1. Introduction

As of the end of 2019, COVID-19 has spread in various countries, cancer patients are a high-risk group of getting severely ill from COVID-19. The tumor of location, type of pathology, shape, size and density are important features for discriminating between benign and malignant tumors. To describe irregular signals with statistical regularity in medical image, two-dimensional fractional Brownian motion (2D FBM) is a good model, which can be easily described by the Hurst exponent [1].

In addition, edge detection of medical images is one of the most important elements of this field, already there is a lot of work which has been done in brain tumor detection. For the detection of tumor CT or MRI images are uses. Corresponding medical equipment of these types brings a noticeable fraction of noise on the obtained images. Therefore, noise suppression is a necessary step to improve the accuracy of the analysis.

A fundamental question for edge detection in noisy images is how faint can an edge be and still be detected [2]. Thus, there are some strategies have been proposed to solve this problem, canny edge detection method can apply to detect the fine edges, Gaussian filter, median filter and mean filter are selected for noise reduction [3][4].

2. Method

The existing methods of tumor detection and evaluation are divided into region-based and contour-based methods. Region-based methods take advantage of only local information for each pixel and do not include shape and boundary information. Contour-based methods rely on the evolution of a curve, such as image gradient, to delineate the boundary of tumor structure or pathology. Many researchers use a wide range of techniques based on segmentation to solve the problem of localizing and analyzing the characteristics of a tumor.

The basic requirement of edge detection is to obtain a lower false judgment rate and a higher positioning accuracy as far as possible. However, in order to achieve the above goal, the algorithm needs to eliminate the noise interference as much as possible, detect the correct edge at the same time, or find a balance between the two. Most of the well-study methods are to improved the classical Canny edge detection by the different strategies.

Hamad et al. [3] proposed a algorithmic scheme for processing and analyzing the medical images. During the image enhancement step, noise reduction and contrast enhancement is conducted. To improve the contrast for highlighting the area of interest we proposed to use Balance Contrast Enhancement Technique (BCET). After image enhancement, it is suggested to use preliminary segmentation of the medical image to determine the boundaries of the area of interest most accurately. Fuzzy C-Means (FCM) clustering method and Canny edge detector were chosen.

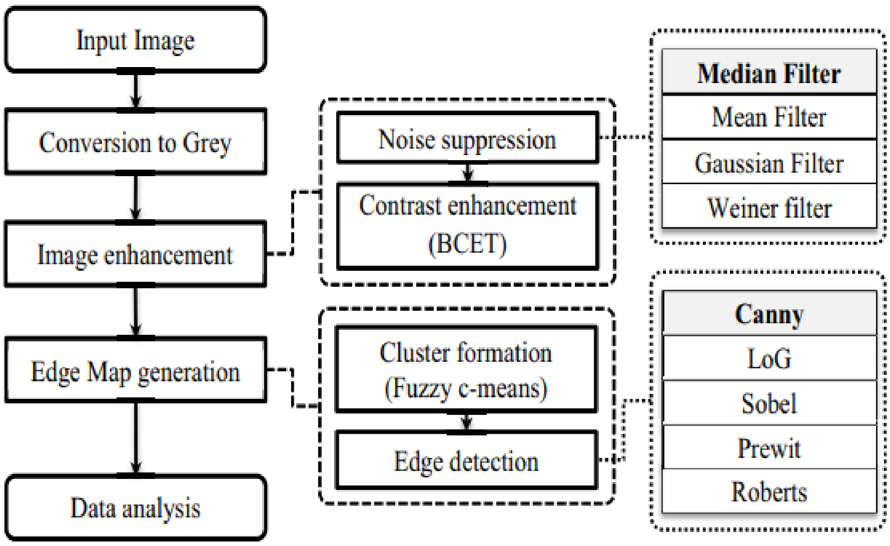


Fig1. The proposed procedure for image preparation

Zorana et al. [4] proposed improved Canny algorithm as the first step applies the LoG filter, which emphasizes the region of fastest intensity change. After applying LoG filter, the proposed method follows the usual steps, but modifying kernel for smoothing the image. The proposed kernel emphasizes the edges of the image (α =2, 4 or 8):

$$\begin{pmatrix} 1 & 1 & 1 \\ 1 & \alpha & 1 \\ 1 & 1 & 1 \end{pmatrix}$$

If we change the alpha parameter in Gaussian function, we will see clear differences:

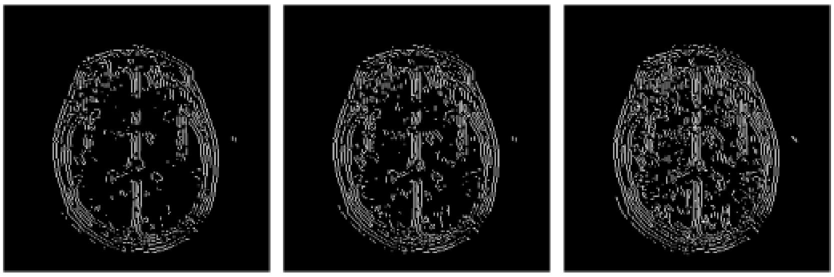


Fig2. The effect of Gaussian filter for α =2, 4 and 8

The second adjustment is modification of gradient magnitude. In this method, a fusion of two images is suggested. The first image is obtained by using the previously mentioned modified kernel, while the second image is obtained by using modified kernel and a modified gradient magnitude. The fusion of these two images guarantees the existence of all edges.

Xu et al. [5] studied a improved Canny operator using Otsu algorithm and double threshold detection method to strengthen the ability of Canny operator edge detection. Otsu algorithm was mentioned in the course, it automatically select a threshold that used to separate the pixel values into two groups, gives the maximum separability. This algorithm is simple to calculate and will not be affected by the contrast and brightness of the image under certain circumstances. On the other hand, the Canny edge detection algorithm uses double thresholding, Otsu algorithm has only one threshold, which is not conducive to double threshold detection. Therefore, this paper multiplied a scale coefficient to convert single threshold into double threshold.

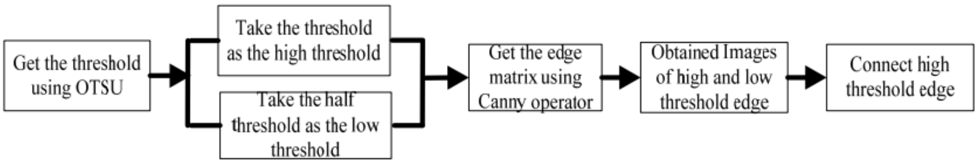


Fig3. The flow chart of user defined double threshold detection

3. Experimental results

- Hamad et al. [3]

The overview of a step-by-step formation of an edge map according to the proposed methodology:

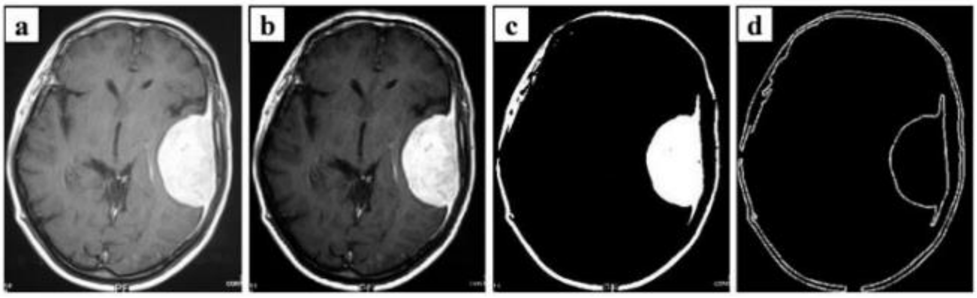


Fig4. Example of the proposed method work: a) original image, b) enhanced image, c) result after FCM clustering, d) final edge map

For the experimental research 35 brain MRI images were selected, they contain tumors that are characterized by different locations and different types of pathologies, shape, size and density.

To show that the proposed methodology has good ability of the edge detection and is resistant to average level of noise, the following studies were carried out. Comparing the proposed method with classical approaches to edge detection is based on simple gradient operators such as Roberts, Prewitt, Sobel and complex methods such as LoG and Canny.

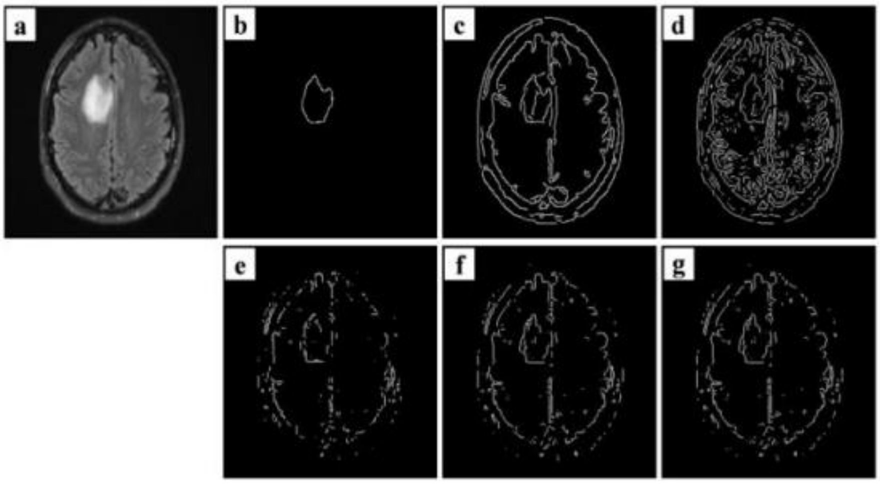
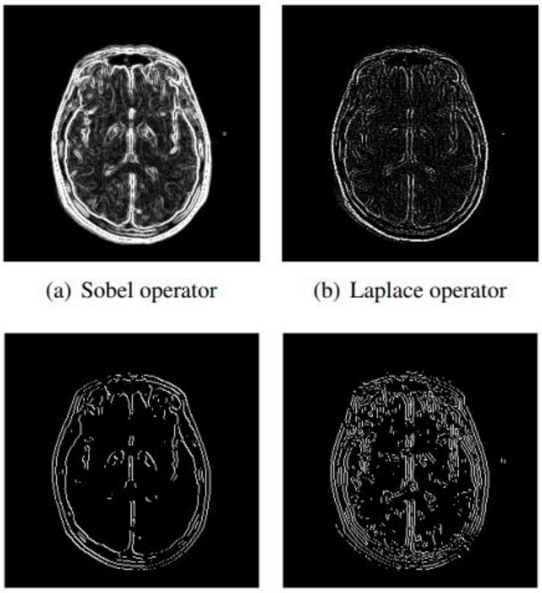


Fig5. Example of edge maps generation: a) original image, b) Proposed method, c) Canny, d) LoG, e) Roberts, f) Prewitt, g) Sobel

- Zorana et al. [4]

  The following figure is the result comparison between the traditional Canny edge detection and the proposed improved methodology for brain MRI images..

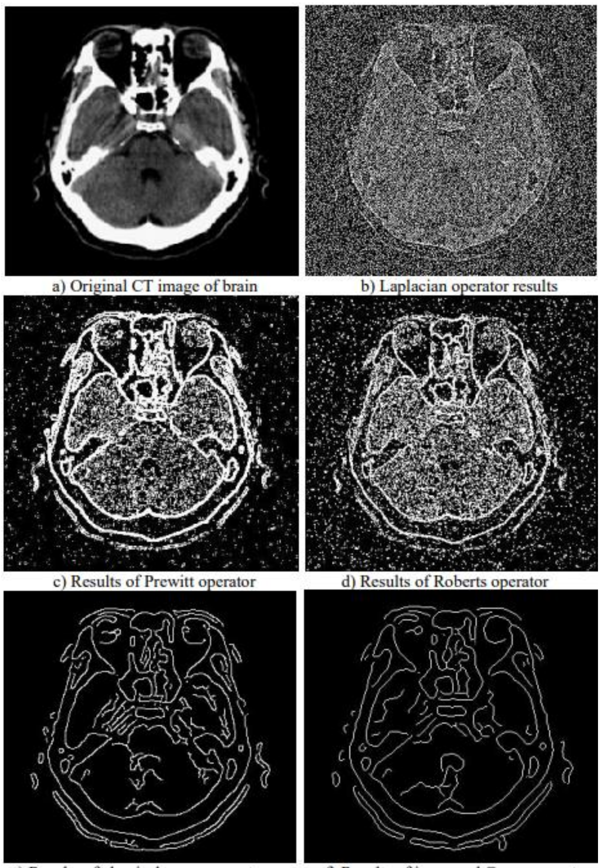


(a) Sobel operator (b) Laplace operator

(c) Canny operator (d) Improved Canny algorithm

- Xu et al. [5]

  In order to verify the practicability of the algorithm, they added the Gaussian noise to the original image. The following figure shows the results of edge detection on brain CT images.



a) Original CT image of brain b) Laplacian operator results c) Results of Prewitt operator d) Results of Roberts operator e) Results of classical canny operators f) Results of improved Canny operator

Table 1. The quantitative comparison of results of various operators in CT edge detection of brain

Algorithm	entropy	Mean square error	Operation time（s）
Noise image	29.224	74.475	—
Laplacian operators	7.712	55.233	0.216
Prewitt operators	59.589	89.631	0.288
Roberts operators	109.396	91.124	0.299
Classical canny operators	110.483	53.889	1.102
Improved Canny operator	78.913	32.798	1.501

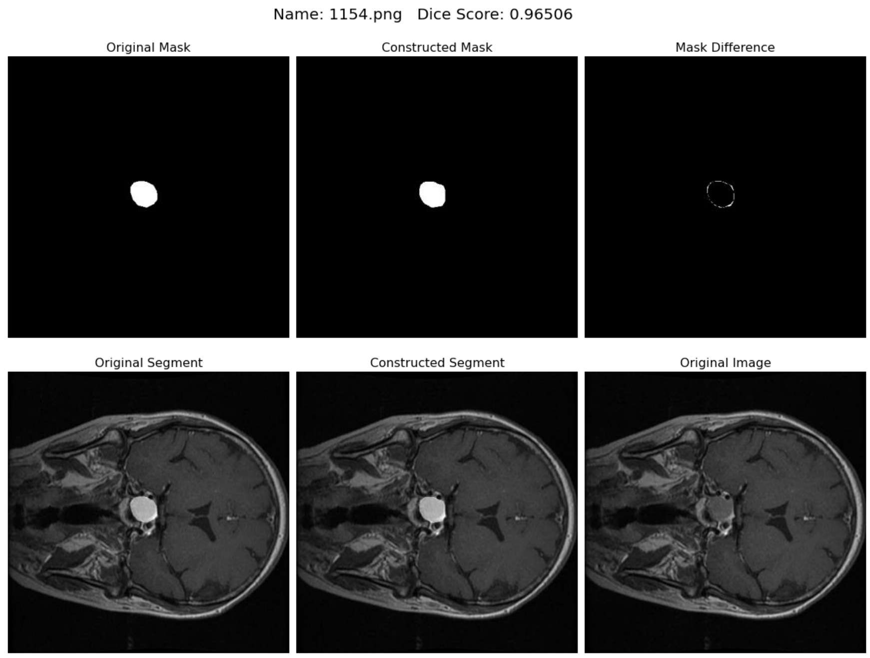
From the Table 1, the operation time of the proposed operator is longer than that of other operators because Canny operator and improved operator need more calculation, but generally speaking, the experimental results can be obtained in a short time.

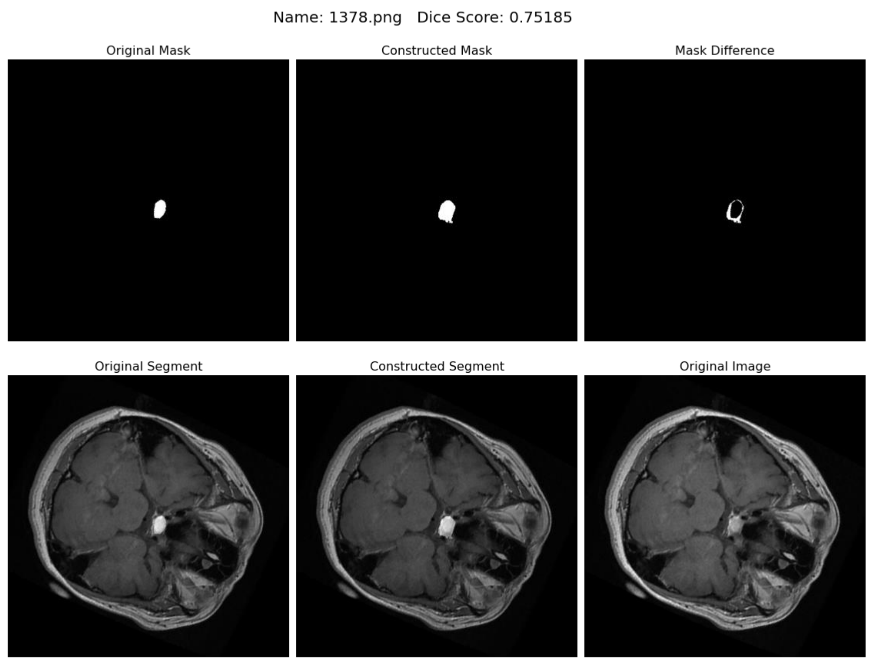
- My experimental results

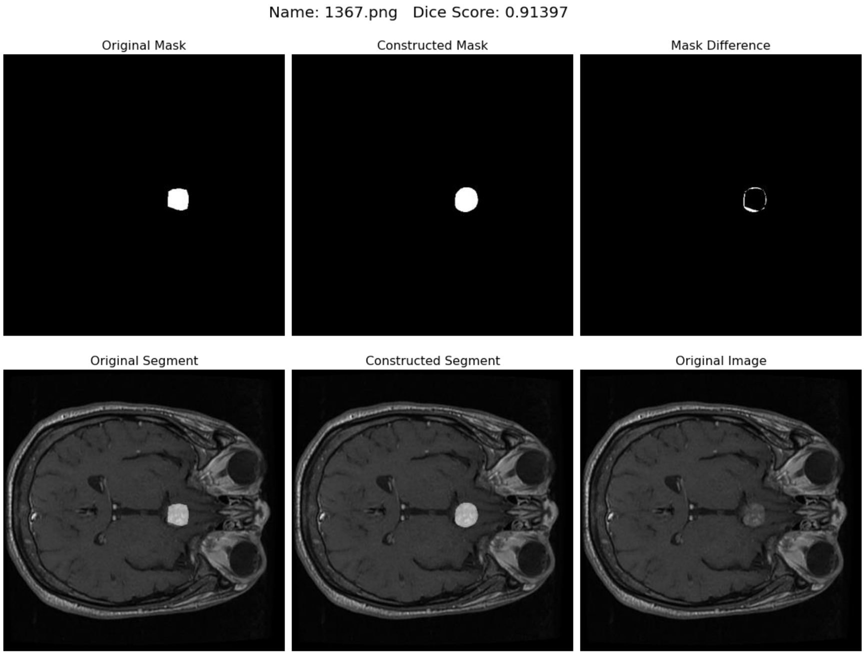
Hamad’s result (the first part of the experimental result) is what we expected, because their method can construct the contour representation of brain tumor. This is true that the other two methodologies improve the classical Canny edge detection, which can detect edges more detailed. However, we still don't know the locations or areas of the brain tumor.

Therefore, I thought of an intuitive method and hoped that it can perform as well as Hamad's method. This method is using UNet to segment the tumor, it is not a edge detection method, I'm still curious about its performance to mask the tumor region.

The codes and my images of the brain used for testing are reference here: https://github.com/sdsubhajitdas/Brain-Tumor-Segmentation. Some samples from our training dataset output are below.







4. Discussion

- Hamad et al. [3]

  From the previous part, we can see that local area of interest obtained by the different approaches of edge detection. From the edge of the whole brain image, the effect of the classical Canny edge detection is the clearest and has less noise effect, but it can't only represent the area of the brain tumor. The proposed method in this paper present an edge map of the object of interest with a better result. Because the input image is denoised with median filter and enhanced by BCET. Then, image is segmented by FCM clustering method and Canny edge detector is applied to construct the edge map of brain tumor.

  The remaining questions I think the generalization of the proposed method, because the images with the least pronounced noise influence were selected for comparison in this paper. Thus, we can't know whether selecting the input image with more noise to evaluate the proposed method can perform as well as the results in this paper.

- Zorana et al. [4]

  From the experimental results, it can be noticed that the improved Canny algorithm has the advantage to find more details than the original one. The proposed method produces small curves that are not fully connected. In this paper, they mentioned the cause of these disconnections is the performance of LoG filter. Laplacian filters have generalized application in area of discontinuity detection and image sharpening, because it reduces sensitivity to the orientation of the edges of the corners and curves whose intensity varies widely.

Although the image result of the improved algorithm can detect more complete information, it is too complex to efficiently recognize where the region of interest (brain tumor), this is a remaining problem in this paper.

- Xu et al. [5]

  Some conclusions can be seen from the experimental results. Except Canny operator and improved Canny operator, the other operators cannot filter out the noise interference well. It is obvious that Laplacian operator cannot recognize the edge, and Prewitt operator and Roberts operator are also seriously disturbed by noise, which leads to edge blur. Focus on Canny operator and the proposed operator, their improved Canny operator has fewer double edges.

  From Table 1, the experimental results are analyzed and compared from two objective evaluation metrics of entropy and mean square error. It can be seen that the entropy of the classical Canny operator is higher than that of other algorithms, which means that Canny operator has more image information and details. For the improved Canny operator, due to Gaussian filtering, many double edges are removed, so the information entropy is slightly less, but the detected edges are more accurate. After adding Gaussian noise, the mean square error of Laplacian operator, Prewitt operator and Roberts operator is greater than that of the noise image because the noise is not filtered well. Moreover, the improved algorithm has better filtering and noise suppression demonstrated by the mean square error of the improved Canny operator is lower than that of the classical Canny operator.

  One of the remaining questions in this paper is similar to the second paper, the output image is much clearer, but the edges of the tumor are still unrecognizable, which may be confused with the outline of other brain tissues.

- My experiment results

  The mean Dice Score our model gained was 0.7881 in testing dataset of 600 images. So, we can conclude that in our testing dataset our constructed mask has a similarity of about 78% with the original mask. I think the above results have met my original expectations. From the output figures, I can see the original mask and our constructed mask at a glance. Then, the similarity can be seen through their mask difference between the two images.

  The future work of this implementation is to try to images of the brain used for testing from web-based medical image depository [6], this can improve the generalization of the model

5. Reference

[1] Yen-Ching Chang, "An Efficient Maximum Likelihood Estimator For Two-Dimensional Fractional Brownian Motion," FRACTALS (fractals), World Scientific Publishing Co. Pte. Ltd., vol. 29(01), pages 1-15, February 2021
[2] N. Ofir, M. Galun, S. Alpert, A. Brandt, B. Nadler and R. Basri, "On Detection of Faint Edges in Noisy Images," in IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 42, no. 4, pp. 894-908, 1 April 2020, doi: 10.1109/TPAMI.2019.2892134.
[3] Hamad, Yousif & Simonov, Konstantin & Naeem, Mohammad. (2018). Brain's Tumor Edge Detection on Low Contrast Medical Images. 45-50. 10.1109/AiCIS.2018.00021.
[4] Zorana Stosic, Petar Rutesic. (2018) An Improved Canny Edge Detection Algorithm for Detecting Brain Tumors in MRI Images. International Journal of Signal Processing, 3, 11-15
[5] Xu, Ziqi & Ji, Xiaoqiang & Wang, Meijiao & Sun, Xiaobing. (2021). Edge detection algorithm of medical image based on Canny operator. Journal of Physics: Conference Series. 1955. 012080. 10.1088/1742-6596/1955/1/012080.
[6] K. A. Johnson and J. A. Becker, “The whole brain atlas [Online].” [Online]. Available: http://www.med.harvard.edu/AANLIB/

dataset.py, model.py, classifier.py and plot.py are reference from https://github.com/sdsubhajitdas/Brain-Tumor-Segmentation

Tumor Segmentation.ipynb

```
import os
import torch
import warnings
import pickle
warnings.filterwarnings('ignore')
from torch.utils.data import SubsetRandomSampler
import numpy as np
%matplotlib inline

import lib.dataset as dataset
import lib.model as model
import lib.classifier as classifier
import lib.plot as plot

device = torch.device('cuda' if torch.cuda.is_available() else 'cpu')

## Hyperparameters setting
# Dataset part used for testing
TEST_SPLIT = 0.2
# Batch size for training. Limited by GPU memory
# BATCH_SIZE = 6
# Dataset folder used
DATASET_USED = 'png_dataset'
# Full Dataset path
DATASET_PATH = os.path.join('dataset',DATASET_USED)
# Training Epochs
# EPOCHS = 100
# Filters used in UNet Model
FILTER_LIST = [16,32,64,128,256]
# Flag to train the model
TRAIN = False
# Flag to load saved model
LOAD_MODEL = True
# Flag to save model trained
SAVE_MODEL = False
```

```
# Model name to save or load.
MODEL_NAME = f"UNet-{FILTER_LIST}.pt"

print(f"Model Name :     {MODEL_NAME}")

## Dataset loading
def get_indices(length, new=False):

    # Pickle file location of the indices.
    file_path = os.path.join('dataset',f'split_indices_{DATASET_USED}.p')
    data = dict()
    if os.path.isfile(file_path) and not new:
        # File found.
        with open(file_path,'rb') as file :
            data = pickle.load(file)
            return data['train_indices'], data['test_indices']
    else:
        # File not found or fresh copy is required.
        indices = list(range(length))
        np.random.shuffle(indices)
        split = int(np.floor(TEST_SPLIT * len(tumor_dataset)))
        train_indices , test_indices = indices[split:], indices[:split]
        # Indices are saved with pickle.
        data['train_indices'] = train_indices
        data['test_indices'] = test_indices
        with open(file_path,'wb') as file:
            pickle.dump(data,file)
    return train_indices, test_indices

tumor_dataset = dataset.TumorDataset(DATASET_PATH)

train_indices, test_indices = get_indices(len(tumor_dataset))
train_sampler, test_sampler = SubsetRandomSampler(train_indices),
SubsetRandomSampler(test_indices)

trainloader = torch.utils.data.DataLoader(tumor_dataset, BATCH_SIZE,
sampler=train_sampler)
testloader = torch.utils.data.DataLoader(tumor_dataset, 1, sampler=test_sampler)
```

```
## Model declaration
unet_model = None
unet_classifier = None
if not LOAD_MODEL:
    # New model is created.
    unet_model = model.DynamicUNet(FILTER_LIST).to(device)
    unet_classifier = classifier.BrainTumorClassifier(unet_model,device)
else:
    # Saved model is loaded on memory.
    unet_model = model.DynamicUNet(FILTER_LIST)
    unet_classifier = classifier.BrainTumorClassifier(unet_model,device)
    unet_classifier.restore_model(os.path.join('saved_models',MODEL_NAME))
    print('Saved model loaded')

## Model training
# Training process
if TRAIN:
    unet_model.train()
    path = os.path.join('saved_models',MODEL_NAME) if SAVE_MODEL else
None
    unet_train_history =
unet_classifier.train(EPOCHS,trainloader,mini_batch=100,save_best=path)
    print(f'Training Finished after {EPOCHS} epoches')

# Testing process on test data.
unet_model.eval()
unet_score = unet_classifier.test(testloader)
print(f'\n\nDice Score {unet_score}')

## Visualize test dataset predictions
# Run this cell repeatedly to see some results.
i = 0
image_index = test_indices[i]
sample = tumor_dataset[image_index]
image, mask, output, d_score = unet_classifier.predict(sample,0.65)
title = f'Name: {image_index}.png     Dice Score: {d_score:.5f}'
# save_path = os.path.join('images',f'{d_score:.5f}_{image_index}.png')
plot.result(image,mask,output,title,save_path=None)
```

```
i += 1
if i >= len(test_indices):
    i = 0
```