A

Major Project

On

BRAIN TUMOUR IMAGE SEGMENTATION USING DEEP LEARNING

(Submitted in partial fulfillment of the requirements for the award of Degree)

BACHELOR OF TECHNOLOGY

In

COMPUTER SCIENCE AND ENGINEERING

By

Rajshree Mondal (187R1A0539)

Under the guidance of

Dr. RAJ KUMAR PATRA

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CMR TECHNICAL CAMPUS

UGC AUTONOMOUS

(Accredited by NAAC, NBA, Permanently Affiliated to JNTUH, Approved by AICTE,
NewDelhi) Recognized Under Section 2(f) & 12(B) of the UGCAct.1956, Kandlakoya
(V), Medchal Road, Hyderabad-501401.

2018-2022

DEPARTMENT OF COMPUTER SCIENCE & ENGINEERING

CMR
GROUP OF INSTITUTIONS
EXPLORE TO INVENT

CERTIFICATE

This is to certify that the project entitled **"BRAIN TUMOUR IMAGE SEGMENTATION USING DEEP LEARNING"** being submitted by **RAJSHREE MONDAL** (187R1A0539), in partial fulfillment of the requirements for the award of the degree of B.Tech in Computer Science and Engineering to the Jawaharlal Nehru Technological University Hyderabad, is a record of bonafide work carried out by him/her under our guidance and supervision during the year 2022-23.

The results embodied in this thesis have not been submitted to any other University or Institute for the award of any degree or diploma.

Dr. Raj Kumar Patra
INTERNAL GUIDE

Dr. A. Raji Reddy
DIRECTOR

Dr. K. Srujan Raju
Head of Department

EXTERNAL EXAMINER

Submitted for Viva voice examination held on ______________

ACKNOWLEDGEMENT

Apart from the efforts of us, the success of any project depends largely on the encouragement and guidelines of many others. We take this opportunity to express our gratitude to the people who have been instrumental in the successful completion of this project. We take this opportunity to express my profound gratitude and deep regard to our guide

Dr. Raj Kumar Patra, for his exemplary guidance, monitoring and constant encouragement throughout the project work. The blessing, help and guidance given by him shall carry us a long way in the journey of life on which we are about to embark. We also take this opportunity to express a deep sense of gratitude to Project Review Committee (PRC) **Mr. J. Narasimha Rao, Dr. T. S. Mastan Rao, Dr. Suwarna Gothane, Mr. A. Uday Kiran, Mr. A. Kiran Kumar, Mrs. G. Latha** for their cordial support, valuable information and guidance, which helped us in completing this task through various stages.

We are also thankful to **Dr. K. Srujan Raju**, Head, Department of Computer Science and Engineering for providing encouragement and support for completing this project successfully.

We are obliged to **Dr. A. Raji Reddy**, Director for being cooperative throughout the course of this project. We also express our sincere gratitude to **Sri. Ch. Gopal Reddy**, Chairman for providing excellent infrastructure and a nice atmosphere throughout the course of this project.

The guidance and support received from all the members of **CMR Technical Campus** who contributed to the completion of the project. We are grateful for their constant support and help.

Finally, we would like to take this opportunity to thank our family for their constant encouragement, without which this assignment would not be completed. We sincerely acknowledge and thank all those who gave support directly and indirectly in the completion of this project

RAJSHREE MONDAL (187R1A0539)

ABSTRACT

Automated segmentation of brain tumour from multimodal MR images is pivotal for the analysis and monitoring of disease progression. As gliomas are malignant and heterogeneous, efficient and accurate segmentation techniques are used for the successful delineation of tumours into intra-tumoural classes. Deep learning algorithms outperform on tasks of semantic segmentation as opposed to the more conventional, context-based computer vision approaches. Extensively used for biomedical image segmentation, Convolutional Neural Networks have significantly improved the state-of-the-art accuracy on the task of brain tumour segmentation. In this paper, we propose an ensemble of two segmentation networks: a 3D CNN and a U-Net, in a significant yet straightforward combinative technique that results in better and accurate predictions. Both models were trained separately on the BraTS-19 challenge dataset and evaluated to yield segmentation maps which considerably differed from each other in terms of segmented tumour sub-regions and were ensembled variably to achieve the final prediction.

CONTENTS

LIST OF FIGURES

1. INTRODUCTION

1.INTRODUCTION

Accurate segmentation of tumours through medical images is of particular importance as it provides information essential for analysis and diagnosis of cancer as well as for mapping out treatment options and monitoring the progression of the disease. Brain tumours are one of the fatal cancers worldwide and are categorized, depending upon their origin, into primary and secondary tumour types. The most common histological form of primary brain cancer is the glioma, which originates from the brain glial cells and attributes towards 80% of all malignant brain tumours. Gliomas can be of the slow-progressing low-grade (LGG) subtype with a better patient prognosis or are the more aggressive and infiltrative high-grade glioma (HGG) or glioblastoma, which require immediate treatment. These tumours are associated with substantial morbidity, where the median survival for a patient with glioblastoma is only about 14 months with a 5-year survival rate near zero despite maximal surgical and medical therapy. A timely diagnosis, therefore, becomes imperative for effective treatment of the patients. Magnetic Resonance Imaging (MRI) is a preferred technique widely employed by radiologists for the evaluation and assessment of brain tumours. It provides several complimentary 3D MRI modalities acquired based on the degree of excitation and repetition times, i.e., T1-weighted, post-contrast T1-weighted (T1ce), T2-weighted and Fluid Attenuated Inversion Recovery (FLAIR). The highlighted subregions of the tumour across different intensities of these sequences, such as the whole tumour (the entire tumour inclusive of infiltrative oedema), is more prominent in FLAIR and T2 modalities. In contrast, T1 and T1ce images show the tumour core exclusive of peritumoral oedema. It allows for the combinative use of these scans and the complementary information they deliver towards the detection of different tumour subregions. The Multimodal Brain Tumour Segmentation Challenge (BraTS) is a

platform to evaluate the development of machine learning models for the task of tumour segmentation, by facilitating the participants with an extensive dataset of 3D MRI images of the gliomas (both LGG and HGG) and associated ground truths annotated by expert physicians. The provided multimodal scans are used for both training and validating the neural networks designed for the particular segmentation task. Manually delineating brain tumour subregions from MRI scans is a subjective task, and therefore it is time-consuming and prone to variability.

platform to evaluate the development of machine learning models for the task of tumour segmentation, by facilitating the participants with an extensive dataset of 3D MRI images of the gliomas (both LGG and HGG) and associated ground truths annotated by expert physicians. The provided multimodal scans are used for both training and validating the neural networks designed for the particular segmentation task. Manually delineating brain tumour subregions from MRI scans is a subjective task, and therefore it is time-consuming and prone to variability. Automated segmentation of gliomas from multimodal MRI images can consequently assist the physicians to speed-up diagnosis and surgical planning as well as provide an accurate, reproducible solution for further tumour analysis and monitoring. The classical methods of automated brain tumour segmentation rely on feature engineering, which involves the extraction of handcrafted features from input images with follow up training of classifier. Convolutional Neural Networks (CNNs) is regarded as the state-of-the-art methods for brain tumour image segmentation as they learn the most useful and relevant features automatically. However, accurate segmentation of tumour remains a challenge; due to heterogeneity in terms of shape, size, and appearance of the gliomas as well as ambiguous and fuzzy boundary existing between cancer and brain tissue. The intensity variability of the MRI data further adds to this difficulty. Therefore, it is still open to improvement, allowing further exploration for better segmentation techniques and accuracy. In this work, we utilize multiple 3D CNN models for brain tumour segmentation from multimodal MRI scans and ensemble their probability maps for more stable predictions. The networks are trained separately, with hyperparameters optimised for each model, on the training dataset acquired from the 2019 Brain Tumour Segmentation (BraTS) challenge. A rigorous evaluation on the BraTS validation set resulted with the proposed ensemble achieving dice scores of 0.750, 0.906 and 0.846 for enhancing tumour, whole tumour, and tumour core, respectively.

2. LITERATURE SURVEY

2.LITERATURE SURVEY

2.2 BRAIN TUMOUR DETECTION WITH CONVOLUTIONAL NEURAL NETWORKS PROPOSED BY DIFFERENT AUTHORS

B. H. Menze et.al [7] have found that different algorithms worked best for different sub-regions (reaching performance comparable to human inter-rater variability), but that no single algorithm ranked in the top for all subregions simultaneously. Fusing several good algorithms using a hierarchical majority vote yielded segmentations that consistently ranked above all individual algorithms, indicating remaining opportunities for further methodological improvements.

M. Goetz et.al [14] have proposed a method that derives high-quality classifiers for the different tissue classes from sparse and unambiguous annotations and employs domain adaptation techniques for effectively correcting sampling selection errors introduced by the sparse sampling. The new approach is validated on labeled, multi-modal MR images of 19 patients with malignant gliomas and by comparative analysis on the BraTS 2013 challenge data sets.

H. Dong et. al [11] have proposed a fully automatic method for brain tumor segmentation, which is developed using U-Net based deep convolutional networks. The method was evaluated on Multimodal Brain Tumor Image Segmentation (BRATS 2015) datasets, which contain 220 high-grade brain tumor and 54 low-grade tumor cases. Cross validation has shown that the method can obtain promising segmentation efficiently.

K. Farahani [15] have introduced a generative probabilistic model for segmentation of tumors in multi-dimensional images. The model allows for different tumor boundaries in each channel, reflecting difference in tumor appearance across modalities. It augments a probabilistic atlas of healthy tissue priors with a latent atlas of the lesion and derive the estimation algorithm to extract tumor boundaries and the latent atlas from the image data. It present experiments on 25 glioma patient data sets, demonstrating significant improvement over the traditional multivariate tumor segmentation.

3. SYSTEM ANALYSIS

3.SYSTEM ANALYSIS

3.1 EXISTING SYSTEM:

➢ Existing system focuses on traditional methods for brain tumor image segmentation. Manual segmentation of such brain tumours requires a great deal of medical expertise, is time-consuming, and prone to human error. Moreover, the manual process lacks consistency and reproducibility, which negatively affects the results and can ultimately lead to incorrect prognosis and treatment.

➢ The classical methods of automated brain tumor segmentation rely on feature engineering, which involves the extraction of handcrafted features from input images with follow up training of classifier.

3.1.1 DISADVANTAGES OF EXISTING SYSTEM:

❖ Manually delineating brain tumour sub regions from MRI scans is a subjective task.

❖ it is time-consuming and prone to variability.

3.3 PROPOSED SYSTEM:

Proposed system is using deep learning methods for brain tumor image segmentation. Deep learning methods is regarded as the state-of-the-art methods for brain tumor image segmentation as they learn the most useful and relevant features automatically, in order to increase the accuracy and efficiency. Using convolutional output layer, the model is an order of magnitude faster than other state of the art methods.

To implement this project, we are using 4 different images and these images are called as FLAIR, T1, T2 and T1CE and the label segmented image. The multi-institutional dataset, acquired from 19 different contributors, contains multimodal MRI scans of each patient, namely T1, T1 contrast-enhanced (T1ce), T2-weighted (T2), and Fluid Attenuated Inversion Recovery (FLAIR), from which the tumoural sub regions are segmented. The data is processed to overcome discrepancies such that they are skull-stripped.

3.2.1 ADVANTAGES OF PROPOSED SYSTEM:

- Deep Convolutional Neural Networks have carved out a niche for achieving the state of the accuracy on the task of brain tumour segmentation.

- The methods used in the system leads to a decrease in computing time and overcomes the overfitting problems.

- Deep learning methods extract crucial features automatically. These approaches have yielded outstanding results in various application domains.

3.3 SYSTEM REQUIREMENTS:

HARDWARE REQUIREMENTS:

• Processor	: i3 intel or above
• Hard Disk	: 128 GB.
• Floppy Drive	: 1.44 Mb.
• Monitor	: 15 VGA Colour.
• Ram	: 512 Mb.

SOFTWARE REQUIREMENTS:

- **Operating System:** Windows
- **Coding Language**: Python 3.7

3.4 SYSTEM STUDY

FEASIBILITY STUDY

The feasibility of the project is analyzed in this phase and business proposal is put forth with a very general plan for the project and some cost estimates. During system analysis the feasibility study of the proposed system is to be carried out. This is to ensure that the proposed system is not a burden to the company. For feasibility analysis, some understanding of the major requirements for the system is essential.

Three key considerations involved in the feasibility analysis are

- ECONOMICAL FEASIBILITY
- TECHNICAL FEASIBILITY
- SOCIAL FEASIBILITY

ECONOMICAL FEASIBILITY

This study is carried out to check the economic impact that the system will have on the organization. The amount of fund that the company can pour into the research and development of the system is limited. The expenditures must be justified. Thus, the developed system as well within the budget and this was achieved because most of the technologies used are freely available. Only the customized products had to be purchased.

TECHNICAL FEASIBILITY

This study is carried out to check the technical feasibility, that is, the technical requirements of the system. Any system developed must not have a high demand on the available technical resources. This will lead to high demands on the available technical resources. This will lead to high demands being placed on the client. The developed system must have a modest requirement, as only minimal or null changes are required for implementing this system.

SOCIAL FEASIBILITY

The aspect of study is to check the level of acceptance of the system by the user. This includes the process of training the user to use the system efficiently. The user must not feel threatened by the system, instead must accept it as a necessity. The level of acceptance by the users solely depends on the methods that are employed to educate the user about the system and to make him familiar with it. His level of confidence must be raised so that he is also able to make some constructive criticism, which is welcomed, as he is the final user of the system.

4.SYSTEM DESIGN

4.SYSTEM DESIGN

4.1 SYSTEM ARCHITECTURE:

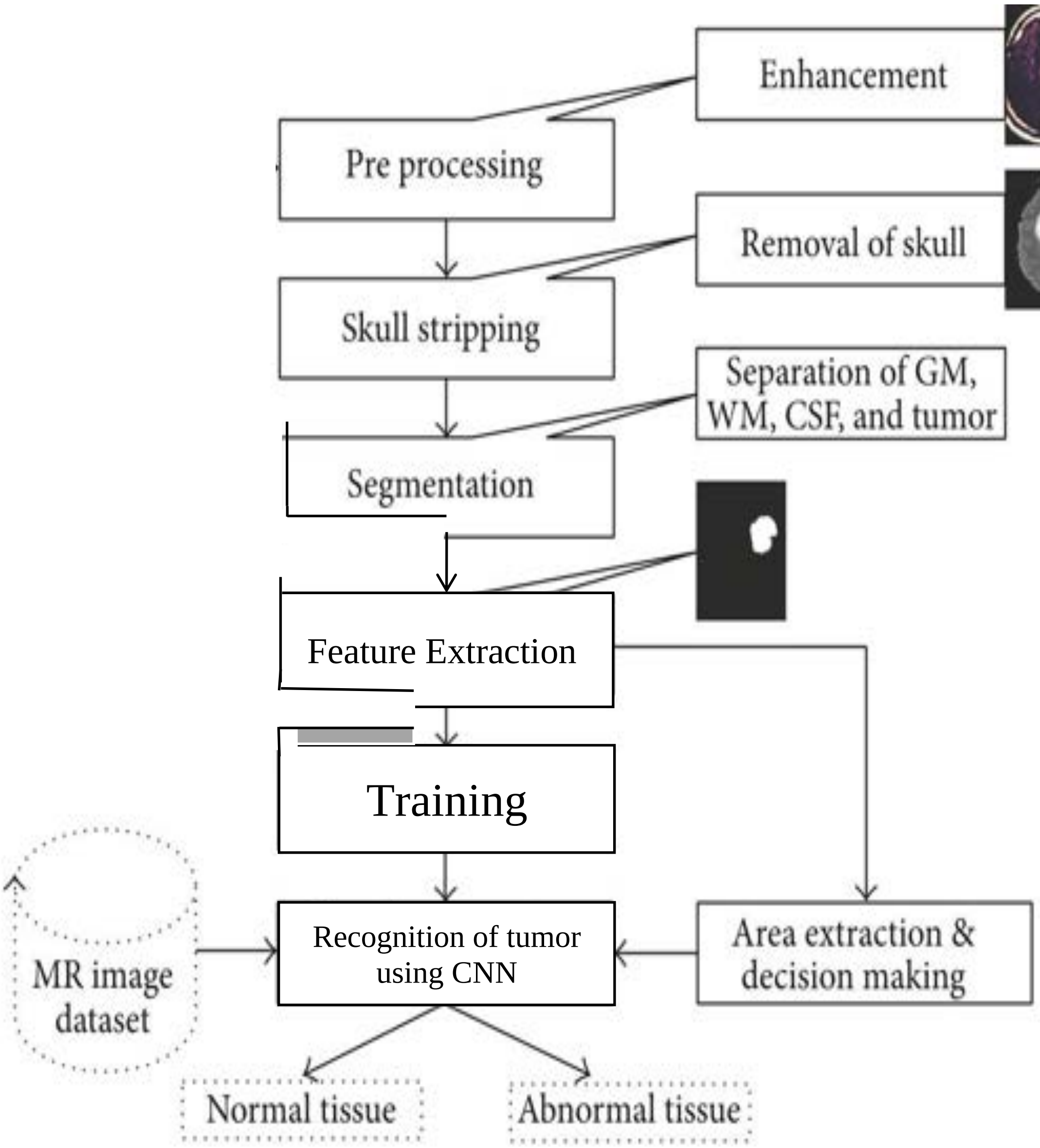


Figure No 4.1.1: SYSTEM ARCHITECTURE

4.2 BLOCK DIAGRAM

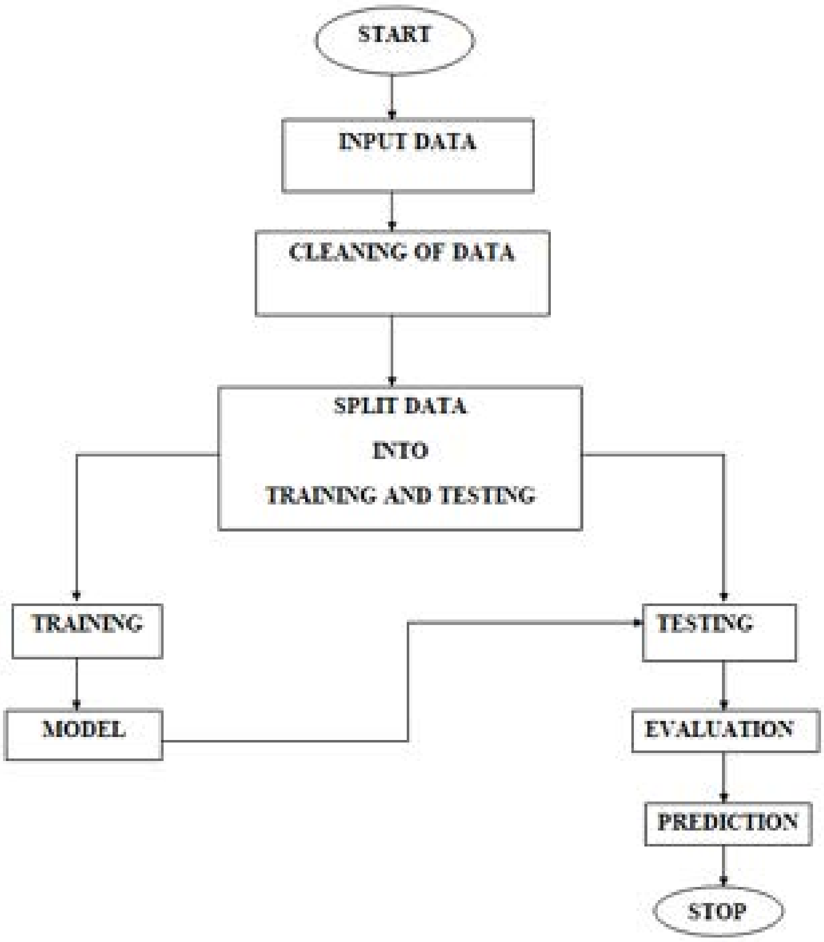


Figure No4.2.1: BLOCK DIAGRAM

4.3 DATA FLOW DIAGRAM:

The data flow diagram (DFD) is one of the most important modeling tools. It is used to model the system components. These components are the system process, the data used by the process, an external entity that interacts with the system and the information flows in the system. DFD shows how the information moves through the system and how it is modified by a series of transformations. It is a graphical technique that depicts information flow and the transformations that are applied as data moves from input to output.

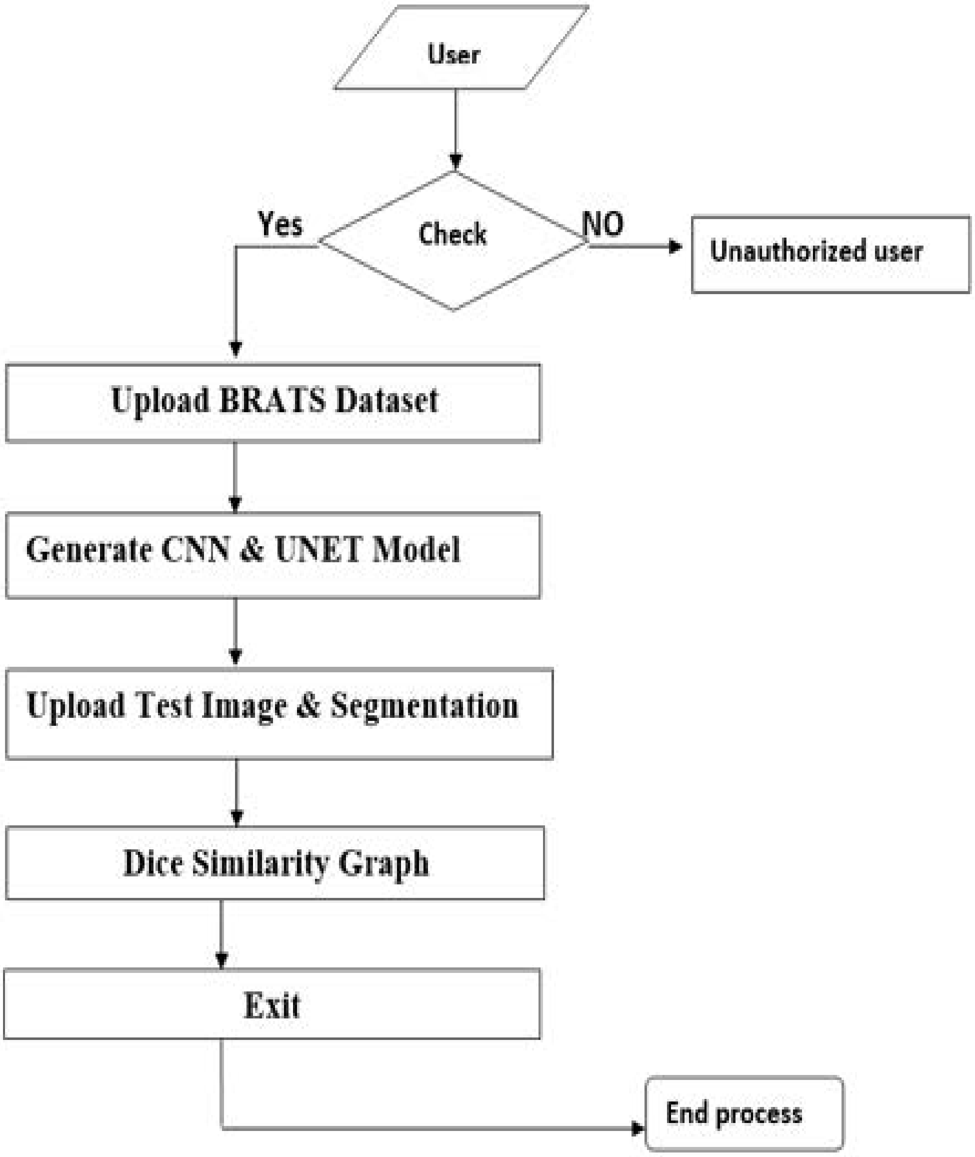


Fig No 4.3.1: DATA FLOW DIAGRAM

4.4 UML Diagram

The Unified Modeling Language (UML) is a standard language for specifying, visualizing, constructing, and documenting the artifacts of software systems, as well as for business modeling and other non-software systems. The underlying premise of UML is that no one diagram can capture the different elements of a system in its entirety.

Hence, UML is made up of eight (8) diagrams that can be used to model a system at different points of time in the software life cycle of a system.

The eight UML diagrams are:

Use case diagram:

The use case diagram is used to identify the primary elements and processes that form the system. The primary elements are termed as "actors" and the processes are called "use cases." The use case diagram shows which actors interact with each use case.

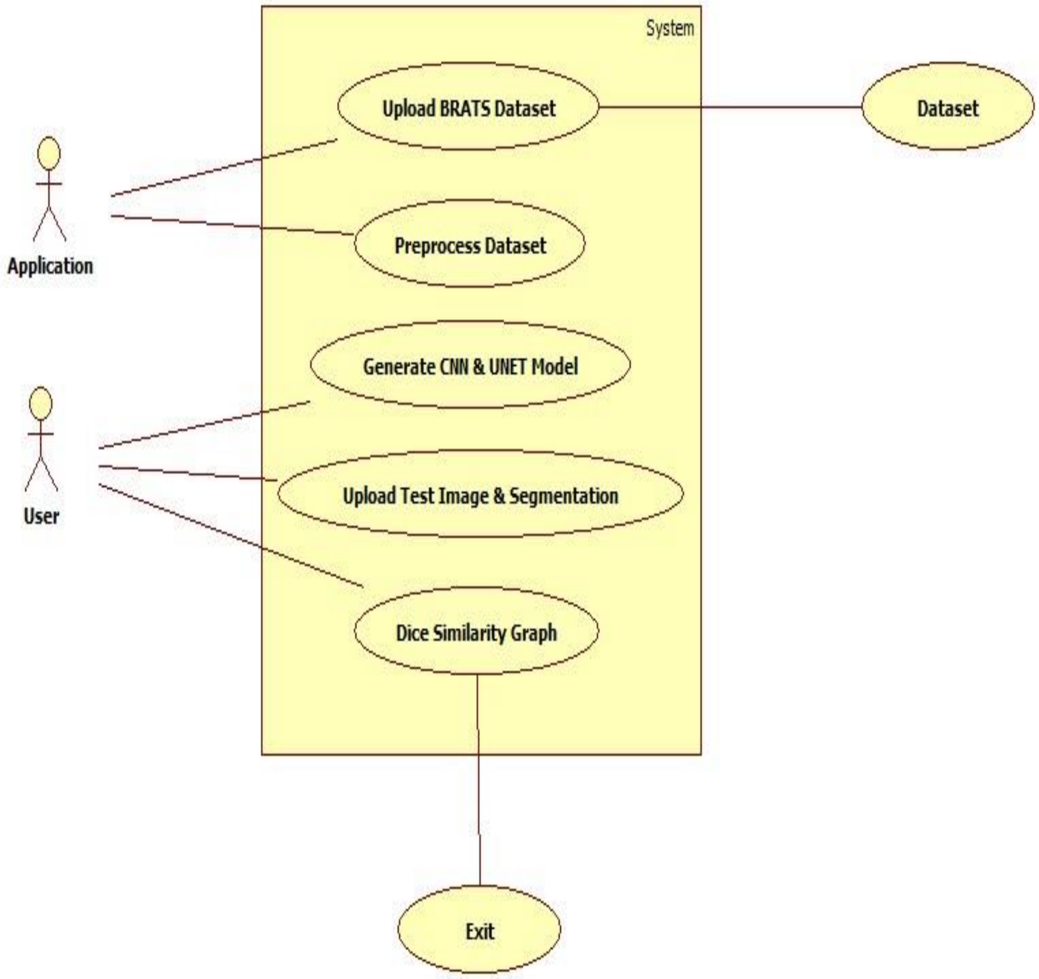


Figure No 4.4.1: USE CASE DIAGRAM

Object diagram:

The object diagram is a special kind of class diagram. An object is an instance of a class. This essentially means that an object represents the state of a class at a given point of time while the system is running. The object diagram captures the state of different classes in the system and their relationships or associations at a given point of time.

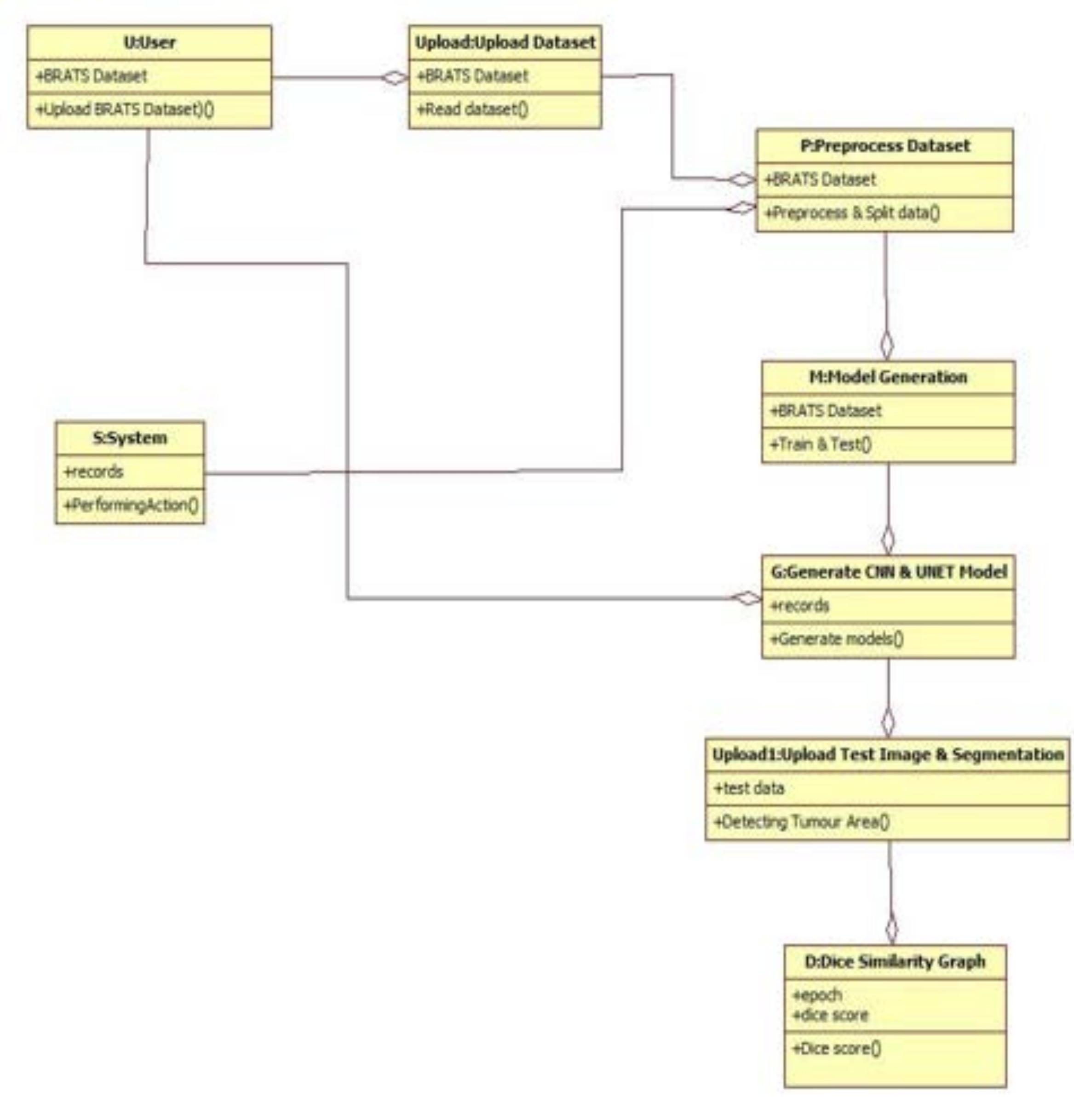


Figure No 4.4.2: OBJECT DIAGRAM

State diagram:

A state diagram, as the name suggests, represents the different states that objects in the system undergo during their life cycle. Objects in the system change states in response to events. In addition to this, a state diagram also captures the transition of the object's state from an initial state to a final state in response to events affecting the system.

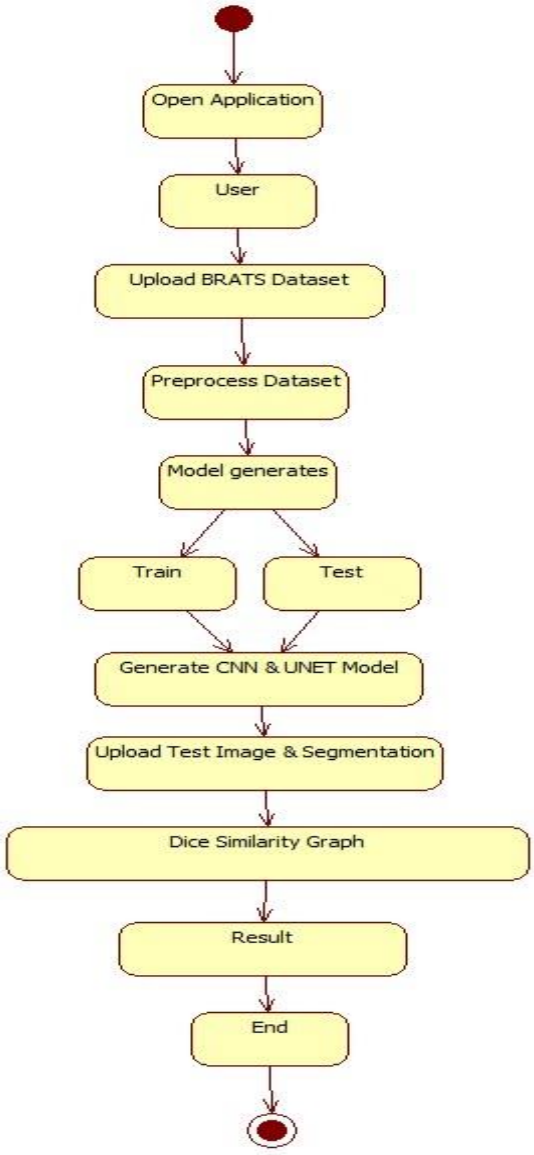


Figure No 4.4.3: STATE DIAGRAM

Activity diagram:

The process flows in the system are captured in the activity diagram. Similar to a state diagram, an activity diagram also consists of activities, actions, transitions, initial and final states, and guard conditions.

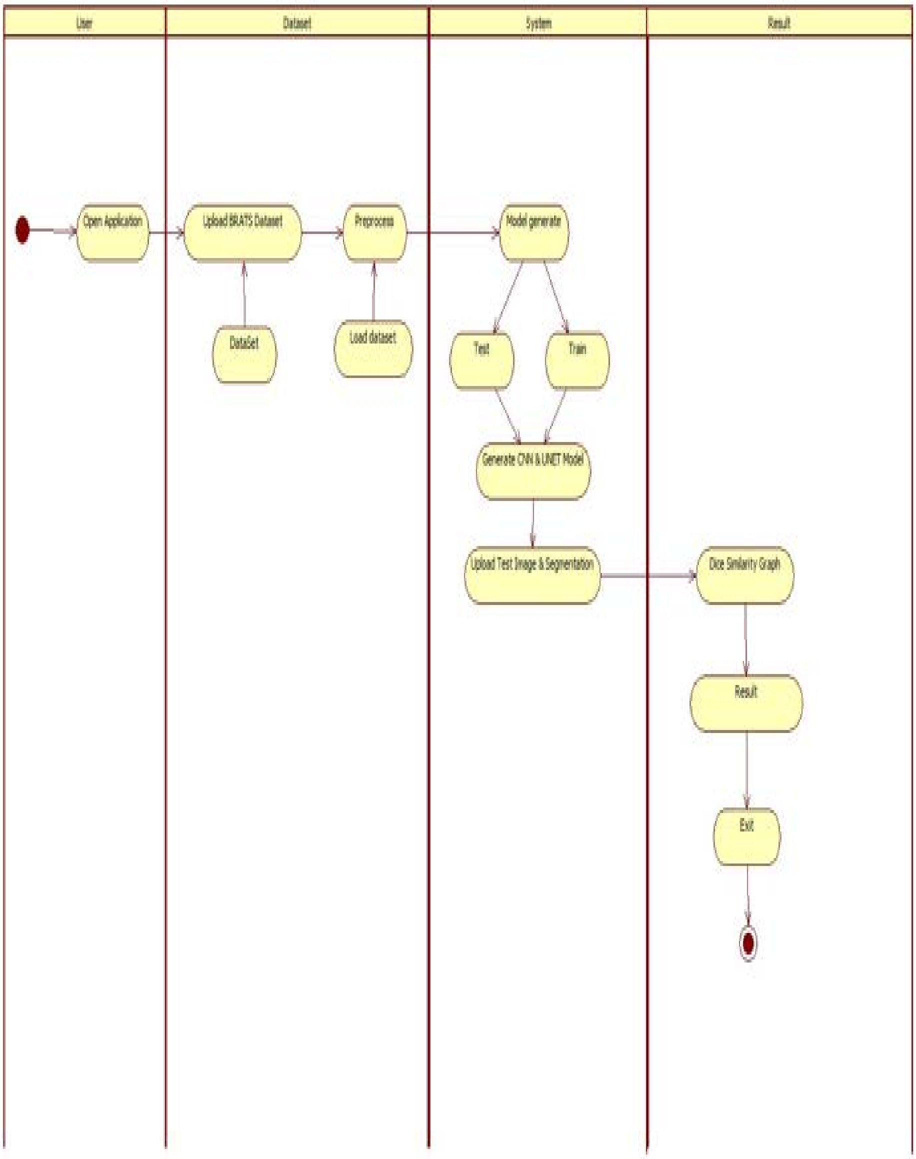


Figure No 4.4.4: ACTIVITY DIAGRAM

Sequence diagram:

A sequence diagram represents the interaction between different objects in the system. The important aspect of a sequence diagram is that it is time-ordered. This means that the exact sequence of the interactions between the objects is represented step by step. Different objects in the sequence diagram interact with each other by passing "messages".

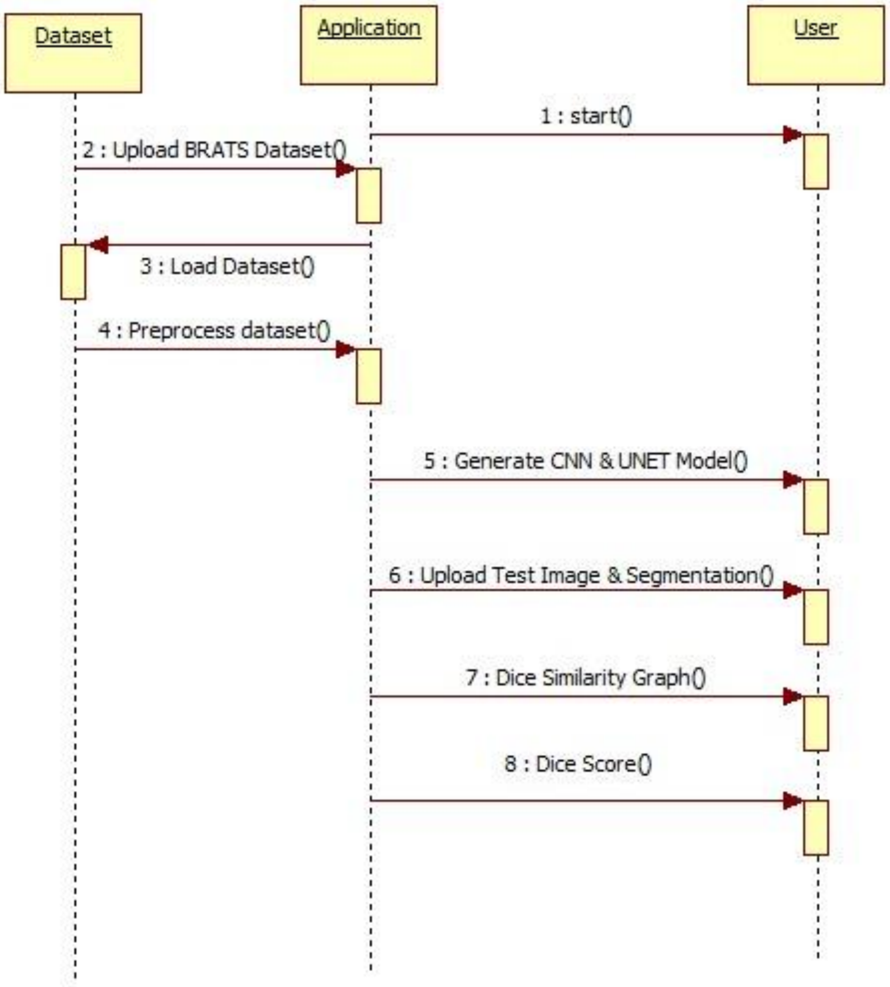

Dataset
Application
User
1 : start()
2 : Upload BRATS Dataset()
3 : Load Dataset()
4 : Preprocess dataset()
5 : Generate CNN & UNET Model()
6 : Upload Test Image & Segmentation()
7 : Dice Similarity Graph()
8 : Dice Score()

Figure No 4.4.5: SEQUENCE DIAGRAM

Collaboration diagram:

A collaboration diagram groups together the interactions between different objects. The interactions are listed as numbered interactions that help to trace the sequence of the interactions. The collaboration diagram helps to identify all the possible interactions that each object has with other objects.

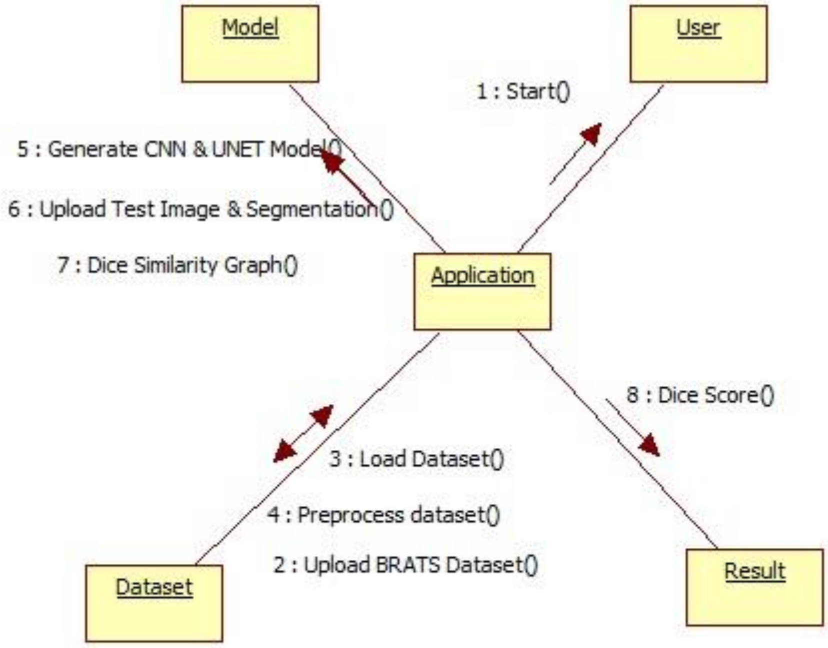

Model
User
1 : Start()
5 : Generate CNN & UNET Model()
6 : Upload Test Image & Segmentation()
7 : Dice Similarity Graph()
Application
8 : Dice Score()
3 : Load Dataset()
4 : Preprocess dataset()
2 : Upload BRATS Dataset()
Dataset
Result

Figure No 4.4.6: COLLABORATION DIAGRAM

Component diagram:

The component diagram represents the high-level parts that make up the system. This diagram depicts, at a high level, what components form part of the system and how they are interrelated. A component diagram depicts the components culled after the system has undergone the development or construction phase.

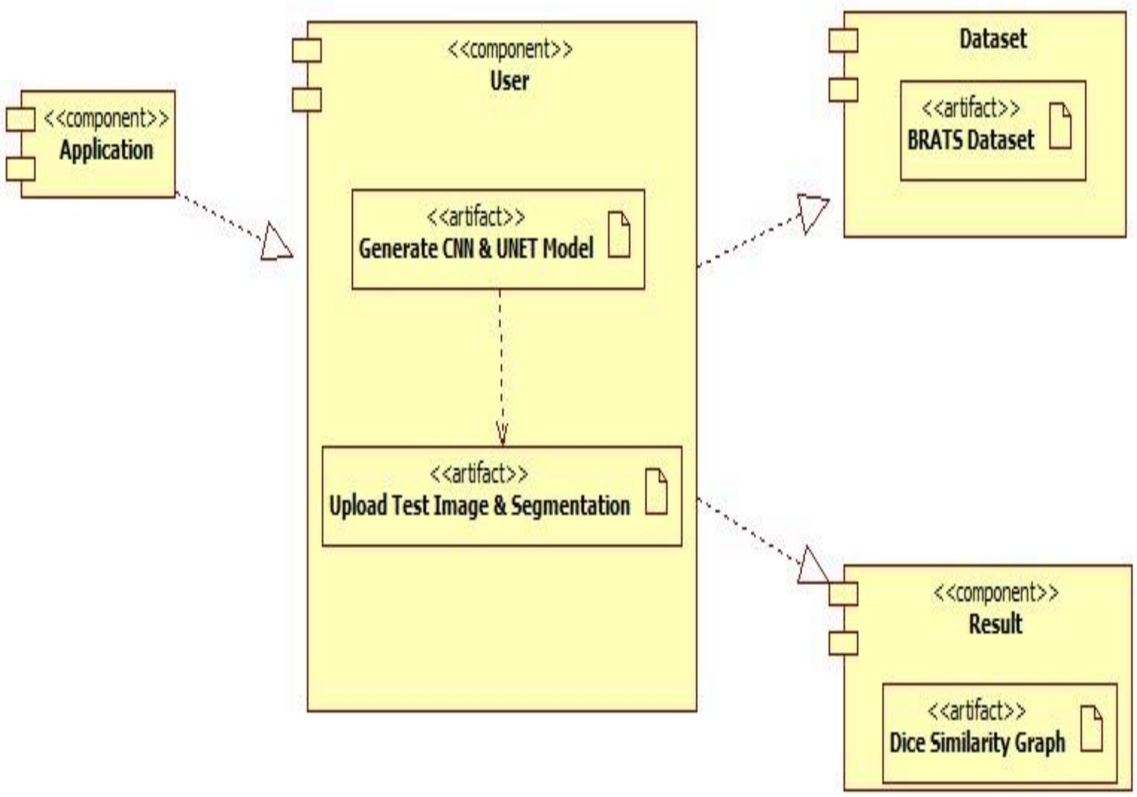


Figure No 4.4.7: COMPONENT DIAGRAM

4.5 SYSTEM IMPLEMENTATION:

MODULES:

- **Upload BRATS Dataset:**

  In this module user upload dataset of BRATS.

- **Generate CNN & UNET Model**

  We can see models are using different size images to filter them and to get best features from it to build efficient model and now model is generated.

- **Upload Test Image & Segmentation**

  In this module user upload test image and top 4 images are the input images such as FLAIR, T1, T2 and T1CE and 5th image is the predicted image with segmented part showing in red colour and this algorithm correctly detecting and marking tumour area.

- **Dice Similarity Graph**

  In this Module, we get final dice score as 0.8 * 100 = 80%. In above graph x-axis represents epoch and y-axis represents dice score

Algorithms

3D-CNN

To demonstrate how to build a convolutional neural network-based image classifier, we shall build a 6-layer neural network that will identify and separate one image from other. This network that we shall build is a very small network that we can run on a CPU as well. Traditional neural networks that are very good at doing image classification have many more parameters and take a lot of time if trained on normal CPU. However, our objective is to show how to build a real-world convolutional neural network using TENSORFLOW.

Neural Networks are essentially mathematical models to solve an optimization problem. They are made of neurons, the basic computation unit of neural networks. A neuron takes an input (say x), do some computation on it (say: multiply it with a variable w and adds another variable b) to produce a value (say; z= wx+b). This value is passed to a non-linear function called activation function (f) to produce the final output(activation) of a neuron. There are many kinds of activation functions. One of the popular activation functions is Sigmoid.

The neuron which uses sigmoid function as an activation function will be called sigmoid neuron. Depending on the activation functions, neurons are named and there are many kinds of them like RELU, TanH.

If you stack neurons in a single line, it’s called a layer; which is the next building block of neural networks. See below image with layers

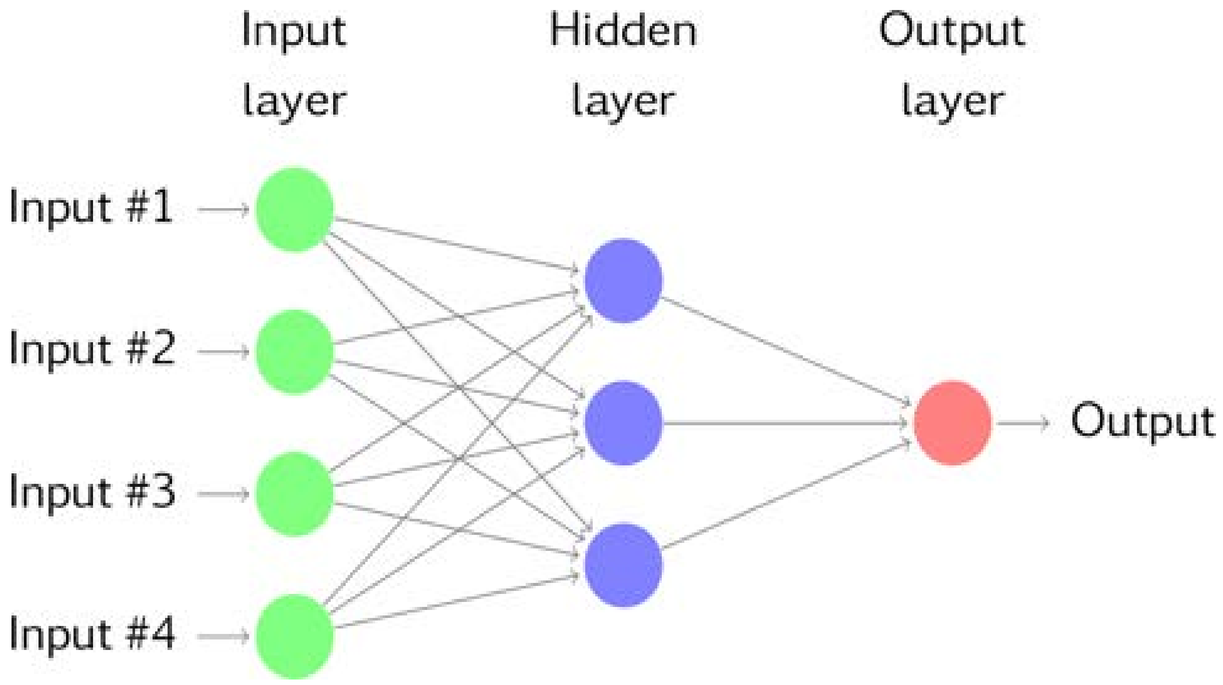


Figure No 4.5.1: 3D-CONVOLUTIONAL NEURAL NETWORK

To predict image class multiple layers operate on each other to get best match layer and this process continues till no more improvement left.

U-net

This deep neural network is implemented with Keras functional API, which makes it extremely easy to experiment with different interesting architectures. Output from the network is a 512*512 which represents mask that should be learned. Sigmoid activation function makes sure that mask pixels are in [0, 1] range.

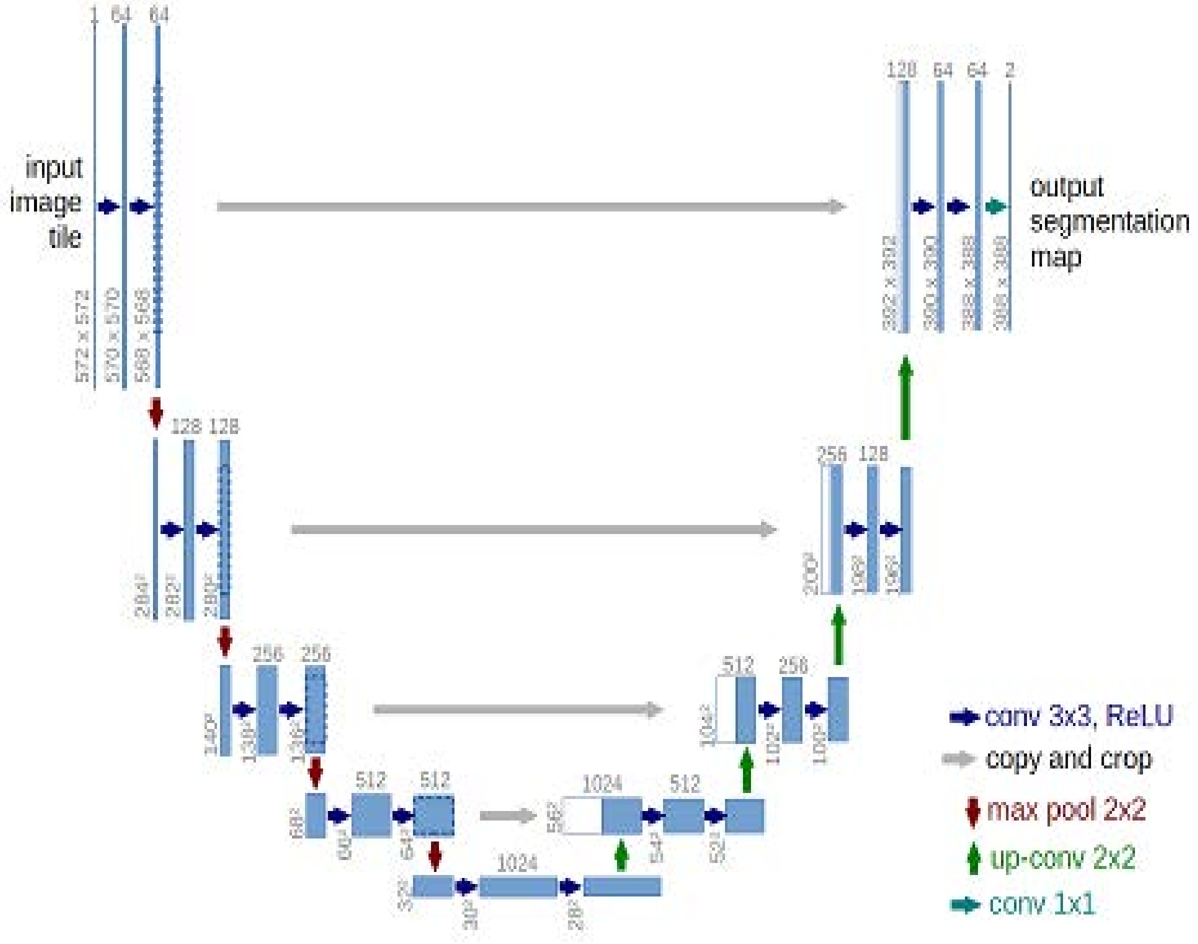


Figure No 4.5.2 : U-NET

5. SYSTEM TESTING

5.SYSTEM TESTING

The purpose of testing is to discover errors. Testing is the process of trying to discover every conceivable fault or weakness in a work product. It provides a way to check the functionality of components, sub assemblies, assemblies and/or a finished product It is the process of exercising software with the intent of ensuring that the Software system meets its requirements and user expectations and does not fail in an unacceptable manner. There are various types of test. Each test type addresses a specific testing requirement.

TYPES OF TESTS

Unit testing

Unit testing involves the design of test cases that validate that the internal program logic is functioning properly, and that program inputs produce valid outputs. All decision branches and internal code flow should be validated. It is the testing of individual software units of the application .it is done after the completion of an individual unit before integration. This is a structural testing, that relies on knowledge of its construction and is invasive. Unit tests perform basic tests at component level and test a specific business process, application, and/or system configuration. Unit tests ensure that each unique path of a business process performs accurately to the documented specifications and contains clearly defined inputs and expected results.

Integration testing

Integration tests are designed to test integrated software components to determine if they actually run as one program. Testing is event driven and is more concerned with the basic outcome of screens or fields. Integration tests demonstrate that although the components were individually satisfaction, as shown by successfully unit testing, the combination of components is correct and consistent. Integration testing is specifically aimed at exposing the problems that arise from the combination of components.

Functional test

Functional tests provide systematic demonstrations that functions tested are available as specified by the business and technical requirements, system documentation, and user manuals.

Functional testing is centered on the following items:

Valid Input : identified classes of valid input must be accepted.

Invalid Input : identified classes of invalid input must be rejected.

Functions : identified functions must be exercised.

Output : identified classes of application outputs must be exercised.

Systems/Procedures : interfacing systems or procedures must be invoked.

Organization and preparation of functional tests is focused on requirements, key functions, or special test cases. In addition, systematic coverage pertaining to identify Business process flows; data fields, predefined processes, and successive processes must be considered for testing. Before functional testing is complete, additional tests are identified and the effective value of current tests is determined.

System Test

System testing ensures that the entire integrated software system meets requirements. It tests a configuration to ensure known and predictable results. An example of system testing is the configuration-oriented system integration test. System testing is based on process descriptions and flows, emphasizing pre-driven process links and integration points.

White Box Testing

White Box Testing is a testing in which in which the software tester has knowledge of the inner workings, structure and language of the software, or at least its purpose. It is purpose. It is used to test areas that cannot be reached from a black box level.

Black Box Testing

Black Box Testing is testing the software without any knowledge of the inner workings, structure or language of the module being tested. Black box tests, as most other kinds of tests, must be written from a definitive source document, such as specification or requirements document, such as specification or requirements document. It is a testing in which the software under test is treated, as a black box. you cannot “see” into it. The test provides inputs and responds to outputs without considering how the software works.

Unit Testing

Unit testing is usually conducted as part of a combined code and unit test phase of the software lifecycle, although it is not uncommon for coding and unit testing to be conducted as two distinct phases.

Test strategy and approach

Field testing will be performed manually and functional tests will be written in detail.

Test objectives

- All field entries must work properly.
- Pages must be activated from the identified link.
- The entry screen, messages and responses must not be delayed.

Features to be tested

- Verify that the entries are of the correct format
- No duplicate entries should be allowed
- All links should take the user to the correct page.

Integration Testing

Software integration testing is the incremental integration testing of two or more integrated software components on a single platform to produce failures caused by interface defects. The task of the integration test is to check that components or software applications, e.g. components in a software system or – one step up – software applications at the company level – interact without error.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Acceptance Testing

User Acceptance Testing is a critical phase of any project and requires significant participation by the end user. It also ensures that the system meets the functional requirements.

Test Results: All the test cases mentioned above passed successfully. No defects encountered.

Test Cases

USER REQUIREMENTS:

1. Home

Home:

Use case ID	Brain Tumour Image Segmentation Using Deep Networks
Use case Name	Home button
Description	Display home page of application
Primary actor	User
Precondition	User must open application
Post condition	Display the Home Page of an application
Frequency of Use case	Many times
Alternative use case	N/A
Use case Diagrams	
Attachments	N/A

Figure No 5.1: USER REQUIREMENTS

6. IMPLEMENTATION

6. IMPLEMENTATION

```
from tkinter import messagebox
from tkinter import *
from tkinter import simpledialog
import tkinter
from tkinter import simpledialog
from tkinter import filedialog
import numpy as np
from tkinter.filedialog import askopenfilename
import pickle
import os
import cv2
import matplotlib.pyplot as plt
from sklearn.model_selection import train_test_split
from keras.models import *
from keras.layers import *
from keras.optimizers import *
gui = tkinter.Tk()
gui.title("Brain Tumour Image Segmentation Using Deep Networks")
gui.geometry("1300x1200")
global filename
global model
global X, Y
def dice_coef(y_true, y_pred):
 y_true_f = keras.flatten(y_true)
 y_pred_f = keras.flatten(y_pred)
 intersection = keras.sum(y_true_f * y_pred_f)
 return (2. * intersection + 1) / (keras.sum(y_true_f) + keras.sum(y_pred_f) + 1)
def dice_coef_loss(y_true, y_pred):
 return -dice_coef(y_true, y_pred)
def getModel(input_size=(64,64,1)):
 inputs = Input(input_size)
conv1=Conv2D(32,(3,3),activation='relu',padding='same')(inputs)
conv1= Conv2D(32, (3, 3), activation='relu', padding='same')(conv1)
pool1 = MaxPooling2D(pool_size=(2, 2))(conv1)
conv2=Conv2D(64, (3, 3), activation='relu', padding='same')(pool1)
conv2 = Conv2D(64, (3, 3), activation='relu', padding='same')(conv2)
 pool2 = MaxPooling2D(pool_size=(2, 2))(conv2)

 conv3=Conv2D(128,(3,3), activation='relu', padding='same')(pool2)
 conv3=Conv2D(128,(3,3),activation='relu',padding='same')(conv3)
 pool3 = MaxPooling2D(pool_size=(2, 2))(conv3)
conv4=Conv2D(256,(3, 3), activation='relu', padding='same')(pool3)
conv4=Conv2D(256,(3,3),activation='relu',padding='same')(conv4)
 pool4 = MaxPooling2D(pool_size=(2, 2))(conv4)
conv5=Conv2D(512,(3, 3), activation='relu', padding='same')(pool4)
```

```
conv5=Conv2D(512,(3, 3), activation='relu', padding='same')(conv5)
up6 = concatenate([Conv2DTranspose(256, (2, 2), strides=(2, 2), padding='same')(conv5), conv4],
axis=3)
conv6 = Conv2D(256, (3, 3), activation='relu', padding='same')(up6)
conv6=Conv2D(256,(3, 3), activation='relu', padding='same')(conv6)
up7 = concatenate([Conv2DTranspose(128, (2, 2), strides=(2, 2), padding='same')(conv6), conv3],
axis=3)
conv7 = Conv2D(128, (3, 3), activation='relu', padding='same')(up7)
conv7=Conv2D(128,(3, 3), activation='relu', padding='same')(conv7)
up8 = concatenate([Conv2DTranspose(64, (2, 2), strides=(2, 2), padding='same')(conv7), conv2],
axis=3)
 conv8 = Conv2D(64, (3, 3), activation='relu', padding='same')(up8)
conv8= Conv2D(64, (3, 3), activation='relu', padding='same')(conv8)
up9 = concatenate([Conv2DTranspose(32, (2, 2), strides=(2, 2), padding='same')(conv8), conv1],
axis=3)
conv9 = Conv2D(32, (3, 3), activation='relu', padding='same')(up9)
conv9= Conv2D(32, (3, 3), activation='relu', padding='same')(conv9)
conv10 = Conv2D(1, (1, 1), activation='sigmoid')(conv9)
return Model(inputs=[inputs], outputs=[conv10])
def uploadDataset():
    global X, Y
    global filename
    text.delete('1.0', END)
    filename = filedialog.askdirectory(initialdir=".")
    text.insert(END,filename+" loaded\n");
    '''
    X = []
    Y = []
    for root, dirs, directory in os.walk(filename):
      for i in range(len(directory)):
        img = cv2.imread(train_directory+"/"+directory[i],0)
        img=cv2.resize(img,(64,64),interpolation=cv2.INTER_CUBIC)
        X.append(img)
        img = cv2.imread("dataset/label/"+directory[i],0)
        img=cv2.resize(img,(64,64),interpolation=cv2.INTER_CUBIC)
        Y.append(img)
        X = np.asarray(X)
        Y = np.asarray(Y)
        '''
        def generateModel():
        global model
        '''
    global X, Y
    dim = 64
    X_train, X_test, y_train, y_test = train_test_split(X, Y, test_size =      0.10, random_state = 1)
    X_train = X_train.reshape(len(X_train),dim,dim,1)
    y_train = y_train.reshape(len(y_train),dim,dim,1)
    X_test = X_test.reshape(len(X_test),dim,dim,1)
```

```
   y_test = y_test.reshape(len(y_test),dim,dim,1)
   images = np.concatenate((X_train,X_test),axis=0)
   mask  = np.concatenate((y_train,y_test),axis=0)
   tr = X_train[12]
   yr = y_train[12]
   cv2.imshow('tr',tr)
   cv2.imshow('yr',yr)
   cv2.waitKey(0)
   '''
   model = getModel(input_size=(64,64,1))
   with open('model/model.json', "r") as json_file:
   loaded_model_json = json_file.read()
   model = model_from_json(loaded_model_json)
   json_file.close()
   model.load_weights("model/model_weights.h5")
   model._make_predict_function()
   print(model.summary())
   text.insert(END,"CNN & UNET model generated. See Black Console for model details\n")
   '''
   model.compile(optimizer=Adam(lr=1e-5),loss=dice_coef_loss, metrics=[dice_coef,
'binary_accuracy'])
   print(model.summary())
   model.compile(optimizer=Adam(lr=2e-4),loss=[dice_coef_loss], metrics = [dice_coef,
'binary_accuracy'])
train_vol,validation_vol,train_seg,validation_seg= train_test_split((images-127.0)/127.0,
(mask>127).astype(np.float32),
 test_size = 0.1,random_state = 2018)
 train_vol,test_vol,train_seg,test_seg=train_test_split(train_vol,train_seg,
test_size = 0.1, random_state = 2018)
 hist = model.fit(x = train_vol, y = train_seg, batch_size = 16, epochs = 50, validation_data
=(test_vol,test_seg))
 model.save_weights('model/model_weights.h5')
 model_json = model.to_json()
 with open("model/model.json", "w") as json_file:
 json_file.write(model_json)
 f = open('model/history.pckl', 'wb')
 pickle.dump(hist.history, f)
 f.close()
 '''
def getSegmentation():
img = cv2.imread('myimg.png')
orig = cv2.imread('test1.png')
gray = cv2.cvtColor(img, cv2.COLOR_BGR2GRAY)
thresh = cv2.threshold(gray, 30, 255, cv2.THRESH_BINARY)[1]
contours=cv2.findContours(thresh,cv2.RETR_TREE,cv2.CHAIN_APPROX_SIMPLE)
contours = contours[0] if len(contours) == 2 else contours[1]
min_area = 0.95*180*35
 max_area = 1.05*180*35
```

```
result = orig.copy()
for c in contours:
 area = cv2.contourArea(c)
 cv2.drawContours(result, [c], -1, (0, 0, 255), 10)
 if area > min_area and area < max_area:
 cv2.drawContours(result, [c], -1, (0, 255, 255), 10)
return result
def TumourSegmentation():
global model
    filename = filedialog.askdirectory(initialdir="testSamples")
    img = cv2.imread(str(filename)+'/t2.png',0)
    img = cv2.resize(img,(64,64), interpolation = cv2.INTER_CUBIC)
    img = img.reshape(1,64,64,1)
    img = (img-127.0)/127.0
    preds = model.predict(img)
    preds = preds[0]
    print(preds.shape)
    orig = cv2.imread(str(filename)+'/t2.png',0)
    orig=cv2.resize(orig,(300,300),interpolation= cv2.INTER_CUBIC)
cv2.imwrite("test1.png",orig)
flair = cv2.imread(str(filename)+'/flair.png',0)
flair=cv2.resize(flair,(300,300),interpolation=cv2.INTER_CUBIC)
t1 = cv2.imread(str(filename)+'/t1.png',0)
t1 = cv2.resize(t1,(300,300),interpolation = cv2.INTER_CUBIC)
t1ce = cv2.imread(str(filename)+'/t1ce.png',0)
t1ce=cv2.resize(t1ce,(300,300),interpolation= cv2.INTER_CUBIC)
preds=cv2.resize(preds,(300,300),interpolation=cv2.INTER_CUBIC)
    cv2.imwrite("myimg.png",preds*255)
    preds = getSegmentation()
    cv2.imshow('Flair Image',flair)
    cv2.imshow('T1',t1)
    cv2.imshow("T1ce Image",t1ce)
    cv2.imshow('T2 Image',orig)
    cv2.imshow("Label Image",preds)
    cv2.waitKey(0)
    def graph():
    f = open('model/history.pckl', 'rb')
    data = pickle.load(f)
    f.close()
    dice = data['dice_coef']
    for i in range(len(dice)):
        dice[i] = dice[i] * 2
    plt.figure(figsize=(10,6))
    plt.grid(True)
    plt.xlabel('Iterations')
    plt.ylabel('Dice Score')
    plt.plot(dice, 'ro-', color = 'green')
    plt.legend(['Dice Score'], loc='upper left')
```

```
#plt.xticks(wordloss.index)
   plt.title('Iteration Wise Dice Score Graph')
   plt.show()
font = ('times', 16, 'bold')
title = Label(gui, text='Brain Tumour Image Segmentation Using Deep Networks')
title.config(bg='LightGoldenrod1', fg='medium orchid')
title.config(font=font)
title.config(height=3, width=120)
title.place(x=0,y=5)
font1 = ('times', 12, 'bold')
text=Text(gui,height=20,width=100)
scroll=Scrollbar(text)
text.configure(yscrollcommand=scroll.set)
text.place(x=10,y=300)
text.config(font=font1)
font1 = ('times', 12, 'bold')
loadButton = Button(gui, text="Upload BRATS Dataset", command=uploadDataset)
loadButton.place(x=50,y=100)
loadButton.config(font=font1)
uploadButton = Button(gui, text="Generate CNN & UNET Model", command=generateModel)
uploadButton.place(x=50,y=150)
uploadButton.config(font=font1)
descButton = Button(gui, text="Upload Test Image & Segmentation", command=TumourSegmentation)
descButton.place(x=50,y=200)
descButton.config(font=font1)
closeButton = Button(gui, text="Dice Similarity Graph", command=graph)
closeButton.place(x=50,y=250)
closeButton.config(font=font1)

gui.config(bg='OliveDrab2')
gui.mainloop()
```

7. RESULTS

7. RESULTS

To run project double click on ‘run.bat’ file to get below screen

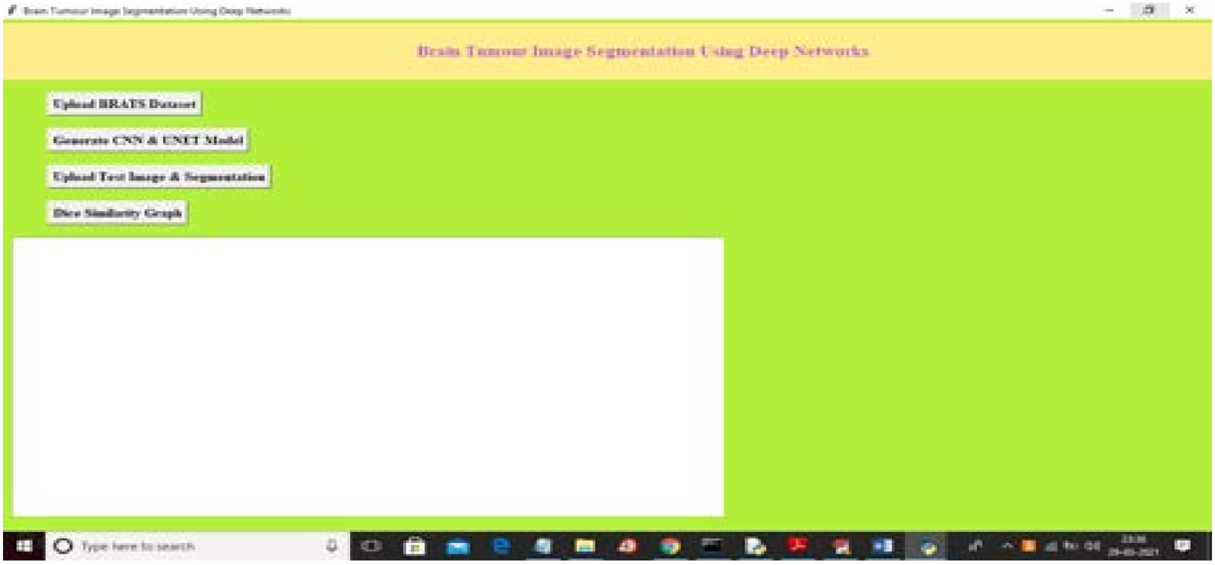

Brain Tumour Image Segmentation Using Deep Networks
Upload BRATS Dataset
Generate CNN & UNET Model
Upload Test Image & Segmentation
Dice Similarity Graph
Type here to search

Figure No 7.1 : CLICKING ‘run.bat’ FILE

In above screen click on ‘Upload BRATS Dataset’ button to upload dataset

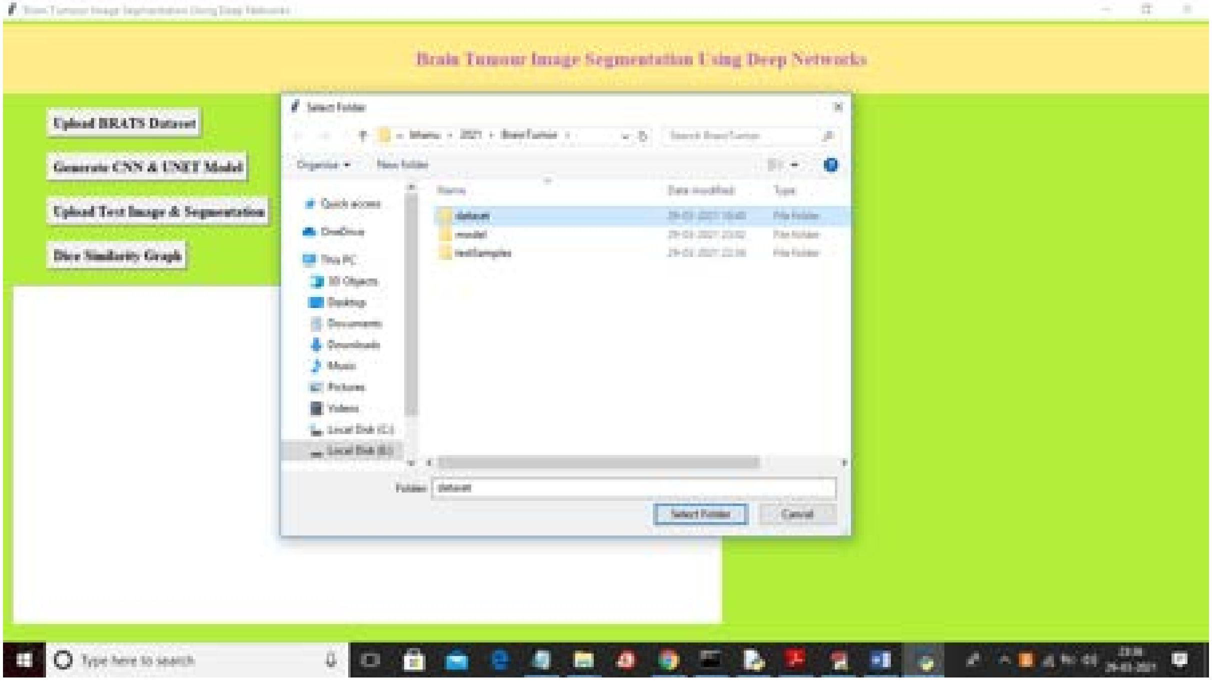
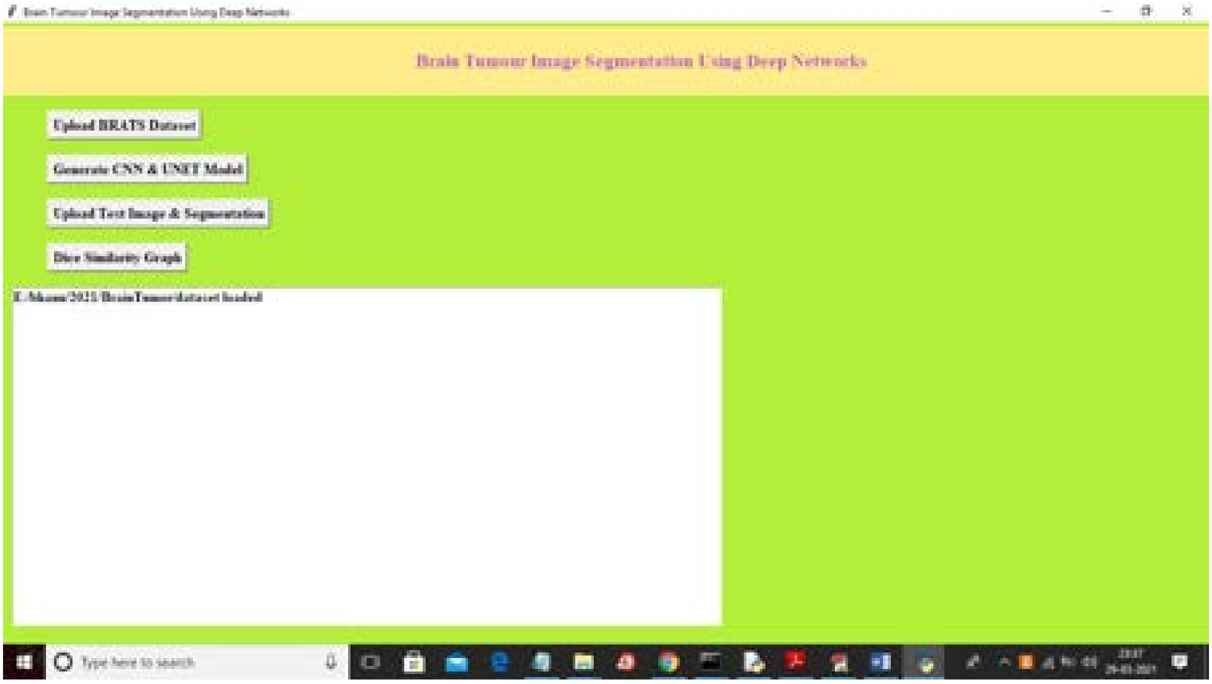


Figure No 7.2: UPLOADING DATASET

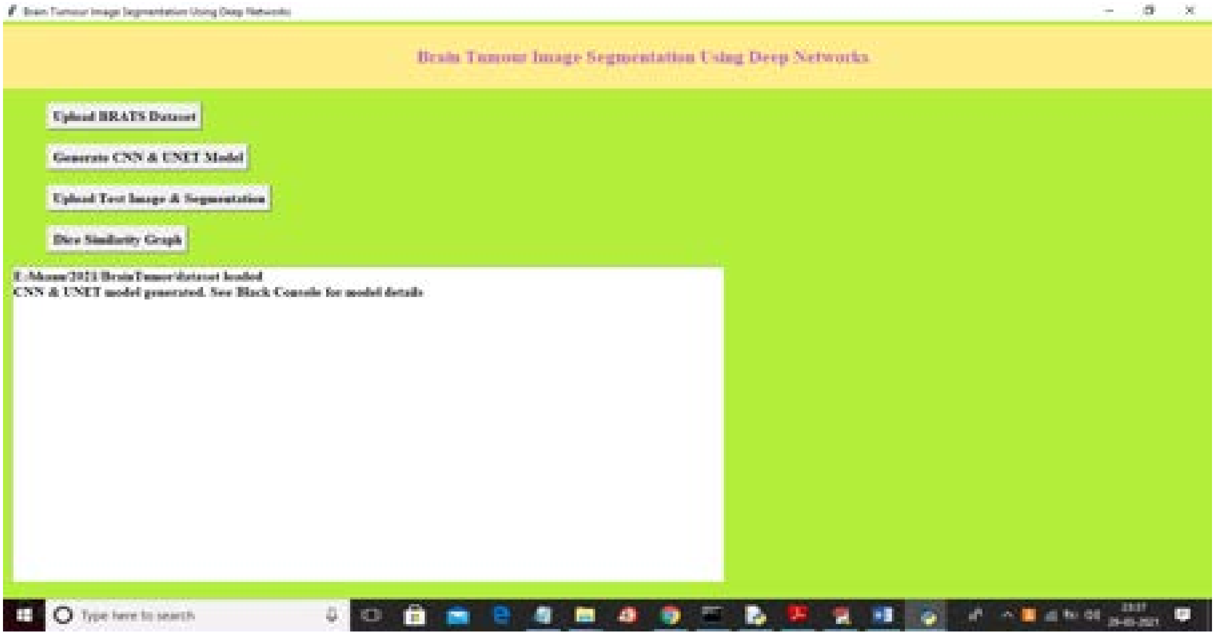
In above screen selecting and uploading ‘dataset’ folder and then click on ‘Select Folder’ button to load dataset and to get below screen


Brain Tumour Image Segmentation Using Deep Networks
Brain Tumour Image Segmentation Using Deep Networks
Upload BRATS Dataset
Generate CNN & UNET Model
Upload Test Image & Segmentation
Dice Similarity Graph
Type here to search

E:/Akram/2021/BrainTumor/dataset loaded

Figure No 7.3: GENERATING CNN & UNET MODEL

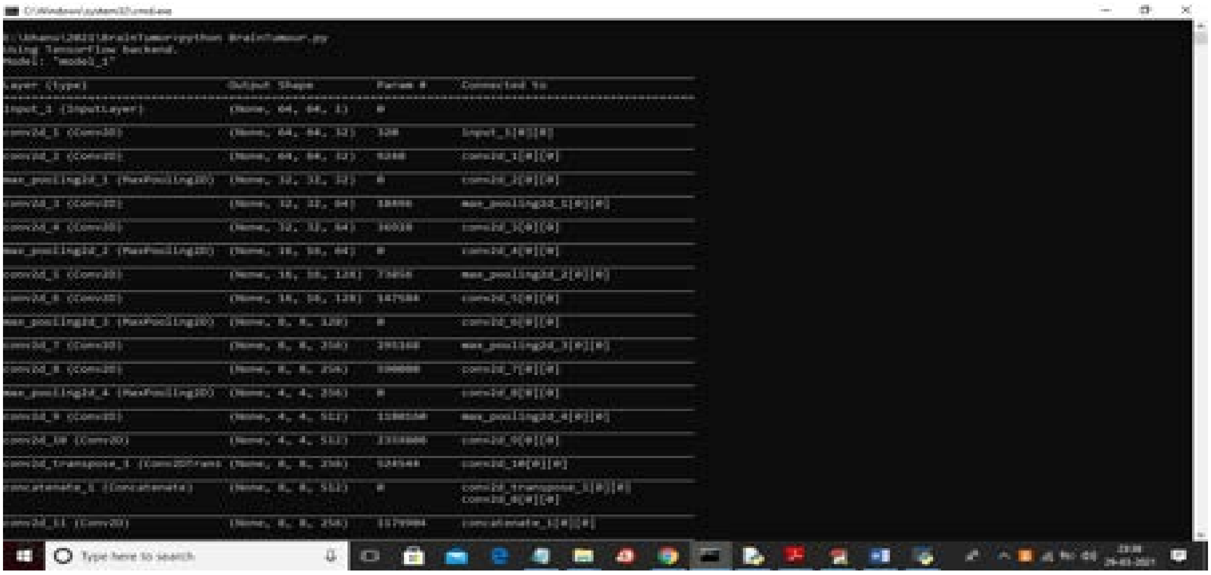
In above screen dataset loaded and now click on ‘Generate CNN & UNET Model’ button to generate models and to get below screen


Brain Tumour Image Segmentation Using Deep Networks
Brain Tumour Image Segmentation Using Deep Networks
Upload BRATS Dataset
Generate CNN & UNET Model
Upload Test Image & Segmentation
Dice Similarity Graph
Type here to search

E:/Akram/2021/BrainTumor/dataset loaded
CNN & UNET model generated. See Black Console for model details

Figure No 7.4 : GENERATING CNN

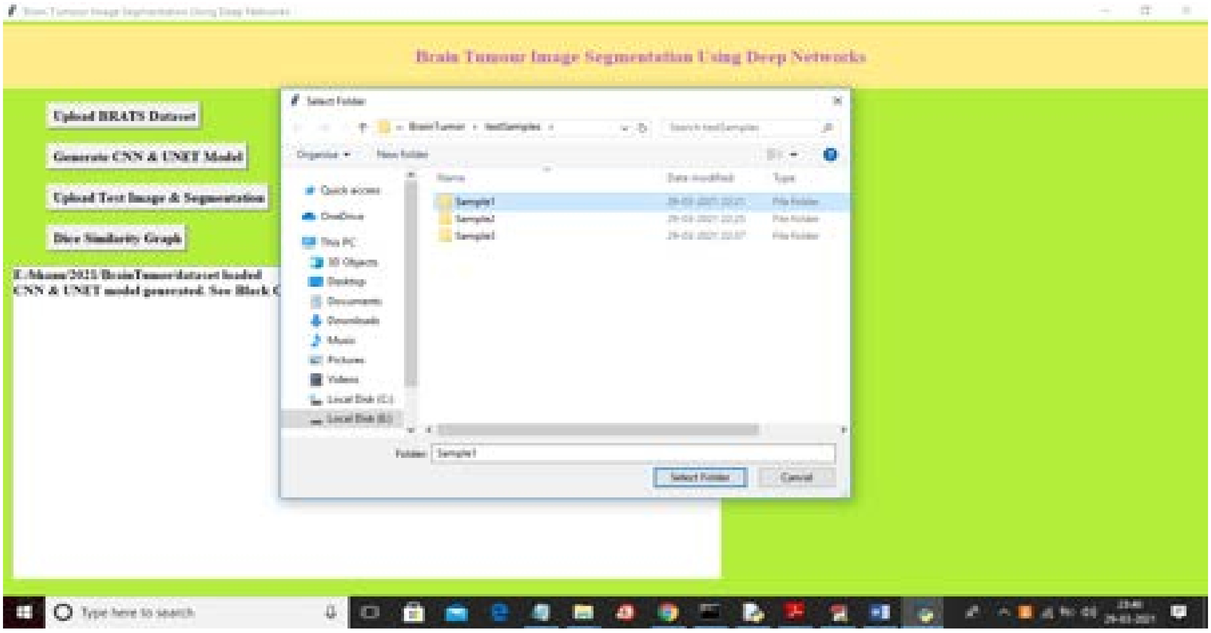
In above screen we can see both models are generated and we can see below black console to see CNN and UNET layer details



```
E:\kkanu\2021\BrainTumour>python BrainTumour.py
Using TensorFlow backend.
Model: "model_1"
__________________________________________________________________________________________________
Layer (type)                    Output Shape         Param #     Connected to
==================================================================================================
input_1 (InputLayer)            (None, 64, 64, 1)    0
__________________________________________________________________________________________________
conv2d_1 (Conv2D)               (None, 64, 64, 32)   320         input_1[0][0]
__________________________________________________________________________________________________
conv2d_2 (Conv2D)               (None, 64, 64, 32)   9248        conv2d_1[0][0]
__________________________________________________________________________________________________
max_pooling2d_1 (MaxPooling2D)  (None, 32, 32, 32)   0           conv2d_2[0][0]
__________________________________________________________________________________________________
conv2d_3 (Conv2D)               (None, 32, 32, 64)   18496       max_pooling2d_1[0][0]
__________________________________________________________________________________________________
conv2d_4 (Conv2D)               (None, 32, 32, 64)   36928       conv2d_3[0][0]
__________________________________________________________________________________________________
max_pooling2d_2 (MaxPooling2D)  (None, 16, 16, 64)   0           conv2d_4[0][0]
__________________________________________________________________________________________________
conv2d_5 (Conv2D)               (None, 16, 16, 128)  73856       max_pooling2d_2[0][0]
__________________________________________________________________________________________________
conv2d_6 (Conv2D)               (None, 16, 16, 128)  147584      conv2d_5[0][0]
__________________________________________________________________________________________________
max_pooling2d_3 (MaxPooling2D)  (None, 8, 8, 128)    0           conv2d_6[0][0]
__________________________________________________________________________________________________
conv2d_7 (Conv2D)               (None, 8, 8, 256)    295168      max_pooling2d_3[0][0]
__________________________________________________________________________________________________
conv2d_8 (Conv2D)               (None, 8, 8, 256)    590080      conv2d_7[0][0]
__________________________________________________________________________________________________
max_pooling2d_4 (MaxPooling2D)  (None, 4, 4, 256)    0           conv2d_8[0][0]
__________________________________________________________________________________________________
conv2d_9 (Conv2D)               (None, 4, 4, 512)    1180160     max_pooling2d_4[0][0]
__________________________________________________________________________________________________
conv2d_10 (Conv2D)              (None, 4, 4, 512)    2359808     conv2d_9[0][0]
__________________________________________________________________________________________________
conv2d_transpose_1 (Conv2DTrans (None, 8, 8, 256)    524544      conv2d_10[0][0]
__________________________________________________________________________________________________
concatenate_1 (Concatenate)     (None, 8, 8, 512)    0           conv2d_transpose_1[0][0]
                                                                 conv2d_8[0][0]
__________________________________________________________________________________________________
conv2d_11 (Conv2D)              (None, 8, 8, 256)    1179904     concatenate_1[0][0]
```

Figure No 7.5: UPLOADING TEST IMAGE AND SEGMENTATION

In above screen we can see models are using different size images to filter them and to get best features from it to build efficient model and now model is generate and now click on ‘Upload Test Image & Segmentation’ button and then upload test samples to get segmented output



E:/kkanu/2021/BrainTumour/dataset loaded
CNN & UNET model generated. See Black C

Folder: Sample1

Figure No 7.6 : SAMPLE FOLDER

In above screen selecting and uploading ‘Sample1’ folder and then click on ‘Select Folder’ button to get below output

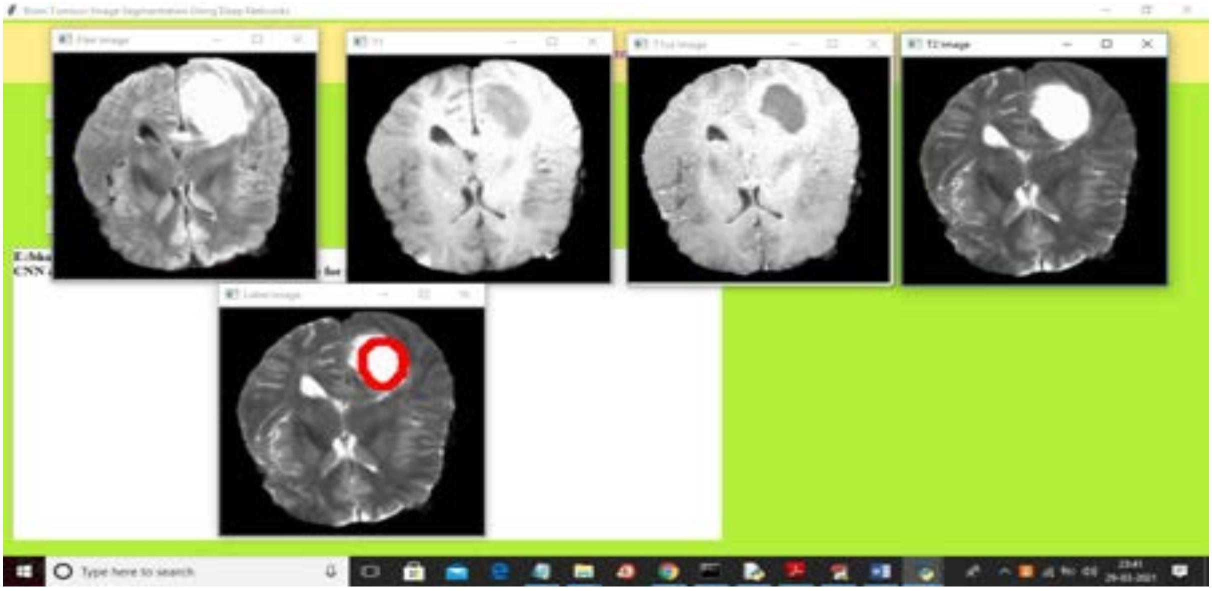
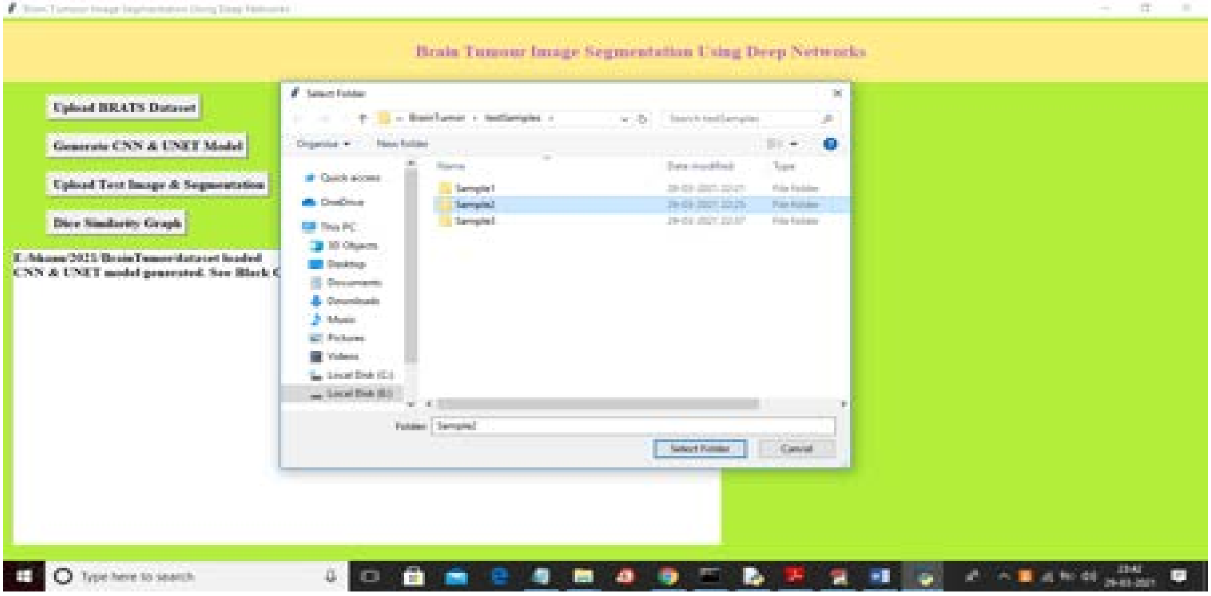


Figure No 7.7: PREDICTED TUMOUR IMAGE

In above screen top 4 images are the input images such as FLAIR, T1, T2 and T1CE and 5th image is the predicted image with segmented part showing in red colour and this algorithm correctly detecting and marking tumour area and now test with other image



E:/kkaau/2021/BrainTumor/dataset loaded
CNN & UNET model generated. See Black C

Figure No 7.8: SAMPLE FOLDER

In above screen I am selecting and uploading ‘Sample2’ folder and then click on ‘Select Folder’ button to load images and to get below output

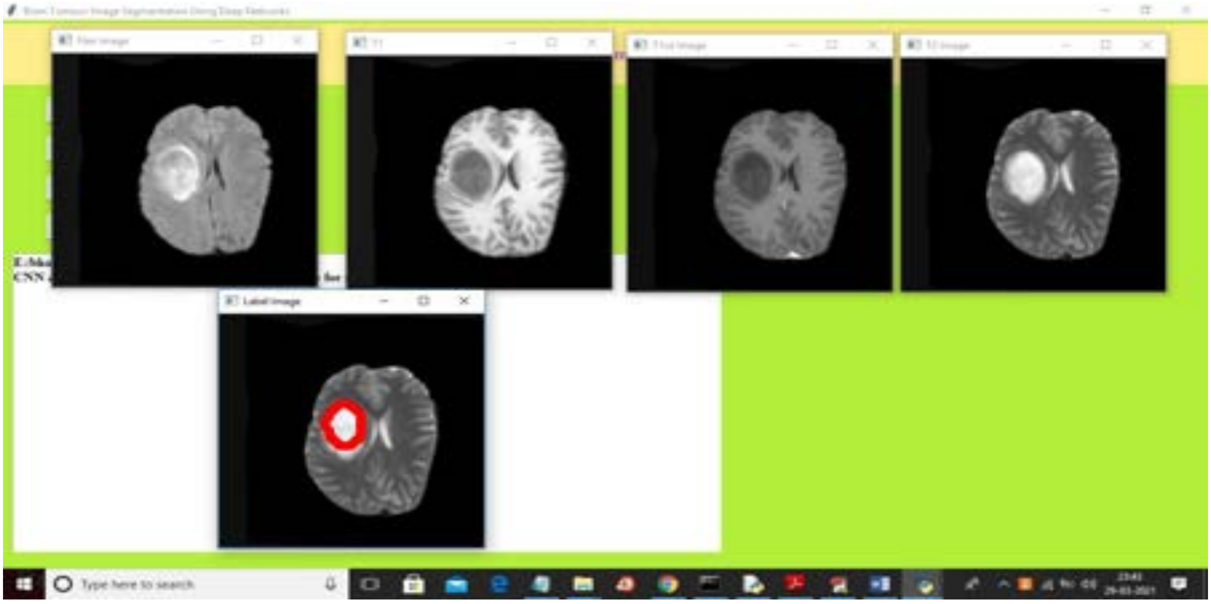


Figure No 7.9 : PREDICTED TUMOUR IMAGE

In above screen first 4 images are the input images and fifth image is the predicted label image with segmented parts around tumour area. Now click on 'Dice Similarity Graph' button to get below graph

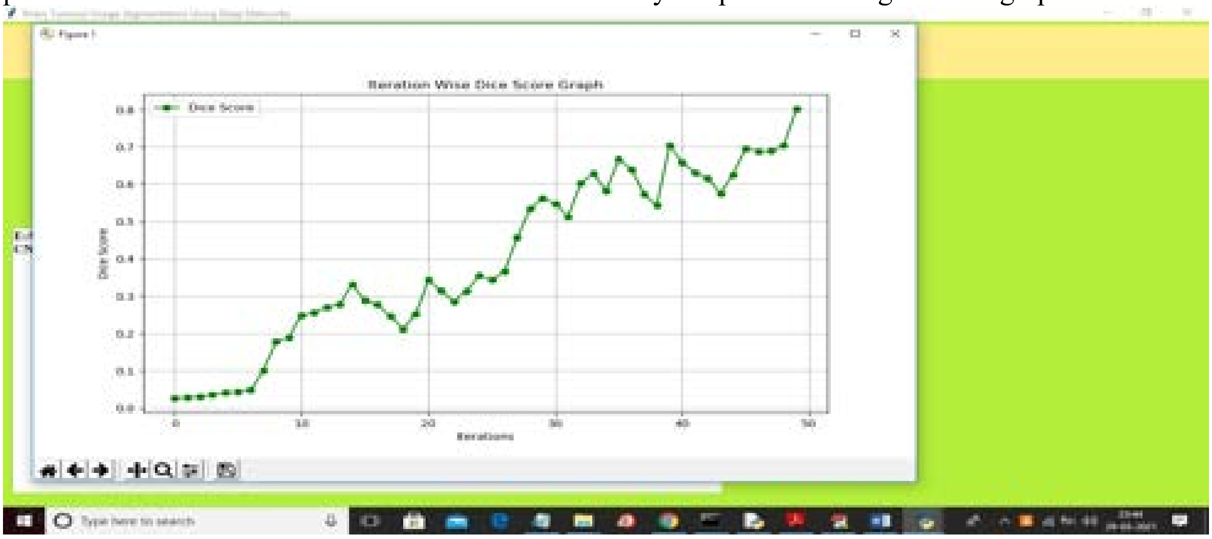


Figure No 7.9.1 : DIICE SIMILARITY GRAPH

To build CNN and UNET model we took 50 epoch or iterations and at each iteration DICE score between training and testing images get better and better and we get final dice score as 0.8 * 100 = 80%. In above graph x-axis represents epoch and y-axis represents dice score

8. CONCLUSION

8.CONCLUSION

In this project, we have described an ensemble of two networks, both of which are individually used frequently on the task of biomedical image segmentation. The ensemble successfully generates highly accurate segmentation of brain tumours from the multimodal MRI scans as provided by the BRATS 2019 challenge, which compares favourably with predictions given from various other state of the art models. We use a method of variable ensembling to combine the respective outputs from the model to achieve the best scores. The proposed ensemble offers an automated and objective method of generating brain tumour segmentation to aid in disease planning and patient management clinically.

9. BIBILOGRAPHY

9.1: REFERENCES

[1] S. Bauer, R. Wiest, L. P. Nolte, and M. Reyes. (2013). A Survey of MRI-Based Medical Image Analysis for Brain Tumour Studies. [Online]. Available: http://www

[2] R. Leece, J. Xu, Q. T. Ostrom, Y. Chen, C. Kruchko, and J. S. Barnholtz-Sloan, ‘‘Global incidence of malignant brain and other central nervous system tumors by histology, 2003–2007,’’ NeuroOncology, vol. 19, no. 11, pp. 1553–1564, Oct. 2017.

[3] T. A. Dolecek, J. M. Propp, N. E. Stroup, and C. Kruchko, ‘‘CBTRUS statistical report: Primary brain and central nervous system tumours diagnosed in the United States in 2005–2009,’’ Neuro. Oncol., vol. 14, no. 5, pp. v1–v49, 2012.

[4] D. N. Louis, A. Perry, G. Reifenberger, A. von Deimling, D. Figarella-Branger, W. K. Cavenee, H. Ohgaki, O. D. Wiestler, P. Kleihues, and D. W. Ellison ‘‘The 2016 World Health Organization classification of tumours of the central nervous system: A summary,’’ Acta Neuropathol., vol. 131, no. 6, pp. 803–820, 2016.

[5] R. Stupp et al., ‘‘Radiotherapy plus concomitant and adjuvant temozolomide for glioblastoma,’’ New England J. Med., vol. 352, no. 10, pp. 987–996, 2005.

[6] S. Bakas et al., ‘‘Identifying the best machine learning algorithms for brain tumor segmentation, progression assessment, and overall survival prediction in the BRATS challenge,’’ 2018, arXiv:1811.02629. [Online]. Available: https://arxiv.org/abs/1811.02629

[7] B. H. Menze, K. van Leemput, D. Lashkari, M.-A. Weber, N. Ayache, and P. Golland, ‘‘A generative model for brain tumor segmentation in multimodal images,’’ in Proc. Int. Conf. Med. Image Comput. Comput.-Assist. Intervent., 2010, pp. 151–159.

[8] S. Bakas, H. Akbari, A. Sotiras, M. Bilello, M. Rozycki, J. S. Kirby, J. B. Freymann, K. Farahani, and C. Davatzikos, ‘‘Advancing the cancer genome atlas glioma MRI collections with expert segmentation labels and radiomic features,’’ Sci. Data, vol. 4, no. 1, Dec. 2017, Art. no. 170117.

[9] S. Bakas et al., ‘‘Segmentation labels and radiomic features for the preoperative scans of the TCGA-LGG collection,’’ Cancer Imag. Arch., vol. 286, 2017.

[10] S. Bakas, H. Akbari, A. Sotiras, M. Bilello, M. Rozycki, J. Kirby, J. Freymann, K. Farahani, and C. Davatzikos ‘‘Segmentation labels and radiomic features for the pre-operative scans of the

TCGA-GBM collection. The cancer imaging archive,'' Nat. Sci. Data, vol. 4, 2017, Art. no. 170117.

[11] H.. Dong et al., ''The multimodal brain tumor image segmentation benchmark (BRATS),'' IEEE Trans. Med. Imag., vol. 34, no. 10, pp. 1993–2024, Oct. 2015.

[12] S. Pereira, A. Pinto, V. Alves, and C. A. Silva, ''Brain tumor segmentation using convolutional neural networks in MRI images,'' IEEE Trans. Med. Imag., vol. 35, no. 5, pp. 1240–1251, May 2016.

[13] M. B. Naceur, R. Saouli, M. Akil, and R. Kachouri, ''Fully automatic brain tumor segmentation using end-to-end incremental deep neural networks in MRI images,'' Comput. Methods Programs Biomed., vol. 166, pp. 39–49, Nov. 2018.

[14] M. Goetz, C. Weber, F. Binczyk, J. Polanska, R. Tarnawski, B. Bobek-Billewicz, U. Koethe, J. Kleesiek, B. Stieltjes, and K. H. Maier-Hein, ''DALSA: Domain adaptation for supervised learning from sparsely annotated MR images,'' IEEE Trans. Med. Imag., vol. 35, no. 1, pp. 184–196, Jan. 2016.

[15] K. Farahani, B. Menze, and M. Reyes. (2014). Brats 2014 Challenge Manuscripts (2014). [Online]. Available: http//www. braintumoursegmentation.org

9.2: WEBSITES:

[16]https://www.nature.com/articles/s41598-021-90428-8

[17]https://www.researchgate.net/publication/341465722_Brain_Tumor_Segmentation_Using_3D_Convolutional_Neural_Network

[18]https://www.ncbi.nlm.nih.gov/pmc/articles/PMC8214948/

[19]https://paperswithcode.com/task/brain-tumor-segmentation

[20]https://arxiv.org/abs/1807.08599