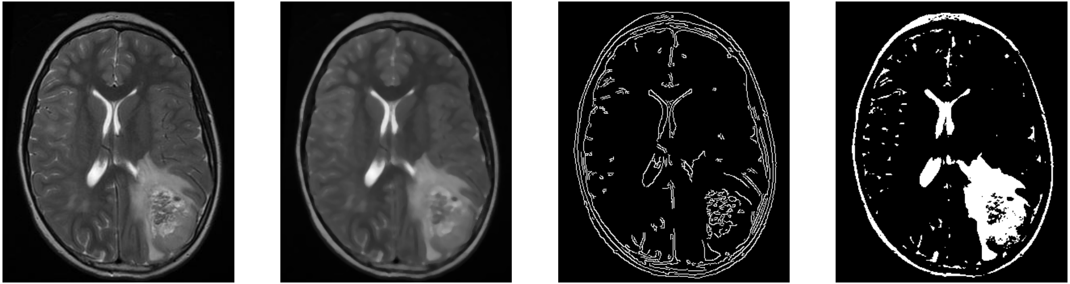
(b) MRI scan of a tumorous cell

Figure: MRI Scans

Preprocessing is needed as it provides improvement in image data which enhances some of the image features which are important for further processing.

Preprocessing includes:

- Binarization
- Filtering
- Edge Detection
- Segmentation



(a) Original Image (b) Filtered Imaage (c) Edge Detection (d) Segmentation

Figure: preprocessing operations

Methodology
Feature Extraction

When input to an algorithm is very large and redundant to be processed, it is transformed into reduced representative set of features called feature vector.

These features are extracted using Gray Level Co-occurrence Matrix (GLCM) as it is robust method with high performance.

The Machine learning algorithms are used for classification of MR brain image either as normal or abnormal. The major aim of ML algorithms is to automatically learn and make intelligent decisions.

For classification we are using **CNN**, **VGG 16** and **ResNet 50**.

For this project we are using binary classification, two classes one for normal and another for abnormal.

Image Classification

Image classification is where a computer can analyze an image and identify the 'class' the image falls under.

What is deep learning?

Deep learning is a type of machine learning; a subset of artificial intelligence (AI) that allows machines to learn from data. Deep learning involves the use of computer systems known as neural networks.

Why use deep learning for image classification

Deep learning allows machines to identify and extract features from images. This means they can learn the features to look for in images by analysing lots of pictures. So, programmers don't need to enter these filters by hand.

CNN or Convolutional Neural Network a class of artificial neural network, most commonly applied to analyze visual imagery.They are also known as shift invariant or space invariant artificial neural networks (SIANN), based on the shared-weight architecture of the convolution kernels or filters that slide along input features and provide translation equivariant responses known as feature maps.

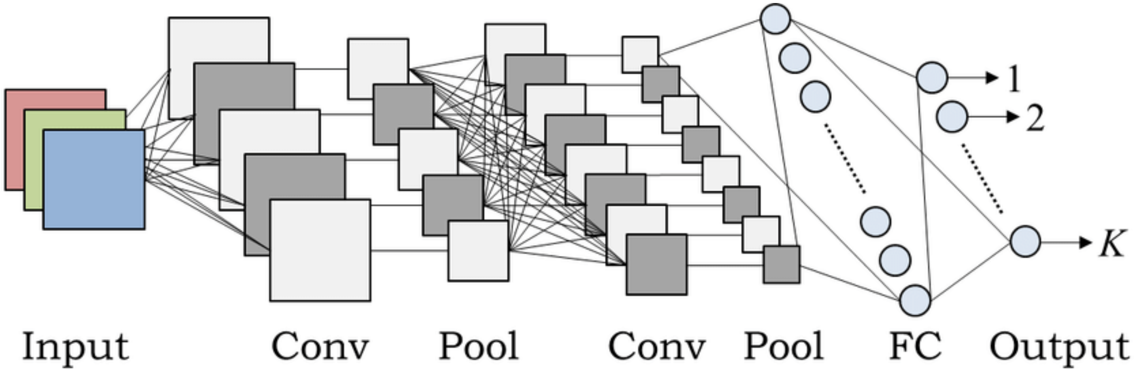


Figure: CNN's architecture

VGG 16 is is a significantly more accurate ConvNet architecture, which not only achieve state-of-the-art accuracy on ILSVRC classification and localisation tasks, but are also applicable to other image recognition datasets, where they achieve excellent performance even when used as a part of a relatively simple pipelines (e.g. deep features classified by a linear SVM without fine-tuning).

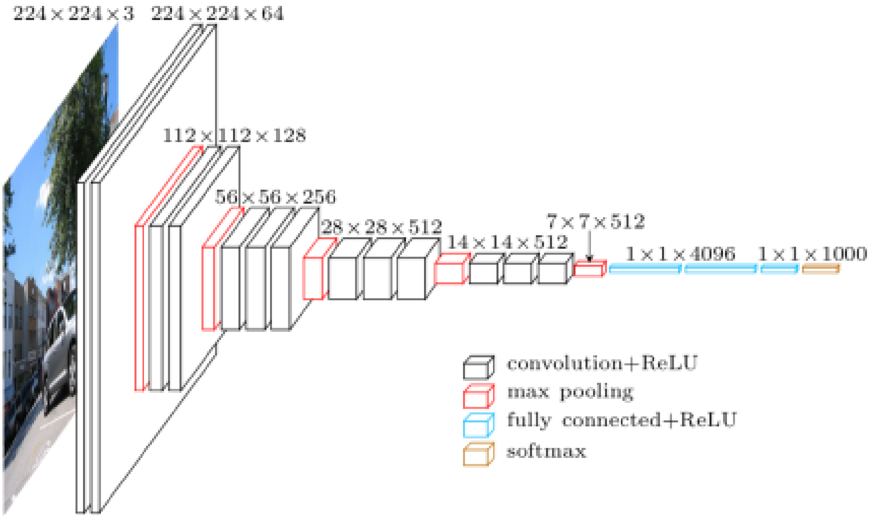


Figure: VGG 16's architecture

Model

ResNet 50

ResNet50 is a variant of ResNet model which has 48 Convolution layers along with 1 MaxPool and 1 Average Pool layer. It has $3.8x10^9$ Floating points operations. It is a widely used ResNet model and we have explored ResNet50 architecture in depth.

layer name	output size	18-layer	34-layer	50-layer	101-layer	152-layer
conv1	112×112	7×7, 64, stride 2				
conv2_x	56×56	3×3 max pool, stride 2				
		$\begin{bmatrix}3\times3, 64\\3\times3, 64\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 64\\3\times3, 64\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 64\\3\times3, 64\\1\times1, 256\end{bmatrix}\times3$
conv3_x	28×28	$\begin{bmatrix}3\times3, 128\\3\times3, 128\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 128\\3\times3, 128\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times4$	$\begin{bmatrix}1\times1, 128\\3\times3, 128\\1\times1, 512\end{bmatrix}\times8$
conv4_x	14×14	$\begin{bmatrix}3\times3, 256\\3\times3, 256\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 256\\3\times3, 256\end{bmatrix}\times6$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times6$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times23$	$\begin{bmatrix}1\times1, 256\\3\times3, 256\\1\times1, 1024\end{bmatrix}\times36$
conv5_x	7×7	$\begin{bmatrix}3\times3, 512\\3\times3, 512\end{bmatrix}\times2$	$\begin{bmatrix}3\times3, 512\\3\times3, 512\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$	$\begin{bmatrix}1\times1, 512\\3\times3, 512\\1\times1, 2048\end{bmatrix}\times3$
	1×1	average pool, 1000-d fc, softmax				
FLOPs		1.8×10^9	3.6×10^9	3.8×10^9	7.6×10^9	11.3×10^9

Figure: Reset's architecture

Implementation

Assumptions

It is assumed that the MRI scans are collected and processed before feeding into the module.

The program is dependent on external modules and are expected to be pre-installed on the system. The modules namely include:

- Tensorflow
- fast.ai
- OpenCV
- matplotlib

- Acquire Data (MRI Scans)
- Dividing the data into two classes while keeping the separate batch for prediction
- Build the models using the aforementioned libraries
- Train the model (feed data into the model)
- Evaluate the trained model on the prediction batch

Result

Comparing the model accuracies

Model	Implementation using TensorFlow	Implementation using fast.ai
CNN	90.31%	92.66%