Deep Learning and its applications to Signal and Image Processing and Analysis

Final Project – Medical Segmentation Decathlon

Yuval Braun, Moshe Davidian, Almog Tawil

1. Abstract

Medical image segmentation is essential for various diagnoses and medical treatments. While any deep learning algorithms can perform successful segmentation, they require adjusting the code for the specific task and its dataset. We present a generic approach based on U-net architecture that can handle various medical segmentation tasks, learn the specific data set, and perform the segmentation. Our predictive model was tested on two different medical segmentation tasks without human interaction and performed high-quality segmentation. In addition, our model was improved by finding the optimal hyperparameters. Moreover, we present a comparison between several augmentation techniques that achieve significant improvement in the results.

2. Introduction and Objective

Medical image segmentation is the task of segmenting objects of interest in a medical image. This task is of great importance for various diagnoses and medical treatments and can be accomplished by different approaches [1]. In recent years, the use of neural networks has become widespread for performing this task. Small data, unbalanced labels, large-ranging object scales, multiclass labels, and multimodal imaging make segmentation of biomedical images complex even for deep learning approaches [2]. This task becomes more complicated when the data is 3D volumes instead of 2D images [2]. The project also deals with multiclass segmentation. Unlike single class segmentation, where we aim to separate the object from the background, in multiclass segmentation, the goal is to identify more than a single area. Therefore multiclass segmentation is an even more difficult task [3].

Many fundamental algorithmic advances in medical imaging are commonly validated on a small number of tasks, limiting our understanding of the generalisability of the proposed contributions. A model which works out-of-the-box on many tasks would have a tremendous impact on healthcare. The Medical Segmentation Decathlon (MSD) challenge [4] aims to provide such resources through the open-sourcing of large medical imaging datasets on several highly different tasks and standardize the analysis and validation process. The goal of the MSD is to provide a generic algorithm that can accept a segmentation task, study it and perform a successful prediction. This process must be completely generic and performed for any task without human intervention. That is, the code must be the same for all tasks. Although the original challenge contains 10 data sets of different tasks, in this project, we focused on only 2 of them: Heart and Hippocampus. Because each task has a different data

set and a different number of labels, we have created an algorithm that identifies these parameters and then creates a model that can handle the specific task.

One of the most popular networks to deal with medical data is U-NET [5], developed for biomedical image segmentation. After identifying the specific task parameters, our code creates a corresponding U-NET and trains it based on the training set. After Training, we evaluate our segmentation results with the Dice Score function. We also compared our results with different hyperparameters to find the optimal hyperparameters. In addition, we try various types of augmentation to enlarge our training set.

This review is organized as follows: Section 3 describes the data we used in our project. Section 4 describes the algorithm and the methods we used to perform the segmentation. Paragraph 5 deals with the challenges we encountered and how we dealt with them. Paragraph 6 illustrates the experiments we performed to find the optimal parameters for our model. Section 7 presents the final results we achieved. Finally, paragraph 8 summarizes our work and offers examples of future work.

3. Data Description

The MSD challenges include ten highly different tasks:

1. brain tumor
2. Heart
3. Liver
4. Hippocampus
5. Prostate
6. Lung
7. Pancreas
8. HepaticVessel
9. Spleen
10. Colon

Our project deals with 2 of the task: Heart and Hippocampus. We choose these tasks because they were the two with the smallest sizes. Since we worked on Google Colab and Google Drive, we cannot deal with the big data sets such as the liver that is 27 GB while the whole space in our drive is 15 GB.

Heart

The heart is a muscular organ in most animals, which pumps blood through the circulatory system. The pumped blood carries oxygen and nutrients to the body while carrying metabolic waste such as carbon dioxide to the lungs. In humans, the heart is approximately the size of a closed fist and is located between the lungs, in the middle compartment of the chest.

In humans, other mammals, and birds, the heart is divided into four chambers: upper left and right atria and lower left and right ventricles. Commonly the right atrium and ventricle are referred together as the right heart and their left counterparts as the left heart. Fish, in contrast, have two chambers, an atrium and a ventricle, while reptiles have three chambers. In a healthy heart blood flows one way through the heart due to heart valves, which prevent

backflow. The heart is enclosed in a protective sac, the pericardium, which also contains a small amount of fluid. The wall of the heart is made up of three layers: epicardium, myocardium, and endocardium.

The heart receives blood low in oxygen from the systemic circulation, which enters the right atrium from the superior and inferior venae cavae and passes to the right ventricle. From here, it is pumped into the pulmonary circulation, through the lungs where it receives oxygen and gives off carbon dioxide. Oxygenated blood then returns to the left atrium, passes through the left ventricle, and is pumped out through the aorta to the systemic circulation−where the oxygen is used and metabolized to carbon dioxide.

The left atrium is one of the four chambers of the heart. The left atrium receives blood full of oxygen from the lungs and then empties the blood into the left ventricle.

The atrium (Latin ātrium, "entry hall") is the upper chamber through which blood enters the ventricles of the heart. There are two atria in the human heart – the left atrium receives blood from the pulmonary (lung) circulation, and the right atrium receives blood from the venae cavae (venous circulation). The atria receive blood while relaxed (diastole), then contract (systole) to move blood to the ventricles. All animals with a closed circulatory system have at least one atrium. Humans have two atria.

The heart dataset was provided by King's College London (London, United Kingdom), originally released through the Left Atrial Segmentation Challenge (LASC) [6], and includes 30 MRI datasets covering the entire heart acquired during a single cardiac phase (free breathing with respiratory and ECG gating). Images were obtained on a 1.5T Achieva scanner (Philips Healthcare, Best, The Netherlands) with voxel resolution 1.25 x 1.25 x 2.7 mm3. The left atrium appendage, mitral plane, and portal vein endpoints were segmented by an expert using an automated tool followed by manual correction.

The dataset contains 30 3D volumes (20 Training + 10 Testing) when the sizes of the volumes are not constant. The training set also contains ground truth binary volumes that are used as labels. Voxel with value 1 represents the left atrium, and 0 represents the background.

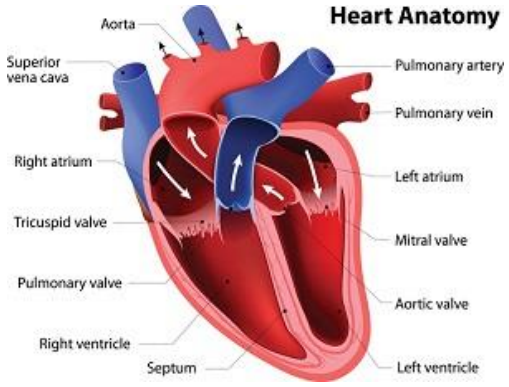

Figure 1. Heart anatomy

Hippocampus

The hippocampus is a major component of the brain of humans and other vertebrates. It is a complex brain structure embedded deep into the temporal lobe. Humans and other mammals have two hippocampi, one in each side of the brain. The hippocampus is part of the limbic system. It has a major role in learning and memory – it consolidated the information from short-term memory to long-term memory and spatial memory that enables navigation. It is a plastic and vulnerable structure that gets damaged by a variety of stimuli. Studies have shown that it also gets affected in a variety of neurological and psychiatric disorders. In Alzheimer's disease (and other forms of dementia), the hippocampus is one of the first regions of the brain to suffer damage; short-term memory loss and disorientation are included among the early symptoms. Damage to the hippocampus can also result from oxygen starvation (hypoxia), encephalitis, or medial temporal lobe epilepsy. People with extensive, bilateral hippocampal damage may experience anterograde amnesia: the inability to form and retain new memories.

The dataset consisted of MRI acquired in 90 healthy adults and 105 adults with a non-affective psychotic disorder (56 schizophrenia, 32 schizoaffective disorder, and 17 schizophreniform disorder) taken from the Psychiatric Genotype/Phenotype Project data repository at Vanderbilt University Medical Center (Nashville, TN, USA). Patients were recruited from the Vanderbilt Psychotic Disorders Program, and controls were recruited from the surrounding community. All participants were assessed with the Structured Clinical Interview for DSM-IV [7]. All subjects were free from significant medical or neurological illness, head injury, and active substance use or dependence. Structural images were acquired with a 3D T1-weighted MPRAGE sequence (TI/TR/TE, 860/8.0/3.7 ms; 170 sagittal slices; voxel size, 1.0 mm3). All images were collected on a Philips Achieva scanner (Philips Healthcare, Inc., Best, The Netherlands). Manual tracing of the head, body, and tail of the hippocampus on images was completed following a previously published protocol [8]. For the purposes of this dataset, the term hippocampus includes the hippocampus proper (CA1-4 and dentate gyrus) and parts of the subiculum, which together are more often termed the hippocampal formation. The last slice of the head of the hippocampus was defined as the coronal slice containing the uncal apex. The resulting 195 labeled images are referred to as hippocampus atlases. Note that the term hippocampus posterior refers to the union of the body and the tail.

The dataset contains 394 3D volumes (263 Training + 131 Testing) when the sizes of the volumes are not constant. The training set also contains ground truth volumes that are used as labels. Since here three areas need to be segmented, each voxel in the ground truth label can be one of {0,1,2}.

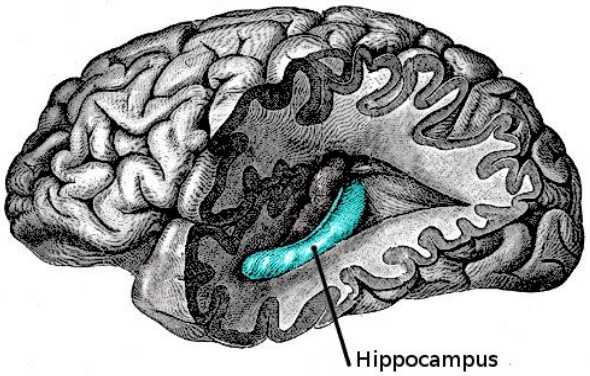

Figure 2. hippocampus' location in the human brain

3D to 2D

Since segmentation in 3d is a difficult task that requires enormous computing resources [3], we cut a 3D volume into several two-dimensional images. That means that instead of 263 volumes for the Hippocampus, we get 4500 slices, and instead of 20 volumes for the Heart, we get 2270 slices. This process is also done for the ground truth labels volumes. Fig 3 shows some of these slices with the ground truth areas mark with colors.

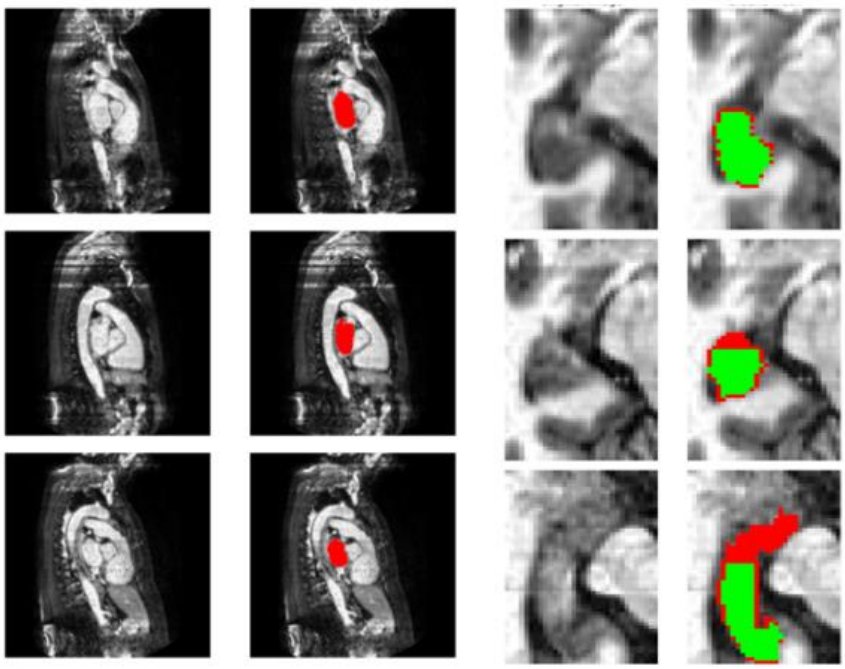

Figure 3. visualization of the 2D slices. From left to right: In the first column, the original image from the Heart data set. The original image from the Heart data set with the left atrium is painted in red in the second column. In the third column, the original image from the Hippocampus data set. The rightmost column is the original image from the Hippocampus data set with the head in red and the body (including tail) in green.

4. Methods and Algorithms

Our project was implemented using Python 3 and Tensorflow on Jupiter notebook. The code is divide into two parts: parameter extraction and the learning process.

4.1. Parameters extractions

Since each task contains different parameters and our code cannot be changed for each task, we build a pipeline to extract the specific parameters of the dataset. First, we cut a 3D volume into slices, as explained in section 3. In this stage, the slices are in various sizes since they produce from different sizes volumes. So that we can work with uniformly sized images, we look for the most common dimensions and resize all images to the power of 2 that is the closest to these dimensions. These dimensions will serve us as the network input and output dimensions. In addition, we found the number of labels in the ground truth labels (2 in Heart, 3 in Hippocampus). This number will be the third dimension in the output layer.

4.2. learning process

4.2.1. architecture

In order to perform the learning stage, we used a U-net network. U-net is a convolutional neural network that was developed for biomedical image segmentation at the Computer Science Department of the University of Freiburg. The original U-net in its two-dimensional version (Fig. 4) is a complete convolution network, i.e., without fully connected layers. The network consists of two symmetrical parts: the contracting path and the expansive path. In the first part, which includes the left side of Fig. 4 and is called the contracting path, convolution is performed at each stage with several filters followed by activation and max pooling, which takes us to the next stage and reduces the resolution. At each stage, the number of filters to perform the convolution increases. Overall, the resolution of the image at each stage decrease, but the features' space increases. The second part, which includes the right side of Fig 4, is called the expansive path and is symmetrical to the first part. At each stage, an upsampling operation is performed instead of max-pooling, thus increasing the resolution while reducing the space of the features until they finally return to the original resolution. This process gives an output that is at the resolution of the input. There are skip connections between parallel layers between the two parts that allow the network not to lose the information from the first layers and reduce the vanishing gradient phenomenon in these layers. In order to reduce overfitting, we also used dropout and batch normalization.

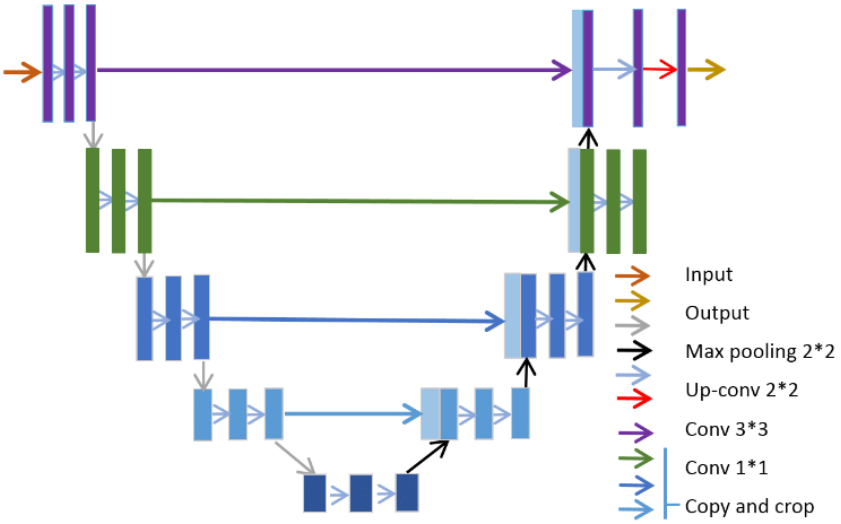

Figure 4. 2D U-Net architecture

4.2.2. Loss function

We used the dice score similarity function to create the loss function and perform the Training. Let p_n^c denote the probability that a pixel belongs to class c, $c \in \{0, \cdots, C\}$, given by the softmax layer, and $t_{n,c} \in \{0, 1\}$ represent the ground truth one-hot label. The dice score for the label c is defined as follows:

$$S_c = \frac{2\sum_n^{N_c} t_{n,c} p_{n,c} + \epsilon}{\sum_n^{N_c} (t_{n,c} + p_{n,c}) + \epsilon} \quad (1)$$

where N_c is the number of pixels labeled as class c and ϵ is a smooth factor.

To perform multiclass segmentation, we defined the loss function as:

$$L = 1 - \sum_c^C S_c \quad (2)$$

After comparing various hyperparameters (see section 6), we used Adam Optimizer, a batch size of 64, a dropout rate of 0.5, and 24 filters.

4.3. augmentations

In order to enlarge the data, we used several augmentations:

- **Horizontal and vertical flip**: In horizontal flip, the flipping will be on the vertical axis. In Vertical flip, the flipping will be on the horizontal axis. Even though only horizontal flips are used in natural images, there is a hypothesis that vertical flips

capture a unique property of medical images [9], namely, invariance to vertical reflection. The flips were also done to the correspondings labels. Figure 5 shows examples of these Horizontal and vertical flips.

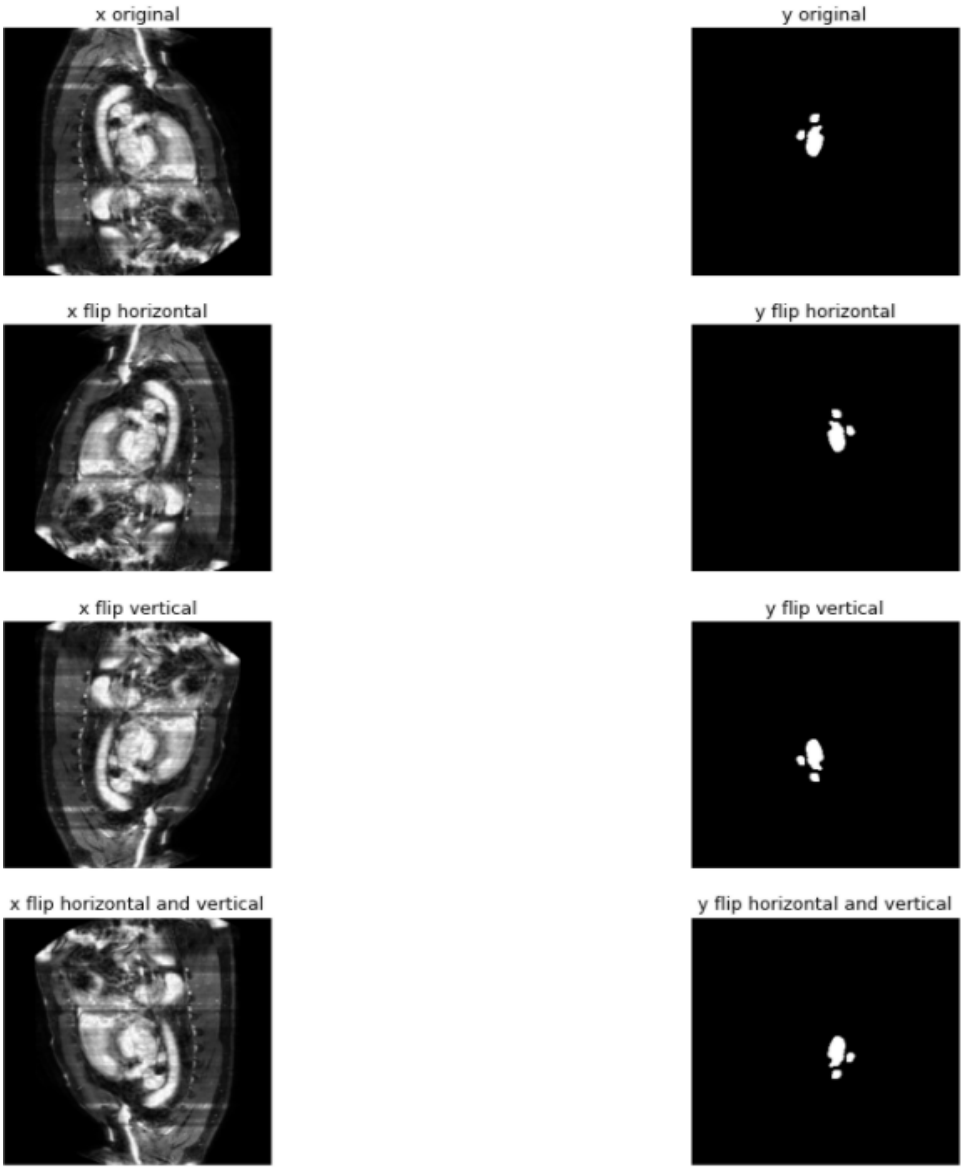

Figure 5. examples of flip augmentation. Left column, the original slice, and its flipped images. Right column: the correspondings flipped labels images.

- **Rotation:** we rotated the images from 0 to 5 degrees and -5 degrees in the clockwise direction. The rotations were also done to the correspondings labels. Figure 6 shows examples of these rotations.

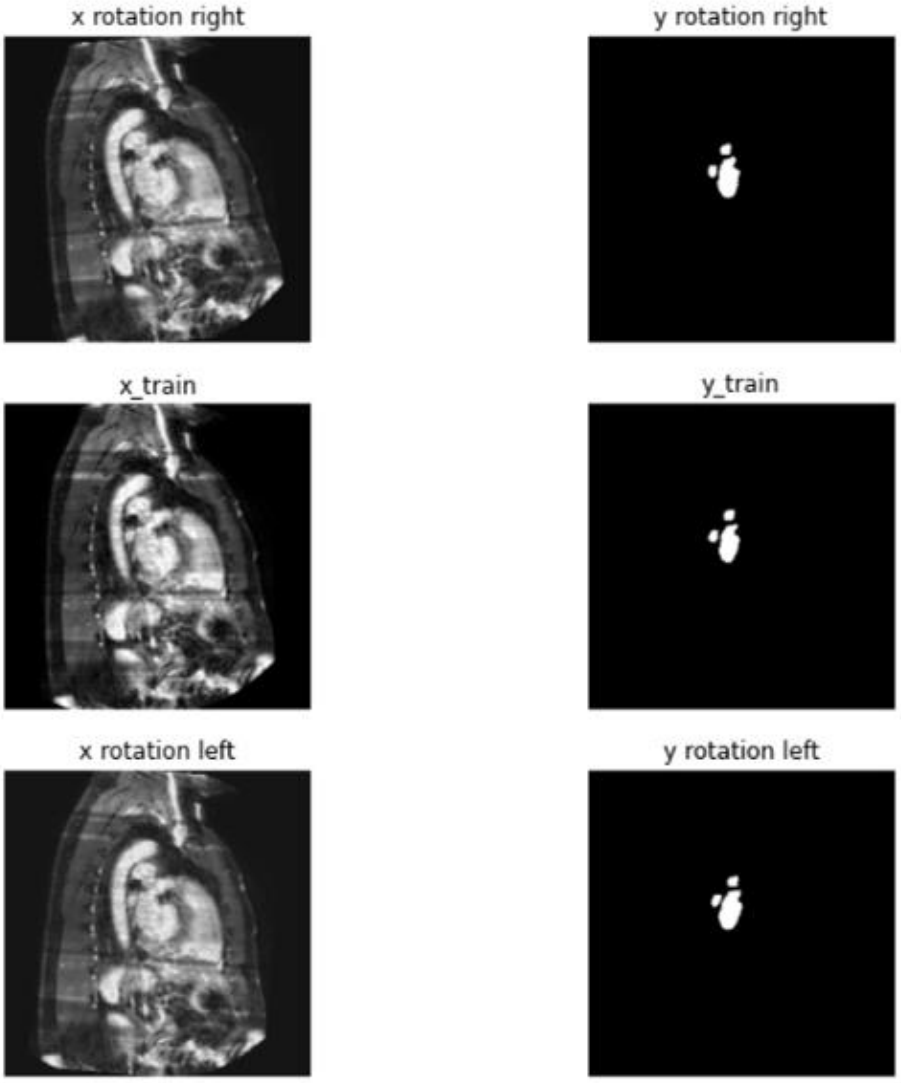

Figure 6. examples of rotation augmentation. Left column, the original slice, and its rotated images. Right column: the correspondings rotated labels images.

- **Adding Noise:** We added Gaussian noise with a $\mu = 0$, $\sigma = 0.1$ the labels are unchanged. Figure 7 shows an example of adding noise to an image.

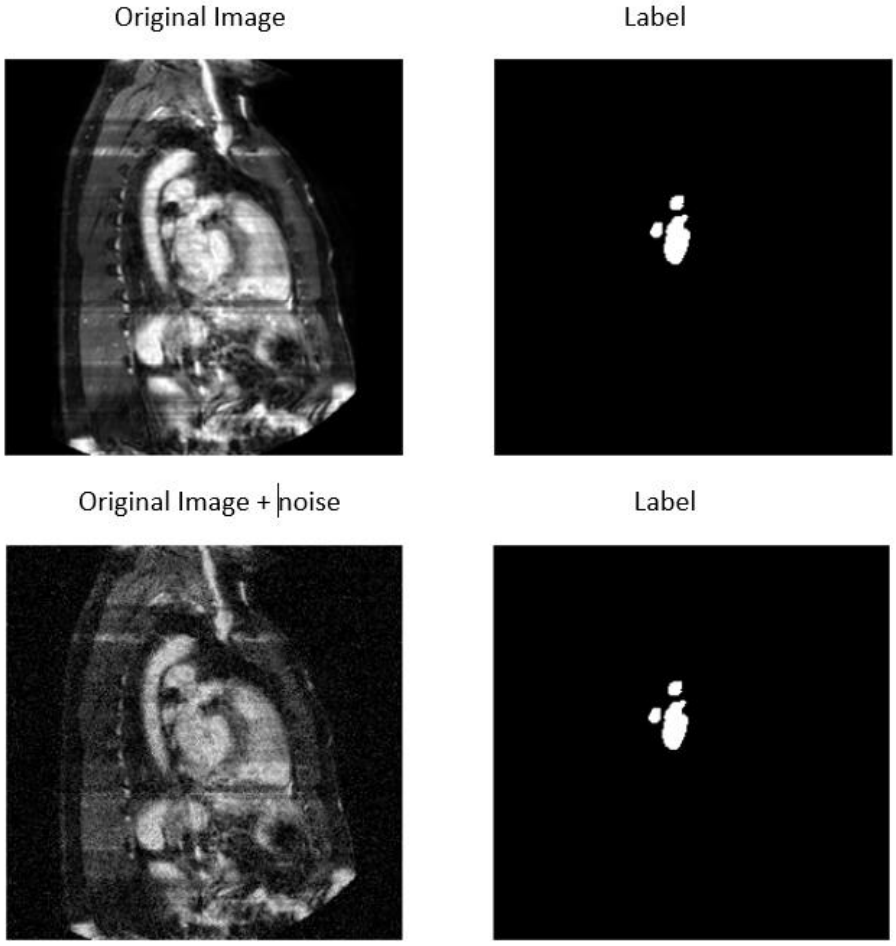

Figure 7. examples of adding noise augmentation. Left column, the original slice, and the original slice+gaussian noise. Right column: the correspondings labels images (unaffected by the noise).

5. Challenges and Difficulties

During the work, we encountered several challenges:

Heavy data

As we mentioned, our data is three-dimensional volumes that require a much larger memory volume than two-dimensional images. In addition, the computational resources needed to deal with such data are much more significant. We selected the least weight (Hippocampus and Heart) challenges to deal with the heavy data. In addition, we cut the 3D volumes into two-dimensional slices. We performed the classification on these slices using a two-dimensional U-NET that requires less computational resources than a three-dimensional U-NET. In addition, we tried to write the most efficient code possible in terms of memory consumption and computational resources.

Low amount train data

The original data contained a low number of examples for the training stage (263 volumes for the Hippocampus and 20 volumes for the Heart). Transferring the data to a two-dimensional form made it possible to produce many examples for the training stage (4500 slices for Hippocampusan 2270 slices for Heart).

In addition, we performed a large number of augmentations to produce additional data for the training stage (see sections 4 and 6).

Imbalance data

The areas in the image that we were required to identify are much smaller than the background of the image. We, therefore, want their impact on the training process to be more significant. So we changed the loss function (see equation 2 in section 4) to give more weight to the less common labels:

$$L = 1 - \sum_c^C \frac{S_c}{N_c} \tag{3}$$

where N_c is the number of pixels labeled as class c and ϵ is a smooth factor. The result of this change can be seen in section 6.

Data is not uniform in its form

As we mentioned, each task has completely different data in its structure (The number of volumes and the size of each volume). in addition, the task varies between segmentation of one region in the Heart and segmentation of two regions in the Hippocampus. As described in section 4, we solve this problem using a pipeline that extracts each dataset's relevant parameters.

Lack of ground truth labels for the Test dataset

The original data set is divided into two parts: train and Test. While the train comes with the ground truth labels for the training stage, the Test comes without ground truth, and its

evaluation should be done by uploading results to the challenge site. Unfortunately, in order to get an evaluation from the site of the challenge it is mandatory to upload results for all ten tasks. Because we only focused on 2, we could not get a score for the Test.

Alternatively, we split the train into two new data sets. Eighty percent of the data will be used for the training stage (Train Set), and twenty percent will be used for evaluation (validation set). We performed the split based on whole volumes so that the data we used for evaluation was foreign to the Train Set. For the original Test set, we performed a visual assessment only (see section 7).

6. Experiments

To find the optimal hyperparameters for our model, we performed several tests. In each experiment, we trained the model with Train Set and performed a prediction of Val Set. To evaluate, we examined the dice coefficient between the ground truth and the network prediction results. The average score of the dice coefficient served us as a measure of the model's success. The training process included 30 epochs that gave good results at a reasonable running time.

The first hyperparameters we examined were the number of filters for the convolution layers and the dropout rate. We try a number of combinations of dropout rates and several filters. The dropout rate was chosen from $\{0.05,0.2,0.5\}$ and the number of filters from $\{12,16,24\}$. In addition, we made a comparison between the Adam optimizer and the SGD optimizer. In this stage, we checked the results for the original data (with no augmentations) and with a batch size equal to 32. The results of these experiments can be seen in tables 1-4.

Number Of Filters	Dropout Value	Validation Accuracy (Dice Score)
8	0.05	0.8999
8	0.2	0.8992
8	0.5	0.9035
16	0.05	0.9058
16	0.2	0.9068
16	0.5	0.9070
24	0.05	0.9034
24	0.2	0.9076
24	0.5	0.9059

Table 1. comparison between various hyperparameters for hippocampus data set with Adam optimizer

Number Of Filters	Dropout Value	Validation Accuracy (Dice Score)
8	0.05	0.5587
8	0.2	0.5193
8	0.5	0.4750
16	0.05	0.7164
16	0.2	0.5937
16	0.5	0.6197
24	0.05	0.8613
24	0.2	0.7487
24	0.5	0.5186

Table 2. comparison between various hyperparameters for hippocampus data set with SGD optimizer

Number Of Filters	Dropout Value	Validation Accuracy (Dice Score)
8	0.05	0.8044
8	0.2	0.8316
8	0.5	0.8359
16	0.05	0.7946
16	0.2	0.8378
16	0.5	0.8214
24	0.05	0.8391
24	0.2	0.8207
24	0.5	0.8367

Table 3. comparison between various hyperparameters for Heart data set with Adam optimizer

Number Of Filters	Dropout Value	Validation Accuracy (Dice Score)
8	0.05	0.4932
8	0.2	0.4910
8	0.5	0.5019
16	0.05	0.4967
16	0.2	0.5010
16	0.5	0.4996
24	0.05	0.5283
24	0.2	0.4999
24	0.5	0.4971

Table 4. comparison between various hyperparameters for Heart data set with SGD optimizer

As can be seen in the results, the Adam optimizer achieves better accuracy than the SGD optimizer. In addition, it can be seen that for Heart, we got the best results with 24 filters and a dropout rate of 0.5. for Heart. Although in the Hippocampus, the results were better with a dropout rate of 0.2, the difference is not significant. Therefore overall, the results with a dropout rate of 0.5 are better. Figures 8-11 show the loss per epochs the accuracy per epoch for both Hippocampus and Heart with the optimal parameters (Adam optimizer, dropout rate of 0.5 and 24 filters).

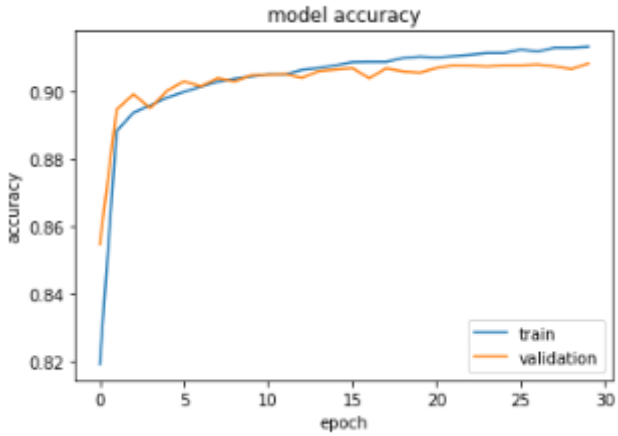

Figure 8. model accuracy per epoch for the Hippocampus data set with batch size 32, a dropout rate of 0.5 and 24 filters. It can be seen that the validation accuracy is relatively close to the training accuracy.

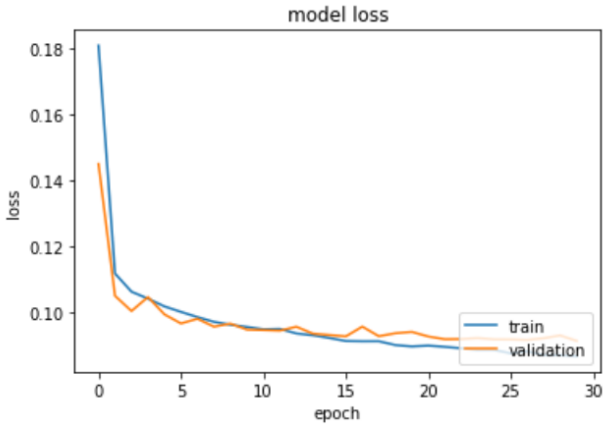

Figure 9. loss per epoch for the Hippocampus data set with batch size 32, a dropout rate of 0.5, and 24 filters. It can be seen that the loss graph is symmetric to the accuracy.

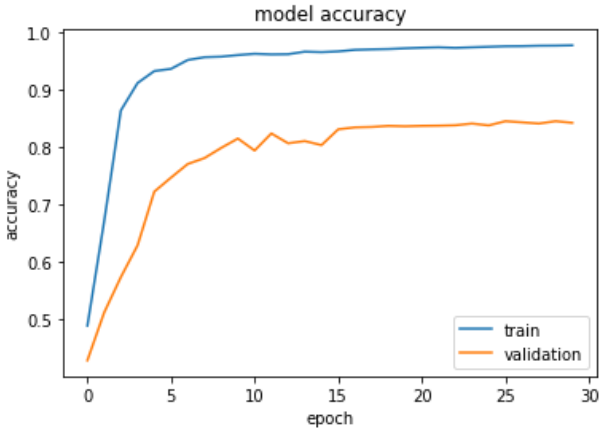

Figure 10. Model accuracy per epoch for the Heart data set with batch size 32, a dropout rate of 0.5, and 24 filters. It can be seen that the validation accuracy is much lower than the training accuracy, so the model suffers from overfitting.

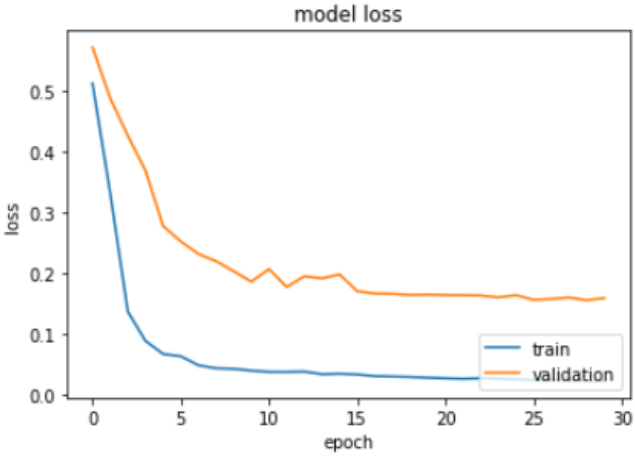

Figure 11. Loss per epoch for the Heart data set with batch size 32, a dropout rate of 0.5, and 24 filters. It can be seen that the loss graph is symmetric to the accuracy.

The next hyperparameter we wanted to test is the batch size. At this stage, we performed the experiment with the Adam Optimizer and the optimal hyperparameters we found in the previous experiment(dropout rate =0.5 and 24 filters). We try three batch sizes: 16, 32, and 64. As can be seen in table 5, we got the best results for batch size 64. Figures 12-15 show the loss per epochs the accuracy per epoch for both Hippocampus and Heart with a batch size equal to 64.

Batch Size	**Heart (Dice Score)**	**Hippocampus (Dice Score)**
16	0.8598	0.8570
32	0.8367	0.9084
64	0.8885	0.9087

Table 5. comparison between various batch sizes for both datasets.

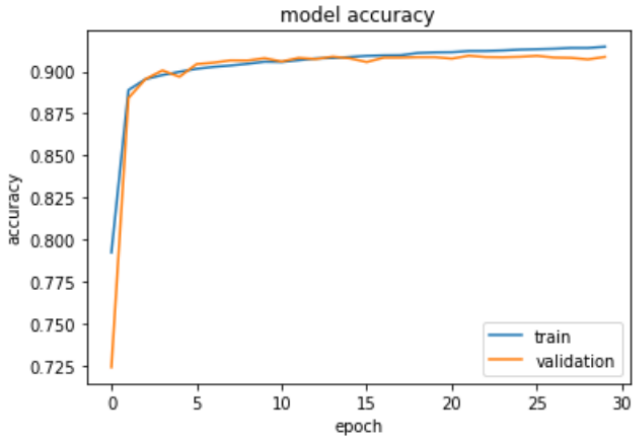

Figure 12. model accuracy per epoch for the Hippocampus data set with batch size 64, a dropout rate of 0.5, and 24 filters. It can be seen that the validation accuracy is relatively close to the training accuracy. In addition, It's converging faster than the run with batch size 32

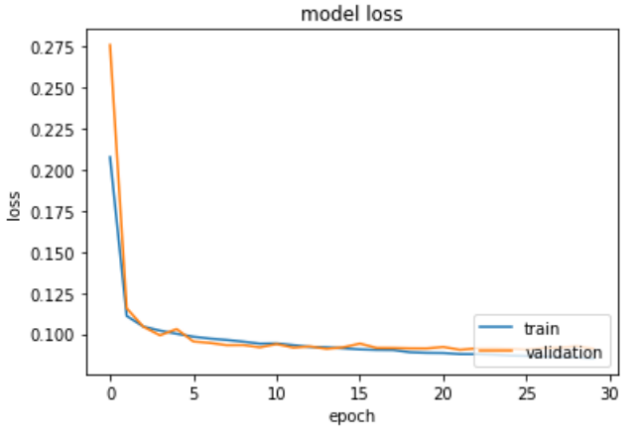

Figure 13. loss per epoch for the Hippocampus data set with batch size 64, a dropout rate of 0.5, and 24 filters. It can be seen that the loss graph is symmetric to the accuracy.

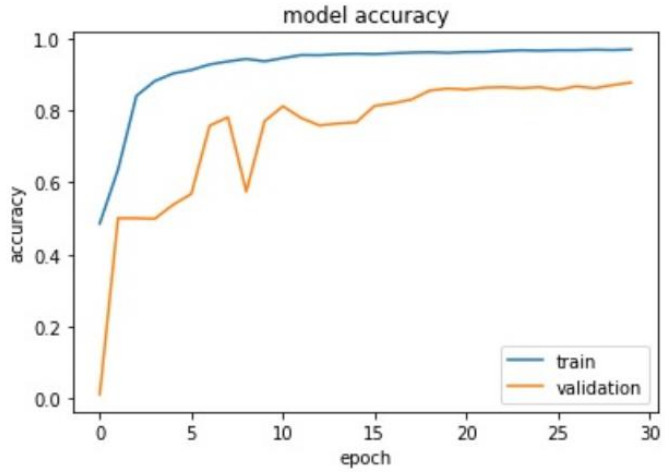

Figure 14. model accuracy per epoch for the Heart data set with batch size 64, a dropout rate of 0.5, and 24 filters. It can be seen that despite the improvement in results, overfitting still exists here

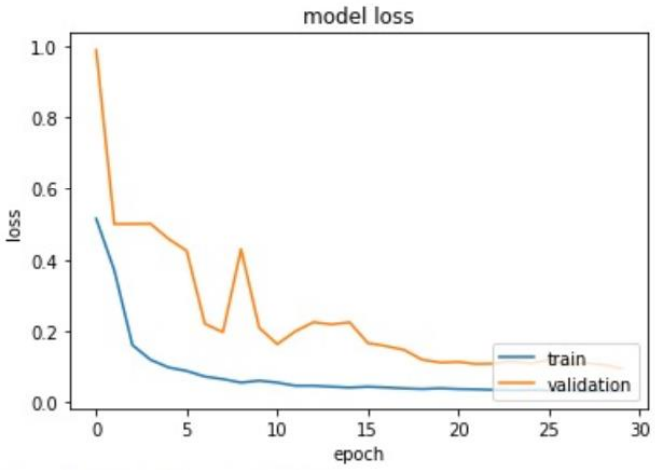

Figure 15. loss per epoch for the Hear data set with batch size 64, a dropout rate of 0.5, and 24 filters. It can be seen that the loss graph is symmetric to the accuracy

After finding the optimal hyperparameters, we wanted to test the effect of the augmentations on the results. We performed the experiment again with the optimal hyperparameters we found in the previous experiments (batch size= 64, dropout rate =0.5, and 24 filters). We tested the model with a different augmentation. Due to RAM limitations, we did not perform a combination of the several augmentations (other than noise and rotation). In addition, we try to use our new weighted loss function (equation 3) with the original data and no augmentations. Table 6 summarizes the results of all augmentation and the experiment with the new weighted loss function.

	Heart (Dice Score)	Hippocampus (Dice Score)
Original data	0.8885	0.9087
Flip vertical	0.8950	0.9089
Flip horizontal	0.9022	0.9114
Flip vertical + horizontal	0.9319	0.9123
Gaussian noise	0.9395	0.9070
Rotation	0.9415	0.9082
Rotation + noise	0.9304	0.9045
Original data with weighted loss	0.9340	0.9084

Table 6. comparison between various augmentations techniques and the new weighted loss function for both datasets.

It can be seen that for Heart, the best results were obtained with the augmentation of rotation. In contrast, for Hippocampus, the best results were obtained for Flip vertical + horizontal, although the difference between the augmentations was minor. That is, it is not possible to decide which of the augmentations is better in general. We speculate that the augmentations have resulted in significantly better results for the Heart because of the small amount of original data. Figures 16-17 show the loss per epochs the accuracy per epoch for Hippocampus with flip horizontal + vertical augmentation. Figures 18-19 show the loss per epochs the accuracy per epoch for Heart with rotation augmentation. It is important to note that our weighted loss function did achieve better results than the original function (with no augmentations), and it can be assumed that if we performed augmentations in combination with the new weighted loss function, we would get better results.

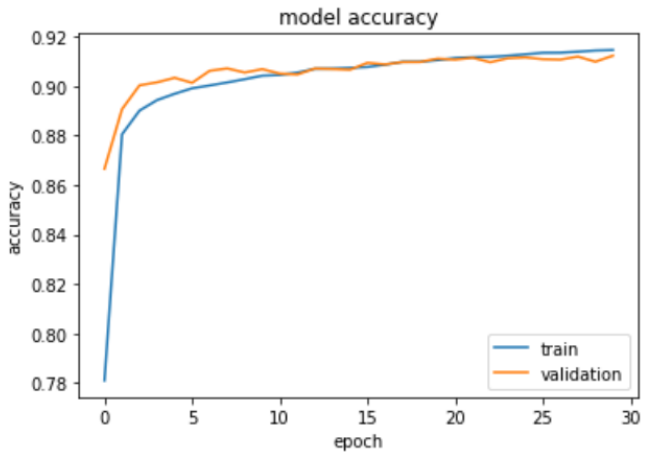

Figure 16. model accuracy per epoch for the Hippocampus data set with batch size 64, a dropout rate of 0.5, 24 filters, and flip horizontal + vertical augmentation. It can be seen that there is a slight improvement in the results compare to the original data.

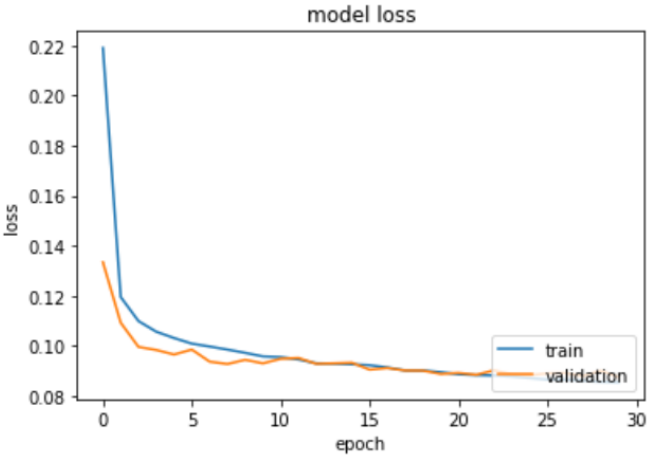

Figure 17. loss per epoch for the Hippocampus data set with batch size 64, a dropout rate of 0.5, 24 filters, and flip horizontal + vertical augmentation.

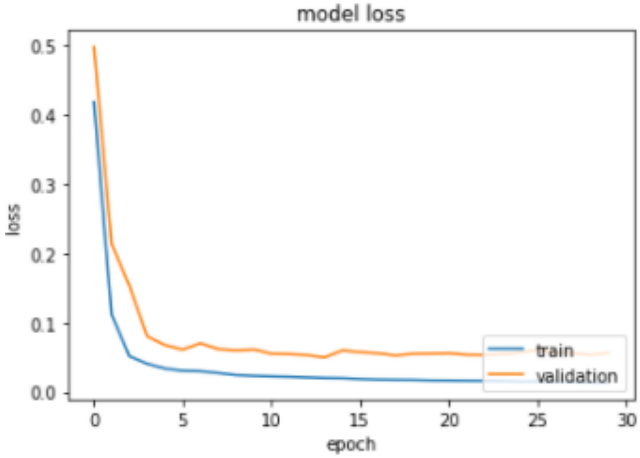

Figure 18. model accuracy per epoch for the Heart data set with batch size 64, a dropout rate of 0.5, 24 filters, and rotation augmentations. It can be seen that the overfitting still exists but is less significant.

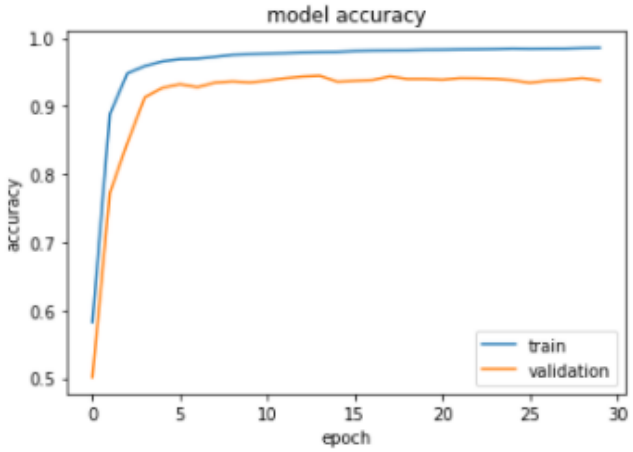

Figure 19. Loss per epoch for the Heart data set with batch size 64, a dropout rate of 0.5, 24 filters, and rotation augmentation.

7. Results

As mentioned in the previous section, our algorithm achieved an accuracy of 88.95 percent in Heart and 90.87 for Hippocampus without augmentations. With augmentations, we got even better results of 94.15 for Heart and 91.23 for Hippocampus. Figure 20 shows the results of our model on Heart, after running with the optimal parameters and rotation augmentation. Figure 21 shows the results of our model on Hippocampus, after running with the optimal parameters and Flip vertical + horizontal augmentation. The slices we show are the middle slices of randomly picked volumes from Val Set. In both cases, it can be seen that we got results relatively close to the ground truth.

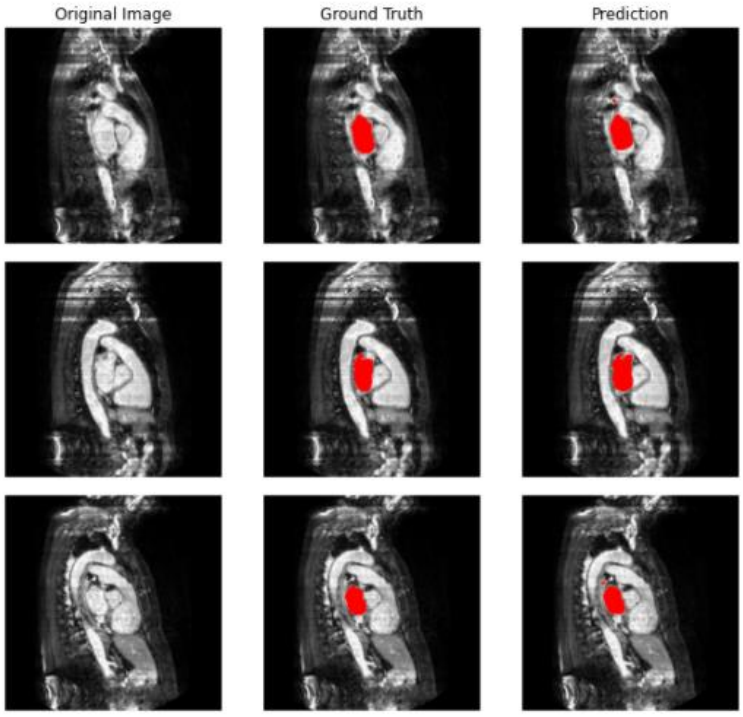

Figure 20. Results of prediction for the validation set of Heart compared to the ground truth. The left atrium is in red.

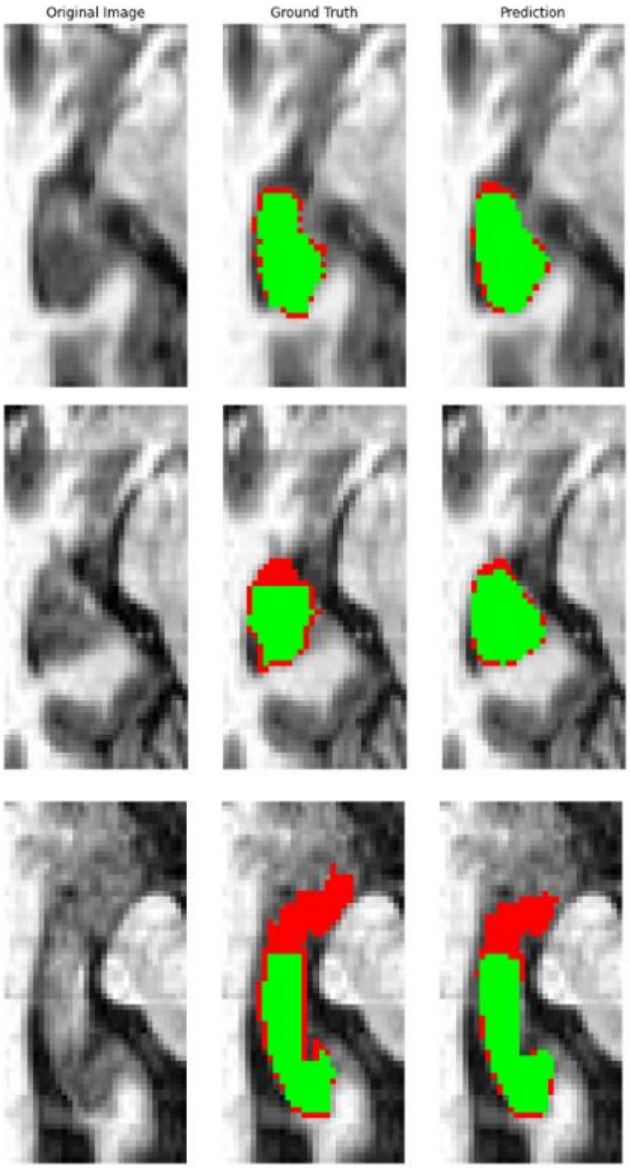

Figure 21. Results of prediction for the validation set of Hippocampus compared to the ground truth. The head of the Hippocampus is in red, and the body is in green.

As mentioned before, the Test set came without ground truth labels. Therefore, we present in figures 22 and 23 the results for the Test set without a comparison to the ground truth. In the case of Heart, it can be seen that the prediction is in the area where the left atrium is really located. In the case of the Hippocampus, it is more difficult for the inexperienced eye.

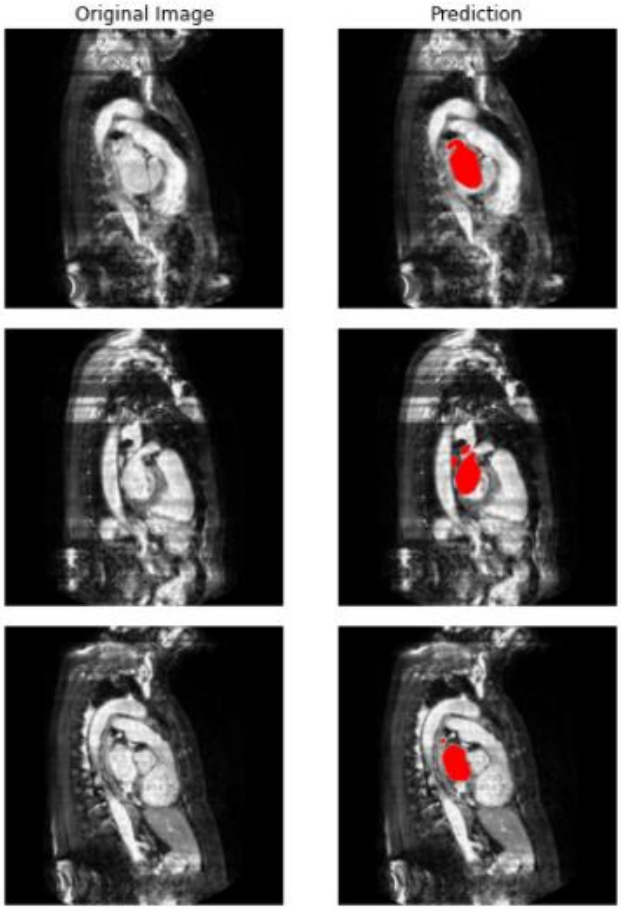

Figure 22. Results of prediction for the test set of Heart. The left atrium is in red.

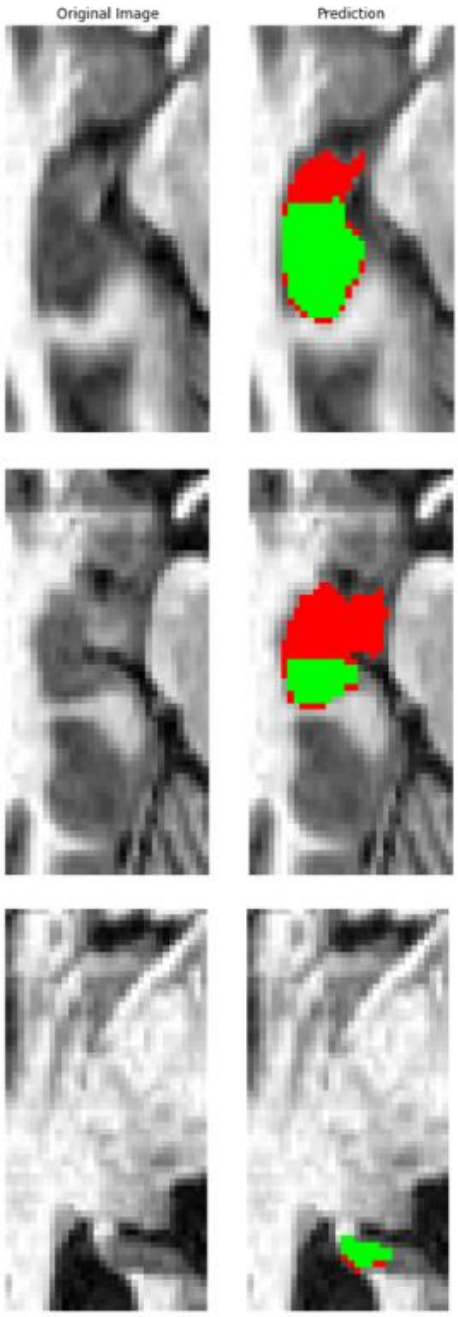

Figure 23. Results of prediction for the validation set of Hippocampus. The head of the hippocampus is in red, and the body is in green.

In addition, we have created a function that presents as a video the segmentation results for a specific volume. The function allows switching between slices and thus an understanding of the three-dimensional shape. Such a demo can be seen at the end of the attached notebook.

8. Conclusions and Summary

In conclusion, we have created a model that allows the segmentation of medical data. Our model knows how to deal with different data sets in various forms and perform successful predictions without human interaction. Although we could not get a score for the Test, we validated ourselves and by splitting the data. In addition, we performed experiments to find the best hyperparameters of the model. We also showed that when the amount of original data is small, augmentations can significantly improve the results. We deal with imbalanced data by creating a new weighted loss function and with heavy 3D data by slicing it into 2D images. The results we obtained are not perfect, but they are impressive and show how effective segmentation using the U-NET is.

Future work can try more combinations of different augmentation. In addition, it may be possible to achieve better results by using several slices at a time rather than on a single slice basis. It will also be interesting to see how our model deals with the rest of the tasks in the challenge.

9. References

[1] M. H. Hesamian, W. Jia, X. He, and P. Kennedy, "Deep Learning Techniques for Medical Image Segmentation: Achievements and Challenges," *Journal of Digital Imaging*, vol. 32, no. 4, pp. 582–596, 2019, doi: 10.1007/s10278-019-00227-x.

[2] Wang, C.; MacGillivray, T.; Macnaught, G.; Yang, G.; Newby, D.E. A two-stage 3D U-net framework for multiclass segmentation on full resolution image. arXiv 2018, https://arxiv.org/abs/1804.04341.

[3] A. A. Novikov, D. Lenis, D. Major, J. Hladuvka, M. Wimmer, and K. Buhler, "Fully Convolutional Architectures for Multiclass Segmentation in Chest Radiographs," *IEEE Transactions on Medical Imaging*, vol. 37, no. 8, pp. 1865–1876, 2018, doi: 10.1109/TMI.2018.2806086.

[4] M. Antonelli *et al.*, "The Medical Segmentation Decathlon," 2021, [Online]. Available: http://arxiv.org/abs/2106.05735

[5] W. Weng and X. Zhu, "INet: Convolutional Networks for Biomedical Image Segmentation," *IEEE Access*, vol. 9, pp. 16591–16603, 2021, doi: 10.1109/ACCESS.2021.3053408.

[6] C. Tobon-Gomez *et al.*, "Benchmark for Algorithms Segmenting the Left Atrium From 3D CT and MRI Datasets," *IEEE Transactions on Medical Imaging*, vol. 34, no. 7, pp. 1460–1473, 2015, doi: 10.1109/TMI.2015.2398818.

[7] J. B. W. Williams *et al.*, "The Structured Clinical Interview for DSM-III-R (SCID) Reliability Description of Sites," *Archives of general psychiatry*, vol. 49, no. 8, pp. 630–6, 1992.

[8] J. C. Pruessner *et al.*, "Volumetry of hippocampus and amygdala with high-resolution MRI and three-dimensional analysis software: Minimizing the discrepancies between laboratories," *Cerebral Cortex*, vol. 10, no. 4, pp. 433–442, 2000, doi: 10.1093/cercor/10.4.433.

[9] Z. Hussain, F. Gimenez, D. Yi, and D. Rubin, "Differential Data Augmentation Techniques for Medical Imaging Classification Tasks," *AMIA ... Annual Symposium proceedings. AMIA Symposium*, vol. 2017, pp. 979–984, 2017.