Brain Tumor Segmentation using Convolutional Neural Network

Project report submitted in partial fulfillment of the requirement
for the degree of Bachelor of Technology

In

COMPUTER SCIENCE AND ENGINEERING

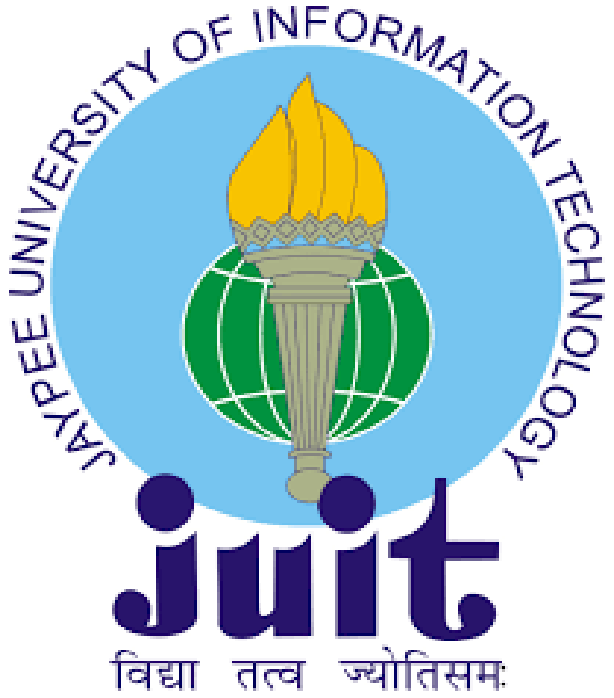
By

CHANDAN DHIMAN (181344)

UNDER THE SUPERVISION OF

Dr. Aman Sharma

To



Department of Computer Science & Engineering and Information Technology

Jaypee University of Information Technology Waknaghat,

Solan-173234, Himachal Pradesh

I

CERTIFICATE

Candidate's Declaration

I hereby declare that the work presented in this report entitled **"BRAIN TUMOR SEGMENTATION"** in partial fulfillment of the requirements for the award of the degree of **Bachelor of Technology** in **Computer Science and Engineering/Information Technology** submitted in the department of Computer Science & Engineering and Information Technology**,** Jaypee University of Information Technology Waknaghat is an authentic record of my own work carried out over a period from January 2022 to May 2022 under the supervision of **Dr. Aman Sharma**, Assistant professor.

The matter embodied in the report has not been submitted for the award of any other degree or diploma.

Student Name: Chandan Dhiman

Roll No.: 181344

This is to certify that the above statement made by the candidate is true to the best of my knowledge.

(Supervisor Signature)

Supervisor Name: Dr. Aman Sharma Designation:

Assistant Professor

Department Name: Computer Science and Engineering Dated:

II

ACKNOWLEDGMENT

I would like to express my heartiest thanks and gratefulness to all my friends, teachers and family members who stood by my side for making it possible to complete this project successfully.

I am very thankful and would like to convey my gratitude to **Dr. Aman Sharma** (Assistant Professor) of Computer Science and Engineering Department of **Jaypee University of Information Technology**. His, scholarly guidance, endless patience, constant and energetic supervision, valuable advice, reading many inferior drafts and correcting them at all stages have made it possible to complete this project.

Your time, guidance and suggestions helped me in n-number of steps of the completion of the project. I will always be very thankful to you in these regards.

I would also like to extend a warm welcome to each of those people who have helped me directly or indirectly in making this project a success. In this unique situation, I would like to thank the various staff, educators and non-educators, who have developed their simple help and assisted with my work.

Chandan Dhiman

III

TABLE OF CONTENTS

IV

LIST OF FIGURES

V

LIST OF TABLES

V

Abstract

Brain Tumor is one of the most harmful disease which basically develops due to the abnormal growth of cells/tissues within the brain. For the study of brain tumor segmentation/detection, MRI images come into play. With the help of MRI images, we are able to detect brain tumor. By looking at the abnormal shape and size of the tissues inside the brain in an MRI image, we can clearly distinguish between a brain having no tumor and a brain having tumor. We also need to check the symmetric and asymmetric shape of brain which tells us about the abnormality.

There is a huge difference in the appearance of Brain tumors and some similarities between tumor and normal tissues. Thus, the extraction of tumor becomes more important so as to help the doctors to clearly identify the main difference. In this paper, we have used Fuzzy C-Mean clustering algorithm followed by the traditional classifiers and CNN (Convolutional Neural Network) for the extraction of brain tumor from the 2D Magnetic Resonance brain Images (MRI). The experiment was carried on a dataset provided by Kaggle.

The dataset is small as it is having 253 Brain MRI Images out of which 98 images have no tumor and 155 images are having tumor. The traditional part contains some classifiers such as Naïve Bayes, Random Forest, K-Nearest Neighbor (KNN), Support Vector Machine (SVM) and Logistic Regression. But with the help of traditional classifiers we were not able to get the desired results and because of that we jumped towards CNN (Convolutional Neural Network) which is implemented using Keras and TensorFlow.

CNN provides us with much more accurate results and better performance as compared to the traditional classifiers. The main aim of this paper is to distinguish between the normal pixels and abnormal pixels.

Chapter 1:

Introduction

1.1 Introduction

Brain tumors square measure are involved in the abnormal growth of cells within the brain. The accuracy of the square root causes has not been identified, yet their square measurement features may increase the risk of the tumor, such as radiation exposure and a history of brain cancer cases. There has been an increase in cases of brain tumors all over the world reported over the past few years. In the United States alone, 78 were for the Associate in Nursing average. Nine hundred and eighty (980) new cases of cancer and non-cancerous tumors were expected to be found in 2018-19. Brain tumors can be cancerous (malignant) or non-cancerous (malignant). As malignant or malignant tumors grow, they can cause pressure inside your skull to increase. This can cause mental damage, and it can put lives at risk.

Imaging imagining techniques, such as resonance imaging (MRI), CT scans,
Positron emission pictorial representation (PET), among other things, plays an important role in plant identification. These are the most common square measurements to detect and evaluate tumor progression before and during treatment. magnetic resonance imaging is often an alternative to the diagnosis and treatment of brain tumors due to their high specificity, soft tissue fragmentation, and non-invasive features. Surgery is the most common way to treat brain tumors, however radiation and treatment may tend to delay tumor growth.

A limited analysis of the affected cells reveals indications of disease progression, its characteristics, and effects on a complex organ. The task seemed daunting, due to the wide variety of form, size, and location of the lesions.

Figure: 1.1

Types of Brain Tumor

Primary brain tumors

The primary brain tumor come from your brain. They can improve from:

- brain cells.
- The membranes that surround your brain, called meninges.
- nerve cells.
- glands, such as the pituitary of the pineal.

The primary tumor can be dangerous or cancerous. In adults, the most common types of brain tumors are gliomas and meningiomas.

Gliomas

Abscess gliomas appear in glial cells. These cells are usually:

- Support the structure of your central nervous system.
- Feed your central nervous system.

- Clean cellular waste.
- breaking dead nerves.

Gliomas can occur in different types of glial cells.

Types of plants that start in glial cells include:

- astrocytic tumors, such as astrocytoma's, from the cerebrum.
- oligodendroglia tissue, which is usually found on the anterior temporal lobes.
- Glioblastomas, which originate in the supporting brain trunk and are not the most aggressive.

Also, in clinical follow-up, tumor screening is usually done by hand. An experienced radiotherapist can quickly study scanned medical images of a patient that divides all affected regions. except that most of the time, the classification by hand depends on the radiotherapist and is subject to a large variety of intra and lay rater. Therefore, manual separation is limited to quality tests or visual inspections only.

At the same time, brain volume testing provides important data for a solid understanding of tumor symptoms and treatment formation. Limited evaluation of the affected cells reveals indications of disease progression, its characteristics, and effects on the complex organ. The task seemed daunting, given the sheer size, size, and location of the sores.

1.2 Problem Statement

How to properly detect the location of a tumor in the human brain so that the remedy and action needed to begin early should prevent it from working negatively in the human body.

Separation of tumors basically represents an accurate identification of the plant area. The classification performed by a radiologist is considered to be the best in the class. Although, specialist classification is not very accurate, it may include undeveloped tissue. It takes time to look at permanent and temporary profiles and explore a wide range of advanced data and pixel profiles while determining the boundary of the wound. A few known methods in the literature for the diagnosis of breast ulcers.

The section is based on a user-defined seed area and a circular circle consisting of a straight back voxel and a wound followed by a boundary that increases the variability of the class of these voxels. The group-based integration approach is developed while shift clustering is used in ROI selection to analyze the related boundary. The separation method is defined based on the following steps: the selection of the intermediate wound and the boundary scale, the analysis of the connected part, and the filling of holes and removal of leaks. However the algorithm is only suitable for weight lifting wounds and will require modification of weightless wounds. The proposed segmentation algorithm requires a complex manual partition and refines this based on graph cutting based on minimizing power. The standard separator method used by subtracting - means (FCM) is used in both the initial division and the appearance of the standard set.

1.3 Objectives

The objectives of brain tumor segmentation is to identify spatial location of tumor in human brain by reading MRI report.
Brain tumor segmentation may be a troublesome task because of the quality of magnetic resonance imaging brain pictures, and it aims to predict tumors by segmenting them through computer science models. we have a tendency to propose BU-Net to phase and classify the tumor regions. For the definite segmentation of brain tumors, we've got planned a completely unique model with modifications in encoder–decoder design. we've got introduced 2 new blocks, namely, residual extended skip (RES) and wide context (WC), into the present U-Net design. Special attention is given to the discourse options of the magnetic resonance imaging scans that have proved to be useful for the segmentation of tumor regions. a rise within the valid receptive field is achieved mistreatment RES block, that improves the general performance.

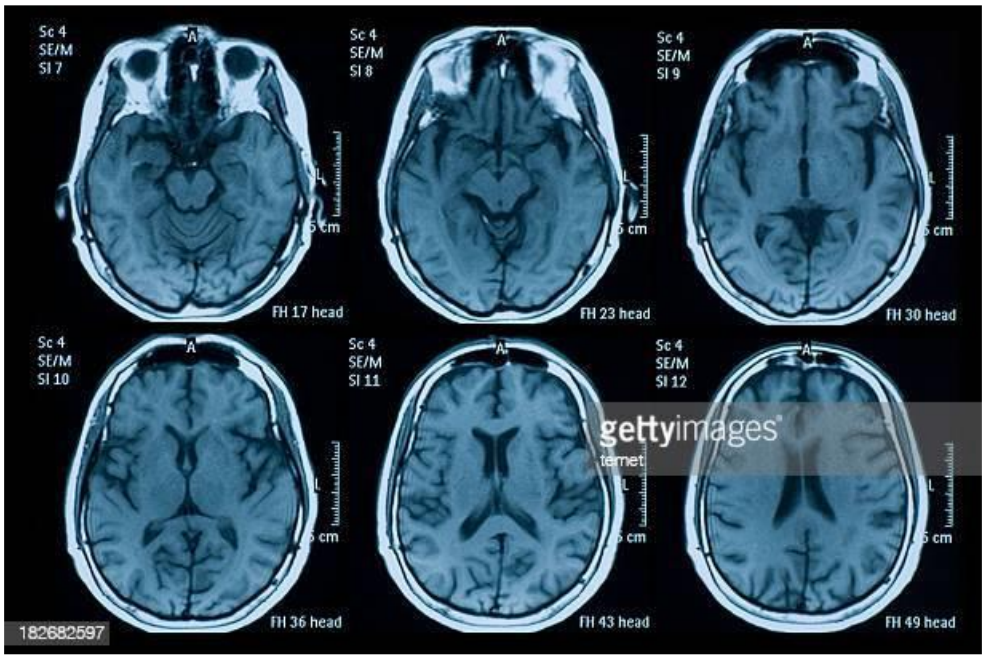


Figure 1.2

1.4 Methodology

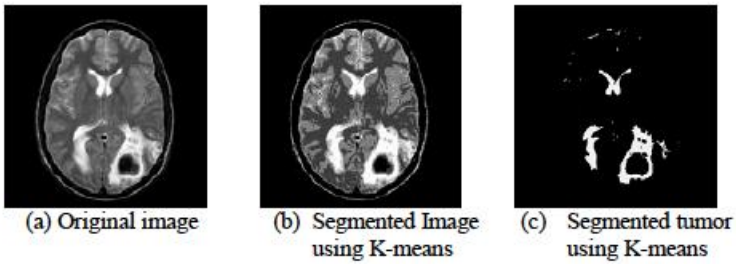
In this section there is an introduction to the various classification algorithms and methodologies in the proposed framework. Before using the main method (CNN) for the classification, other classifiers were used so as to find the accurate results. The main objective/methodology of this project is to detect weather a brain is tumorous or non-tumorous. The dataset which is being used in making of the project is taken from kaggle. The dataset is small in number so that we could get a prescribed result in shorter period of time. We have first used a large dataset and there was a huge loss in accuracy, to overcome that situation we jumped towards a small dataset. There are n-number of methods for preforming the segmentation but we have used CNN (Convolutional Neural Networks) with some other classifiers such as k-mean clustering algorithm, image segmentation, Naïve Bias and many more for gaining accuracy as high as possible.

Image segmentation:

Part of the human brain consists of separating different traditional brain tissues such as Gray substance (GM), white matter (WM) and humor (CSF) and therefore bone from plant tissue. Cluster techniques are ideal for segmentation of brain master images because it supports organizing objects into teams of comparable options, attributes and features. group strategies are divided into supervised and neglected strategies that do not agree on the type of learning. In supervised strategies, the terms of the collection are required to be better known anywhere within the unchecked strategies, only the number of collections needed to be better known to make a collection.

K-Means cluster:

K-means cluster is that the simplest unattended cluster technique which will work for big range of variables and classifies the input file into multiple categories supported their inherent distance from one another. In k-means cluster, it clusters a given set of knowledge employing a bound range of categories based on the similarity between the given knowledge and therefore the classes' centers. Following figure illustrates brain imaging segmentation victimization k-means, the amount of categories was given as 5.



(a) Original image (b) Segmented Image using K-means (c) Segmented tumor using K-means

Figure 1.3

Naïve Bayes

Naive Bayes classifiers is a collection of classification algorithms based on Bayes' Theorem. It is not a single algorithm but a family of algorithms in which they all share the same goal, i.e. all the separated components are independent of each other.

The Bayes Theorem finds opportunities to happen when considering the possibility that another event has already taken place. The Bayes theorem is mathematically defined as the following equation:

$P(A \mid B) = P(B \mid A)\, P(A) / P(B)$

where A and B are events and $P(B) \neq 0$.

• Basically, we are trying to find opportunities for event A, as event B is a reality. Event B is also referred to as evidence.
• P (A) is the subject of A (pre-event, i.e. pre-event probability). Evidence is an attribute value of an unknown event (here, event B).
• P (A | B) is the probability after B, i.e. the probability of an event after the discovery of evidence.

Random forest

Random Forest is a supervised machine learning algorithm that is widely used for planning and retrieval problems. Build decision-making trees from different samples and take their majority vote to split and rate in the event of a decline.
One of the most important features of the Random Forest Algorithm is that it can manage a set of data containing continuous variables such as reversal and segmentation as a separate segment. It produces the best results of classification problems.

Before understanding the random operation of the forest, we should look at the ensemble

technology. Integration simply means combining multiple models. Therefore, a set of models is used to make predictions rather than each model.
The Ensemble uses two types of methods:

1. **Bagging** - Creates a sub-training set from the training sample data that is adaptable and the end result is based on a majority of votes. For example, Random Forest.

2. **Boosting** - Incorporates weak students into strong students by building sequential models so that the final model has the highest accuracy. For example, ADA BOOST, XG BOOST

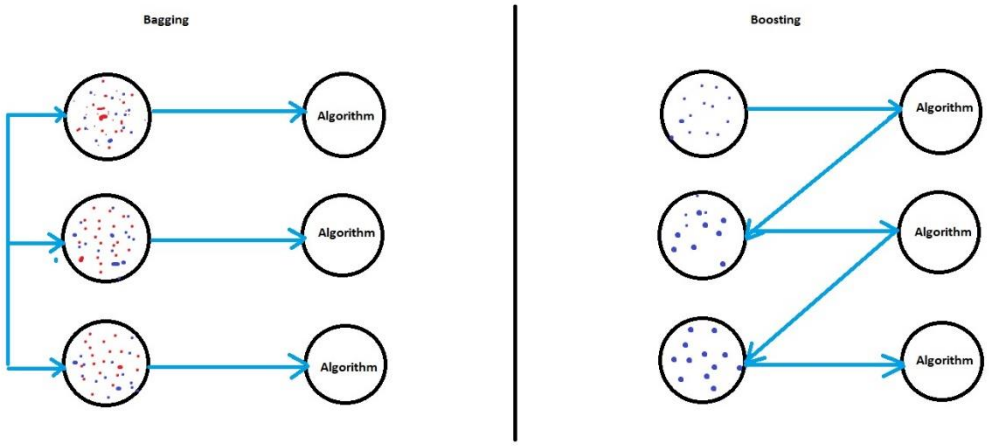


Figure 1.4

KNN (K Nearest Neighbor)

The Close Neighbor algorithm falls under the category of Supervised Reading and is used for division (frequency) and regression. It is a flexible algorithm and is used to insert missing values and re-sample data sets. As the name (K's nearest neighbor) suggests it assumes K neighbors (Data points) to predict the class or continuous value of the new Datapoint.

The readings of the algorithm are:

1. **Story-based learning**: In this part we do not study weights from training data to predict output (as in algorithms-based models) but use all training scenarios to predict the outcome of intangible data.

2. **Lazy Learning**: The model is not read using pre-training data and the learning process is postponed the time when speculation is requested in the new context.
3. **Non-Parametric**: For KNN, there is no predefined type of map function.

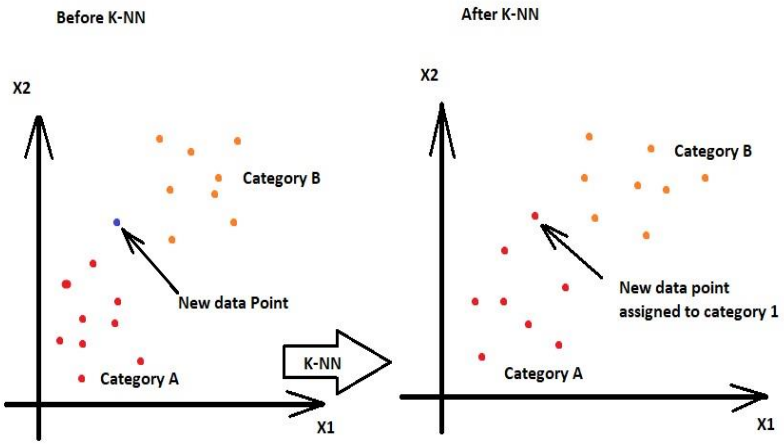


Figure 1.5

CNN (Convolutional Neural Network)

A convolutional neural network (CNN) is a type of artificial neural network used for image detection and processing designed to process pixel data.

CNN is a powerful imaginative, artificial intelligence (AI) system that uses in-depth learning to perform productive and descriptive tasks, often using machine vison that integrates image and video recognition, as well as NLP complementation and processing programs.

The neural network is a hardware system and / or software that controls the pattern after the activation of neurons in the human brain. Traditional neural networks are not suitable for image processing and should be fed images with fragmented fragments. CNN has its own "sensory" structures, such as those of the previous record, the site for processing human and other visual aids. The layers of neurons are arranged in such a way that they cover the entire viewing field to avoid the problem of processing image of traditional neural networks.

CNN uses a multilayer perceptron system designed for reduced processing needs. CNN layers contain input layer, output layer and hidden layer that includes multiple flexibility layers, integration layers, fully integrated layers and custom layers. Removal of limitations and increased efficiency of image processing results in a more efficient, easy-to-use training system that limits image processing and natural language processing.

Suppose we have a N * N layer of neuron squares followed by our convolutional layer. If we use the m * m

filter, the output of our layer will be the size (N − m + 1) * (N − m + 1). In order to calculate pre-aligned input

lines for a particular unit xℓij in our layer, we need to summarize the contributions (rated by filter components)

from previous layer cells:

$$x_{ij}^{\ell} = \sum_{a=0}^{m-1} \sum_{b=0}^{m-1} \omega_{ab} y_{(i+a)(j+b)}^{\ell-1}.$$

Then, the convolutional layer applies its nonlinearity:

$$y_{ij}^{\ell} = \sigma(x_{ij}^{\ell}).$$

Backward Propagation

$$\frac{\partial E}{\partial \omega_{ab}} = \sum_{i=0}^{N-m} \sum_{j=0}^{N-m} \frac{\partial E}{\partial x_{ij}^{\ell}} \frac{\partial x_{ij}^{\ell}}{\partial \omega_{ab}} = \sum_{i=0}^{N-m} \sum_{j=0}^{N-m} \frac{\partial E}{\partial x_{ij}^{\ell}} y_{(i+a)(j+b)}^{\ell-1}$$

$$\frac{\partial E}{\partial x_{ij}^{\ell}} = \frac{\partial E}{\partial y_{ij}^{\ell}} \frac{\partial y_{ij}^{\ell}}{\partial x_{ij}^{\ell}} = \frac{\partial E}{\partial y_{ij}^{\ell}} \frac{\partial}{\partial x_{ij}^{\ell}} \left(\sigma(x_{ij}^{\ell})\right) = \frac{\partial E}{\partial y_{ij}^{\ell}} \sigma'(x_{ij}^{\ell})$$

$$\frac{\partial E}{\partial y_{ij}^{\ell-1}} = \sum_{a=0}^{m-1} \sum_{b=0}^{m-1} \frac{\partial E}{\partial x_{(i-a)(j-b)}^{\ell}} \frac{\partial x_{(i-a)(j-b)}^{\ell}}{\partial y_{ij}^{\ell-1}} = \sum_{a=0}^{m-1} \sum_{b=0}^{m-1} \frac{\partial E}{\partial x_{(i-a)(j-b)}^{\ell}} \omega_{ab}$$

Max-Pooling Layer

The layers of high integration do not do any reading on their own. Instead, reduce the size of the problem by introducing a minimum. In forward distribution, k * k blocks are reduced to a single value. Then, this single value gets the calculated error from streaming back from the previous layer. This error is then forwarded to the default location. As it appears in only one place in the k * k block, the spread errors from multi-layer layers are very small.

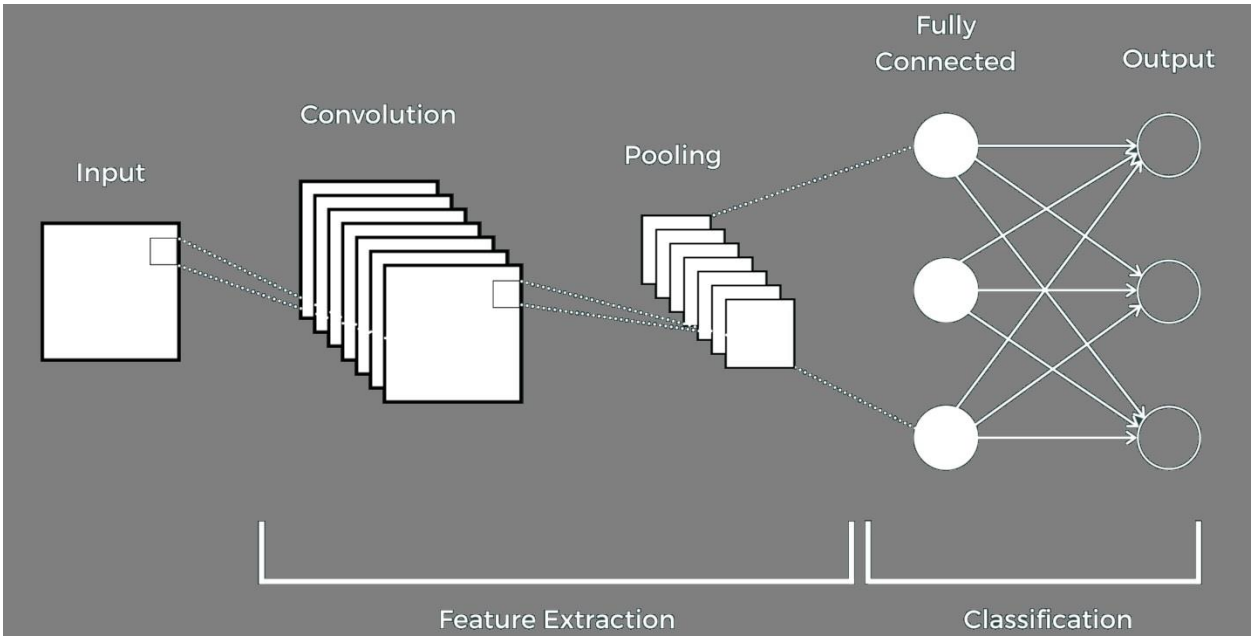


Figure 1.6

In recent studies, CNN became a de-factor model for brain tumor differentiation due to its record-breaking performance on chronic computer vision problems and medical image analysis compared to other models. CNN models are able to read the local layout of elements within the data, for example, the first layer will learn smaller local patterns, such as edges, the second layer will read larger patterns formed by the elements of the previous layer and so on. This ability equips them to work well on image analysis. In addition, convolutional layer units share weights, thereby reducing the parameter value to read and improve network efficiency.

- **Building Blokes of CNN**

The convolutional layer contains a set of readable filter or filters (typical size is usually 3 × 3 or 3 × 3 × 3, depending on which input is a two-dimensional image (2D) or three-dimensional (3D), respectively) used to slide over the entire input volume, create a dot product between filter input and instant input. Therefore, the convolutional operation first removes the patches from the input in the form of a smooth window, and then applies the same line conversion to all of these episodes. The output of the convolution function is sometimes called a feature map. The network will read filters that detect certain visual patterns present in the input data. When the convolutional layers are stacked sequentially, the network is able to learn the sequence of complex growing features, from the simple edges to the ability to detect the presence of a face as an example.

Over the past few years, there have been various efforts aimed at improving the performance of in-depth learning models by replacing the common layer of conversion with blocks that increase network capacity while using smaller computational resources. For example, Szegedy et al. introduced the first block that captures small correlation patterns while using multiple measurement reception fields. Their network architect, Google Net, which won ILSVRC 2014, has fewer network limitations and requires fewer calculation resources than its predecessors Alex Net or VGG. The residual block was another significant development, which made very deep networks unaffected by the issue of gradient collapse. Hu et al. introduced the Squeeze-and-Excitation (SE) block that captures interdependence between maps of network features.

- **Pooling Layer**

A pooling layer usually follow a convolutional layer or a set of convolutional layers. The goal is to reduce the dimensions of the feature maps, and at the same time, keep important features. A pooling operation is applied to a rectangular neighborhood in a sliding window fashion. For example, the max pooling is used in order to produce a maximum of a rectangular neighborhood. Other popular pooling operations include average and weighted average pooling.

- **Non-Linearity Layer**

Basically, CNN uses 3 steps, which are:

1. Convolutional operations are applied on the input function/features maps for the production of linear activation.
2. After that, for the output feature maps, a non-linear transformation is applied/performed.
3. At last for the modification of the output a pooling layer is applied.

Non-line conversions can be achieved through a special category of functions, called activation functions. Non-linearity gives the network the ability to read less important presentations. Therefore, enabling the network to withstand minor changes or noise in the input data and to improve the efficiency of the representation computer.

In the past, sigmoid and hyperbolic tangent functions were commonly used for the non-linearity layer. Today, the most popular activation function is the rectified linear unit (ReLU), which is expressed as $f(z)=\max(z,0)$. It was observed that where ReLU typically learns faster in network with many layers and does not suffer from vanishing/exploding gradients, as with the sigmoidal activations. However, ReLU presents some potential drawbacks when the network saturates with a constant zero gradient causing the network to converge slowly. As a solution, Maas et al. proposed a Leaky ReLU (LReLU) that allows for small, non-zero gradient to flow when the network is saturated. This function is defined as:

$$f(z)=\max(z,0) +\alpha \min (0, z)$$

where α is a constant leakiness parameter (typically 0.01). Another common variant of ReLU is Parametric Rectified Linear Unit (PReLU). This activation function adaptively learns the parameter α in Equation, thus improving the accuracy with less computational cost.

- **Fully Connected Layer**

The convolutional layers are used as feature extractors. The features that they produce are then passed to the fully connected (FC) layers for classification. Each unit in the FC layer is connected to all of the units in the previous layer, as shown in Figure 1.7. The final layer is usually a softmax classifier, which produces a probability vector map over the different classes. All of the features are converted in to a one-dimensional feature vector before being passed to a FC layer. By doing so, spatial information inherent in image data is lost. Another issue with the FC layers is that they have a larger number of parameters as compared to other layers that increase the computational costs and require input images to be of the same size.

As a solution to above problems, Long et al. proposed converting FC layers to 1×1 convolutional layer, thus transforming the network into a fully convolutional network (FCN). The network takes the input of any arbitrary sizes and outputs a grid of classification maps.

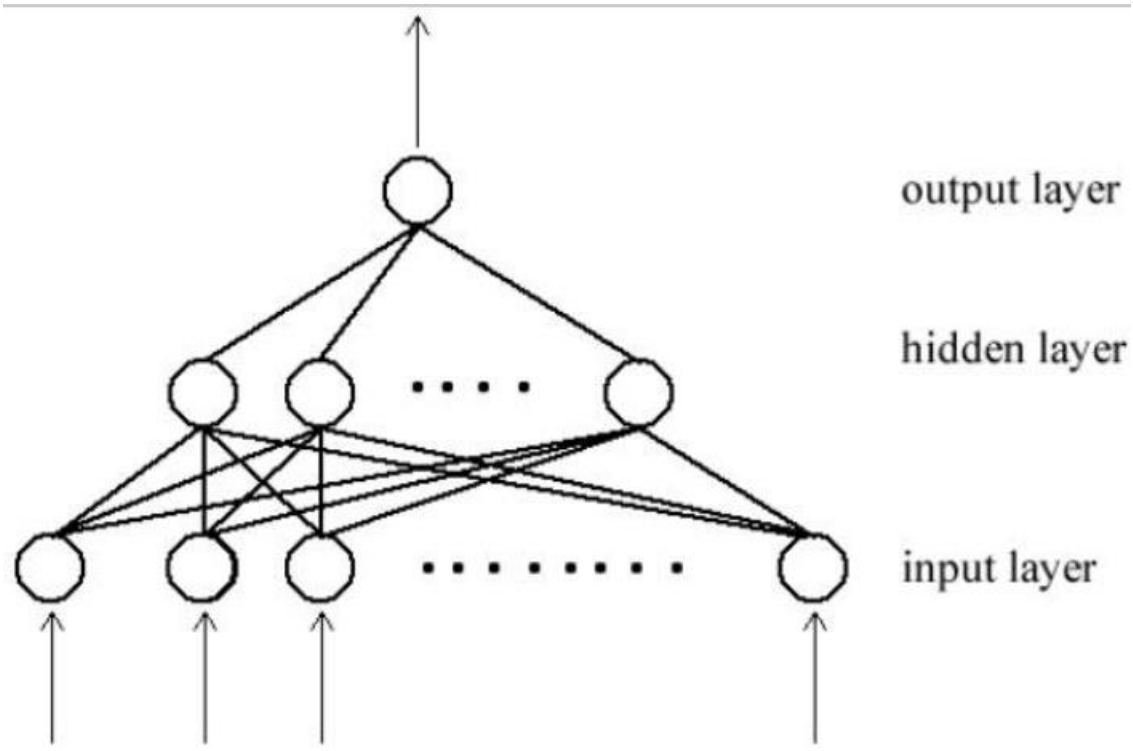


Figure 1.7

- **Optimization**

The performance of the deep CNN can be improved (or optimized) by training the network on a large dataset. Training involves finding the parameters θ of the model that significantly reduce a cost function J(θ). Gradient descent is the widely used method for updating network parameters through a back-propagation algorithm. Optimization can be done per single sample, subset, or full set of the training samples. Thus, stochastic, mini-batch, or batch gradient descent, respectively. Today, many optimization algorithms for deep learning use mini-batches and it is now common to just call them stochastic methods.

Stochastic gradient descent (SDG) comes with few notable challenges. Choosing an appropriate learning rate can be difficult. A learning rate that is too small leads to very slow convergence (tiny updates to the model parameters) and, at the same time, too large will result in undesired divergence behavior in the loss function. All of the parameter updates are based on the same learning rate, disregarding the fact that some of the features might have higher frequency than other. Another key challenge is that optimization can be trapped in sub-optimal local minima or saddle points, especially for non-convex optimization.

Various variants of SDG have been proposed in the literature that address the aforenoted challenges. Momentum-based SDG methods can help in accelerating SDG in relevant direction, dampening undesirable oscillations in local optima. Adagrad addressed the issue of manually turning the learning by adapting the learning rate to the parameters, performing larger updates for infrequent parameters as compared to frequent ones. However, Adagrad suffers from monotonically decreasing learning rate to a point at which the algorithm stops learning. Adadelta, RMSprop, and Adam addressed the shortcomings of Adagrad by dividing the learning rate by an exponentially decaying average of past gradients.

- **Loss Function**

In machine learning, the loss function is used to test whether a specific algorithm for data models is provided. If the output is far from the real value, the loss will be much higher and lower if the predictions are closer to real values. The main goal of training a neural network is to minimize the loss (or cost) of network activity as much as possible and, at the same time, to ensure that the network integrates seamlessly with intangible data.

The choice of cost function depends on the location of the problem, whether it is a separation or deceleration problem and the choice of the output unit. Most image classification algorithms use SoftMax Loss, which has a combination of SoftMax and CE loss or log loss. The SoftMax function generates distribution opportunities over the number of output classes provided,

while the loss of CE takes chances to predict and punishes reliable but inaccurate predictions. Class inequality is one major problem in medical analysis, where one class will have fewer cases than others. For example, a brain tumor plays a small part in comparison to healthy tissue. As a result, class dividers will tend to favor class divisions. One way to deal with such a dilemma is to redress the inequalities of the class. Some operations have raised a loss function based on the Dice coefficient. Ronneberger et al. has proposed weight loss for CE, which gives more value to other pixels in training data.

- **Parameter Initialization**

Algorithms for developing in-depth learning algorithms are very repetitive in nature, thus requiring the user to specify the first basic location of the algorithms. Choosing a startup will affect how quickly the learning can come together if it does not work at all. Intensive research has shown that a carefully chosen startup program significantly improves the level of integration, while gradient-based improvements from random startups can be trapped near bad solutions.

The study proposed a standard start-up scheme (Xavier implementation), which ensured that weight lifting should not get too small or too large, thus reducing space filling and perishable gradients, thus improving coherence. This method was later developed to make the best use of Relu or PRElu with more advanced models.

- **Hyperparameter Tuning**

Hyperparameters are parameters that are supplied by the user to control the algorithm's behavior before training commences, such as learning rate, batch size, image size, number of epochs, kernel size etc. While the learning algorithms do not adapt these parameters, their choice has varying effects on the resulting model and its performance. The majority of the works studied in this review set their hyperparameters manually or perform a grid search while using the validation set. However, these approaches will become impractical when the number of hyperparameters is large and they rely on human expertise, intuition, or guessing. As a solution to these challenges, automated approaches, like AutoML and Keras Tuner, are beginning to gain much attention.

- **Regularization**

Regularization is a technique for improving the performance of a machine learning algorithm on unseen data. It is a way of reducing over-fitting on training set. Over-fitting occurs when the gap between the training error and test error is too large. When that happens, the model performs well on training data, but poorly on previously unseen data. There are various techniques that can be employed in order to reduce the generalization error, such as reducing the model capacity, which is, reducing the number of learnable parameters in the model; adding L2 or L1 weight decay regularization term to the cost function to force the model to only take small weight values; introducing early stopping whenever the model performance stops improving on validation dataset; randomly dropping out (skipping) the output of some units during training. The last approach is one of the most effective and most commonly used technique, mainly because it is computationally inexpensive and prevents interdependent learning amongst units.

Batch Normalization can also be used as a regularized by ensuring that the distribution of non-linearity inputs remains more stable as the model trains, thereby improving the training of the model.

Training a machine learning model with more data is the best way to reduce the generalization error. However, in the medical domain, acquiring a training dataset is time-consuming, more expensive, and requires highly trained personnel to annotate ground truth labels. Data augmentation can increase the dataset and reduce over-fitting by flipping, applying small rotations, warping, and using the non-rigid deformation transformation of images. However, great care must be taken when performing transformations of the medical image dataset since the patch's label is determined by the center of pixel. Some recent works used generative models that include variational autoencoders and generative adversarial networks to act as additional regularization that deals with data scarcity.

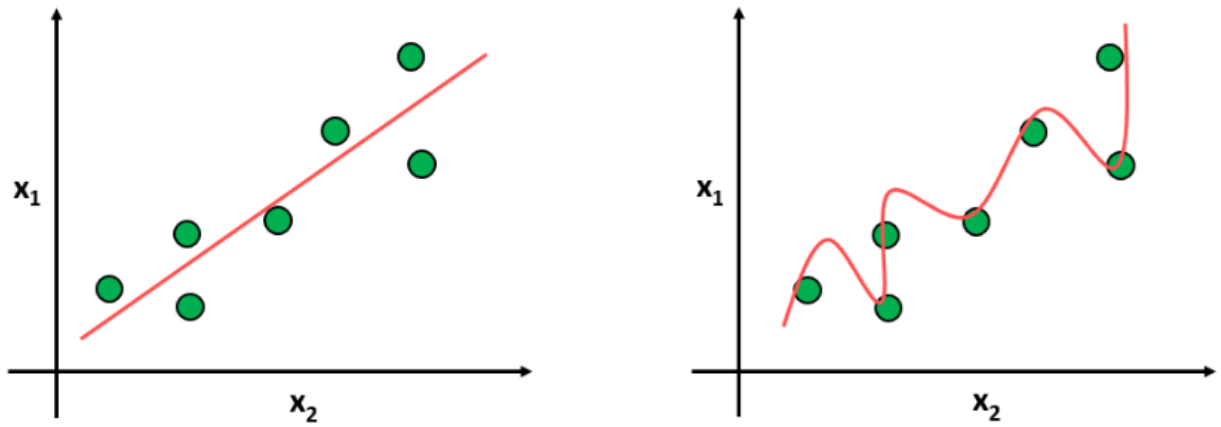


Figure 1.8

Chapter 2
Literature Review

This section of the paper throws a light on different research works and their approach to the problem with methodologies. There is a moderate growth in brain tumor cases all around the world. Researchers have been experimenting with a wide range of approaches and algorithms to predict brain tumor with more accuracy. There is a small comparison table given below for different years telling us the reviews done in 2016, 2017, 2018 and 2019.

In these books, the authors introduce an automated system of targeted cropping, non-fixed square division to match hands effects. This isolation of the machine-controlled plant will help alleviate the difficulties associated with personally analyzing brain tumors. this can speed up the process of analyzing the brain image, improving the outcome of the appointment, and building an accurate diagnosis of the disease by assessing progressive growth.

In this section, among the methods of plant classification considered within the literature; regional growth, machine learning, and in-depth learning based on the strategies to be evaluated for diagnostic testing, pre-processing, feature extraction, classification formula, and efficiency.

Sr. No.	Title	Year	Review
1.	Research on feature extraction of tumor image based on convolutional neural network.	2019	This paper proposes to use the local binary mode and the convolutional neural network based on the rotation of the image to cut and separate the image, and then extract the element to achieve the purpose of the separation.
2.	An Automatic Brain Tumor Image Segmentation Method Based on the U-net model.	2018	A basic U-Net model with a valid parameter structure is proposed. Worth to be separated. 1x1 is a flexible layer, added to the basic model to re-integrate the top layer features, enrich the extracted features, reduce network

			parameters, and improve segmentation results.
3.	DRINet for Medical Image Segmentation.	2018	Convolutional neural networks (CNNs) have recently changed the spectrum of classification of medical images over the past few years. In this DRINet novel paper for CNN architecture are proposed that reduce the hassle of learning different features in images.
4.	Interactive Medical Image Segmentation Using Deep Learning with Image-Specific Fine Tuning.	2017	A collaborative framework is proposed when a binding box is made in the plant area with a well-organized segment-based arrangement, which further helps to locate the tumor region. The 2015 Brats Data Set was used in this study. BIFSeg gets a better result.
5.	Advanced Brain Tumor Segmentation from MRI Images.	2016	Various classification methods have been discussed including the Fuzzy C-means method, the classification using the regional augmentation method and the Genetic Algorithm Based Method. Image Distribution with the help of a computer can improve the efficiency of a tumor detection.
6.	Watershed Algorithm based DAPP features for Brain Tumor Segmentation and Classification.	2016	In this study the watershed segmentation algorithm is used in images to separate a tumor from the rest of the brain. The Watershed Dynamic Angle Projection Technique presents. Abnormal brain image is detected, and research is being done successfully with the BRATS database.

Table 1

- **Shallow and Region-Based Unsupervised Machine Learning Technique**

One of the most widely used methods of distinguishing image-driven image applications is regional-based separation. Image squares measure a gaggle of connected pixels that meets the conditions of bound homogeneity, such as values of limb stiffness, shape, and texture. in the region-based section the image is divided into different regions so that the specified area is easily accessible. The area-based component takes into account the values of the poles, such as gray matter and contrast, and the approximate area of the pixels, such as the Euclidian distance and the regional interaction in pixel collection. In tumor classification, regional growth, and a host of square algorithms are standardized measurements commonly used in the basic classification process.

The K-means bunch is an unreadable degree reading formula for the corresponding degree and is usually a general category of area of interest from the rest of the image. K-means have been extensively tested for tumor differentiation and have shown acceptable accuracy. The need for a line machine, ease of use for large databases, adaptability to new models, and secure square integration measure the number of benefits that produce a K-mean fashion segmentation formula. However, k-means suffers from an incomplete definition of a plant region, choosing the first weight center is wrong, and sensitive to outsiders. as a result of these limitations various solutions are developed, including, equitable distribution of first-class institutions (k-means ++), breeding ik-methods and different collective methods, flexible implementation of collective institutions, such as k-flexible, modified k-adaptional (MAKM) methods, and a bar chart primarily based on k-methods.

Fuzzy c-means that it works with share membership values for all pixels in the image compared to collective centers that calculate specific terms of similarity. In fuzzy c-means (FCM) bulk items will be more than one group based on its membership level. Therefore, in such a sensitive group process, photographic pixels will capture multiple collections. As a result, compared to hard-welding techniques such as k-means, FCM performs relatively high compared to soundless images. However, in medical imaging such as brain magnetic resonance imaging that is immediately attacked by unknown sounds, the performance of FCM is severely affected. Various studies are being conducted to improve FCM restrictions.

In the growing region of plant differentiation, tissue and plant regions are a square measure divided into a matter of similarity based, such as homogeneity, texture, sharpness, and gray levels. The process begins with selecting first degree seeds based on the methods described earlier. Then, the neighboring square pels weigh more according to the seed pixel. A region that grows in phases will essentially divide regions with similar structures and regions geographically. However, it is sensitive to noise and is influenced by the condition of similarity. Therefore, it will find itself with disconnected circuits and leads to a hole in the middle of the separated zone. moreover, finding the first reliable seeds is not a straightforward task. Growing region and traditional supervised machine learning based primarily on plant classification techniques presented in a square measure of literature in a nutshell. The table shows the data imaging magnetic resonance data used in the test, the center of the big data format strategies, the targeted performance, hence the performance of the segment.

- **Supervised Shallow Machine Learning Algorithm**

Machine-based classification methods based on machine learning also corrected the partial reversal of the image to partial duplication of the plant. The input vector for these monitored study models includes a variety of output options, and the output may be a vector for the desired classification categories. In the implantation of the plant, wherever the square-scale circuits of the plant are usually distributed throughout the image, partial separation rather than the traditional methods of square measurement is generally preferred. Therefore, standard machine-readable algorithms are used to separate the tumor in a scanner to detect the appearance of a magnetic field in the head.

In this section, the most appropriate textbooks on plant-based learning strategies, such as artificial neural network (ANN), vector support (SVM), random forest (RF) square measure of information acquisition, pre-processing factor. output algorithms, segmentation model, and whether post-processing processing is mandatory or not.

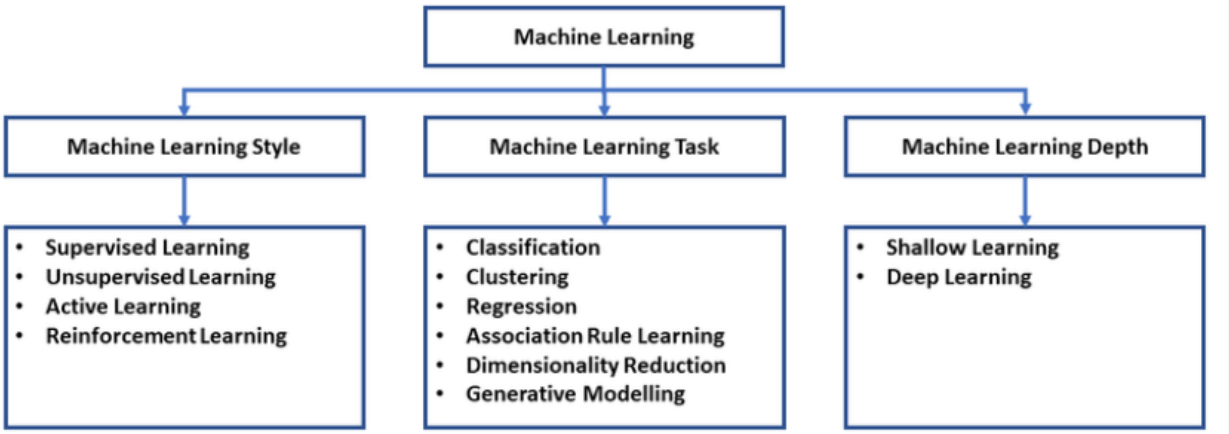


Figure 2.1

- **Some of the literature reviews mentioned in Table 1.**

Research on feature extraction of tumor image based on convolutional neural network. (2019)

The medical image is the main carrier and important carrier of a modern definition of another medical language. Little by little plays an important role in going to the medical field with Accurate expression and definition of purpose. Medical image analysis is a modern technology for image analysis covering mathematical model, practical wisdom, integrated medical photography, digital imaging processing and some multidisciplinary.

Defining whether a plant is dangerous or dangerous according to in the picture and it is a topic that the medical community has it has not stopped. The first image release feature split the image, then remove the texture elements from photo after splitting. There is a lot of texture in trees, fabrics and clothes. These invisible objects are taken from their structure. Of course, medically, due to abscesses, it is not as accurate as trees or clothing. It can only be cut into tumors, depending on the imaging machine to turn its texture into medical imaging such as CT or MRI. Texture is a feature of the basic and most useful information in an image. It is an important parameter for describing image content, as well academic research has gradually become a new topic.

In the idea of typing, the texture of the image is the best rotation, translation, and consistency of scale, i.e. regardless of image shape, position, shape and the size of the texture is permanent. With the release of the text element, the most commonly used methods of statistical analysis, model

analysis, structural analysis and spectrum analysis. The first successful application for image element release texture features were in 1972, when Sutton and Hall identified normal and irregular lungs. Taleb Ahmed uses typing theory to produce a CT scan of the bone, and finds a picture difference between a patient with osteoporosis and osteoporosis. Normal human bone. In addition, a patient with osteoporosis can be discriminated against. Ganeshan et al. analyzed MRI image of the liver and obtained the diagnosis of patients with cirrhosis. In 2014, Sun et al. improved face recognition system using the extracted feature the figure, with an error rate of only 3.55%, reached high global impact. Similarly, Chen et al. he also learned face to face recognition.

An Automatic Brain Tumor Image Segmentation Method Based on the U-net model. (2018)

A major challenge in brain tumor planning and dose evaluation is tumor level determination. Rare magnetic resonance imaging (MRI) techniques have emerged as an advanced diagnostic tool for brain tumors without ionizing radiation. The actual separation of the brain tumor length from the 3D MRI volume is a time-consuming task and the performance depends largely on what the user is doing. In this context, a completely reliable method of differentiating the brain tumor is needed to accurately measure the quality of the tumor. In this study, we propose a fully automatic method of brain tumor classification, which is developed using in-depth conversion networks based on U-Net. Our method was analyzed in the Multimodal Brain Tumor Image Segmentation (BRATS 2015) data sets, which contained 220 advanced brain tumors and 54 low-grade tumor tumors. The opposite confirmation has shown that our approach can get a promising split well.

DRINet for Medical Image Segmentation. (2018)

Convoluted Neural Networks (CNN) has changed the whole medical image predictions in the last few years. UNET Architecture is very well-known CNN architectures, for semantic segmentation. It has achieved a huge success in many diff. medical image segmentation. The U-Net architecture consists of standard convolution layers, pooling layers, and upsampling layers. These convulsion levels learn the representative properties of the input images and create segmentation based on the properties. However, while the differences between the different categories in terms of density, location, shape and size are unclear, the features learned by the standard convolution levels are not specific. In this article, we propose a new CNN architecture called Dense-Res-Inception Net (DRINet) that addresses this challenging problem. The proposed DRINet consists of three blocks, a convoluted block with close connections, a deconvolution block with the remaining start modules,

and an unpulling block. Our proposed architecture outperforms U-Net in three different challenging applications: multiclass segmentation of cerebrospinal fluid (CSF) on CT images of the brain, multi-organ segmentation on abdominal CT images, and MR on segmentation.

Interactive Medical Image Segmentation Using Deep Learning with Image-Specific Fine Tuning. (2017)

Convoluted Neural Network (CNN) have achieved the art of performance for automatic medical image breakup. However, they did not show sufficiently accurate results for the clinical requirement. Furthermore, they are limited by the lack of image-specific acquiesce and the lack of generalizations for previously unseen object classes (aka zero-shot learning). To address these issues, we propose a novel Deep Learning-Based Interactive Segmentation Framework by incorporating CNN into bounding boxes and scribble-based segmentation pipelines. We propose image-specific fine tuning to adapt the CNN model to a specific test image, which can be either unmanaged (without additional user interactions) or monitored (with additional scribbles). We also propose weight loss work for fine tuning considering network and interaction-based uncertainty. We have applied this framework to two applications: 2-D segmentation of multiple organs from fetal magnetic resonance (MR) slices, where only two types of these organs for training and 3-D segmentation of the brain tumor core (excluding edema) and whole brain from different MR sequences. Tumors (including edema), where only one MR sequence was noted for tumor core training.

		PC-Net	PC-Net+CRF	BIFSeg(-w)	BIFSeg
Dice (%)	TC	82.66±7.78	85.93±6.64	85.88±7.53	**87.49±6.36***
	WT^	83.52±8.76	85.18±6.78	86.54±7.49	**88.11±6.09***
T_m(s)	TC	-	**0.14±0.06***	3.33±0.86	4.42±1.88
	WT^	-	**0.12±0.05***	3.17±0.87	4.01±1.59

TC: Tumor core in T1c, WT: Whole tumor in FLAIR.

Table 2

Python

Python Dynamic Systems is a highly developed, deciphered by Semantics. Its elevated level of specific data structures, with an attractive build and assurance, is involved in rapid performance improvement, as well as the relative use as a writing or paste language to connect existing parts together. Python is clear, simple teaching of the structure of the language emphasizes undeniable and appropriate quality reduces program funding costs. Supports parts and packages, such as System Protection Renew and Code Reuse.

The language comes with a large standard library that covers fields such as string processing (regular expression, Unicode, calculation of differences between files), Internet protocols (HTTP, FTP, SMTP, CGI programming), software engineering (unit testing, logging, profiling). Is. Python code), and the operating system interface (system calls, filesystems, TCP / IP sockets).

TensorFlow

TensorFlow is an environment which offers different types of useful libraries so you can choose the right one for your needs. Build and train models by using the high-level Keras API, which makes getting started with TensorFlow and machine learning easy.

If you need more flexibility, eager execution allows for immediate iteration and intuitive debugging. For large ML training tasks, use the Distribution Strategy API for distributed training on different hardware configurations without changing the model definition.

NumPy

NumPy is a python library which basically deals with the mathematical functions, generation of different/random numbers, Fourier transformation, algebraic calculations and many more. NumPy is that type of package which is used for the basic and many other scientific computations in Python. It is that type of library which provides a multidimensional array objects.

Epoch

Epoch is a time frame. It is also very well known as POSIX time, Unix time. It basically indicates the number of seconds gone since 1st Jan 1970 in which the leap is not counted. It contains 10 digits and it can also highlight all the time zone at once.

Machine learning (ML)

Machine learning enables PCs to learn without pointing in the right direction (Arthur Samuel, 1959) This is the area under the programmatic structure. Can read machine learning data in all aspects of scale development and create opportunities. Such controls follow changed rules, but they can also make data predictions or decisions based on the data. Build models from test inputs. Machine learning is done when it is inconceivable for master minding and programming calculations. Models include spam filtering.

Deep Learning Approach

Associate in Nursing Tomography Brain Scan Pediatrics Machine learning algorithms under shallow supervision have also led to true growth in identifying brain tumors in their various forms, while associate in nursing tomography scans are still problematic in identifying brain abnormalities. The unit of parts of this problem is primarily due to the field of interest discovery, and the victimization of descriptive material is not economically due to historically hand-made specialty extraction algorithms. These unskilled techniques arise mainly due to the complex structure of the advanced anatomy of the human brain and hence the high-complex nature of the

human brain.

Unlike shallow machine learning techniques, deep learning techniques belief on learning knowledge representations and ranked feature learning. In deep learning-based algorithm tumor segmentation, the deep learning models found the descriptive data that optimally shows totally different brain tumors. This nature of deep learning algorithm transforms the tumor segmentation from handmade feature-driven features into data-driven downside . From all the deep learning models, a convolutional neural network (CNN) is widely used in tumor segmentation tasks, and a considerable result is achieved.

In the all reviewed literature, there area unit differentiation within the techniques used for the segmentation of brain tumors. The differentiation consists: (i) the dataset used for segmentation and growth sorts , (ii) the knowledge augmentation algorithms and imposed pre-processing (iii) whether yes or not the region of interest segmentation was used as a earlier steps within the segmentation, (iv) either or not a custom-designed deep learning or pre-trained structure is employed.

The data-set contains neoplasm, glioma, and pituitary tumor sorts scanned beside the 3 anatomical views, i.e., axial, sagittal, and lei. the photographs were pre-processed victimization techniques, like social control and re-sizing. additionally, pictures within the data-set area unit increased with 90o rotation and vertical flipping to extend the coaching data-set. what is more, they used a custom-designed CNN model trained with Adam optimizer with a mini-batch size of sixteen and tested with 10—fold cross-validation. The weights of the convolution layers area unit initialized employing a Glorot initialization. The model performance was live victimization sensitivity, specificity, accuracy, precision, recall, and F1-score. The sensitivity for neoplasm, glioma, and pituitary is eighty nine.8%, 96.2%, and 98.4%, severally. The specificity of the model for neoplasm, glioma, and pituitary is ninety.2%, 95.5%, and 97.7%, severally. what is more, the models' overall accuracy, average preciseness, average recall, and F1-score area unit ninety five.4%, 94.81%, 95.07%, and 94.94%, severally.

Chapter 3

System Development

3.1 Design and Algorithms

For this project we have using various python algorithms to make the result more accurate and give the desired output CNN could be a powerful algorithmic program for image process. These algorithms are presently the simplest algorithms we've for the machine-driven process of pictures. several corporations use these algorithms to try and do things like distinguishing the objects in a picture.

Images contain knowledge of RGB combination. Matplotlib is accustomed import a picture into memory from a file. The PC doesn't see a picture, all it sees is Associate in Nursing array of numbers. Color pictures are hold on in three-dimensional arrays. the primary 2 dimensions correspond to the peak and dimension of the image (the variety of pixels). The last dimension corresponds to the red, green, and blue colors gift in every constituent.

Three layers of CNN

As we already know each and everything about CNN, there are basically three layers in CNN which are the main pillars for the execution of the program. These are:

- **Convolutional Layer**

  In a normal neural network, all incorporated vegetative cells are connected in a series of hidden secretions. In CNN, only a small area of vegetative layer cells of the input layer are connected to the hidden layer of the neuron.

- **Pooling layer**

  Pooling layer or Blend layer is used to reverse the feature map location. They will be multiple layers of opening and consolidation within CNN's hidden layer.

❖ **Connected layer**

Fully Connected Layers type a few previous layers within a network. The input is in a fully connected layer that the output from the Final Addition or Conversion, which is a program that has been eaten and inserted into a fully connected layer.

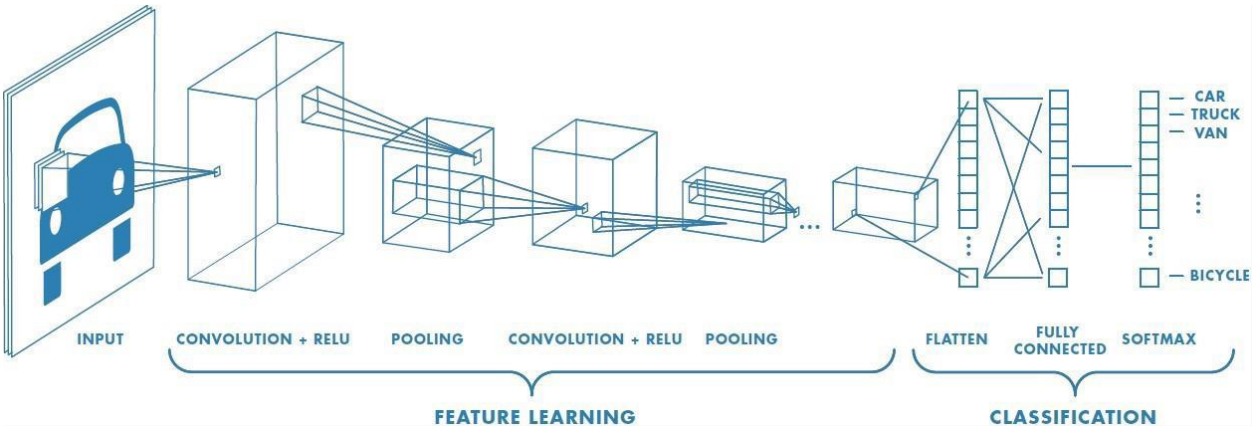


Figure 3.1 Structure of CNN

As we know that for the image detection, we have to use such type of algorithms which are easy to use, easy to understand and which can provide great results. The main algorithm used in making of the project is K-mean clustering algorithm. Before coming into the explanation of what is K-mean clustering algorithm, let's see some useful stages of our model, i.e. Raw image input, image processing, ground truth image, data set, training & testing data and modified CNN model. Now let's have a look on the performance behavior of the model.

First of all, there is a row MRI image/images which is to be executed. After that, we have to convert the image DICOM (Digital Imaging and Communication in Medicine) to another reliable format. The converted image gets to main part which is image processing where the row image gets processed in such a way that it could provide us with an accurate result. Then the processed image is compared with the ground truth images and then the data set is compared with them. After that we have to train and test our data set so as to get accuracy as high as possible. After that the CNN is applied so as to crop the image in such a way that we will be able to see the tumor clearly. It provides us with the output which shows us the images having tumor and the images having no tumor.

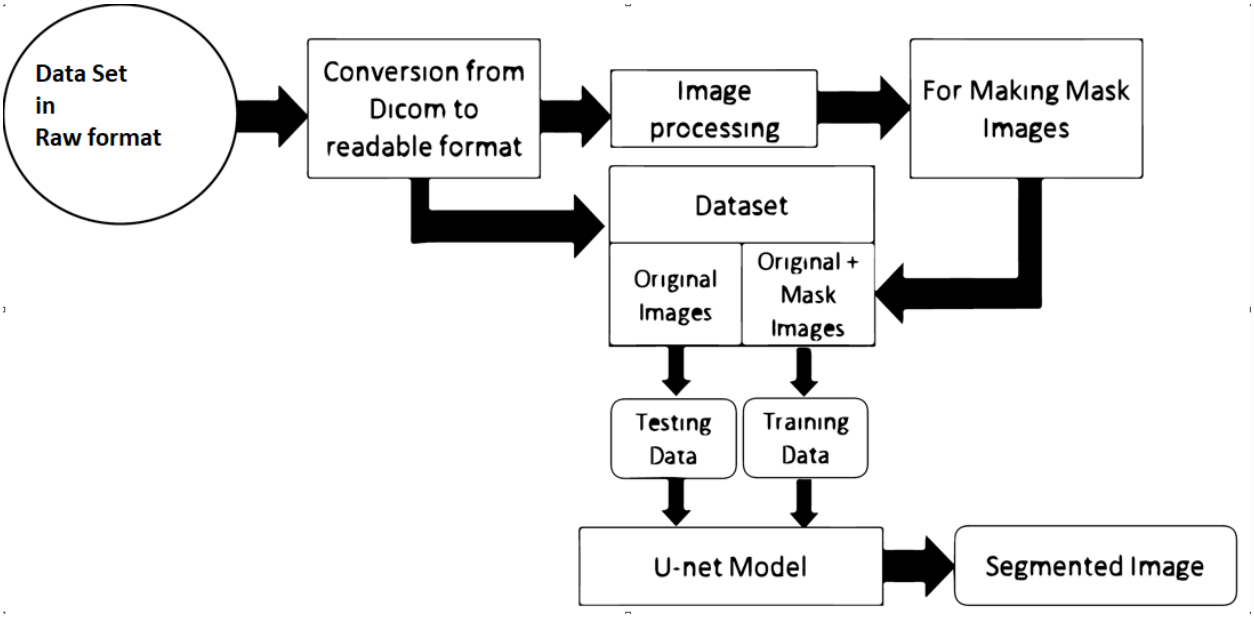


Figure 3.2 Flow Chart

Now let's have a small conversation about K-mean clustering algorithm.

The K merging algorithm comprises inches and repeats until the correct centroid is found. Also known as a flat clustering algorithm. The number of clusters found in the data path is defined by the letter 'K' by K-method. In this method, data points are grouped in such a way that the total number of distances between the data points and the centroid is as small as possible. It is important to note that the reduced diversity between groups leads to very similar data points in the same collection.

Similarly, as the definition of K-mean clustering algorithm we have also applied it on the MRI images in such a way that we could get a clear view of the location of the tumor inside the brain. The K-mean algorithm works in such a way that it divides the MRI image pixels in such a way that after applying the CNN we could clearly differentiate between the images which are having tumor and the images which don't have any trace of tumor in it.

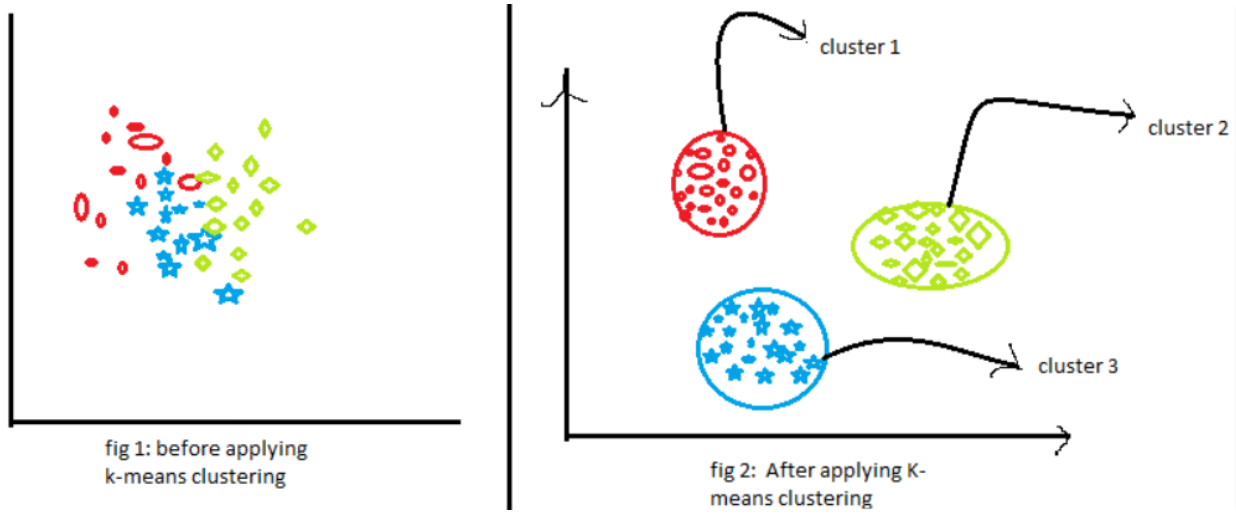


Figure 3.3

2.1.1 Data Augmentation

It is a strategy that allows experts to maximize the diversity of data available from training models, without having to collect new data. Data augmentation techniques such as cropping, scrolling, and horizontal scrolling are often used to train large number of networks.

The dataset which is being used in the implementation of the project is small and to create more images I have used data augmentation technique. We could also solve the data imbalance issue using data augmentation. In the data set 61% of the data belongs to the tumorous class. In the image 3.4 we can clearly see how we are able to create more number of images using data augmentation.

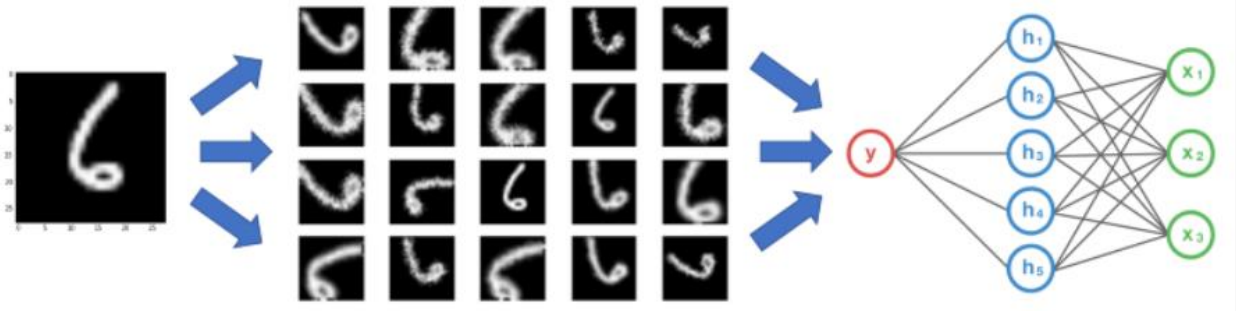


Figure 3.4

The whole process can be divided into the following categories:

● **Database acquisition:**

The data collection for a specific application is called a data set. As a machine learning model it relies entirely on data so we have to make sure we get the best data we can.

● **Importing libraries:**

Some of the python libraries we need for our specific project must be integrated. These are called foreign libraries. In particular there is the use of three python libraries

Numpy: Used for code coding.

Panda: The most common and useful library. It is used to manipulate data and analyze data.

Matplotlib: Used to edit various charts using python. It works with the help of a small library pyplot.

● **Importing data sets:**

The step in which we import the required databases of the model.

2.2 Model Development

Database collected at Kaggle.com. Kaggle is a company owned by google, a web community of information scientists and ML staff. Kaggle allows customers to discover and download recording units, find and build models with an online recording science environment, collaborate with other data scientists and participate in competitions to solve any data science issues. We are convinced that the database is original and authentic because kaggle is a reliable website used by over a million users worldwide.

Data set

We have basically used an image dataset because CNN works on the image dataset as we already know. Here are some of the sample images of the data set on which the whole process is going on.

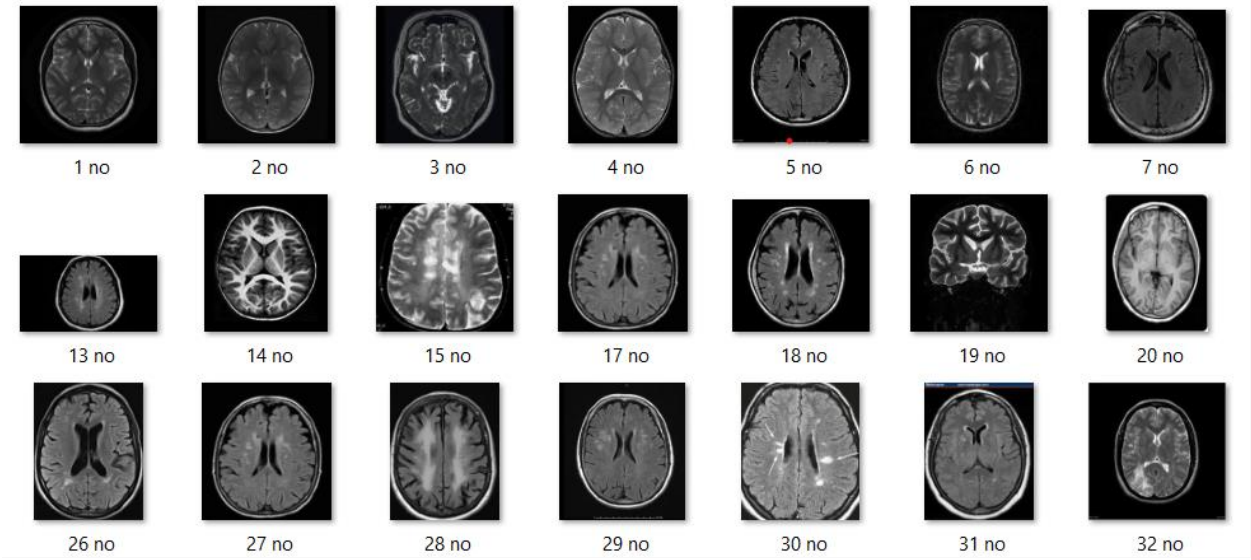


Figure 3.5 Dataset having no tumor

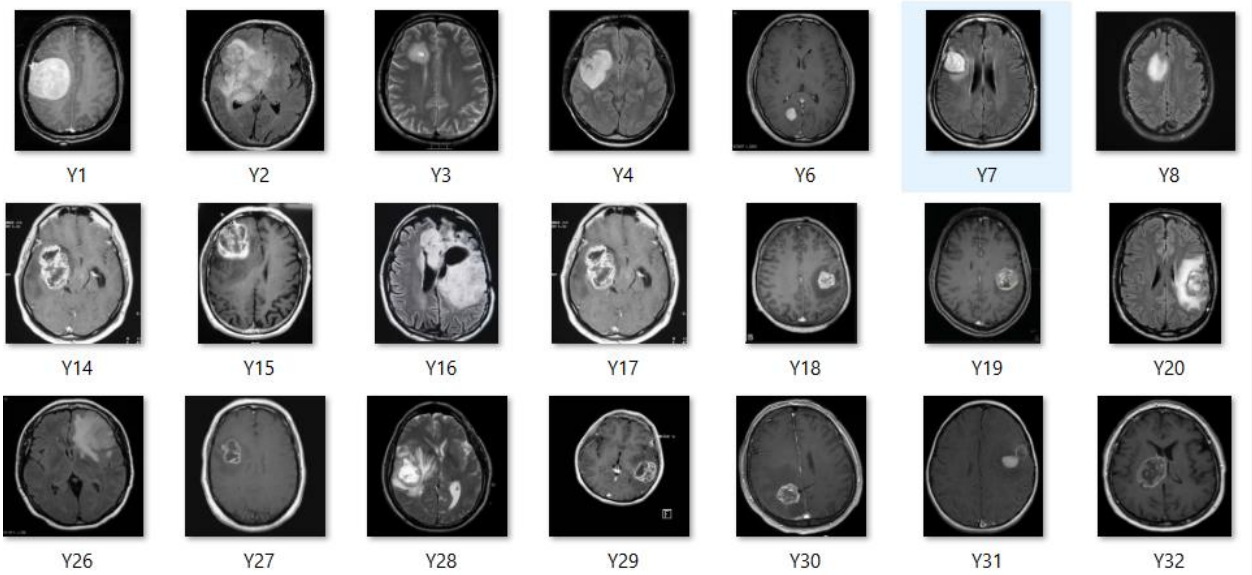


Figure 3.5 Dataset having tumor

2.2.1 Data Preprocessing

For every image, the following preprocessing steps were applied:

1. Crop the part of the image that contains only the brain (which is the most important part of the image).

2. Resize the image to have a shape of (240, 240, 3) = (image_width, image_height, number of channels): 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 0-1.

2.2.2 Data split

The data was split in the following way:

1. 70% of the data for training.

2. 15% of the data for validation.

3. 15% of the data for testing.

2.2.3 Neural Network Architecture

This is the architecture that I've built:

![Neural Network Architecture](convnet_architecture.jpg)

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 with a pool size of (2, 2).

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 activation layer.

5. A Max Pooling layer with f=4 and s=4.

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).

These packages mentioned above were used in their latest up to date editions. The code works properly and would not cause any issue until any further updates in them.

Here are some of the quantitative results of various forms of our planed structure:

Method	Dice score (mean)			Sensitivity (mean)		
	Enh	Whole	Core	Enh	Whole	Core
Two-route CNN	0.2531	0.2796	0.2143	0.2456	0.2569	0.2007
Global route CNN + Attention mechanism	0.3128	0.3410	0.3025	0.3343	0.2947	0.2896
Local route CNN + Attention mechanism	0.3412	0.3671	0.3625	0.3356	0.3819	0.3808
Two-route CNN + Attention mechanism	0.4136	0.3754	0.3988	0.3910	0.3951	0.3822
Global route CNN + Preprocessing	0.7868	0.7916	0.7867	0.7426	0.7965	0.7448
Local route CNN + Preprocessing	0.8602	0.8343	0.8516	0.8751	0.8569	0.8485
Two-route CNN + Preprocessing	0.8756	0.8550	0.8715	0.8941	0.9036	0.8512
Proposed method	0.9113	0.9203	0.8726	0.9217	0.9386	0.9712

Table 3 Quantitative results

Chapter 4

Performance Analysis

4.1 System Configuration

4.1.1 Software Requirements

- Windows 7, 10 ,11
- Python 3.8 or above
- PIP
- NumPy 1.22.3

Python:

Python is a translated, high-quality, common-sense, language developed by Guido Van Rossum and first released in 1991, Python's design philosophy emphasizes the readability of the code and its remarkable use of Whitespace. Its language-building and object-oriented approach aims to help programmers write clear, logical code for small and large projects. Python is spontaneously written and garbage collection. It supports multiple planning paradigms, including process, object-oriented, and functional program.

Pip:

Pip is nothing but package management system which is used to install and manage software packages which are written in Python language.

NumPy:

NumPy is a standard purpose-packed package. It provides high performance for multidimensional object, and tools for working with these components. It is a basic computer science package via Python. Contains a variety of features that include the following:

 Powerful N-dimensional array

 Complex (broadcast) functions

 C / C ++ and Fortran code integration tools

 Useful line algebra, Fourier transform, and random number power

Pandas:

Pandas is a popular python library used for data analysis. Provides excellent performance with background source code written in C or Python only. We can analyze data on pandas by

1. Series

2. Data frames

Anaconda:

Anaconda is a free distribution and open source source of Python and R computer programming science that aims to simplify package management and use. Package versions are managed by the conda of the package management system. Anaconda distribution includes data science packages suitable for Windows, Linux, and macOS. Anaconda Distribution comes with 1,500 PyPI selected packages as well as a conda package and a visual environment manager. It also includes a GUI, Anaconda Navigator, as an alternative to the command line interface (CLI).

Jupyter Notebook:

Anaconda Distribution comes with 1,500 PyPI selected packages as well as a conda package and a visual environment manager. It also includes a GUI, Anaconda Navigator, as an alternative to the command line interface (CLI). The Jupyter Notebook document is a JSON document, which follows a version schema, and contains a list of input / output cells that can contain code, text statistics, episodes and rich media, often ending with “. pynb "extension.

Tensorflow:

Tensorflow is a free software library with an open source of data flow and a diversified system across a wide range of functions. It is a mathematical library, and is used in machine learning applications such as neural networks. It is used for both Google search and production.

Keras:

Keras is an open source neural-network library written via Python. It can work on TensorFlow, Microsoft Cognitive Toolkit, R, Theano, or Plaid ML. It is designed to enable rapid exploration through deep neural networks, focusing on ease-of-use, modular, and scalability. Keras contains a wide range of commonly used neural-network building blocks such as layers, objectives, unlocking functions, layouts, and a host of tools to make image processing and text data easier to make the required code easier to write in-depth neural network code.

OpenCV:

OpenCV (open source computer view) is a library of highly targeted editing activities for real-time computer vision. It was originally developed by Intel, later supported by a willow garage and then by Itseez (later acquired by Intel). The library is a crossroads and is free to use under the BSD open source license. OpenCV supports specific models from in-depth learning frameworks such as TensorFlow, Torch, PyTorch (after converting to ONNX model) and Caffe with a detailed list of supported layers. Encourages Open Vision Capsules. which is a portable format, compatible with all other formats.

4.1.2 Hardware Requirement

For this project we have used the machine with the following specifications at the training time:

CPU:

The computer we used had the following specs:

Parameters	Specifications
CPU Model name	Intel core i5 8th Gen
CPU Frequency	1.60GHz to 1.80GHz
No. of CPU Cores	4
RAM	12GB
Disk Space	1TB

Table 4 CPU Specifications

These are the CPU specs of the machine we used to do computations. Most of the computational work mostly happens on the GPU. But CPU takes care of most of the preprocessing. The large amount of RAM did not put loads of pressure and made it easier for the whole dataset to be loaded in time and wedid not have to worry about any system crashes occurring.

GPU:

Parameters	Specifications
GPU	AMD Radeon 540
GPU Memory	4GB
GPU Core Clock	1046 MHz
GPU Memory Clock	1300 MHz
BIOS Date	2019/1/17
Cores	2
Available RAM	12
Disk Space	1TB

Table 5 GPU Specifications

4.2 Sample Code

- **Importing Libraries**

```
import tensorflow as tf
from keras.preprocessing.image import ImageDataGenerator
from tensorflow.keras.layers import Conv2D, Input, ZeroPadding2D, BatchNormalization,
Activation, MaxPooling2D, Flatten, Dense
from tensorflow.keras.models import Model, load_model
from tensorflow.keras.callbacks import TensorBoard, ModelCheckpoint
from sklearn.model_selection import train_test_split
from sklearn.metrics import f1_score
from sklearn.utils import shuffle
import cv2
import imutils
import numpy as np
import matplotlib.pyplot as plt
import time
from os import listdir

%matplotlib inline
```

- **Segment**

```
def crop_brain_contour(image, plot=False):

    #import imutils
    #import cv2
    #from matplotlib import pyplot as plt

    # Convert the image to grayscale, and blur it slightly
    gray = cv2.cvtColor(image, cv2.COLOR_BGR2GRAY)
    gray = cv2.GaussianBlur(gray, (5, 5), 0)

    # Threshold the image, then perform a series of erosions +
    # dilations to remove any small regions of noise
    thresh = cv2.threshold(gray, 45, 255, cv2.THRESH_BINARY)[1]
    thresh = cv2.erode(thresh, None, iterations=2)
    thresh = cv2.dilate(thresh, None, iterations=2)

    # Find contours in thresholded image, then grab the largest one
```

```
    cnts = cv2.findContours(thresh.copy(), cv2.RETR_EXTERNAL,
cv2.CHAIN_APPROX_SIMPLE)
    cnts = imutils.grab_contours(cnts)
    c = max(cnts, key=cv2.contourArea)


    # Find the extreme points
    extLeft = tuple(c[c[:, :, 0].argmin()][0])
    extRight = tuple(c[c[:, :, 0].argmax()][0])
    extTop = tuple(c[c[:, :, 1].argmin()][0])
    extBot = tuple(c[c[:, :, 1].argmax()][0])

    # crop new image out of the original image using the four extreme points (left, right, top,
bottom)
    new_image = image[extTop[1]:extBot[1], extLeft[0]:extRight[0]]

    if plot:
        plt.figure()

        plt.subplot(1, 2, 1)
        plt.imshow(image)

        plt.tick_params(axis='both', which='both',
                        top=False, bottom=False, left=False, right=False,
                        labelbottom=False, labeltop=False, labelleft=False, labelright=False)

        plt.title('Original Image')

        plt.subplot(1, 2, 2)
        plt.imshow(new_image)

        plt.tick_params(axis='both', which='both',
                        top=False, bottom=False, left=False, right=False,
                        labelbottom=False, labeltop=False, labelleft=False, labelright=False)

        plt.title('Cropped Image')

        plt.show()

    return new_image


ex_img = cv2.imread('E:\study material\Brain-Tumor-Detection-master\yes\Y1.jpg')
```

```
ex_new_img = crop_brain_contour(ex_img, True)
```

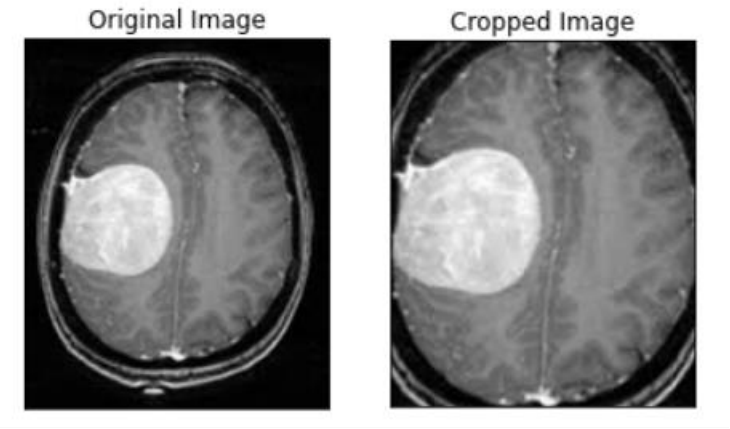


Figure 4.1 Gray Scale Images and Filtered images

- **Loading the data:**

The following function takes two arguments, the first is a list of folders of 'yes' and 'no' folders containing image data and the second argument is image size, and in all images in both directions and does the following:

1. Read the picture.
2. Trim the part of the image that represents the brain only.
3. Resize the image (because images in the database come in different sizes (i.e. width, length and # channels).
4. Use to make it normal because we want pixel values to be measured in the range 0-1.
5. Insert an image on the X and its label on y.

After that, Shuffle X and y, because the data commands (meaning that the same members contain the first part belonging to one class and the second part belong to another class, and we do not want that).
Finally, Replace X and y.

```python
def load_data(dir_list, image_size):

    # load all images in a directory
    X = []
    y = []
    image_width, image_height = image_size

    for directory in dir_list:
        for filename in listdir(directory):
            # load the image
            image = cv2.imread(directory + '\\' + filename)
            # crop the brain and ignore the unnecessary rest part of the image
            image = crop_brain_contour(image, plot=False)
            # resize image
            image = cv2.resize(image, dsize=(image_width, image_height), interpolation=cv2.INTER_CUBIC)
            # normalize values
            image = image / 255.
            # convert image to numpy array and append it to X
            X.append(image)
            # append a value of 1 to the target array if the image
            # is in the folder named 'yes', otherwise append 0.
            if directory[-3:] == 'yes':
                y.append([1])
            else:
                y.append([0])

    X = np.array(X)
    y = np.array(y)

    # Shuffle the data
    X, y = shuffle(X, y)

    print(f'Number of examples is: {len(X)}')
    print(f'X shape is: {X.shape}')
    print(f'y shape is: {y.shape}')

    return X, y
```

Now as we have used the augmented data for the generation of new images with the original images, we have to load that data also:

```python
augmented_path = 'augmented data/'

# augmented data (yes and no) contains both the original and the new generated examples
augmented_yes = augmented_path + 'yes'
augmented_no = augmented_path + 'no'

IMG_WIDTH, IMG_HEIGHT = (240, 240)

X, y = load_data([augmented_yes, augmented_no], (IMG_WIDTH, IMG_HEIGHT))
```

The output of the data augmentation is:

```
Number of examples is: 2065
X shape is: (2065, 240, 240, 3)
y shape is: (2065, 1)
```

As we see, we have 2065 images. Each images has a shape of **(240, 240, 3)=(image_width, image_height, number_of_channels)**

After loading the data augmentation part, now we have to load/plot the image:

```
def plot_sample_images(X, y, n=50):

    for label in [0,1]:
        # grab the first n images with the corresponding y values equal to label
        images = X[np.argwhere(y == label)]
        n_images = images[:n]

        columns_n = 10
        rows_n = int(n/ columns_n)

        plt.figure(figsize=(20, 10))

        i = 1 # current plot
        for image in n_images:
            plt.subplot(rows_n, columns_n, i)
            plt.imshow(image[0])

            # remove ticks
            plt.tick_params(axis='both', which='both',
                            top=False, bottom=False, left=False, right=False,
                           labelbottom=False, labeltop=False, labelleft=False, labelright=False)

            i += 1

        label_to_str = lambda label: "Yes" if label == 1 else "No"
        plt.suptitle(f"Brain Tumor: {label_to_str(label)}")
        plt.show()
```

This is the output after plotting the images:

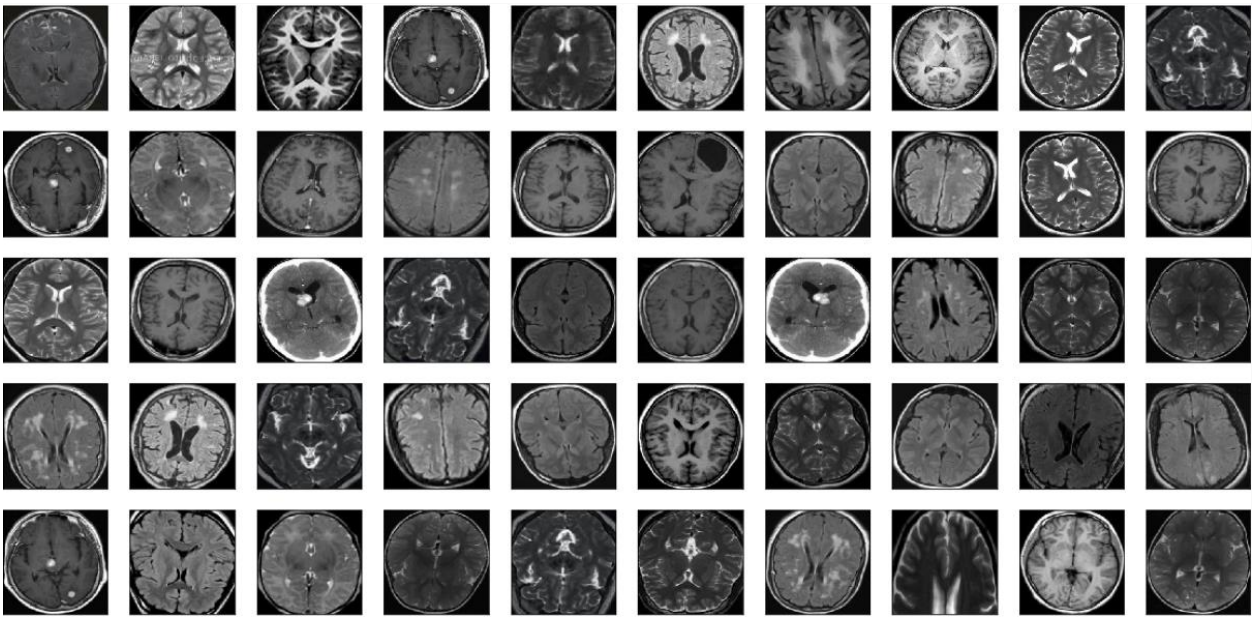


Figure 4.2 Brain tumor: no

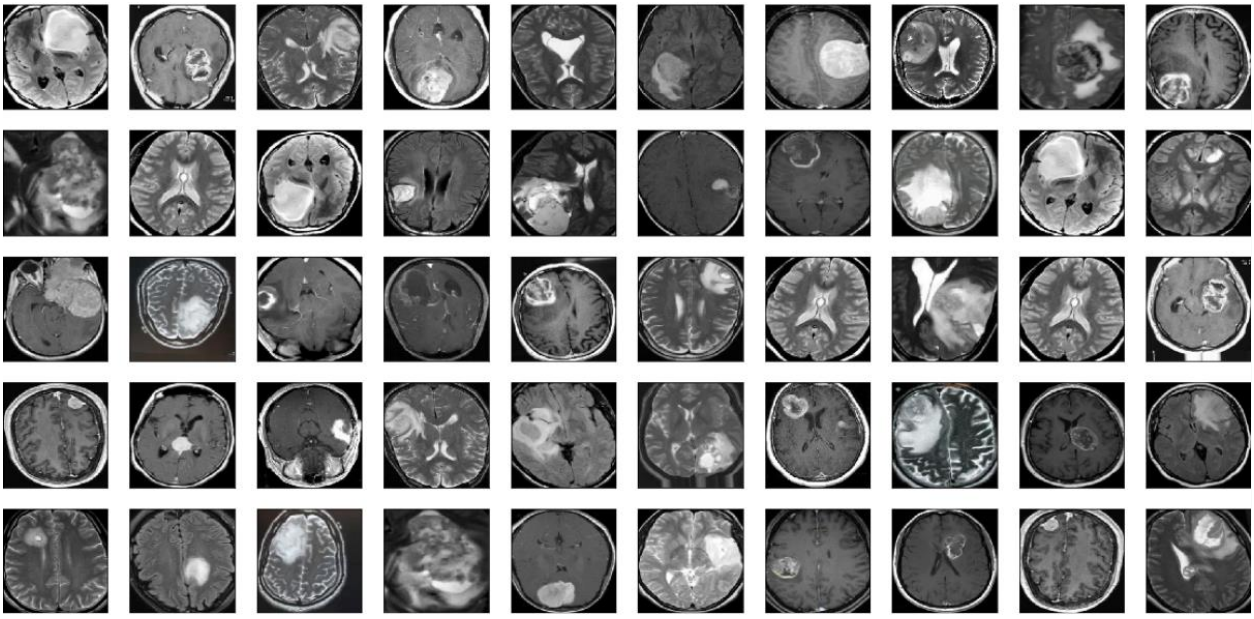


Figure 4.3 Brain Tumor: yes

After plotting the images, we need to build the model before the model training so that we could have an idea of what is going on and what will be the next step.

For building the model we have to use padding, alignment, maxpool and at last create the model. After that we have to define the image shape. The same is mentioned in the table below:

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

Table 6. Shape of the images

After performing these tasks, now we have to compile the model so that it could be easy to train the model.

- **Training the Model**

As we have already loaded the training data, now we only have to start the time for the model training. Here I have used 46 training dataset samples instead of 1445 so as to utilize the time or for the time management. The accuracy also depends upon the training dataset. The huge the dataset, less will be the accuracy and visa-versa.

```
# Train model
start_time = time.time()

model.fit(x=X_train, y=y_train, batch_size=32, epochs=10, validation_data=(X_val, y_val), callbacks=[tensorboard, checkpoint])

end_time = time.time()
execution_time = (end_time - start_time)
print(f"Elapsed time: {hms_string(execution_time)}")
```

The output for the same is:

```
Epoch 1/10
46/46 [==============================] - ETA: 0s - loss: 0.7808 - accuracy: 0.6180INFO:tensorflow:Assets written to: models\cnn
-parameters-improvement-01-0.59.model\assets
46/46 [==============================] - 156s 3s/step - loss: 0.7808 - accuracy: 0.6180 - val_loss: 0.6529 - val_accuracy: 0.58
71
Epoch 2/10
46/46 [==============================] - ETA: 0s - loss: 0.5078 - accuracy: 0.7571INFO:tensorflow:Assets written to: models\cnn
-parameters-improvement-02-0.52.model\assets
46/46 [==============================] - 164s 4s/step - loss: 0.5078 - accuracy: 0.7571 - val_loss: 0.6788 - val_accuracy: 0.51
94
Epoch 3/10
46/46 [==============================] - ETA: 0s - loss: 0.4075 - accuracy: 0.8138INFO:tensorflow:Assets written to: models\cnn
-parameters-improvement-03-0.69.model\assets
46/46 [==============================] - 166s 4s/step - loss: 0.4075 - accuracy: 0.8138 - val_loss: 0.5933 - val_accuracy: 0.69
03
Epoch 4/10
46/46 [==============================] - ETA: 0s - loss: 0.4275 - accuracy: 0.8090INFO:tensorflow:Assets written to: models\cnn
-parameters-improvement-04-0.72.model\assets
46/46 [==============================] - 167s 4s/step - loss: 0.4275 - accuracy: 0.8090 - val_loss: 0.5526 - val_accuracy: 0.71
94
Epoch 5/10
46/46 [==============================] - ETA: 0s - loss: 0.3716 - accuracy: 0.8325INFO:tensorflow:Assets written to: models\cnn
-parameters-improvement-05-0.78.model\assets
46/46 [==============================] - 158s 3s/step - loss: 0.3716 - accuracy: 0.8325 - val_loss: 0.4939 - val_accuracy: 0.77
74
Epoch 6/10
46/46 [==============================] - ETA: 0s - loss: 0.3105 - accuracy: 0.8637INFO:tensorflow:Assets written to: models\cnn
-parameters-improvement-06-0.57.model\assets
46/46 [==============================] - 149s 3s/step - loss: 0.3105 - accuracy: 0.8637 - val_loss: 0.7155 - val_accuracy: 0.57
10
Epoch 7/10
46/46 [==============================] - ETA: 0s - loss: 0.3218 - accuracy: 0.8588INFO:tensorflow:Assets written to: models\cnn
-parameters-improvement-07-0.64.model\assets
46/46 [==============================] - 144s 3s/step - loss: 0.3218 - accuracy: 0.8588 - val_loss: 0.6589 - val_accuracy: 0.63
55
Epoch 8/10
46/46 [==============================] - ETA: 0s - loss: 0.2427 - accuracy: 0.9024INFO:tensorflow:Assets written to: models\cnn
-parameters-improvement-08-0.73.model\assets
46/46 [==============================] - 146s 3s/step - loss: 0.2427 - accuracy: 0.9024 - val_loss: 0.5152 - val_accuracy: 0.72
90
Epoch 9/10
46/46 [==============================] - ETA: 0s - loss: 0.2414 - accuracy: 0.9073INFO:tensorflow:Assets written to: models\cnn
-parameters-improvement-09-0.81.model\assets
46/46 [==============================] - 149s 3s/step - loss: 0.2414 - accuracy: 0.9073 - val_loss: 0.4270 - val_accuracy: 0.81
29
Epoch 10/10
46/46 [==============================] - ETA: 0s - loss: 0.2216 - accuracy: 0.9183INFO:tensorflow:Assets written to: models\cnn
-parameters-improvement-10-0.84.model\assets
```

After training the epoch, we have to train some more epoch until and unless we could achieve accuracy as high as possible. After that we just have to plot loss and accuracy graph.

```
# Plot loss and accuracy
def plot_metrics(history):

    train_loss = history['loss']
    val_loss = history['val_loss']
    train_acc = history['accuracy']
    val_acc = history['val_accuracy']

    # Loss
    plt.figure()
    plt.plot(train_loss, label='Training Loss')
    plt.plot(val_loss, label='Validation Loss')
    plt.title('Loss')
    plt.legend()
    plt.show()

    # Accuracy
    plt.figure()
    plt.plot(train_acc, label='Training Accuracy')
    plt.plot(val_acc, label='Validation Accuracy')
    plt.title('Accuracy')
    plt.legend()
    plt.show()
```

The graphs achieved are:

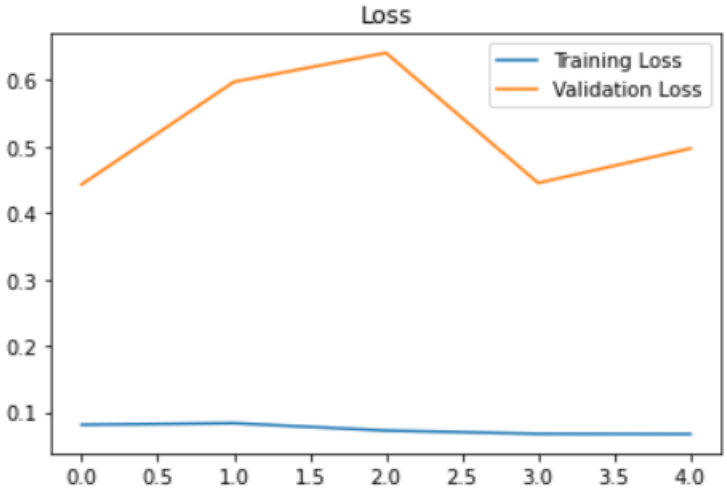


Figure 4.4

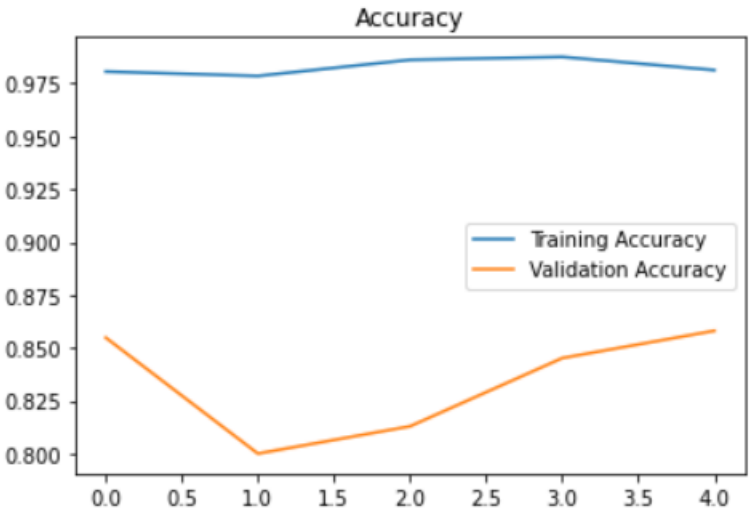


Figure 4.5

4.3 Results

Concretely, the model after 32 iterations with accuracy of 98%. Which is way better than the other accuracies. Now we only just have to load the best model to achieve the test loss and the test accuracy. The test loss and accuracy are mentioned in the table below:

Test loss	0.3941361904144287	39%
Test accuracy	0.8709677457809448	87%

Table 7.

After the computation we achieved the f1 score of 0.8481012658227848 which is approximately equals to 85%

Now the percentage of whole data is:

```
Number of examples: 2065
Percentage of positive examples: 52.54237288135593%, number of pos example
s: 1085
Percentage of negative examples: 47.45762711864407%, number of neg example
s: 980
```

```
print("Training Data:")
data_percentage(y_train)

print("Validation Data:")
data_percentage(y_val)

print("Testing Data:")
data_percentage(y_test)
```

Output:

```
Training Data:
Number of examples: 1445
Percentage of positive examples: 52.94117647058823%, number of pos example
s: 765
Percentage of negative examples: 47.05882352941177%, number of neg example
s: 680
Validation Data:
Number of examples: 310
Percentage of positive examples: 53.225806451612904%, number of pos exampl
es: 165
Percentage of negative examples: 46.774193548387096%, number of neg exampl
es: 145
Testing Data:
Number of examples: 310
Percentage of positive examples: 50.0%, number of pos examples: 155
Percentage of negative examples: 50.0%, number of neg examples: 155
```

As expected, the percentage of Positive examples are around 50% which is a very good percentage.

Performance Table

	Validation set	**Test set**
Accuracy	98%	87%
F1 score	0.9875	0.85

Table 8.

Chapter 5

Conclusion

5.1 Conclusion

An abscess on the brain, known as an intracranial tumor, is a rare tissue where cells grow and reproduce uncontrollably, seemingly uncontrolled by processes that control normal cells. More than 150 brain tumors are listed, but the two major groups of brain tumors are called primary and metastatic.

In this paper, we have designed a plant to differentiate a plant based on a four-dimensional model of thinking. It means that all methods have different features to help the network effectively distinguish between categories. it is undeniable that working in the area of the brain image only near growing tissues allows the CNN model to achieve performance prepared for human viewers. In addition, the simple but economical CNN model is expected to release traditional and international options with 2 alternatives with completely different output sizes. On our way, when the expected site of the plant is removed using a powerful advance method, those tracts are selected to feed the network its center located in this space. This results in a reduction in machine time and the ability to make quick predictions to differentiate the clinical picture because it removes a large number of insignificant pixels from the image within the previous processing step. Complete testing has demonstrated the effectiveness of Distance-Wise Attention in our algorithmic law alike due to the outstanding power of our entire model when followed by continuous methods.

Although the outstanding results of the proposed method are compared with the recently revealed models in contrast, our algorithmic rule still has limitations when it meets the growth volume of a third of the full brain. this is usually due to an increase in the expected size of the plant space which leads to a decrease in the effect of the extraction factor.

5.2 Future Scope

It is noteworthy in concluding that the proposed approach requires extensive training for better results; in the field of medical imaging, medical data collection is a tedious task, and, in a few cases, data sets may not be available. In all such cases, the proposed algorithm should be strong enough to accurately identify the plant regions from MR Images. The proposed method can be further developed by working with weakly trained algorithms that can detect abnormalities with less training data and self-learning algorithms that can help improve algorithm accuracy and reduce computational time.

References

1. Van Meir, E. G. et al. Exciting new advances in neuro-oncology: the avenue to a cure for malignant glioma. CA. Cancer J. Clin. 60(3), 166–193. https://doi.org/10.3322/caac.20069 (2010).
Article
PubMed
PubMed Central
Google Scholar

2. Bakas, S. et al. Advancing The Cancer Genome Atlas glioma MRI collections with expert segmentation labels and radiomic features. Sci. Data 4(1), 1–13. https://doi.org/10.1038/sdata.2017.117 (2017)
MathSciNet
Article
Google Scholar

3. Khosravanian, A., Rahmanimanesh, M., Keshavarzi, P. & Mozaffari, S. Fast level set method for glioma brain tumor segmentation based on superpixel fuzzy clustering and lattice boltzmann method. Comput. Methods Programs Biomed. 198, 105809. https://doi.org/10.1016/j.cmpb.2020.105809 (2020).
Article
PubMed
Google Scholar

4. Tang, Z., Ahmad, S., Yap, P. T. & Shen, D. Multi-atlas segmentation of MR tumor brain images using low-rank based image recovery. IEEE Trans. Med. Imaging 37(10), 2224–2235. https://doi.org/10.1109/TMI.2018.2824243 (2018).
Article
PubMed
PubMed Central
Google Scholar
5. Bakas, S. Segmentation labels and radiomic features for the pre-operative scans of the TCGA-LGG collection. Cancer Imag. Arch. https://doi.org/10.7937/K9/TCIA.2017.GJQ7R0EF (2017).
Article
Google Scholar

6. Ramli, N. M., Hussain, M. A., Jan, B. M. & Abdullah, B. Online composition prediction of a debutanizer column using artificial neural network. Iran. J. Chem. Chem. Eng. 36(2), 153–174. https://doi.org/10.30492/IJCCE.2017.26704 (2017).
CAS
Article
Google Scholar

7. Q. V. Le and T. Mikolov, Distributed representations of sentences and documents. 2014, doi: https://doi.org/10.1145/2740908.2742760.

8. Kamari, E., Hajizadeh, A. A. & Kamali, M. R. Experimental investigation and estimation of light hydrocarbons gas-liquid equilibrium ratio in gas condensate reservoirs through artificial neural networks. Iran. J. Chem. Chem. Eng. 39(6), 163–172. https://doi.org/10.30492/ijcce.2019.36496 (2020).
CAS
Article
Google Scholar

9. Ganjkhanlou, Y. et al. Application of image analysis in the characterization of electrospun nanofibers. Iran. J. Chem. Chem. Eng. 33(2), 37–45. https://doi.org/10.30492/IJCCE.2014.10750 (2014).
CAS
Article
Google Scholar

10. Chen, G., Li, Q., Shi, F., Rekik, I. & Pan, Z. RFDCR: Automated brain lesion segmentation using cascaded random forests with dense conditional random fields. Neuroimage 211, 116620. https://doi.org/10.1016/j.neuroimage.2020.116620 (2020).
Article
PubMed

11. A. Jalalifar, H. Soliman, M. Ruschin, A. Sahgal, A. Sadeghi-Naini, A brain tumor segmentation framework based on outlier detection using one-class support vector machine," in Proceedings of the Annual International Conference of the IEEE Engineering in Medicine and Biology Society, EMBS, Jul. 2020, vol. 2020-July, pp. 1067–1070, doi: https://doi.org/10.1109/EMBC44109.2020.9176263.

12. Torabi Dashti, H., Masoudi-Nejad, A. & Zare, F. Finding exact and solo LTR-retrotransposons in biological sequences using SVM. Iran. J. Chem. Chem. Eng. 31(2), 111–116. https://doi.org/10.30492/IJCCE.2012.5998 (2012)
CAS
Article
Google Scholar

13. Partovi, S. M. A. & Sadeghnejad, S. Reservoir rock characterization using wavelet transform and fractal dimension. Iran. J. Chem. Chem. Eng. 37(3), 223–233. https://doi.org/10.30492/IJCCE.2018.27647 (2018).
CAS
Article
Google Scholar

14. Antonelli, M. et al. GAS: A genetic atlas selection strategy in multi-atlas segmentation framework. Med. Image Anal. 52, 97–108. https://doi.org/10.1016/j.media.2018.11.007 (2019).
Article