DATA MINING AND MACHINE LEARNING
BRAIN SEGMENTATION PROJECT

Introduction to the Methods Used:

U-Net Model:

U-Net is a particularly suitable segmentation network architecture where it handles demanding tasks like medical imaging. It follows an encoder-decoder structure as mentioned below.

Encoder: It plays a role similar to the feature detector. It sends the input MRI image for processing using a subsequent chain of convolution and pooling layers. By the layers, these extraction sets the high-level features from stipulated images, in this way, the system can give aided with patterns and shapes that are to be used in segmenting of cancer tissues.

Decoder: In this part, the procedure passes the source of information on to the pertinent features, which are subsequently upsampled to produce better image resolution. To do this, it then converges these upsampled features with the coresponding features from the decoder tracked at the different resolutions. This enables the tumour area within the original image can be captured more precisely.

Unlike many other models that employ fully connected layers, U-Net introduced non-linearity into the architecture by using activation functions such as ReLU which allow the model to learn complex relationships between the image features and the segmentation mask.

Along with these, the model comprise dropout layers which essentially prevent overfitting. Overfitting (also known as memorisation) happens when model becomes too specific to the training data and has problems with the unseen data. Dropout causes the model to recognize more general features by means of randomly dropping neurons during the training.

Custom Data Generator:

A custom data creator is implemented to manage data loading and preprocessing during the machine learning models. This approach offers several benefits like memory efficiency, on-the-fly preprocessing and customization as mentioned below.

Memory Efficiency: By loading data in batches, the generator reduces the memory footprint required to train the model, especially when dealing with large medical images.

On-the-fly Preprocessing: The generator processes the data in "near time" (e.g. doing reshaping of MRI slices, performing normalisation of intensity values) during the training. This significantly reduces the need for advanced preprocessing by computing it directly from the data itself, thus saving time and computational power.

Customization: The generator allows for flexibility in defining preprocessing steps specific to the needs of the model and data.

Description of the preprocessing steps:

Steps in preprocessing the data are specifically dedicated to ensuring that brain tumour localization would be well-grounded on effective training of the U-Net model using brain tumour segmentation. Here's a breakdown of the key steps implemented in the provided code.

Data Path Definition: The main step is defining the path name to the dataset directory where the MRI scan datasets are mostly in different modalities (e.g. T1CE and FLAIR) and the segmentation masks (usually in .numpy format). This lets the code to look for the training data where it is stored and access it.

Custom Data Generator: Data generation is one of the core the components of preprocessing, and we use a custom data generator class to conduct this. It is responsible for pulling and assembling data in batches when training. Below are the steps involved in the data generator.

- **MRI Slice Resizing:** The generator will adapt every MRI image slice to a given dimension (e.g., 128×128px). Hence the input data format for the U-Net model remains constistent, which always requires the input to be a fixed input size.

- **Data Loading and Preprocessing:** It loads both MRI data (like T1CE and FLAIR modalities) and corresponding segmentation masks.

- **Segmentation Mask Processing:** In majority of algorithms, the segmentation masks identify separate regions with alternative tumour subtypes and then label them clearly. Using this generator, the labels are converted into a formatted output that one-hot encoded the values. One-hot vector is a method that represented each word by a binary vector with items of them set to zero all the time except for the element corresponds to the word itself, and it would set to one.

- **Data Normalisation:** The generator can normalize the MRI data by dividing the intensity values by the maximum data value in the set. This process is called normalisation, and it does so by keeping all data segments on the same scale thus enabling the model to learn better.

Detailed methodology of the segmentation processes:

Segmentation of the segmenting agent in this project relies on the U-Net model, making a particular architecture to advance the image segmentation tasks. Methodology is mentioned below.

1. U-Net Architecture:

The U-Net model follows the below architecture with an encoder-decoder structure.

Encoder:

- Process requires magnetic resonance imaging, which is in this mode usually performed more than one channel, where each channel represents a different modality information (for example T1CE, FLAIR).
- The network architect employs the convolutional layers for hiding details from image. They impose structure and bring into the picture shapes and forms which are directly relevant to the process of brain tumour segmentation.
- Uses pooling layer for feature maps downsampling. As a result of this, we get a modest image resolution but the number of feature channels that enable the model to recognize higher level features that are invariant to small shifts in spatial status has increased.
- Layers in a stack (convolving and pooling layers) capture higher order and more complex patterns than basic features from the input image.

Decoder:

- Takes features of size of embedding dimension from the encoder branch.
- Up samples the features at a high-resolution level by using techniques like transposed convolutions. This results in signature enhancement for higher accuracy in pinpointing the tumour region in the initial image.
- Concatenating the up sampled features together with the matching feature maps from encoder path at different resolutions along the route. This is a technique to preserve the spatial details that are lost upon motion while being down sampled in the encoder.
- Clears features and perfects the mask while the convolutional layers work with concatenated features.

```
def U_Net(inputs, ker_init, dropout):
    'model definition'
    conv1 = Conv2D(32, 3, activation='relu', padding='same',
kernel_initializer=ker_init)(inputs)
    conv1 = Conv2D(32, 3, activation='relu', padding='same',
kernel_initializer=ker_init)(conv1)

    pool1 = MaxPooling2D(pool_size=(2, 2))(conv1)
    conv2 = Conv2D(64, 3, activation='relu', padding='same',
kernel_initializer=ker_init)(pool1)
    conv2 = Conv2D(64, 3, activation='relu', padding='same',
kernel_initializer=ker_init)(conv2)

    up1 = Conv2D(32, 2, activation='relu', padding='same',
kernel_initializer=ker_init)(UpSampling2D(size=2)(conv2))
    merge1 = concatenate([conv1, up1], axis=3)
    conv3 = Conv2D(32, 3, activation='relu', padding='same',
kernel_initializer=ker_init)(merge1)
    conv3 = Conv2D(32, 3, activation='relu', padding='same',
kernel_initializer=ker_init)(conv3)

    conv4 = Conv2D(4, 1, activation='relu')(conv3)
    'model definition ended'
    return Model(inputs=inputs, outputs=conv4)
```

Above mentioned UV Net architecture used for the Brain segmentation project implementation.

2. Segmentation with U-Net:

- During training, the U-Net model receives pre-processed MRI data (e.g., resized and normalised) as input.
- The encoder part extracts features from the MRI data, progressively capturing more high-level information about the brain anatomy and potential presence of a tumour.
- The decoder part takes these features and up samples them, while also incorporating spatial information from the encoder. This allows the model to reconstruct a segmentation mask that identifies the tumour region within the original image resolution.
- The segmentation mask is typically a multi-channel image where each channel represents a specific tumour sub-region or background.

3. Loss Function and Metrics:

- To train the U-Net model effectively, a loss function is used to evaluate the difference between the predicted segmentation mask and the ground truth segmentation mask (provided in the training data).
- The code utilises categorical cross entropy, a suitable loss function for multi-class classification problems like segmenting different tumour sub-regions.
- Additionally, the model is evaluated using various metrics to assess its segmentation performance:
- **Accuracy:** Measures the overall proportion of correctly classified pixels in the segmentation mask.
- **Mean IoU (Intersection over Union):** Calculates the average Jaccard similarity coefficient between the predicted and ground truth segmentation masks. IoU reflects the overlap between the two masks, with higher values indicating better segmentation accuracy.
- **Dice coefficient (custom implementation):** Similar to IoU, this metric also measures the overlap between the predicted and ground truth segmentation masks. It can be particularly useful for imbalanced datasets where some tumour sub-regions might occupy a smaller area.

4. Model Compilation:

- U-Net model is set up with Adam optimizer – its role is to control the update of the weights during the training process. The Adam optimizer does calculations of parameters of the model in a quick and effective way so to minimise the loss function.
- a loss function (eg., categorical cross entropy) is determined. This function is basically a comparison of the masks that were predicted by the model with the ground truth mask. It is used by the optimizer as feedback to adjust the model.

- Metrics (such as accuracy, mean IoU, dice coefficient) are the things that are chosen to check the model performance when the model is being trained. These metrics provide the overview of how well the learning process is progressing through the segmentation of tumour regions.

5. Training with Custom Data Generator:

- The custom data generator is used to load and preprocess MRI data and corresponding segmentation masks in batches during training. This approach improves memory efficiency compared to loading the entire dataset at once.
- The model is trained for a specific number of epochs (iterations over the entire training data). In each epoch, the model processes training data batches, updates its weights based on the loss function and optimizer, and refines its ability to segment tumours.

Results:

For the below Training Images the segmented images are given:

```
'BRATS_020.nii','BRATS_019.nii','BRATS_018.nii','BRATS_017.nii'
```

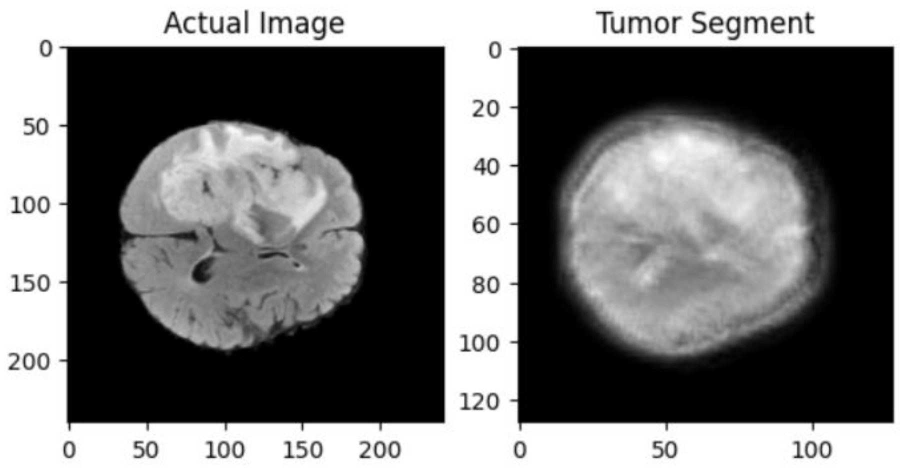

Fig 1: BRATS_020

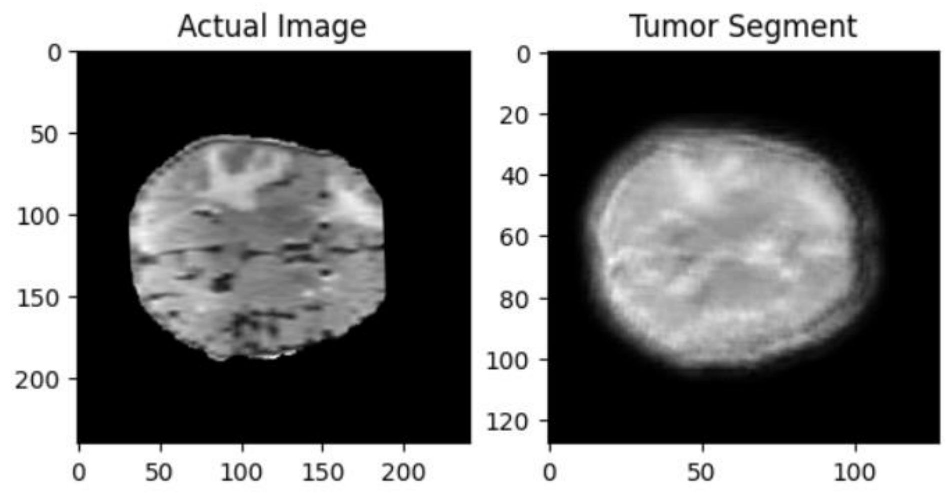

Fig 2: BRATS_019

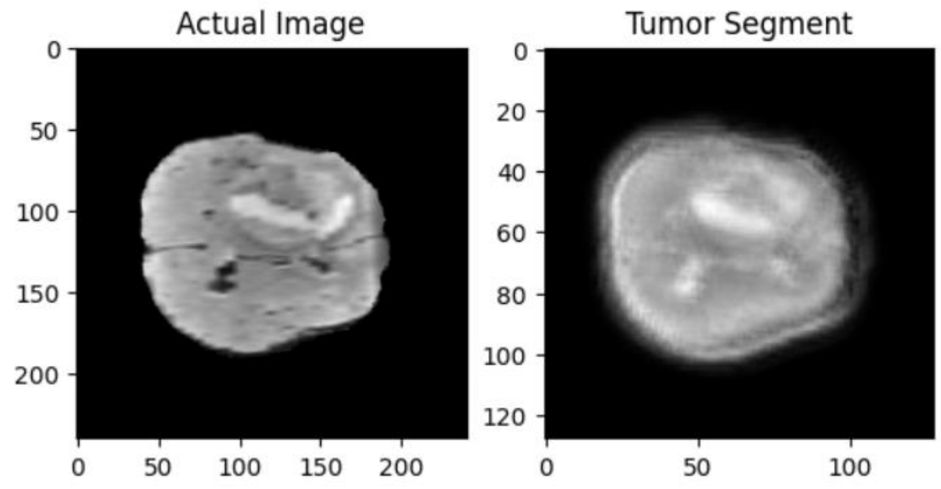

Fig 3: BRATS_018

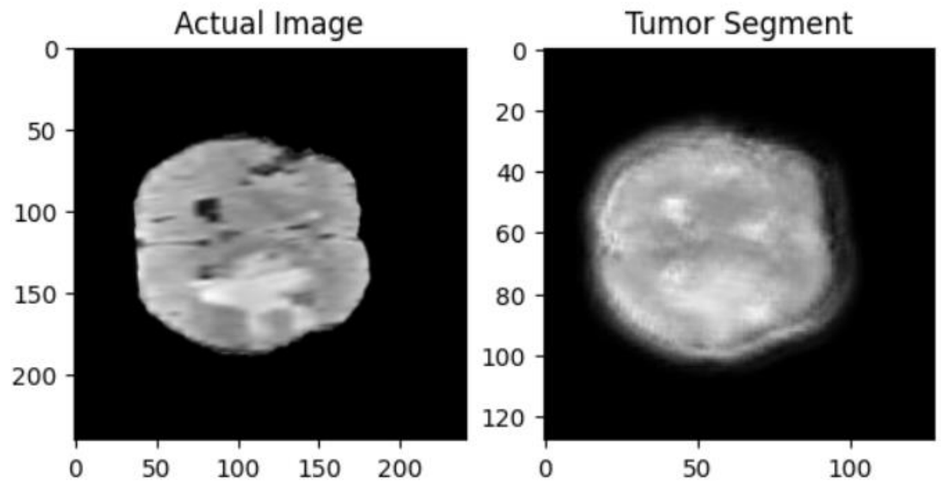

Fig 4: BRATS_017

For the below Testing Images the segmented images are given:

```
'BRATS_485.nii','BRATS_486.nii','BRATS_487.nii','BRATS_488.nii'
```

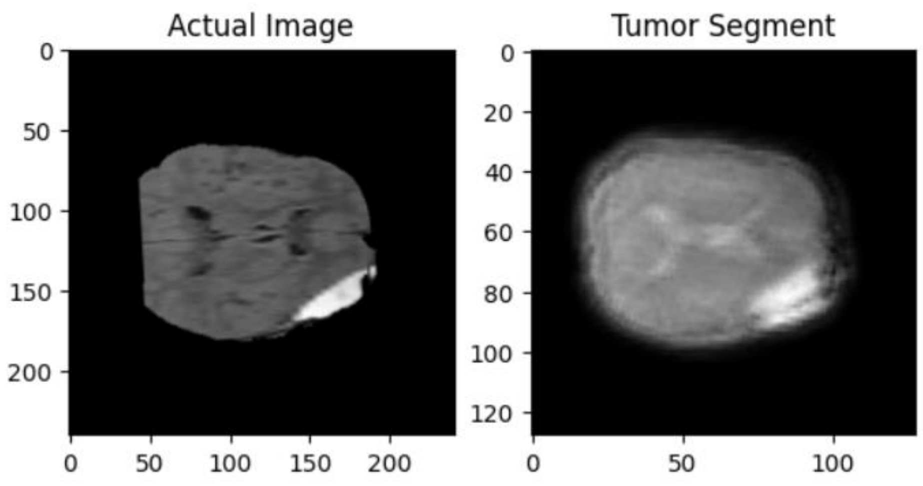

Fig 5: BRATS_485

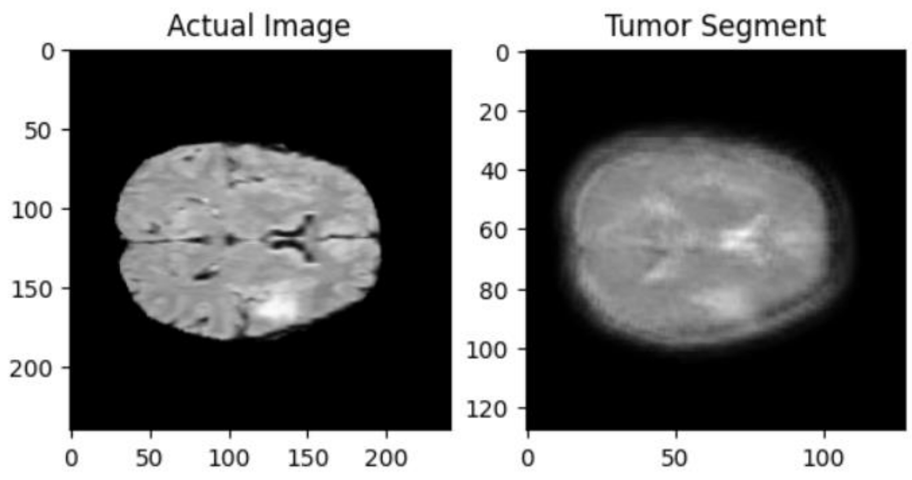

Fig 6: BRATS_486

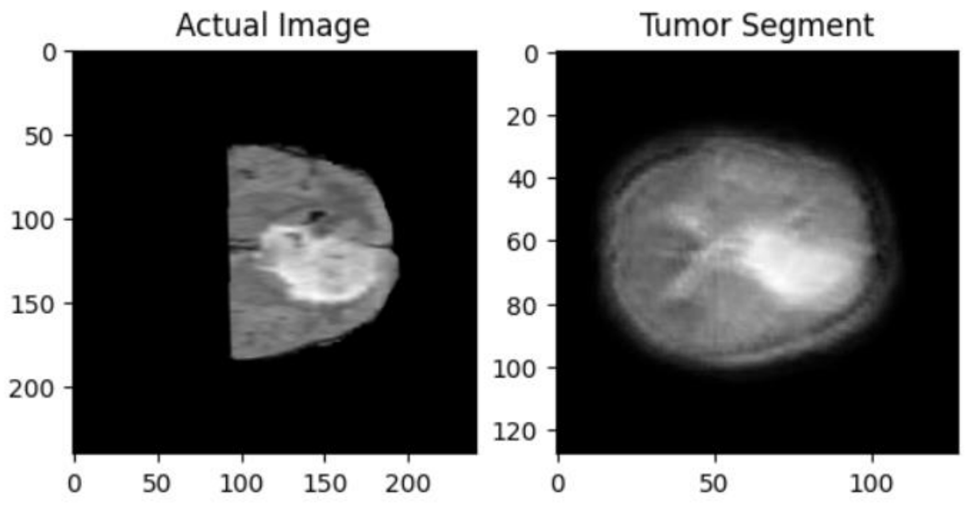

Fig 7: BRATS_487

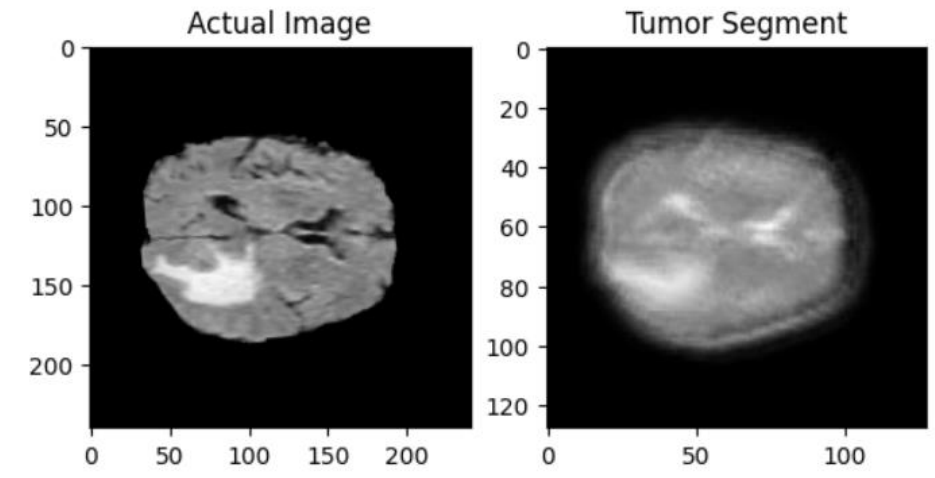

Fig 8: BRATS_488

4 images from the given data are fed to the model for the predictions, and above are the segments identified by the model. We can see that in each segment (models prediction) the tumour region is little whiter than the surrounding region, which shows that our model is learning to identify the tumour gradually.

CONCLUSION:

I have implemented the code with two models. Model 1 has the learning rate of 0.0001 and the training accuracy is 98.63% and the model 2 has the learning rate of 0.00001, the training accuracy is 98.33%.

Brain segmentation project with UV net architecture gave the good results as the accuracy of the model is very high and it has segmented the images very well as shown in the results section.

From this assignment, I can conclude that deep learning methods, especially CNN’s are very powerful techniques for image segmentation in medical areas. With little data and model complexity our model was able to produce good results, if I had trained the model with more data and for more epochs, the model's performance might increase drastically.

And also, preprocessing is also an important step in image segmentation, I believe that since MRI images has lot of background, our model is biassed towards background, so it is necessary to use extensive data preprocessing steps like identifying only the brain and feeding to the model, instead of passing the background as well, that way model will only be focussed on the MRI and not on any background.

In conclusion, I have successfully pre-processed the data, utilised a data generator for efficient handling of the data for training, researched and implemented custom loss functions, implemented a small U-Net for segmentation of Tumour from MRI images, trained the model using the data generator or data loader and obtained predictions from the model. And from the results I can say that machine learning techniques, especially CNN are very powerful for neuroimaging analysis and segmentations.