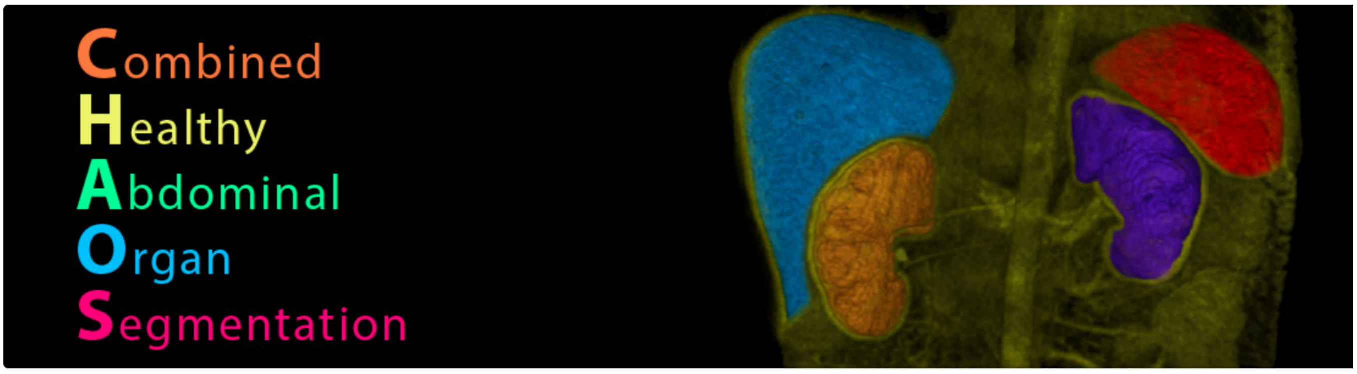
FINAL PROJECT: CHAOS

Liver Segmentation



Group 2 - Deep Learning Methods for Medical Image Analysis

Mariona Carrasco and Gerard Castells

1. Introduction

In this course project, the aim is to put our acquired knowledge into action by developing a precise segmentation algorithm. The focus is on tackling the liver segmentation task within the CHAOS (Combined Healthy Abdominal Organ Segmentation) challenge, using CT (computed tomography) images. The objective is to construct a reliable segmentation model, incorporating the methods gleaned from deep learning and medical image analysis courses.

2. Design and Results

2.1 Data Processing

In this challenge, we have a dataset consisting of medical imaging data from 20 patients. Each patient's data includes 16-bit DICOM images with a resolution of 512x512 pixels. These images have a spacing between 0.7-0.8 mm in the x-y direction and an inter-slice distance ranging from 3 to 3.2 mm, representing different CT slices. Additionally, for each slice, there is a corresponding mask in .jpg format.

The CT images are stored in DICOM files, while the masks are in .jpg format. The first part of our code is responsible for loading these images and masks, and then organizing them into a dictionary named "patient_data." In this dictionary, the keys represent patient identifiers, and the values consist of lists containing pairs of (DICOM image, mask) converted into NumPy arrays for each patient. This approach ensures that slices from the same patient are not included in both the training and validation datasets.

To create the training, validation, and testing sets, we use dictionaries. We randomly shuffle the patient identifiers and allocate them into sets: 15 for training, 4 for validation, and 1 for testing. Finally, we generate lists of image and mask pairs from these three dictionaries, which are subsequently utilized for various image segmentation tasks.

2.2 Model and training

For this challenge, we employed a pre-existing U-Net model whose parameters were fine-tuned through a trial-and-error approach to achieve the optimal learning curve. Once the model was implemented, we compiled it using the Adam optimizer, binary cross-entropy loss, and assessed its performance using metrics like binary accuracy, precision, recall, and the dice coefficient.

Subsequently, the "data generator" function is applied to the lists of training and validation data pairs generated earlier. This function generates batches of augmented images and masks, normalizes them, expands their dimensions, and applies data augmentation.
Finally, the model is trained using the previously obtained data pairs and evaluated using the "plotcurve" function, which generates two plots for learning curves (loss and accuracy) and one for evaluation metrics.

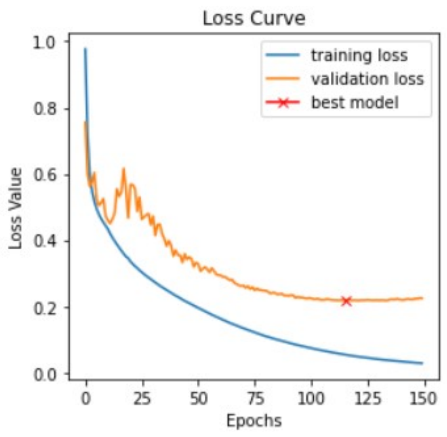


Figure 1: Loss curve

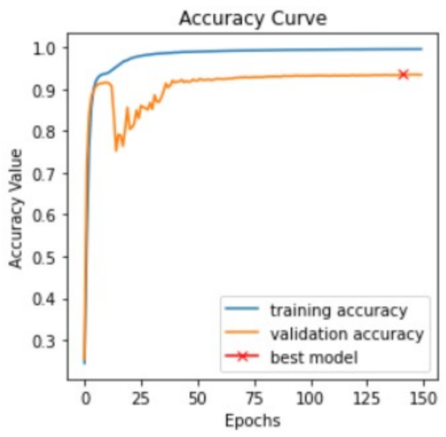


Figure 2: Accuracy curve

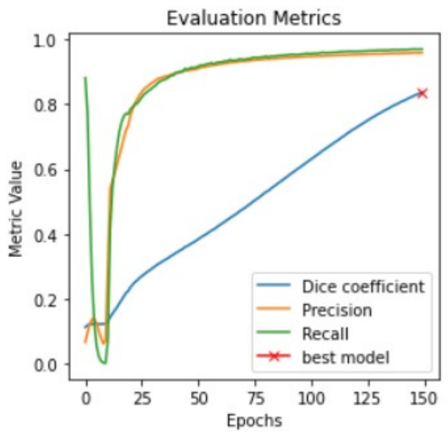


Figure 3: Evaluation metrics

The values obtained from the loss and accuracy curves indicate that the model has been trained well, with low values around 0 for training loss and close to 1 for training accuracy. However, the validation values don't reflect the same level of performance. Consequently, a slight overfitting has occurred, which could be rectified by incorporating additional epochs and augmenting the dataset size. Analyzing the evaluation metrics Recall and Precision, which have been obtained around 1, indicates that the model has been well-trained because it captures relevant instances of the positive class and every positive prediction made by the model is correct. Observing a Dice coefficient of 0.8 indicates a relatively high level of performance, considering that 1 represents perfection. In many cases, a value higher than 0.6 is generally considered satisfactory.

2.3 Predicted masks

To generate predicted segmentation masks using the pre-trained model, we will follow a series of steps. First, we'll save the trained model, and subsequently, we'll load it for inference. Then, we will take the testing patient dataset and divide it into two lists: the images and the corresponding ground truth masks.

The images will be fed into the model, slice by slice, to generate predicted masks for each of them. These predicted masks will be saved in a designated directory, along with the corresponding true masks for each slice. This way, we will have both the model's predictions and the actual masks readily accessible for further analysis.

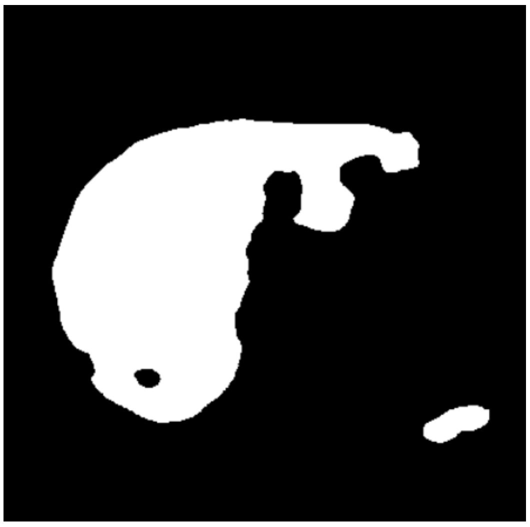


Figure 4 : Predicted mask

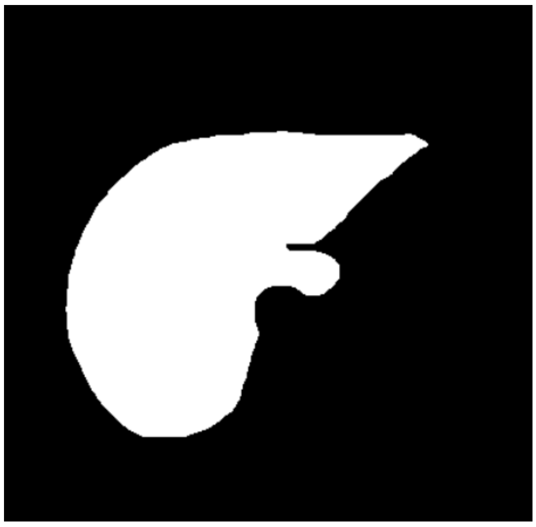


Figure 5: True mask

This comparative image displays the anticipated mask on the left (Image 4) and the actual mask on the right (Image 5). It illustrates liver segmentation using a binary mask. Upon analysis and comparison of the two masks, it is evident that they are not identical. Therefore, adjustments and clarification through morphological operations are necessary to achieve an optimal outcome and enhance the performance of the machine learning model. However, it is important that the model is able to segment the liver in a useful way, even with minor errors.

2.4 Dice score

To compute the Dice score for each patient, we will create a dictionary named "patient_dice_scores." This dictionary will store the average Dice score for all the slices belonging to each patient. The Dice score is calculated by comparing the predicted mask generated by the model with the ground truth mask.

To arrive at the average Dice score for each patient, we will first calculate the Dice score for each slice within a patient's dataset. Subsequently, we will compute the mean of these individual slice scores and associate this average with the respective patient's identifier. This approach allows us to assess the overall segmentation performance for each patient in a clear and organized manner.

Patient id	30	28	24	27	8	22	23	21	10	6	2	14	19	18	1	16	25	5	26	29
Average dice score	0.285	0.588	0.351	0.355	0.576	0.339	0.239	0.496	0.512	0.485	0.42	0.549	0.595	0.468	0.56	0.18	0.552	0.418	0.326	0.567

Table 1: Dice scores

The average of Dice scores for each patient is presented in Table 1. Upon calculating the median, an overall average of 0.43 is obtained, suggesting a moderate level of performance. It is important to note that a score of 1 represents perfection, so it's a good result considering that it has been noted that masks could be adjusted better. Despite the acknowledgement that masks could be adjusted for better precision, achieving this score of 0.43 is considered a positive result.

3. Conclusion

In conclusion, this project has served as a valuable application of the knowledge acquired throughout the course. It not only provided a real-world example through the CHAOS challenge but also presented an opportunity to navigate and learn from the encountered challenges. While the implemented U-Net model showcased commendable performance, there's acknowledgment of the potential for improvement. To enhance the predictive accuracy of masks, future endeavors could explore employing a high-performance model and expanding the training dataset with a larger volume of CT images.

Even with certain obstacles that could be solved with other operations or modifications, the final result is a significant achievement and we were successful in properly training a model. Nevertheless, despite the hurdles, we are deeply satisfied with the overall outcomes, appreciating the practical insights gained not only from the project itself but also from the entire course experience.