BABEŞ-BOLYAI UNIVERSITY CLUJ-NAPOCA

FACULTY OF MATHEMATICS AND COMPUTER SCIENCE

SOFTWARE ENGINEERING

DISSERTATION THESIS

Web Cardiac MRI Segmentation and Visualization

Supervisor

Prof.Univ.Dr. Dioșan Laura

Author

Ivanov Silviu-Gabriel

2020

UNIVERSITATEA BABEŞ-BOLYAI CLUJ-NAPOCA

FACULTATEA DE MATEMATICĂ ŞI INFORMATICĂ

SOFTWARE ENGINEERING

LUCRARE DE DISERTAŢIE

Aplicație Web pentru Segmentarea și Vizualizarea Cardiacă a unui RMN

Supervisor

Prof.Univ.Dr. Dioșan Laura

Author

Ivanov Silviu-Gabriel

2020

Abstract

Nowadays one of the hardest tasks in the field of medicine is to interpret and analyse the MRI scans in order to understand them. In this paper, we will present an web application which will be able to compute a 3D volume and provide an easy way to visualize the heart chambers based on an MRI scan. The intelligent component of the application is based on a DenseVNet fully convolutional neural network which was trained on a set of MRI which are already labeled by medical experts. This labels are split into three classes: heart, blood vessels and the rest of the MRI chest scan. The software assistant we are going to implement is targeted at medicine students or anybody else who is working on chest scans and needs an easy to use tool for cardiac segmentation. Also, we are hoping that, by bringing the field of medicine and AI together we will have a great impact on the learning curve of those people and we will facilitate as much as possible the already cumbersome process of reading the MRIs. The first part of the paper presents the field of AI and serves as an introduction to common terms we are going to use. Fallowed by our approach and implementation of the application. And last but not least, we will analyse the results we achieved. This work is the result of my own activity. I have neither given nor received unauthorized assistance on this work.

Contents

List of Figures

Chapter 1

Introduction

Nowadays cardiac segmentation has a very high impact on various applications related to the health and pathology domain. Magnetic resonance images (MRI) that are composed by various grey shades, are used worldwide in the medical field. Usually, students are facing multiple challenges when they are trying to recognise various organs in MRI scans. Without having a professional point of view, the process of identifying the position of the organs and segmenting them, may prove highly troublesome.

On the other hand, researches shows that Convolutional Neural Networks are very efficient in solving tasks like classification, segmentation and object detection. The vastly anomalies among patients requires a segmentation process to be applied to the MRIs in order for the texture analysis to be done properly and efficiently [7].

In this paper we will present our approach on helping the segmentation process, based on a fully convolutional neural network. One of the biggest challenges will be to train and test our architecture with a small data-set. Even if, segmentation tasks nowadays can be easily done, in the field of medicine there is still room for improvements due to the lack of already labeled images, and also because current algorithms requires a considerable amount of computing power. In consequence, we will need to augment our data before the training phase. The final product will be an web application which will facilitate the dificult process of segmentation, by using a Dense V-Net neural network trained on images that were already labeled by medical experts.

In the next chapter we will briefly present the field of AI. It will later serve as an introduction to common terms we are going to use throughout the paper. The third chapter will analyse the current state of the art implementation for heart segmentation. Here we will focus on the award winning paper of HVSMR[1] 2016 challenge. The forth chapter will focus on the software development process of our implementation of the Dense V-Net and the web application. In the next chapter, we will talk about the results we achieved and the refinements that can be made on both the neural network and the web application. In the last chapter we will present our conclusion and future work.

[1]MICCAI Workshop on Whole-Heart and Great Vessel Segmentation from 3D Cardiovascular MRI in Congenital Heart Disease

Chapter 2

Theoretical Background

Artificial intelligence or AI is a field of study in which tries to understand the intelligent entities of our world. AI currently encompasses a variety of sub-domains, from specific requirements such as playing chess, writing poetry, diagnosing diseases or demonstrating mathematical theorems to more general areas such as perception and logical reasoning. Scientists in this vast field often tend to apply their methods and algorithms in any other area that requires human intellectual effort, thus demonstrating the universality applicability of this field [9].

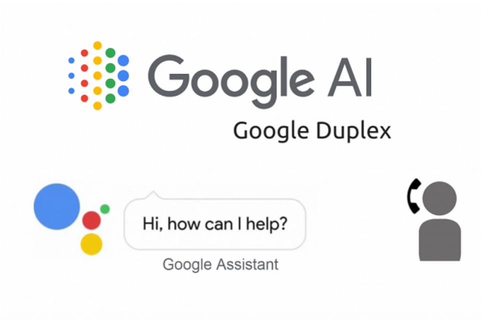


Figure 2.1: Google Duplex [13]

What does it mean to be intelligent and how can we test this? Proposed in 1950 by Alan Turing[1], the test that bears his name, *"The Turing Test"* was designed to bring a satisfactory definition of computational intelligence. The test consists of the computer's ability to achieves human cognitive performance in order to be able to mislead a human interrogator. Some computer capabilities required to successfully pass this test would be: *natural language processing*, to communicate and master a certain language completely; *representation of information*, ability to store information during a question; *machine learning*, to adapt to new circumstances and finally *automatic reasoning*, to be able to use stored information to draw new conclusions or answer

[1] Personality of great influence in the field of computer science, former mathematician, logician and philosopher, can be identified as one of the parents of artificial intelligence.

questions.

Figure 2.1 is part of the Google presentation in May 8, 2018, where a new technology was presented called ”Google Duplex” capable of having sophisticated conversations in a completely autonomous manner in order to create a reservation. Although, it is acknowledged that the system can not autonomously complete reservations that have a high degree of difficulty, from my point of view this technology is among the first that are able to pass the turing test in a natural conversation.

2.1 Ethics

”The ethics for humane technology framework provides lenses to understand human rights in the digital age, to understand the technological phenomena that threaten these rights and principles to build technology so as to create a beneficial future for humanity.”[2]

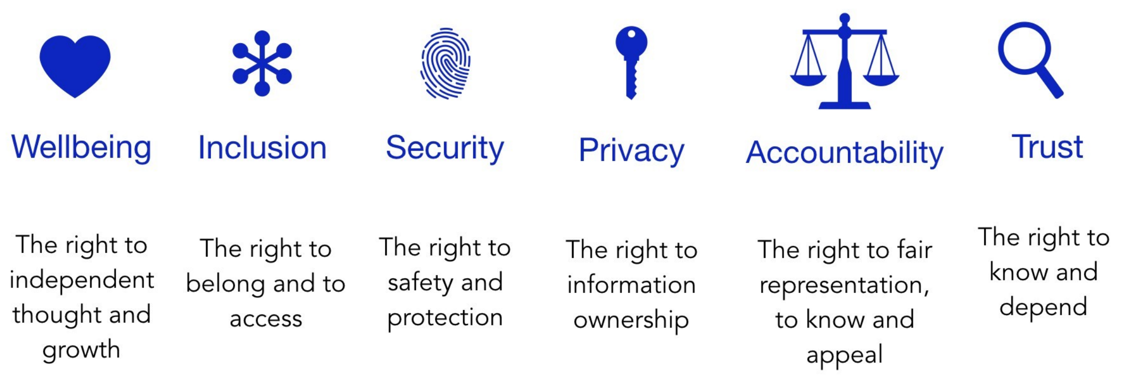


Figure 2.2: Six lenses of human ethical problems [2]

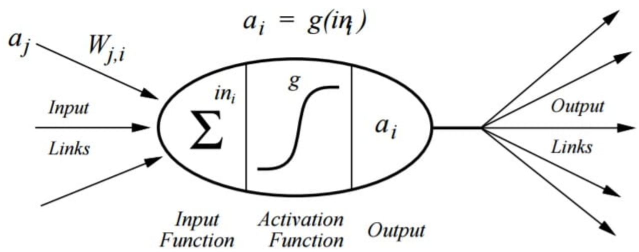
Nowadays intelligent machines can do almost anything, from cooking, recognizing faces while authorizing phone access, to optimizing traffic and also making art. With this tools the world becomes better and better, but with all of these benefits, some ethical values can be violated.

The primary concern of the population is that AI can take their working place. The McKinsey Global Institute report [6] mentions that until 2030 up to 14% of the workers will be affected or will lose their jobs due to automation and AI. At the same time there is the fear that these intelligent machines will do some very serious mistakes that can lead to disasters.

In this paper we only purpose to develop a medical helper, for students, that can be scaled in the future for helping doctors. Thus, this will not affect any particular job in this field of work, also because the specialist are necessary in order to detect anomalies, diseases and other parts that AI couldn't recognise. Due to the fact that, students are at the beginning of their experience with MRI's this assistant is a very appropriate and ethical compliant application.

2.2 Machine Learning

Machine Learning often represents the changes that we bring to a system that deals with meeting certain requirements in the field of artificial intelligence. Even if a system's activity is observing and modeling the environment in order to determine the correct actions that must be done, autonomously without the need for outside interaction, it does not necessarily mean that it is intelligent [8]. We will further analyze what changes to the components of this "agent" are required in certain situations, in order for it to be called "intelligent".



$$a_i = g(\sum_j W_{j,i} a_j)$$

Figure 2.3: Perceptron: The Artificial Neuron [1]

- The possibility to adapt to everything that appears new. Because the world is in a continue motion and evolution, it would not be practically to have to adapt and reshape the system every time. Thus, the implementation of machine learning becomes mandatory in order to cope with the flow of continuous changes.

- We would like the ***"agent"*** to be able to adjust his internal structure by himself in order to arrive to the desired result, depending on the examples we provide. Thus, after a great number of iterations and processed examples, it is able to get very close to the desired result, even if the input data is new.

- It often happens that the developed machines do not have a suitable structure for the environment in which it operates either due to logical or functional errors. In this case various machine learning methods can be used to aid in design of these machines.

Nowadays, the interest in the field of machine learning is constantly growing, a large part of it is due to the need to create models capable of using existing data sets to train. One of the most common approaches is the use of DNN[2] for Deep Learning [5]. This approach being suitable for a variety of tasks, some of them being even generating content such as images, videos and audio recordings.

2.3 ANN

We consider an artificial neural network as a simplified model for the biological network's structure of the neurons. An ***ANN***[3] is composed of interconnected processing units. The model consists of a component for summing and a component for returning the result [14]. The component that has the purpose of summation receives *N* values as input data, after which assigns to each of them a weight and finally calculates their sum. A second component takes this *activation value* and returns it as a *signal*. Depending on the sign of each weight, the input is said to be either *excitatory* or *inhibitory* [11]. Inputs and outputs can also be discrete or continuous. In Figure 2.3 it can be seen how a neuron works. These neurons are grouped in layers, the network being formed from one or more interconnected layers. In most cases the network contains a input layer, one output layer and a few layers between the two called *hidden* layers.

One of the most popular learning methods is *backpropagation*. This process involves the propagation of the errors back, starting from the output layer to the hidden layers in order to recalculate the weights for processing units contained in those layers. The error is calculated using the difference between the desired results and the results obtained from each output unit.

The simplest neural network is one in which the neurons on a layer communicate only with those on the next layer (**feedforward**), the information traveling from the input to the

[2]Deep Neural Network
[3]Artificial Neural Network

output. The convolutional neural network is also popular, where data is process in convolutions. This network possesses a high learning ability and is often used for more complex purposes such as classification or segmentation. Classification is the process in which a neural network receives an input (e.g. an image) and outputs a **class** or in some cases a **probability** of being a specific class. Each of this classifications can be measured by using several metric. The most common metric that used in this process is the Accuracy($\frac{Numberofcorrectpredictions}{Totalnumberofpredictionsmade}$). Whilst, segmentation is the process in which unit of the input (e.g. pixel-2D image, voxel-3D image) is classified.

2.4 V-Net

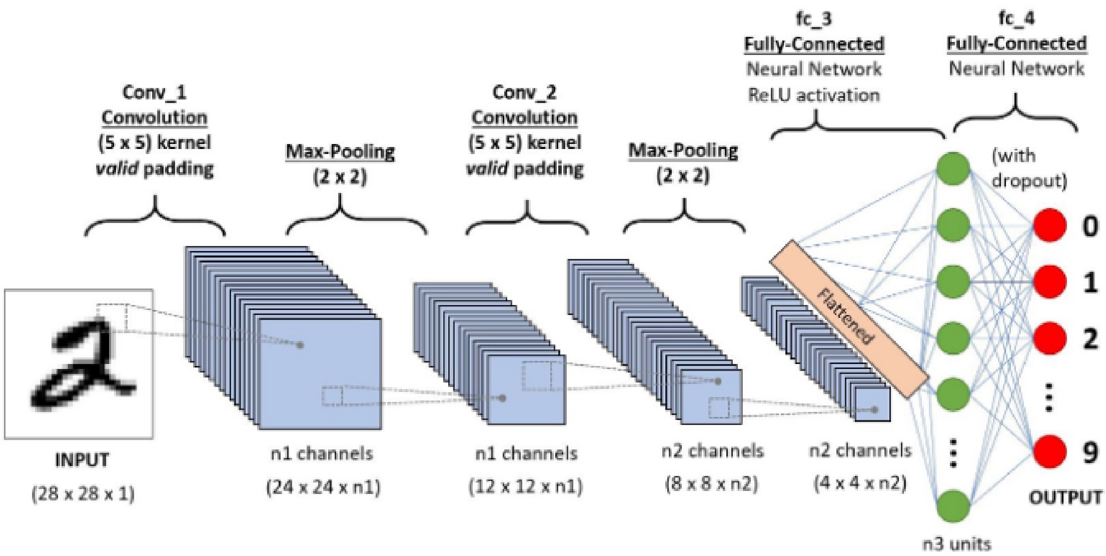


Figure 2.4: A CNN sequence to classify handwritten digits [10]

In order to describe a fully convolutional neural network for volumetric medical image segmentation, we first need to understand the architecture of a CNN[4]. A **Convolutional Neural Network (CNN/ConvNet)** can be described as a algorithm which receives as input an image, and assigns various weights and biases to multiple aspects/objects from that image, in order to differentiate them one from the other. In order to achieve this aspect of the algorithm, a **convolution layer** is needed to extract the features from this input image using filters (Kernals). An example of a convolutional neural network can be seen in Figure 2.4.

[4]Convolutional Neural Network

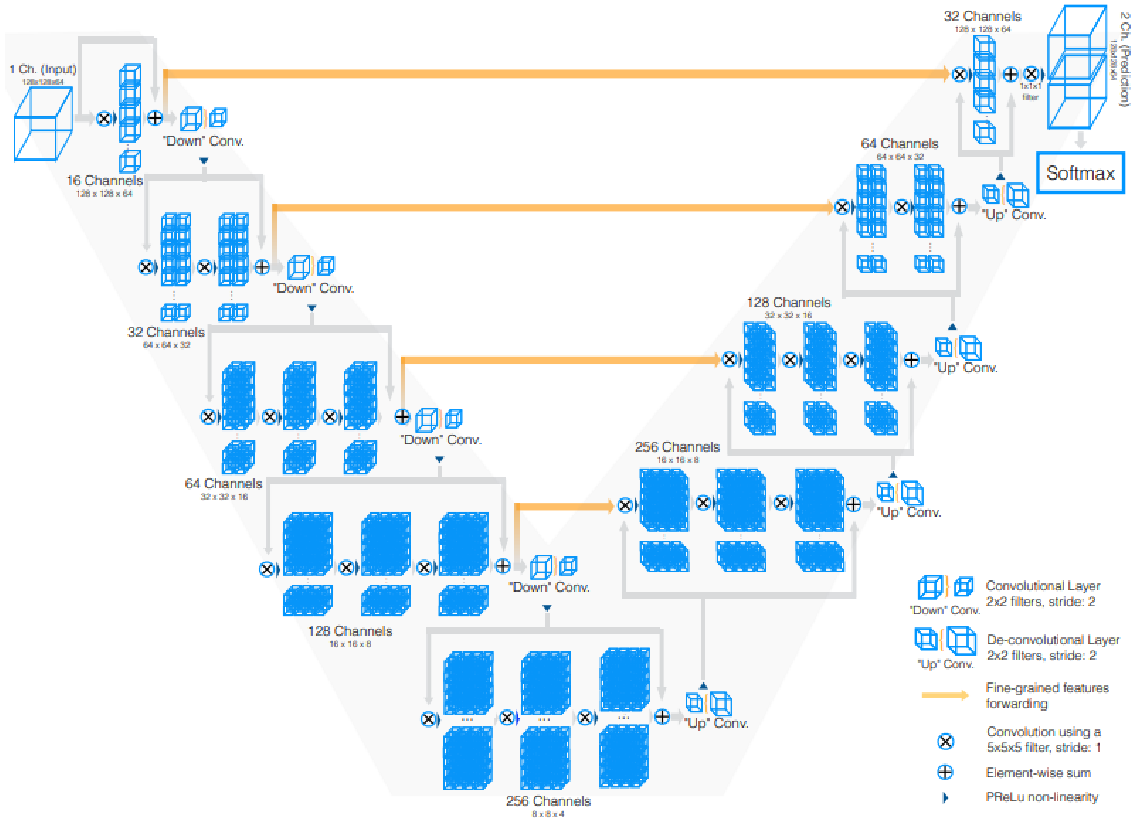


Figure 2.5: Schematic representation of the V-Net architecture

Despite of the CNN's popularity, the majority of the approaches are tackling only the 2D images, but for the medical field of work it is needed an architecture that is able to process 3D volumes. To cope with this aspect, we will be using a convolutional neural network for volumetric medical image segmentation or simply called ***V-NET*** [7]. The architecture of this neural network can be seen in Figure 2.5.

2.5 Dense V-Net

A V-network architecture contains two main parts: the downsampling[5] and upsampling[6] subnetworks. These two are connected in order to propagate to the final segmentation the higher resolution data. Typically, the v-nets use shallow strided-convolution downsampling and transpose-convolutional upsampling units with concatenating skip connection in each resolution.

[5]The process of downscaling images or results in order to reduce the storage size, increase the processing speed and manageability

[6]The process of increasing the size of a image or result with the purpose of reconstruction or enhancement

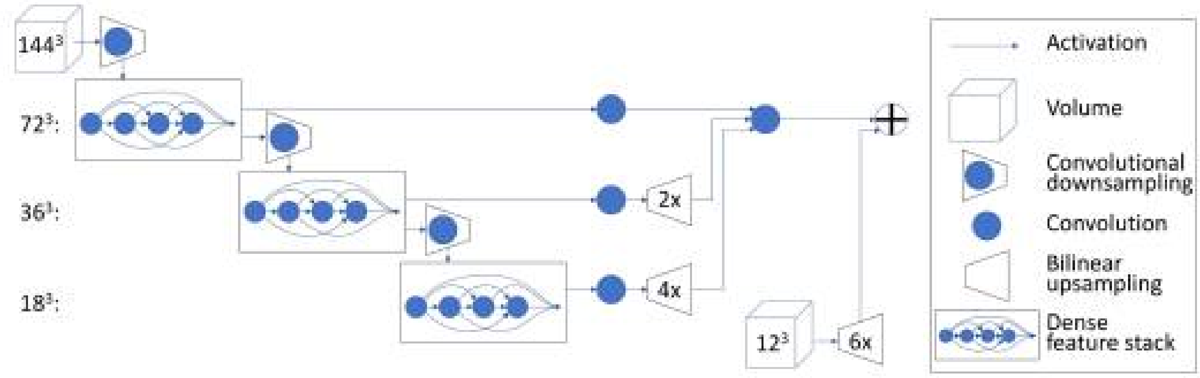


Figure 2.6: DenseVNet network architecture [4]

The DenseVNet differs from this by having the downsampling as a sequence composed of three dense feature stacks connected between them by downsampling strided convolutions. In the first step, the 72^2 feature maps are created by using a strided convolution. A stride represents the amount of movement between applications of the filter on the input image. Secondly, a *"cascade"* composed of strided convolutions and dense feature stacks is generating at three different resolutions the activation maps. In the third step, a convolution unit is used for each resolution with the purpose of reducing the number of features. After the bilinear upsampling, the final maps are connected for generating the likelihood logits. The final step is to add these to the upsampled spatial before the segmentation logit is generated. The architecture from Figure 2.6 will be the one we will use for our neural network because after several experiments [4], the dense connections structure greatly improves the performance and the accuracy of the segmentation.

Chapter 3

Scientific Problem and Research Questions

The MRI scans are gray-scale and very noisy, with a lot of organs. To be able to delimit the heart and the chambers from the rest of the image, a specialized doctor is needed. When a medical student starts to learn about the heart and its chambers, he needs a lot of research and a doctor to show him where everything is and how it looks like. Therefore, we are trying to replace the specialized doctor by automating the whole process of segmentation. For this purpose, we will train a Dense V-Net on 3D images provided by the data-set from HVSMR 2016, the axial full volumes. A label is calculated for each voxel of the volume, based on the already labeled volumes by medical experts. Thus, at the end of the process, the output will be a several regions with the same label, representing the: heart, blood vessels and the background.

During the training process, several challenges can be identified. First challenge is the difficulty to process whole volumes, each one of them occupying several megabytes of data. Secondly, each volume being a gray-scale 3D image, and not 100% accurate, it presents a lot of noise, making the delimitation process (where a segment ends) very ambiguous. The last main challenge we identified is the lack of data, even after the augmentation process, we still don't have enough data in order to train a reliable model, which will be able to segments all sorts of chest MRIs. In this way, another common training problem will appear, the overfitting of the neural network on the few data we are showing to it. In consequence, the neural network will not be able to properly segment new volumes, from the real world, that were never presented to it in the training process.

The purpose of the application is to facilitate the activity of learning the heart, of the medical students or anybody who is curious about this subject. The final product will be a web application, in this way, it can be accessed by anybody from any device. As main functionality, the student will be able to upload an MRI image, and as a result the student will be able to see in a 3D view the final segmentation of the heart.

3.1 Related Work

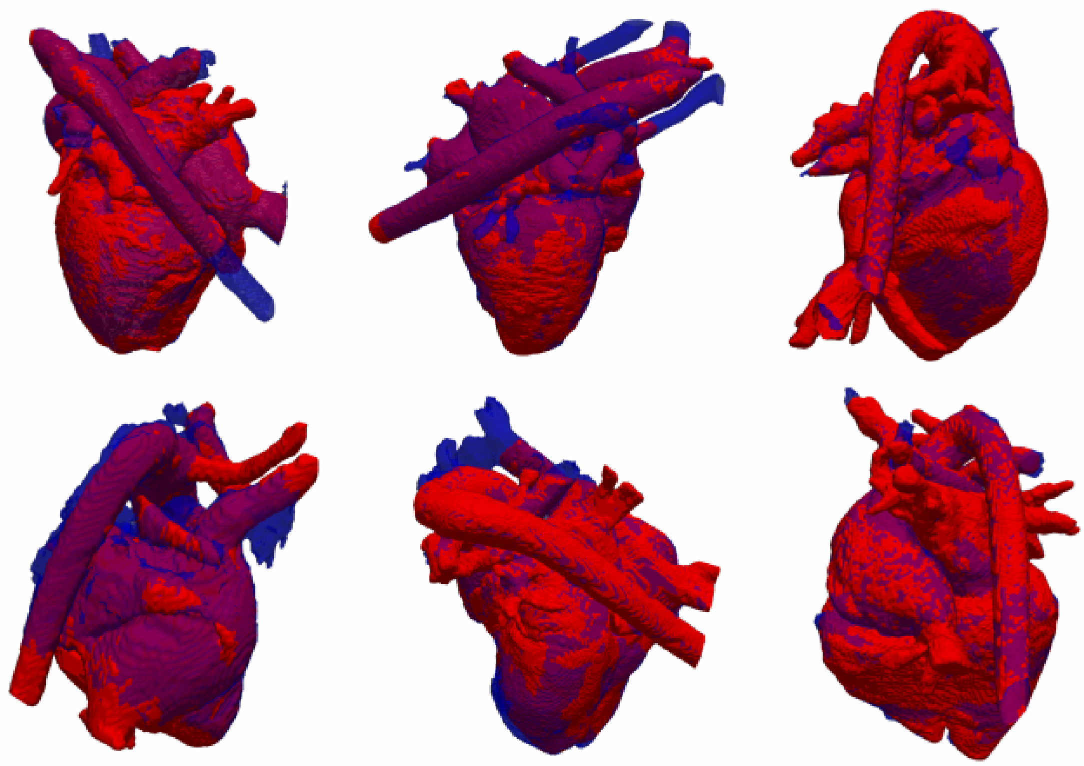


Figure 3.1: Explicit surface-to-surface comparison of segmentation results (blue) with ground truth (red) of different hearts in training dataset[15]

3.1.1 State of the art

For this topic we consider the paper from the winners of HVSMR 2016 challenge as state of the art [15]. This paper describes their approach to segment 3D cardiac images by using a deeply supervised fully convolutional architecture. The main advantage of this architecture is the input, because the processed image can be arbitrary-sized. This network can put volumetric labels directly on a 3D input.

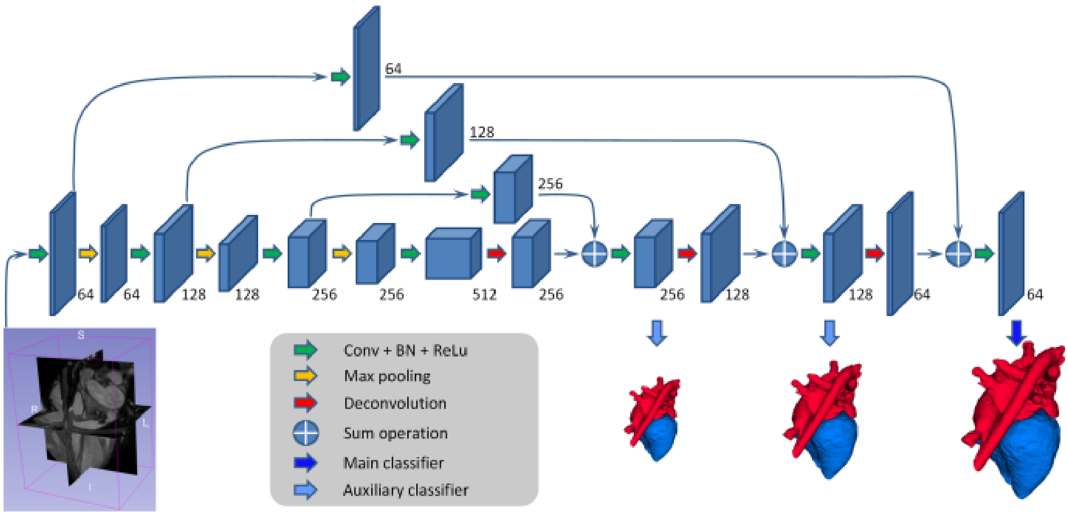


Figure 3.2: 3D FractalNet Proposed Architecture [15]

They have started with a base case, and expanded it recursively until they have finished the 3DFractalNet architecture. It is based on convolutional, max-pooling and deconvolutional 3D layers. Also they have introduced in the learning process, auxiliary classifiers, which have different output dimensions, therefore the deconvolutional layers were added. As data-set they have used the 3D cropped axial images and cropped short-axial images of HVSMR 2016 Challenge data-set, and increased it with data augmentation from 20 cardiovascular magnetic resonance images to 80. The augmentation process was performed by using rotations and axial flips of the images [15]. Their final segmentation results coincide well with ground truth (Figure 3.1), the average dice coefficient being 0.930. In Figure 3.2 we can see the architecture they have used for their neural network.

3.1.2 Other Related Work

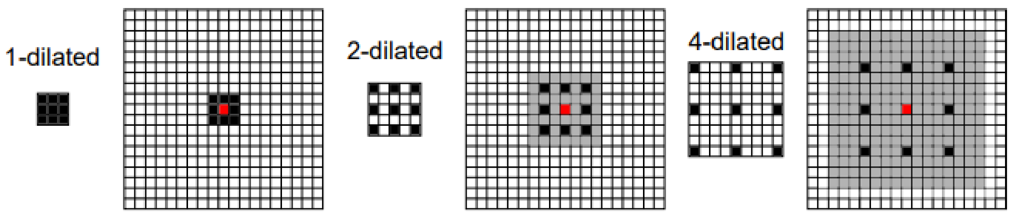


Figure 3.3: Dilatation process

"Dilated Convolutional Neural Networks for Cardiovascular MR Segmentation in Congenital Heart Disease". This paper presents an automatic method of using dilated convolutional neural networks for segmentation of the myocardium and blood pool. For patients that have severe heart disease or those who are required to have a surgery from childhood. The use of 3D models has been very helpful, mainly for the preoperative planning of the surgery. They trained a CNN and assigned a class label for every voxel in the image. The CNN used dilated convolutions that allow large receptive field with a few trainable parameters. In Figure 3.3 we can see an example of the dilatation process.

An experiment with a five-fold cross-validation has been performed, in which each fold contained two CMR[1] scans. To segment the set for test, a single CNN has been trained using all the training images. As data-set for training they used 3D images, total of 10 MRI's, cropped around the heart and thoracic aorta. The network parameters have been optimized with Adam using as a cost function the categorical cross-entropy. To compare the performance of a CNN with dilated convolutions and a CNN without dilated convolutions, the segmentation was performed using and identical CNN architecture containing 72,643 trainable parameters[12]. In order to measure their results, they used the Dice coefficient, the distance to boundaries (ADB) and the Hausdorff distance. We can observe in Figure 3.4 that the average Dice index for training and testing process is 0.80 for the segmentation of the myocardium and 0.92 for the segmentation of the blood pool.

		Myocardium			Blood pool		
		Dice	ADB	Hausdorff	Dice	ADB	Hausdorff
Training	Average	0.80 ± 0.06	1.01 ± 0.43	6.70 ± 3.52	0.92 ± 0.03	0.81 ± 0.28	5.86 ± 3.36
Test	Patient 10	0.72	1.34	10.75	0.94	0.74	5.23
	Patient 11	0.81	0.68	2.50	0.93	0.94	9.17
	Patient 12	0.87	0.60	3.94	0.93	0.83	9.74
	Patient 13	0.88	1.03	10.19	0.94	0.94	10.62
	Patient 14	0.71	1.33	8.69	0.90	1.07	4.21
	Patient 15	0.76	1.07	3.97	0.89	1.44	11.78
	Patient 16	0.76	0.80	3.14	0.91	0.77	6.12
	Patient 17	0.87	0.70	4.14	0.95	0.61	4.27
	Patient 18	0.85	0.61	2.19	0.94	0.64	3.29
	Patient 19	0.79	1.41	11.76	0.93	0.87	6.28
	Average	0.80 ± 0.06	0.96 ± 0.32	6.13 ± 3.76	0.93 ± 0.02	0.89 ± 0.24	7.07 ± 3.01

Figure 3.4: Dilatation process

[1]Cardiovascular magnetic resonance

Chapter 4

Application

This chapter will cover the main stages of the developing process for our application, more precisely, an web application that will offer the possibility to visualize directly online the 3D MRIs, and also an application for training and computing the segmentation model. Although these methodologies are often used in creating standalone software, we believe that by applying these steps, it will lead to an easier further development of the finished product.

- ***Analysis:*** In this development phase we will analyze the requirements, independently of the implementation or design. Here we will define the problem we want to solve and create diagrams with the different use cases of the application. Also here we will want to analyze how different systems communicate with each other and the order in which they do so.
- ***Design:*** In the design phase we will establish the system architecture based on the requirements of the analyze stage. Here we will configure the system components as well as their behavior. We will also analyse the implementation plan of the requirements, where we will establish details such as used technologies, development environment, programming languages, data structures, etc.
- ***Implementation:*** In this stage we will build the system based on the requirements of the previous stages. Here we will manage issues related to product quality, libraries used as well as performance of the application.
- ***Testing:*** Because quality is an important factor for a software product, we want to be able to ensure the good operation of the system for a long time. Thus the implementation

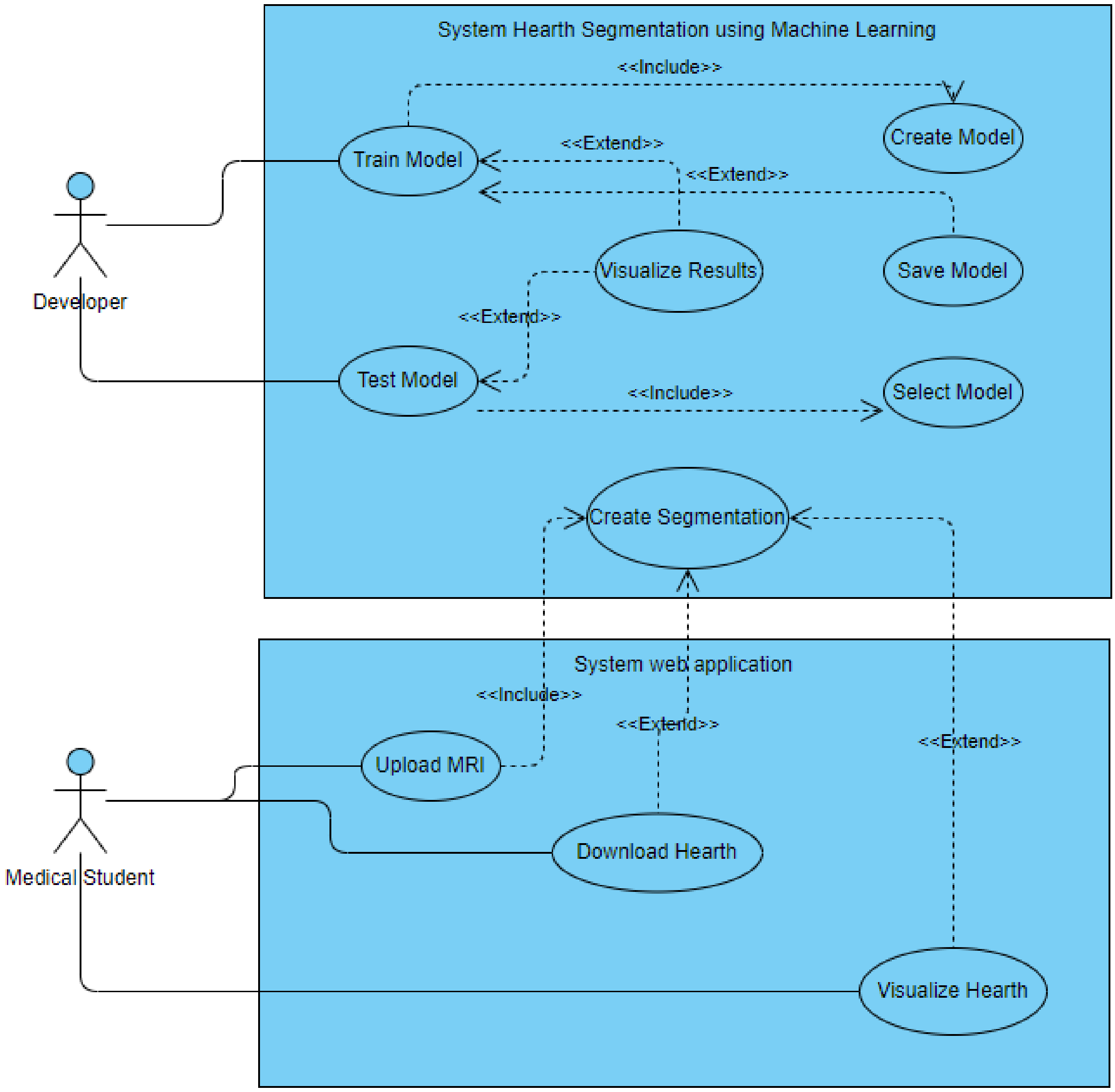


Figure 4.1: Use cases diagram

and application of tests on the desired product is necessary, in this way we are avoiding a higher cost for subsequent changes.

4.1 Requirements analysis

The first step in the requirements analysis is to draw up the use case diagram. As we can see, Figure 4.1 captures the main cases in which actors interact with the application.

Model training is one of the most important use case. We want the application to be able to create and train a model that can be later used for heart segmentation. Furthermore, as it is presented in Figure 4.1, we can see that after the model is trained we want to be able to see the statistics and the results of the training process.

Another important use case is the possibility to ***test the model*** created by the developer. Thus, we can see how well the model handles other data than those on which he was trained. We also want to be able to view statistics and the results of the testing process.

In the system of the web application, we can observe three use cases. ***Uploading the MRI***, in which the medical student uploads an chest 3D scan, fallowed immediately by the segmentation process, in which the Machine Learning System extracts the heart from the image. After the heart is segmented, the medical student can ***download the heart*** or ***visualize the heart*** on the web platform.

4.2 Application Design

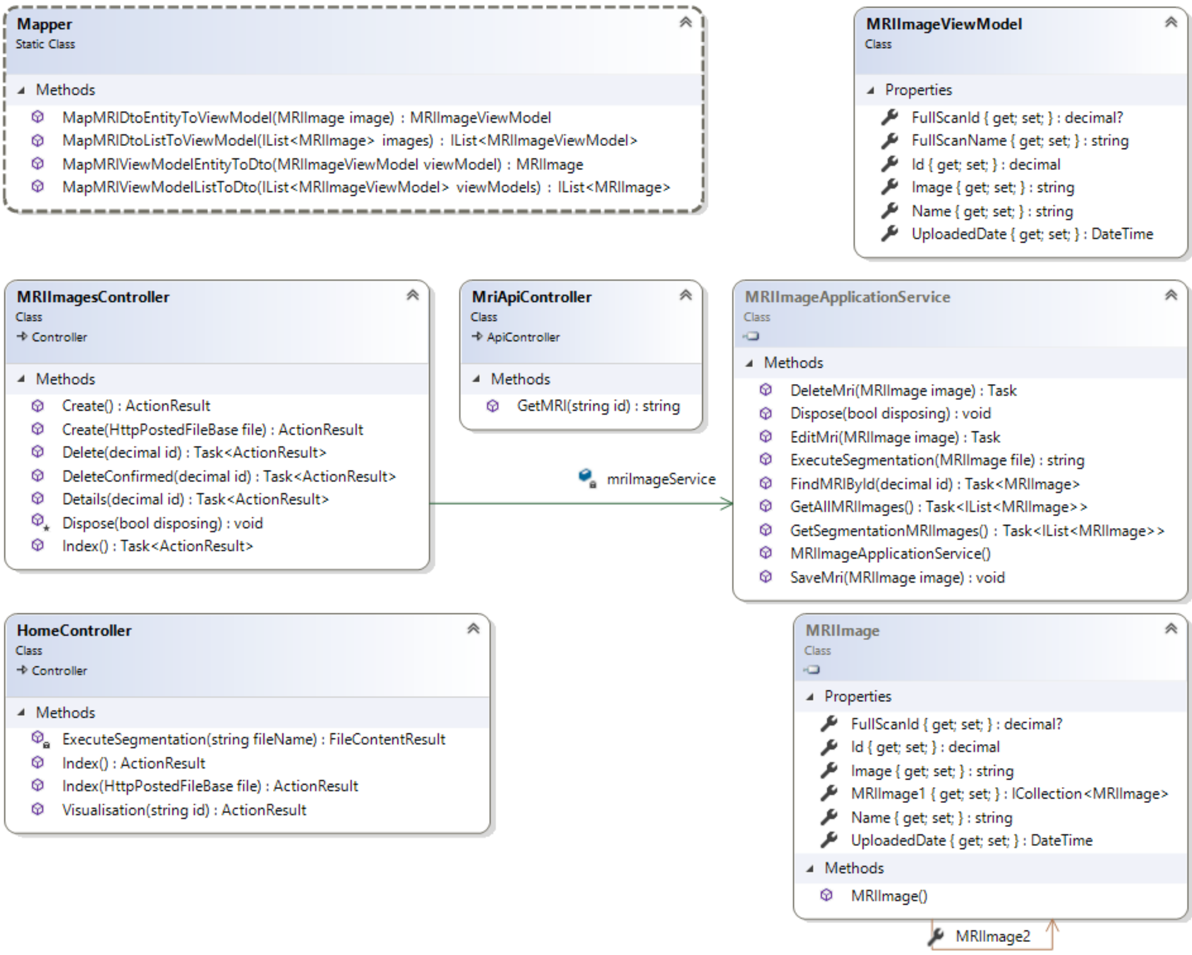


Figure 4.2: Class Diagram

Following the analysis performed in the previous stage, we can design the architecture of the system which we want to implement as well as the choice of technologies and programming languages used. Furthermore, we will need to implement a system that makes possible the communication between segmentation model and web application. In Figure 4.3 we tried to capture the main components of the application and how they will be structured. The first component represents the device from which the student is interacting with the application. The MRIs will be managed from a web page that will communicate with the server which will segment and process the data. Communication will be done via HTTP (Hypertext Transfer Protocol) thus allowing more clients to communicate with our server [3].

The second node is the server to be implemented. It will contain a Web API that will make it possible to receive requests from the client and send the answer back, and an application that will take care of the segmentation process. The web server and the application will both use the database in order to manage the MRIs. When an MRI is uploaded by the client, the web browser will call the web API which will trigger the segmenter. After the segmentation process is completed, it will be saved into the database, thus being available for download or online visualization.

For the web application we will approach the class diagram from Figure 4.2 as fallows: implement an MVC controller for the CRUD operations of the MRIs, a web API for serving the images to the visualization tool integrated in the page and a service which will facilitate the communication between the application server and the database.

4.3 Technologies and tools

The programming languages we choose for the software development are C# for the web application and Python for the neural network.

ASP.NET

In order to create the service we will use ASP.NET, which is a Microsoft technology used to create services and web applications. The service architecture will be a RESTful type (Representational State Transfer) because it gives us the flexibility in terms of service development. With its help and with the ASP.NET Web API Framework we build the service that will deal with the transmission of segments between the client and server. For the implementa-

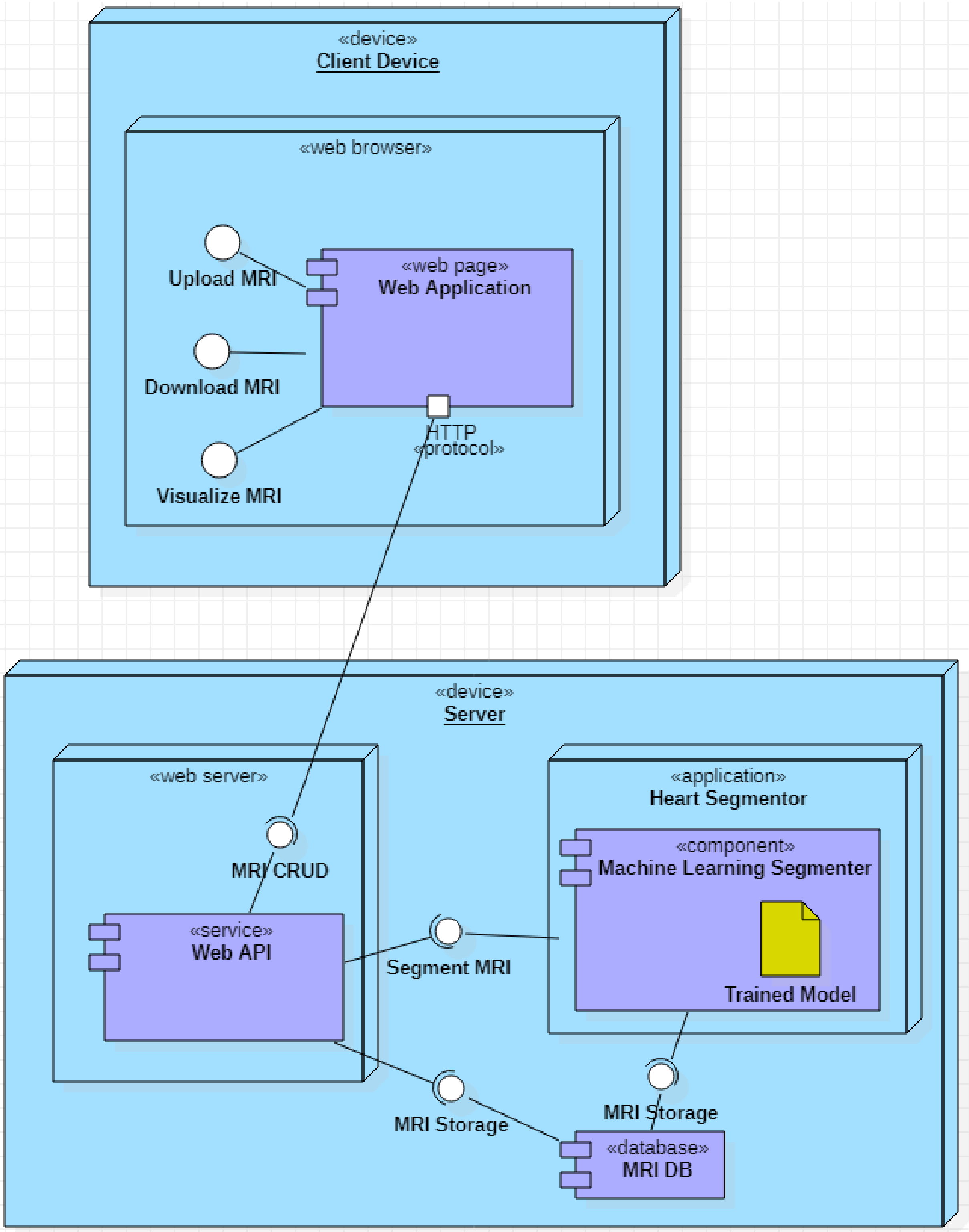


Figure 4.3: Component Diagram

tion and development of ASP.NET type applications we will need a specific environments and tools.

Visual Studio

This integrated development environment (IDE) offers a whole package of tools that allows us to develop an ASP.NET application. At the same time, Visual Studio offers access to essential technologies that facilitate the process of developing a web application, thus benefiting from the features of the .NET framework.

Visual Paradigm

Visual Paradigm is a CASE (Computer aided-software engineering) UML (Unified Modeling Language) tool offering an environment where we can model and analyze the application that already exists or must be implemented. I have used this tool to generate diagrams and ensure a complete and correct model of the application.

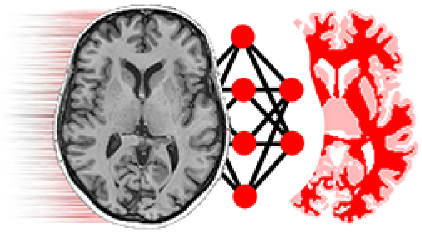
Postman

How testing is an essential step in the process of development, we will need a method to test the API. Furthermore, because testing within the user interface requires a high level of complexity, we need to be able to test the service in a disconnected way, in isolation. Postman is built from a set of tools that allows web service developers to test applications. It is used to determine if the services designed by us return the correct answer, in the desired format. In this way we can ensure the correct behaviour of the application.

Anaconda

In order to configure our neural network we will need several python libraries and their dependencies. In this way, we decided that Anaconda might be a good choice for our project. It is a free and open-source Python distribution, used for package management and simplify deployment. In this way, it will automatically download for us all the required packages and libraries, and also, it will create separate environments that we can use in the development process, each environment having its own versions and packages.

NiftyNet



For developing our neural network, we will need a tool that offers us various ways to interact with medical images. NiftyNet is a deep learning open-source library that specialize in medical imaging, it is build on TensorFlow and consists of APIs for convolutional neural networks (CNNs) that can be later configured and adapted for our cause. We can mimic models from literature easily, and quickly build our new solution, without having to go trough the troublesome process of building everything from scratch.

Git

Git is a version control system which is dealing with tracking changes within an application. To have complete control over the application versions we will use Git together with SourceTree. The latter will provide us with an easy-to-understand interface that will facilitate the maintenance process.

Mango

Mango (Multi-image Analysis GUI) is a tool we used for viewing the medical images. The main features of this application are the analysis tools and a friendly user interface that makes possible the navigation trough image volumes. Furthermore, it supports Mac, Windows, and Linux operating systems, thus being perfect for our scenario.

4.4 Implementation

4.4.1 Intelligent Component

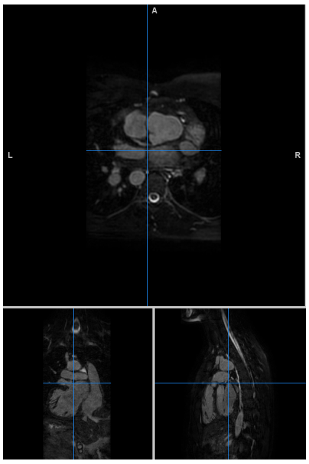


Figure 4.4: Chest CT

The first step in the development process of the intelligent component will be to create a neural network, which will be based on a NiftyNet DenseVnet, that can be trained on already labeled images.

For the dataset on which to train our neural network, we are using the HVSMR 2016 axial full-volume training data-set. The total amount of data being 20 MRI scans, from which 10 will be used in the training and 10 in testing. In order to maximise the results we will add an augmentation step. The augmentation pre-processing is performed in three different ways: adding random angles for each axis, flipping the image axes and randomly scaling the volumes. In the training step we will use 80% of the dataset for training and 20% for validation. An example of an input used in the training process can be seen in Figure 4.4, and for each input, we will use an already labeled heart, as in Figure 4.5.

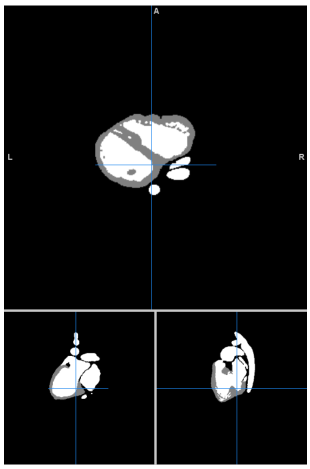


Figure 4.5: Label image

The architecture of the network is composed of two parts. The first half of the convolutions represents the compression path and the right part is the decompression path until the voxel reaches it's original size. Due to the fact that in anatomy our interest point is relatively small, our neural network will be using the Dice coefficient as the loss function ($\frac{2|A \cap B|}{|A|+|B|}$). The voxels will be passing trough a softmax layer, a probabilistic algorithm, which will define the voxel as heart, blood vessel or background. For the efficiency of the algorithm we used the GPU as the main processing unit and installed the CUDA framework for the parallel computations.

The neural network will be trained on a batch size of 2 samples, using a re-sizable window sampling and a learning rate of 0.001. Trough experimentation, we set the queue length to 10, which

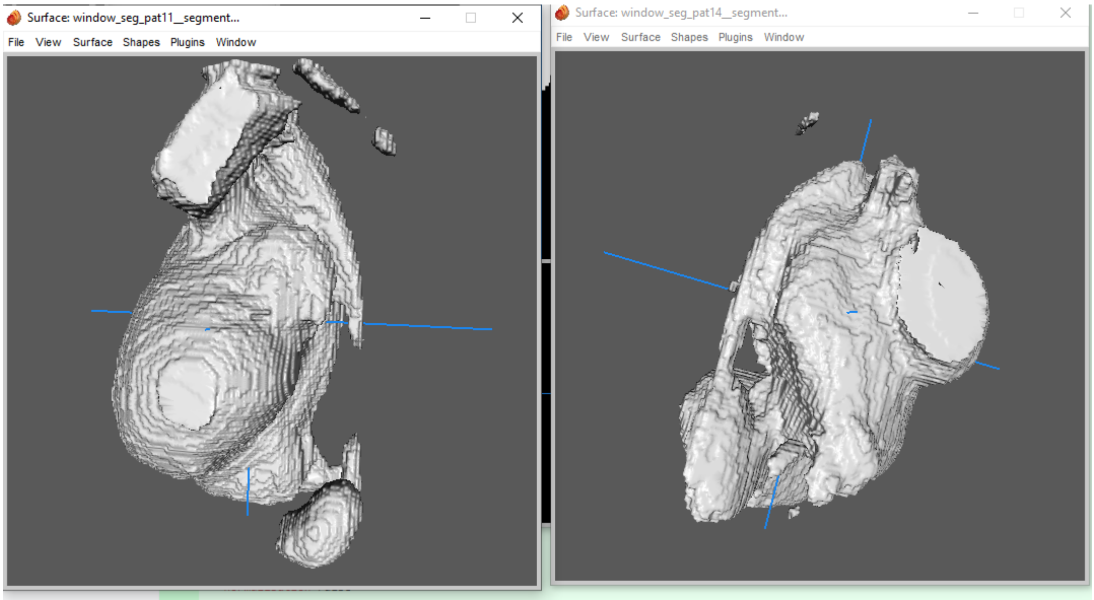


Figure 4.6: 3D heart segmentation after 100 iterations

was the biggest number which didn't cause a memory overflow. The augmentation process will rotate the image at 90, 180 and 270 degree angle. Also, we will scale each image at 80% and 120% of the original size. For the first training session we set the interpolation level to linear. Every tenth iteration we will save the model with the best results. We will create a configuration file for each of the steps of development of the neural network: training, inference and evaluation.

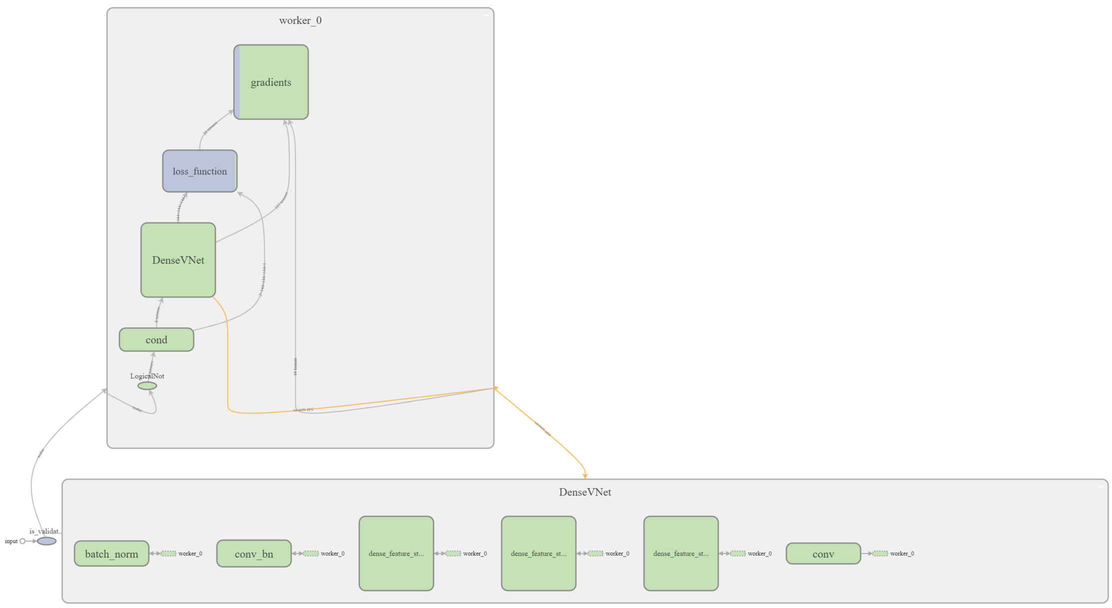


Figure 4.7: Neural Network Implementation Design

After 100 training iterations (Figure 4.6), we managed to generate MRIs in which the heart is recognizable. On the other hand, the image is still not refined enough in order to distinguish each heart chamber or the surface texture. In addition, it is currently impossible to determine where the heart ends, looking like it is still connected to other body parts.The average value of the dice index during this process being 0.3.

For the next step we will let the neural network train for 300 iterations, and after analyse the results. By using the tensorboard we managed to compute the neural networks implementation design, Figure 4.7, which summerise the computational path and the CPU's and GPU's load.

After we finalised the training process and the loss coefficient is stabilising around a certain value (in our case 0.40) , we will save the best model we achieved and use it later during the usage of the web application.

4.4.2 Web Component

For the web component we will create a MVC Web Application in .NET Framework. By using this application, the student will be able to upload, download and visualize online the 3D images. Each MRI will be saved in a SQL database, thus the application can be used as a management tool for the full images and their segmentations. We will have only one table, for storing the images (Figure 4.8). The table references itself for the purpose of having a connection to the full scan after the segmentation was done successfully.

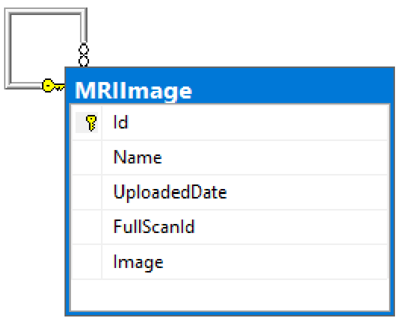


Figure 4.8: Images Table

For mapping the database to our code we will use the Entity Framework ORM[1], database first, in order to create our classes. Each class will be configured and implemented according to our class diagram (Figure 4.2).

In order to be able to visualize the 3D image in the browser, we integrated in our application a pure JavaScript library named *Papaya*[2]. This library was developed by the *Uni-*

[1]Object-relational mapping
[2]https://github.com/rii-mango/Papaya

```
<script type="text/javascript">
    var params = [];
    params["encodedImages"] = [];
    $.ajax({
        type: "GET",
        url: "@Url.Action("GetMRI", "MriApi", new { id = @Model.Id, httproute = "API Default" })"
    }).done(function (data) {
        window.MRI = data;
        var params = { encodedImages: ['MRI'] };
        papaya.Container.resetViewer(0, params);
    });
</script>
```

Figure 4.9: Retrieving the Base64 Image and preparing the content for rendering

versity of Texas Health Science Center, and it is supporting DICOM and NIFTI formats. After we integrated the script into our page, we created an API that can retrieve the image from the database, and returns it encoded in base64. After we retrieve the image in base64, we can pass the content to the papaya container, in order for it to render the 3D image in the browser. The necessary code for it can be observed in Figure 4.9.

4.5 Testing

Being one of the most important stages of programming, we will implement tests for all components of this application: API web service, model training and image segmentation.

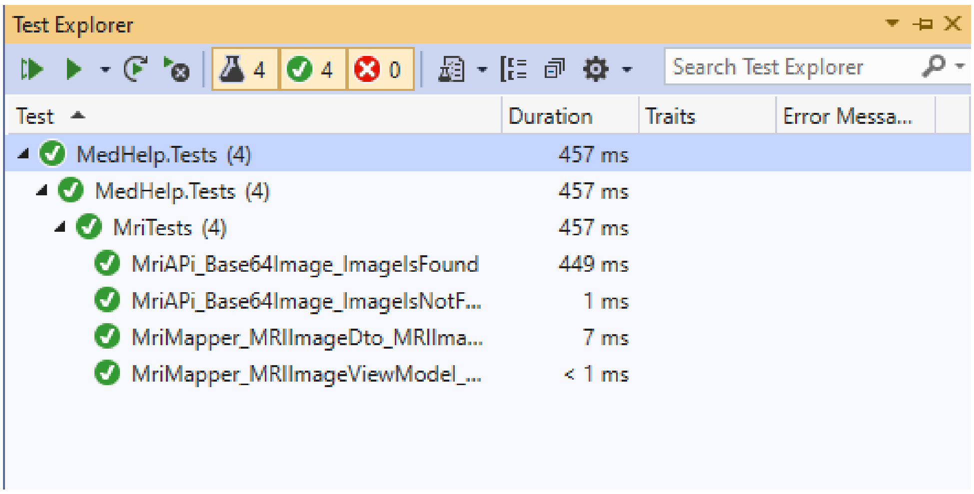


Figure 4.10: Unit tests (MSTests) in the Test Explorer

Testing the Web API

For testing the Web API we will write several unit tests (Figure 4.10) which will cover two scenarios:

- Endpoint - we test if the endpoint returns the image in base64 format
- Mapping - we check if the mapping between the view model and the transfer object is done correctly

Testing the training process

Here we will check if the training validation is done correctly and as we progress trough the iterations the results are getting better and better. We also need to check that the most accurate models are being saved at the checkpoints, and we do not lose them trough the iterations. In Figure 4.11 we can observe the performance of each model troughout the iterations.

Testing the segmentation process

At this point, we are interested in testing the final product. More precisely, if we run the model separately on a full body scan, we are getting the correct segmentation. We check if the segmentation is being centered accordingly in the image and that the generated file has the correct format, is complete and uncorrupted.

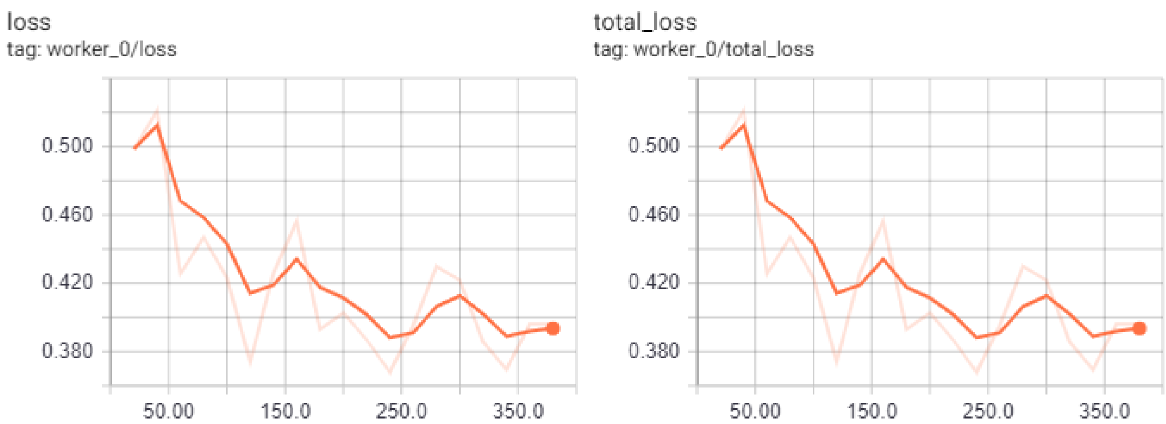


Figure 4.11: Loss Graph for 380 iterations

Chapter 5

Results

We will analyse the results from two perspectives: the segmentation and the web application.

5.0.1 Segmentation Process

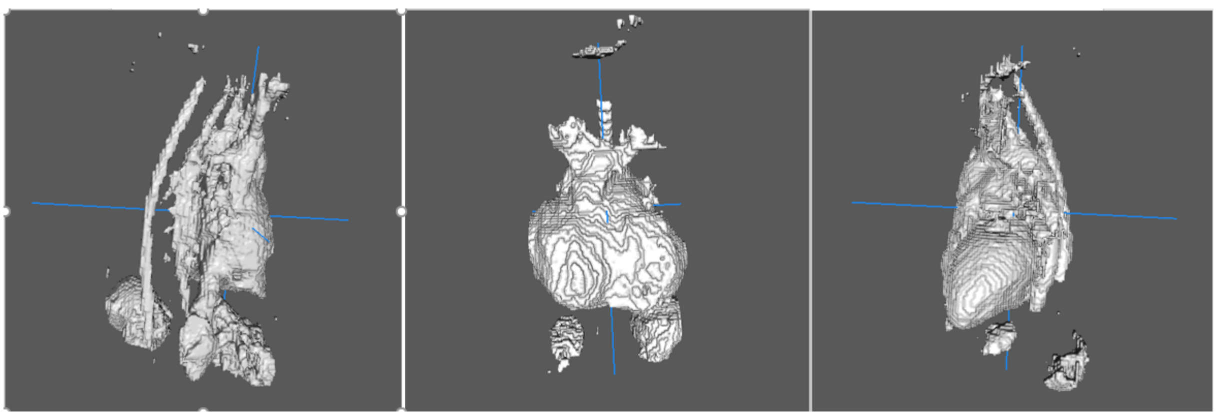


Figure 5.1: 3D heart image after 380 iterations

After we increased the number of iterations from the previous session (from 100 iteration to 380), we can confirm that the heart is starting to be recognizable among the other organs (Figure 5.1). We can see an improvement from the previous results and we also are able to analyse the structure of the heart. For the final model we decided that a higher interpolation order is required during the inference process. Thus, we changed the interpolation level from linear to trilinear, even if this requires a longer time for the training process. On further analysis we observed that the loss function is stabilising around the 0.40 value, meaning that the model will not get as accurate as we desired. This might be caused by the small amount of data and also by the low number of training iterations.

Blood Vessel Class	Dice Loss	Jaccard
part 0	0.328467	0.196506
part 1	0.040482	0.020659
part 2	0.285799	0.166724
part 3	0.185206	0.102053
part 4	0.125614	0.067016
part 5	0.06335	0.032711
part 6	0.2312	0.13071
part 7	0.33787	0.203275
part 8	0.394185	0.245473
part 9	0.459523	0.298299
Avg	0.24517	0.146343

Blood Vessel Class	Dice Loss	Jaccard
part 0	0.452646	0.292529
part 1	0.254755	0.145971
part 2	0.521458	0.352684
part 3	0.300254	0.176646
part 4	0.426547	0.27109
part 5	0.256987	0.147438
part 6	0.418703	0.264785
part 7	0.524872	0.355815
part 8	0.462187	0.300548
part 9	0.658741	0.491136
Avg	0.427715	0.279864

Figure 5.2: Evaluation results before and after the re-centering with trilinear interpolation

Several result images weren't perfectly centered and had different dimensions labeled segments, thus making impossible the process of evaluation. Therefore, in order to evaluate the results we need to apply a two step post-processing: re-sample the 3D volumes to make them have the same dimensions and re-center the heart in the middle of the volume. In Figure 5.2 we can see the Dice and Jaccard coefficient for each volume, before and after this process.

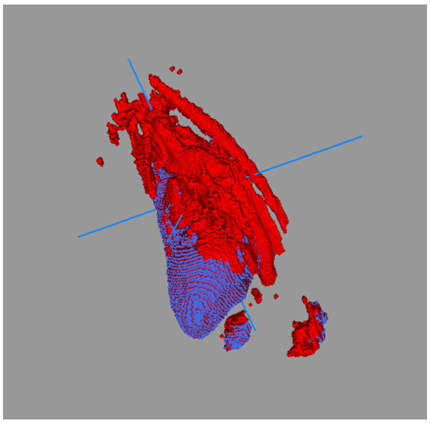


Figure 5.3: Best heart segment

The best segment we managed to obtain can be seen in Figure 5.3. The blue color represents the heart and the red color the blood vessels, in accordance with our 3 classes. We can still observe several imperfections, but on the other hand we think that this MRI is detailed enough for understanding how the heart looks like in a chest volume.

5.0.2 Web Application

The main screen of the web application can be seen in Figure 5.4, here we can observe the main functionalities of the application. After the user uploads an MRI, he can see it in this list, and also he has the possibility to visualize it online or to download it later.

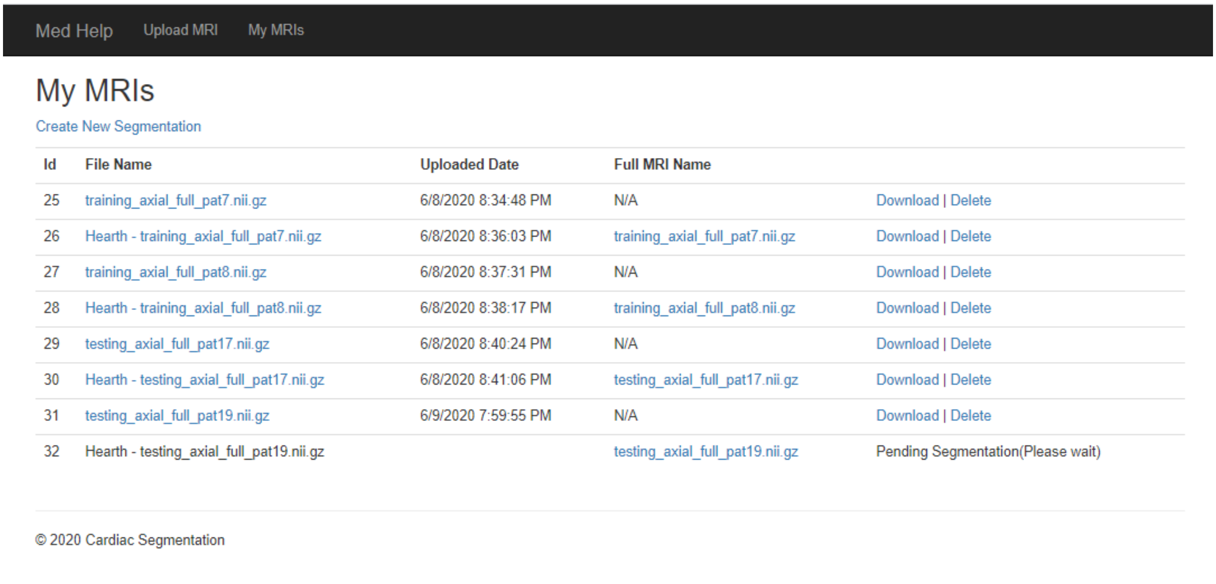


Figure 5.4: MRI's CRUD Interface

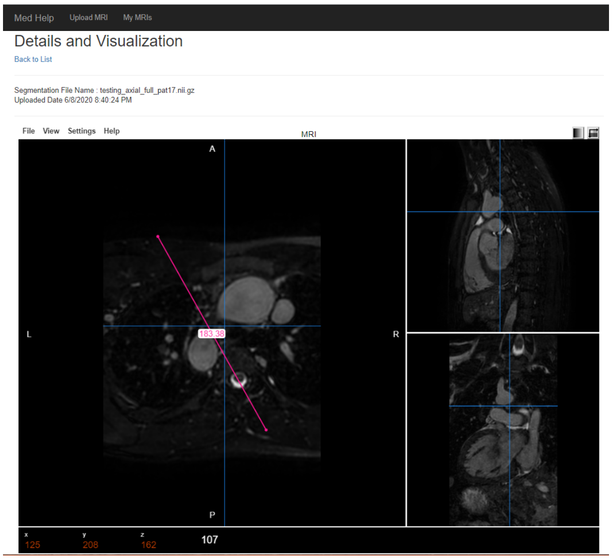


Figure 5.5: Online Chest Visualization

During the development process we had some difficulties regarding the asynchronous processing of the image. The main challenge was the UI freeze during the segmentation process. Due to this fact, we decided to implement a ***"fire and forget"*** pattern. In this way, after the image is successfuly uploaded, the user is returned to the MRI's list and after the segmentation is completed the "Pending Segmentation(Please wait)" message is replaced with "Download Delete", and also, the name of the heart becomes a link to it's visualization (Figure 5.6). In Figure 5.7 we can see the application during the 3D visualization on a mobile device.

5.0.3 Discussion

We can observe from the beginning that the model cannot achieve a high grade of accuracy due to the small amount of data available in the field. This proves to be an even grater challenge when the difference between volumes are considerable and the model is not able to differentiate between the background and the main point of interest.

Another topic of discussion can be the augmentation process which proves to be the main reason in the increased accuracy, nevertheless from our point of view that data is produced in an artificial way, thus making the results of the research debatable.

Furthermore, after analysing the evaluation table we can see that some volumes were much more closer to the original label volumes than the others. This result might be caused by the fact that several labels might look similar, and by the lack of data the more unique ones are much harder to be cropped by the neural network.

Figure 5.6: Heart Visualization

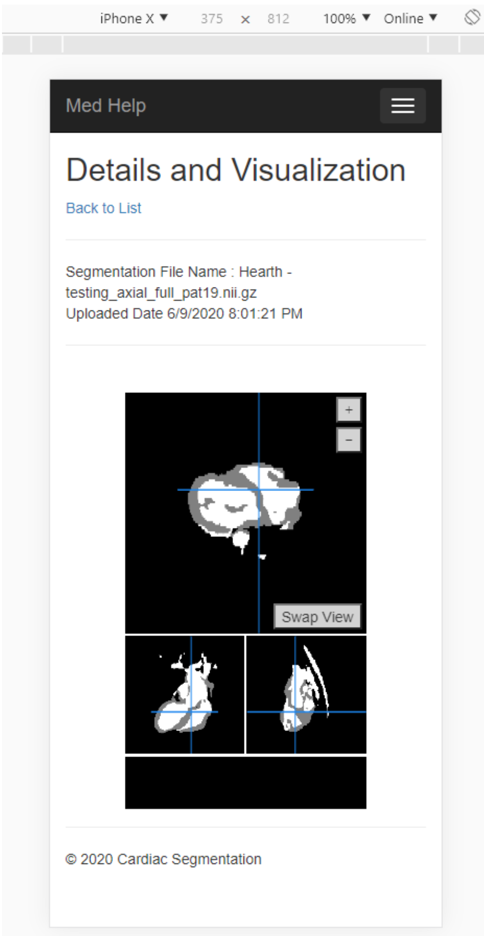


Figure 5.7: Mobile heart visualisation

Chapter 6

Conclusion and Future Work

We presented in this paper a web application, that can help different users to crop and better visualize a 3D image of a heart. The application has a volatile behaviour which let the user to chose the desired visualization tool for the 3D volumes by downloading the image, or to visualize the image directly in the web application. Furthermore, after we evaluated the volume results and analyse the possible causes of the loss stagnation at a relative high value during the training process, we considered that is caused by the lack of data. The challenge created by the small amount of data, was partially solved by the augmentation process, but may prove that the results cannot be as valuable as we desired. We hope that in the future we will have access to more labeled images. Even if those images will not be as high quality resolution as those from HVSMR 2016, it will have a major impact on the credibility of the final neural network model and also on its performance.

For future work, we plan to improve the neural network model and increase the speed of execution by creating a better distribution of the processing load among different units. One possible improvement can be training and testing the neural network on multiple GPU's. Also, we would like to improve the user interface component by making it more responsive to the end user. As the first step we want to deploy the application on a server and have real people testing it. Also, by having a focus group composed of medical experts and even medical student, we can easily detect possible improvements and modifications. As a future perspective we want to evolve this application and make it available for the doctors also.

Bibliography

[1] What is perceptron: A beginners tutorial for perceptron, 2018 (accessed May 3, 2016). `https://www.simplilearn.com/what-is-perceptron-tutorial`.

[2] Aparna Ashok. Ethical principles for humane technology, 2018 (accessed May 2, 2020). `https://blog.prototypr.io/ethical-principles-for-humane-technology-19f4fb3b0ba2`.

[3] Roy Fielding, Jim Gettys, Jeffrey Mogul, Henrik Frystyk, Larry Masinter, Paul Leach, and Tim Berners-Lee. Hypertext transfer protocol–http/1.1. 1999.

[4] Eli Gibson, Francesco Giganti, Yipeng Hu, Ester Bonmati, Steve Bandula, Kurinchi Gurusamy, Brian Davidson, Stephen P Pereira, Matthew J Clarkson, and Dean C Barratt. Automatic multi-organ segmentation on abdominal ct with dense v-networks. *IEEE transactions on medical imaging*, 37(8):1822–1834, 2018.

[5] Ian Goodfellow, Yoshua Bengio, and Aaron Courville. *Deep learning*. MIT press, 2016.

[6] James Manyika, Susan Lund, Michael Chui, Jacques Bughin, Jonathan Woetzel, Parul Batra, Ryan Ko, and Saurabh Sanghvi. Jobs lost, jobs gained: Workforce transitions in a time of automation. *McKinsey Global Institute*, 150, 2017.

[7] Fausto Milletari, Nassir Navab, and Seyed-Ahmad Ahmadi. V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 Fourth International Conference on 3D Vision (3DV)*, pages 565–571. IEEE, 2016.

[8] Nils J Nilsson. Introduction to machine learning: An early draft of a proposed textbook, 1996.

[9] Stuart Russell and Peter Norvig. Artificial intelligence: a modern approach. 2002.

[10] Summit Saha. A comprehensive guide to convolutional neural networks, 2018 (accessed May 1, 2020). `shorturl.at/pwX24`.

[11] Robert J Schalkoff. *Artificial neural networks*. McGraw-Hill Higher Education, 1997.

[12] Jelmer M Wolterink, Tim Leiner, Max A Viergever, and Ivana Išgum. Dilated convolutional neural networks for cardiovascular mr segmentation in congenital heart disease. In *Reconstruction, segmentation, and analysis of medical images*, pages 95–102. Springer, 2016.

[13] Yossi Matias Yaniv Leviathan. Google duplex: An ai system for accomplishing real-world tasks over the phone, 2018 (accessed May 3, 2020). `https://ai.googleblog.com/2018/05/duplex-ai-system-for-natural-conversation.html`.

[14] Bayya Yegnanarayana. *Artificial neural networks*. PHI Learning Pvt. Ltd., 2009.

[15] Lequan Yu, Xin Yang, Jing Qin, and Pheng-Ann Heng. 3d fractalnet: dense volumetric segmentation for cardiovascular mri volumes. In *Reconstruction, segmentation, and analysis of medical images*, pages 103–110. Springer, 2016.