Brain Anomaly Segmentation, A Medical Image Application

Daniel Xavier[pg50310] and Diogo Rebelo[pg50327]

[1] University of Minho, Informatics Department, 4710-057 Braga, Portugal
[2] email:{pg50310,pg50327}@alunos.uminho.pt

Abstract. This project focuses on the development of an image segmentation model using the U-Net architecture for medical image analysis. The objective is to accurately identify and segment regions of interest in medical images, aiding in diagnosis and treatment planning. The model was trained on a dataset of medical images and corresponding ground truth masks, with augmentation techniques used to enhance performance. Evaluation metrics such as Dice coefficient and IoU were utilized to assess the model's accuracy. The results demonstrate the model's effectiveness in segmenting target regions with high precision. This work has the potential to improve medical image analysis, enabling better clinical decision-making and patient care.

Keywords: Deep Learning · Segmentation · Medical Image · Brain MRI

1 Introduction

In this report, the key points of the work developed were explored and described. The methodology and data used are described in an initial phase, followed by the analysis of their processing and the study of the architecture and model, and finally the evaluation of the results and conclusions.

The slogan of the whole project is the segmentation of medical images. It is defined as the task of identifying specific regions of interest in medical images, such as tumours, anatomical structures and lesions. Thus this is a fundamental task in various areas of medicine, being our focus the areas of neurology and oncology since we used a dataset with magnetic resonance images of the brain and associated with each image we have a mask that was developed using FLAIR technology (used the signal from the cerebrospinal fluid to highlight the contrast between white matter and gray matter in the brain) and that allowed us to evaluate and properly train our model for automatic segmentation of anomalies in the brain.

2 Context

The case study thus focuses on neuroimaging, paramount in the diagnosis and treatment of neurological diseases. Magnetic resonance imaging (MRI) is one of the most widely used neuroimaging techniques, providing detailed information about the structure and function of the brain. However, manual interpretation of MRI images can be time-consuming and error-prone, especially when it comes to identifying and segmenting brain lesions.

With the advancement of artificial intelligence and deep learning, automated approaches for the analysis of brain MR images have emerged. One such approach is automated image segmentation, which involves the precise identification and delineation of structures of interest in the images, such as brain tumours or areas of inflammation.

In this context, the use of public datasets, such as the brain MRI segmentation dataset from The Cancer Imaging Archive (TCIA) that was used in the project, becomes relevant.

Moreover, the automatic segmentation of brain lesions can contribute to other diverse research and studies since it can provide insights into the relationship between the characteristics of brain lesions and other diseases, helping to understand them and contributing to the personalisation of treatment.

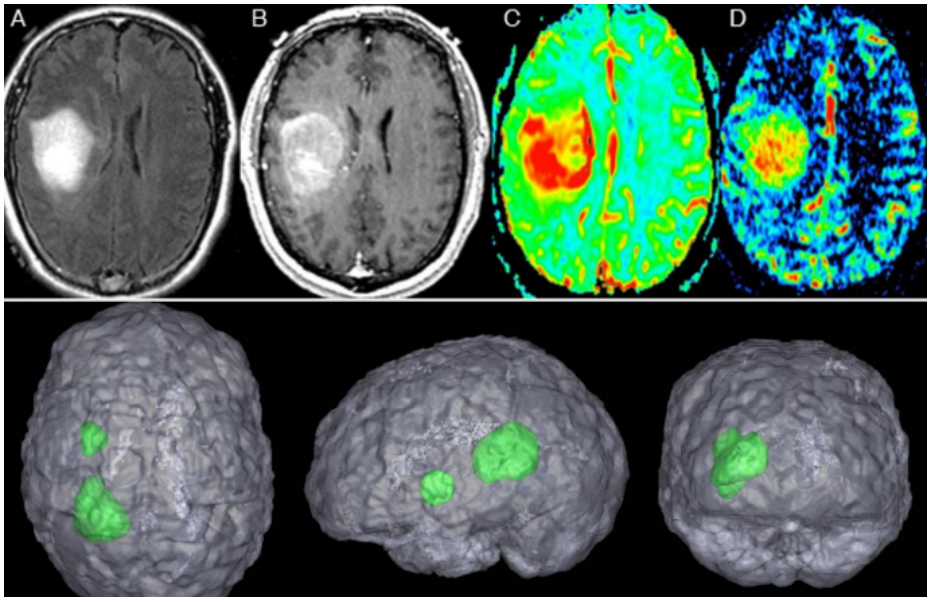

Fig. 1: Brain anomaly segmentation.

3 Goals

As described the main goal of this work is to develop a model for automatic segmentation of low-grade brain anomalies using the deep learning techniques learned throughout the curricular unit.

To achieve the main goal we have defined a series of objectives:

- Prepare and explore the dataset, understanding the structure of the images and segmentation masks.
- Implement and train a segmentation model.
- Evaluate the model performance using loss metrics, Jaccard coefficient and Dice coefficient.
- Perform a qualitative analysis of the results by viewing random samples from the test set and comparing the predicted segmentation masks with the reference masks.
- Investigate the possibility of improving the segmentation accuracy through post-processing techniques.

By achieving these goals, we hope to contribute to the advancement in the field of automatic segmentation of low-grade brain abnormalities, providing an effective and accurate tool to aid in the diagnosis and treatment of these complex diseases.

4 Methodology

The methodology employed in this project follows the CRISP-DM (Cross-Industry Standard Process for Data Mining) framework. CRISP-DM is a widely accepted and well-documented approach for solving data mining and analytic problems, providing a structured and iterative process that guides the project from initial business understanding to the final deployment of the solution. The following sections generally outline the key steps of the CRISP-DM methodology and how they were applied in this project. All the work was performed in google collabs notebooks because of the complexity of the data and necessity to process and work with a large amount of information.The link is available `here`. Note that it was used Google Drive to store the data, so the paths are the default path to select files from Google Drive. To execute the code locally, it is necessary to change the paths accordingly.

4.1 Business Understanding

In this phase, the project objectives and requirements were identified, along with a clear understanding of the business problem to be addressed. The goals of the project were defined, and the scope and constraints were established.

This phase was strongly promoted by previous research on generative models in medical imaging. One of the focus topics of the respective work was the segmentation of tumors in order to make their identification more effective.

4.2 Data Understanding

During this phase, the available data sources were identified and assessed. Data collection methods were determined, and the relevant datasets were acquired. The data was then thoroughly analyzed to understand its structure, quality, and completeness.

Our main focus was to comprehend the data that would be utilized in the project, specifically the set of magnetic resonance imaging (MRI) images. This phase was crucial in terms of managing the project timeline while accommodating the complexity of the data.

4.3 Data Preparation:

In this phase, the data was prepared for analysis. The main concern was to be able to create a dataset of image paths and their masks, and use them later to generate the new images through augmentation. The ultimate task was to classify brain tumors as malicious or benign.

4.4 Modeling

During this phase, various modeling techniques were employed. This involved selecting appropriate algorithms based on the nature of the problem and the available data. The models were trained using the prepared dataset and evaluated using suitable performance metrics. It is important to note that we have constructed our own U-net network in order to apply the segmentation.

4.5 Evaluation

The model generated in the previous phase were evaluated against the project's objectives. Their performance was assessed using different evaluation measures, such as accuracy and dice loss, both implemented in a separate module.

5 Data

The dataset used in this project consists of brain MR images (256x256) with manual FLAIR abnormality segmentation masks. These images were obtained from The Cancer Imaging Archive (TCIA). Specifically, the dataset includes data from 110 patients who were part of The Cancer Genome Atlas (TCGA) lower-grade glioma collection.

The brain MR images in the dataset were acquired using the fluid-attenuated inversion recovery (FLAIR) sequence, which is a specialized MRI technique commonly used for detecting and visualizing abnormal tissue in the brain. FLAIR imaging is particularly effective in identifying abnormalities associated with gliomas, such as areas of edema or tumor infiltration.

The images in the dataset are three-channel images, represented in the RGB color space. The three channels correspond to different imaging modalities or image processing techniques used to enhance specific features of interest within the brain. The inclusion of multiple channels provides additional information and potentially improves the accuracy of subsequent analysis and segmentation tasks.

Each image in the dataset represents a specific slice of the brain. The slices are selected to capture relevant anatomical information and pathological characteristics. It is important to note that the dataset consists of preprocessed images, where preprocessing steps such as normalization, registration, and noise reduction may have been applied. These preprocessing steps aim to enhance the quality and consistency of the images, enabling more reliable analysis and segmentation.

Accompanying each MR image, the dataset also provides manual FLAIR abnormality segmentation masks. These masks indicate the regions of abnormality, such as tumor areas or edematous regions, as annotated by experts. The segmentation masks serve as ground truth for evaluating and training automated segmentation algorithms.

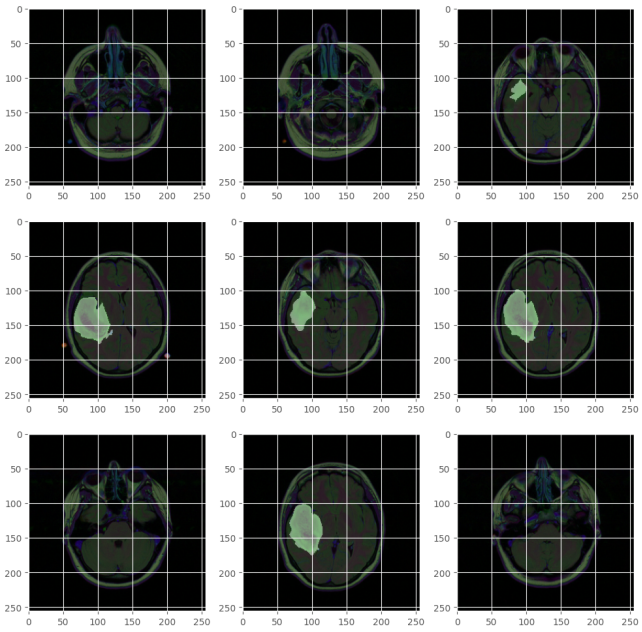

Fig. 2: Dataset of images and corresponding masks.

6 Processing

In this section, we describe any and all treatments applied to the data, either before or after model training. In order to make the training process faster, google collab notebooks were used, using google machines. We start by defining the properties of the images, then we prepare the dataset and, finally, we train the model. After training, we still performed a post-processing, which facilitated the evaluation of the classification.

6.1 Pre-processing

The first step in the processing phase involves setting the size parameters for the images to ensure consistency. In this case, the width and height of the images are set to 256 pixels each (`im_width = 256`, `im_height = 256`).

Next, a list of image filenames for training is created. This is accomplished by searching for files that contain the word 'mask' in their names within the specified directory. The corresponding image filenames are obtained by removing the `'_mask'` substring from the mask filenames. This ensures that each image filename is associated with its respective mask file. The list of image filenames is then used to create a DataFrame called `'df'` which contains the image filenames for training along with their corresponding mask file paths. This DataFrame serves as a mapping between the image filenames and their respective masks.

To split the dataset into training, validation, and testing sets, the DataFrame is further divided using the `train_test_split` function from scikit-learn. Initially, the DataFrame is split into a training set and a test set, with a test size of 0.1 (10%). Subsequently, the training set is further split into a validation set and the final training set, with a validation size of 0.2 (20%).

The shapes of the resulting DataFrames (`df_train, df_test, df_val`) are printed respectively to verify the sizes of the training, testing, and validation sets:

```
(2828, 2)
(393, 2)
(708, 2)
```

6.2 Post-processing

After generating masks using the trained model, a post-processing step is applied to refine the masks. The `postprocess_mask` function is defined to perform this operation. The function takes the generated mask as input and converts it to the correct data type, specifically an 8-bit unsigned integer. This ensures compatibility and consistency with further processing steps. The function returns the post-processed (closed) mask as the output.

Next, a closing operation is applied to the mask using an appropriate structuring element. In this case, an elliptical structuring element with dimensions

5x5 is used. The closing operation helps fill in any small holes or gaps in the mask, resulting in smoother and more accurate mask boundaries.

By applying the post-processing step, the generated masks are improved in terms of smoothness and accuracy, enhancing the quality of the final segmentation results.

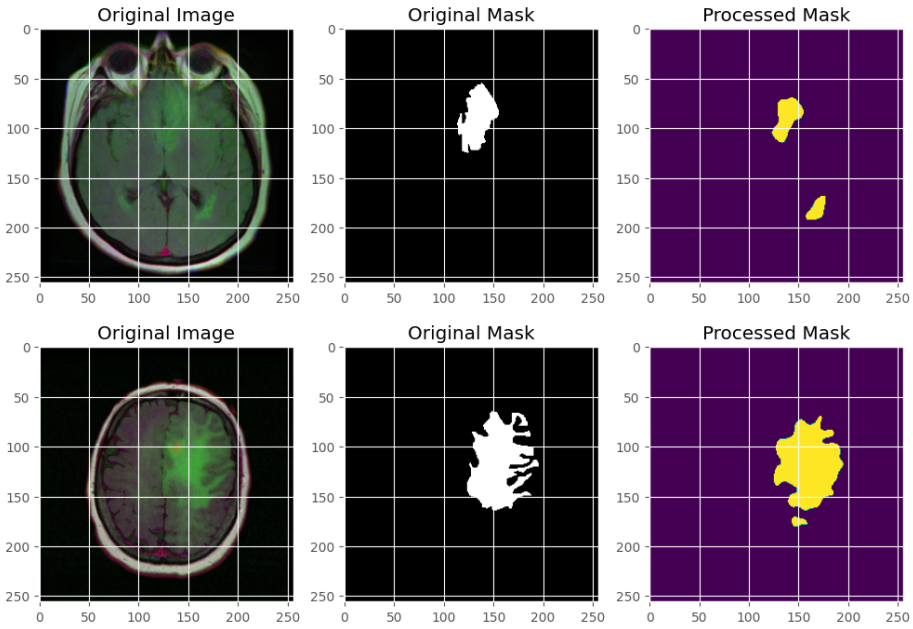

Fig. 3: Unified predicted masks.

7 Architecture

The architecture used in this project is based on a convolutional neural network known as U-Net. U-Net is widely used in image segmentation tasks due to its ability to capture contextual information and perform accurate segmentations. This architecture has two main parts:

Encoder The U-Net contraction path is responsible for capturing the relevant features of the input images. Composed of several convolutional layers, each followed by a non-linear activation function (ReLU). As the convolutional layers are stacked, the spatial resolution of the features is reduced by pooling or convolution operations with a larger stride, allowing a more abstract and global view of the image.

Decoder Conversely, the U-Net decoder path aims at reconstructing the segmented image from the features learned in the contraction path. Each layer in the expansion path is composed of a transposed convolution layer, which performs upsampling operation to increase spatial resolution, followed by a concatenation with the corresponding features of the contraction path. This concatenation is an important feature of U-Net, as it allows the fusion of detailed information from the contraction layers with the more abstract context information from the expansion layers.

Unlike an VAE, it does not learn the latent representation of the data so they have different different purposes.

In addition, U-Net also uses a technique called skip connections. These direct connections between layers in the contraction path and the expansion path allow low-level, spatially detailed information to be transmitted directly to subsequent layers in the expansion path. This helps preserve fine details during upsampling, improving segmentation accuracy. During the downsampling process in the contraction path, the information is scaled down and higher level features are learned. However, these higher level features may have lost detailed information that is important for accurate reconstruction of the segmented image.

Furthermore, the U-Net architecture uses regularisation techniques, such as batch normalisation and dropout, to avoid overfitting during model training.

During training, the U-Net is optimized using the Adam optimizer and the dice coefficient loss function. The dice coefficient is a metric commonly used to evaluate the overlap between the segmented mask and the reference mask. The closer the dice coefficient value is to 1, the better the segmentation.

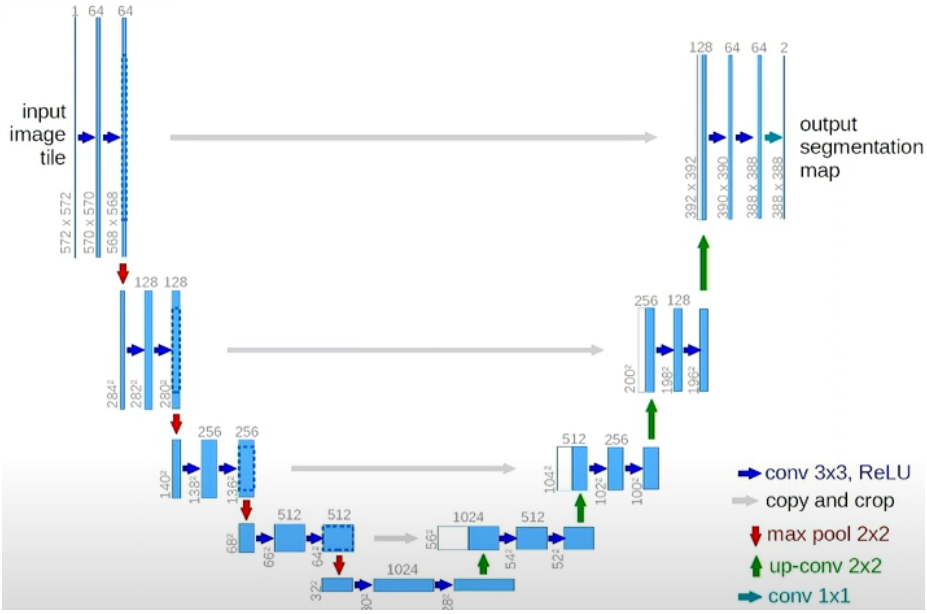

Fig. 4: Architecture of the model.

8 Model

The model architecture used in this project is a U-Net, which is a convolutional neural network (CNN) commonly used for image segmentation tasks. The U-Net architecture consists of an encoder-decoder structure with skip connections that allow for capturing both local and global information in the input images.

8.1 Generators

The code defines a function called `train_generator` which generates image and mask pairs for training. The function utilizes the `ImageDataGenerator` class from the Keras library to perform data augmentation. This allows for generating augmented versions of the input images and their corresponding masks, which helps improve the model's generalization capability.

Inside the `train_generator` function, two `ImageDataGenerator` objects are created, one for the images and another for the masks. These generators are initialized with the provided augmentation dictionary, which specifies the augmentation techniques to be applied, such as rotation, shifting, shearing, zooming, and horizontal flipping.

The `flow_from_dataframe` method is then called on each generator to retrieve batches of augmented images and masks. The data is loaded from the given dataframe, where the image filenames and mask paths are specified. The

generators are set to operate in the same order by using the same random seed (`seed=1`).

The resulting image and mask generators are combined using the `zip` function, creating a generator that yields image-mask pairs. Before yielding each pair, a function called `normalize_and_diagnose` is applied to normalize the pixel values of the images and masks and perform additional preprocessing: this function divides the pixel values of the images and masks by 255 to bring them into the range of [0, 1]. Additionally, the masks are thresholded, setting values greater than 0.5 to 1, representing the presence of the tumor, and values less than or equal to 0.5 to 0, representing the absence of the tumor.

8.2 Parameters

After defining the generator, we set the number of training epochs (`EPOCHS`), the batch size (`BATCH_SIZE`), the learning rate (`learning_rate`), and a smoothing parameter (`smooth`). These parameters are crucial for controlling the training process and optimizing the model's performance.

The model architecture is instantiated using a function called `unet`, which represents our custom implementation of the U-Net architecture. The U-Net architecture is well-suited for image segmentation tasks due to its ability to capture both local and global information through skip connections.

Next, we define the a dictionary with augmentation parameters for the training generator:

- `rotation_range`: Specifies the range of random rotations to apply to the images, expressed in degrees.
- `width_shift_range`: Determines the range of horizontal shifts to randomly apply to the images, given as a fraction of the total width.
- `height_shift_range`: Controls the range of vertical shifts to randomly apply to the images, represented as a fraction of the total height.
- `shear_range`: Defines the range of shearing transformations to randomly apply to the images.
- `zoom_range`: Specifies the range of random zooming to apply to the images.
- `horizontal_flip`: Determines whether horizontal flipping of the images is performed randomly.
- `fill_mode`: Specifies the strategy for filling in newly created pixels resulting from image transformations, such as shifts or zooming. The "nearest" fill mode assigns the value of the nearest pixel to the newly created pixel.

Two generators are created using the `train_generator` function: one for the training set (`train_gen`) and another for the validation set (`test_gen`). The training generator is initialized with the training dataframe (`df_train`), the batch size, the augmentation parameters, and the target size of the input images.

The model is compiled using the Adam optimizer with the specified learning rate, decay rate, and other parameters. The loss function used is the Dice coefficient loss, which is a common metric for image segmentation tasks. Additional metrics such as binary accuracy, intersection over union (IoU), and Dice coefficients are also calculated during training.

8.3 Callbacks

A checkpoint callback is set up to save the model weights after each epoch. The callback specifies the checkpoint path where the weights will be saved, with the epoch number included in the filename for identification. We then can load the last model that was saved.

8.4 Fit

Finally, the `fit` method is called on the model, providing the training generator, the number of steps per epoch, the number of epochs, the validation generator, and the number of validation steps. The training process is monitored by the specified callbacks, and the history object captures the training and validation metrics for analysis and visualization.

9 Evaluation

A number of metrics were used to evaluate the model so it is necessary at an early stage to explain each metric:

- Dice Coefficient: The Dice coefficient is a widely used metric to evaluate the overlap or similarity between two binary masks. It is calculated as the ratio of the intersection to the sum of the areas of the two masks. In the context of image segmentation, the Dice coefficient is a useful measure to assess the segmentation accuracy, i.e. the overlap between the mask segmented by the template and the reference mask.
- Binary Accuracy: Binary accuracy is a metric that calculates the proportion of pixels correctly classified as belonging to the class of interest (in this case, the segmented mask) relative to the total number of pixels. This metric is useful for evaluating the model's ability to correctly classify pixels as part of the segmented region or not.
- Intersection over Union (IoU): The IoU is another commonly used metric to evaluate the overlap between the segmented mask and the reference mask. It is calculated as the ratio of the intersection to the union of the areas of the two masks. IoU provides a more comprehensive measure of segmentation quality, taking into account both the overlap and the discrepancy between the two masks.

At this stage, it remains for us to evaluate the model, so we will make a more detailed analysis of the model performance graphs and discuss the results obtained.

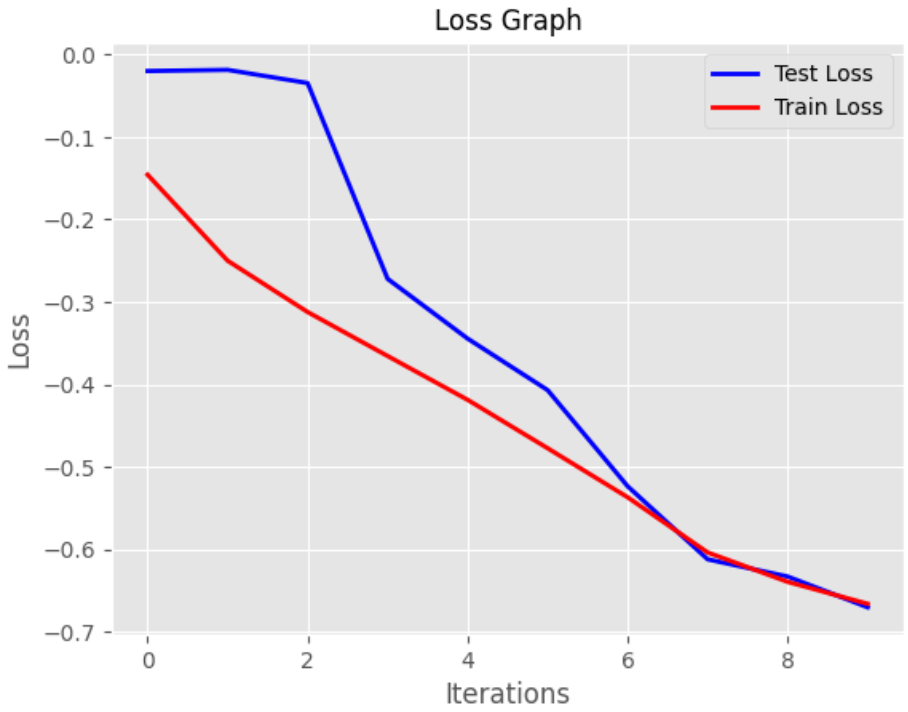

Fig. 5: Graph to evaluate the loss.

The first graph represents the loss of the model over the iterations. We observe that both the loss on the training set (red line) and the test set (blue line) decrease gradually over iterations. At the beginning, the loss on the test set was -0.02 and on the training set was -0.15. These negative values may indicate that the model is obtaining better results than initially expected. As iterations progress, the loss lines approach and stabilise around -0.65 after 8 iterations. This suggests that the model has achieved a good level of fit to the training data and is able to generalise well to the test set.

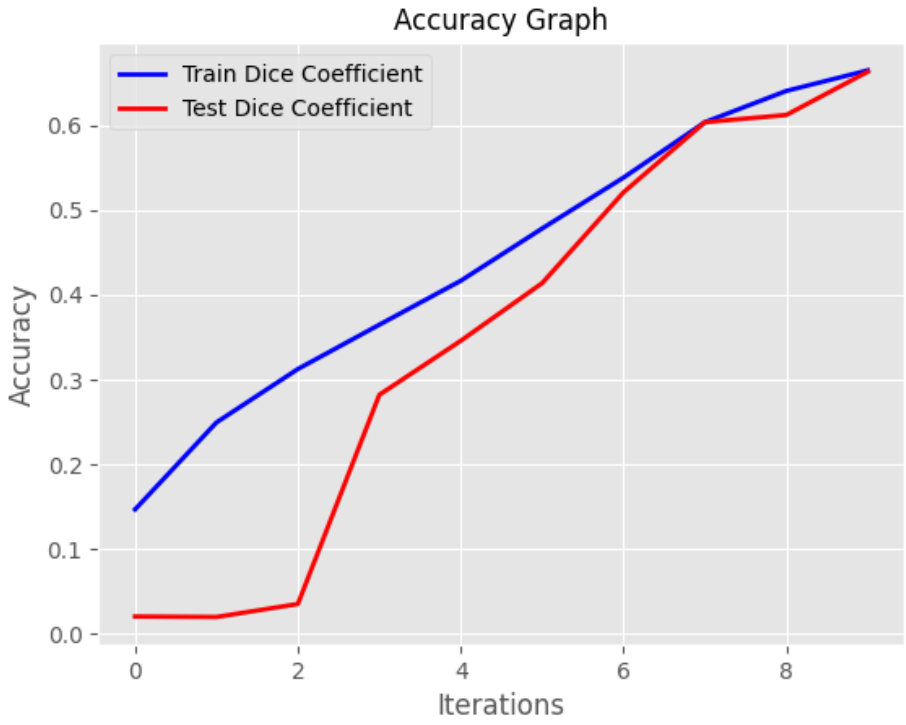

Fig. 6: Graph to evaluate the accuracy.

The second graph represents the Dice coefficient of the model over the iterations. The Dice coefficient is a commonly used metric to evaluate the overlap or similarity between the segmented mask and the reference mask. At the beginning, the precision (Dice coefficient) on the test set was 0.1, while on the training set it was 0.15. Over the iterations, both lines increase and find themselves around 0.6 precision after 7 iterations. The value of the Dice coefficient continues to rise, reaching 0.7 after 8 iterations. These values indicate that the model was able to segment the regions of interest in the brain MRI images with considerable accuracy.

Regarding the analysis of the results, we can consider the following aspects:

- Loss: The decrease in loss over the iterations indicates that the model is adjusting to the training data and learning to correctly map the input images to the segmented masks. The final loss value on the test set indicates that the model was able to significantly reduce the discrepancy between the segmented masks and the reference masks.
- Dice coefficient: The gradual increase in the Dice coefficient indicates that the model is improving the overlap between the segmented masks and the reference masks. The final value of the Dice coefficient (0.7) suggests that

the model was able to correctly segment about 70% of the regions of interest in the brain MRI images.

In addition to the graphs, the following metric values were also obtained for the trained model:

- Test Loss: -0.6689
- Test IoU: 0.9955
- Test Dice Coefficient: 0.5183

These values indicate that the model performed well in the brain MRI image segmentation task, with a high accuracy and a reduced loss. The IoU value indicates that there was a significant overlap between the segmented mask and the reference mask. The Dice Coefficient also presented a satisfactory value, indicating a good similarity between the segmented mask and the reference mask.

We believe that with a greater computational power, we could train longer and better the model, with more epochs for example, and thus obtain better results

10 Conclusions

Once the project is concluded, a critical, reflected and thoughtful view of the work done will make sense.

Overall, we are happy with the project developed and the results obtained, following the techniques learned in this that was our deepest contact with Deep Learning and at a time when this area is very powerful and promising.

We have thus achieved the goal of developing a system capable of identifying and delimiting regions of interest in medical images, which can be of great use in several health applications.

The U-Net architecture proved to be suitable for the segmentation task, providing promising results. The results obtained indicate that the model was able to learn and generalise well the segmentation task, presenting satisfactory accuracy values. These results suggest that the model can be applied in different contexts and be useful for medical professionals, helping in diagnosis, treatment planning and image analysis.

Thus as future work there are several opportunities to explore new techniques and approaches, such as using larger datasets, applying different pre-processing and post-processing techniques, and combining different neural network architectures.

In conclusion, the work presented an effective approach for medical image segmentation, demonstrating the potential of this technique in assisting healthcare professionals. Through accurate and automated segmentation, it is possible to optimize the process of medical image analysis, contributing to faster and more accurate diagnoses, besides facilitating clinical decision-making.