POLITECNICO DI TORINO

Master's Degree in Biomedical Engineering

Master's Degree Thesis

Muscle segmentation in MRI volumes of lower limbs: Centralized or Federated learning?

Supervisors

Prof. Kristen Meiburger

Eng. Francesco Marzola

Candidate

Giorgia Passione

March 2025

Abstract

The purpose of this work is to segment muscles in MRI volumes of lower limbs, comparing the performances of two different techniques: a 3D neural network, trained locally on a manually segmented dataset, and DAFNE (Deep Anatomical Federated Network), an open-source software trained in a decentralized collaborative manner using federated learning.
Federated learning is a novel technique that involves different clients that collaborate without sharing the data. Single models are trained locally by the centers. These models are then aggregated together on the server side and the updated model is shared with each center.
The dataset has been granted by Radbound University Medical Center and contains 166 lower limb volumes from patients with two different pathologies, Fascioscapulohumeral muscular Distrophy (FSHD) and Myotonic Dystrophy Type 1 (MD1), and healthy volunteers acquired using the Dixon technique. It includes 82 volumes of thigh and 84 volumes of leg, both unilateral and bilateral. For MD1 group there can be from one to three different volumes of the same patient, while for the other two groups only one volume per patient is provided. Two kinds of division of the dataset into training, validation and test set have been tested: the first one randomly extracts volumes according to set specific percentages from each type of pathological and healthy group, while the second one randomly divides the pathological volumes into training and validation set and assigns all the healthy volumes to test set. At the end of each division the training set consists of aproximatly 60 volumes and the validation and test set of about 10 volumes each. Thigh and leg volumes have been considered separately, so two 3D neural networks of the SwinUNETR type, one for the leg, that segments 9 different muscles, and one for the thigh, that segments 12 different muscles, have been trained using the MONAI library, with a pipeline designed specifically for the dataset. Five different loss functions have been tested and the results have been compared using DICE and Haussdorff Distance at the 95° percentile (HD95).
Despite the unquestionable potential of DAFNE, the locally trained 3D Neural Networks, both for thigh and leg, allowed to obtain higher values of DICE and lower values of Haussdorff Distance than the segmentations given by DAFNE.
In particular, for the segmentations of the volumes in test set with the 3D neural network for thigh it can be achieved a DICE of 0.7625±0.0458 and a HD95 of 5.980±1.3858 pixel, while the neural network for leg allows to obtain a DICE of 0.7516±0.0270 and a HD95 of 6.0544±0.9937 pixel.
According to the performances obtained in this work, even if with DAFNE there are great advantagies in therms of data protection and the way in which the available

model can be updated and shared, at the moment a locally trained 3D neural network remains the most suitable choice for the provided dataset.

Table of Contents

Acronyms

AI

Artificial Intelligence.

CNN

Convolutional Neural Network.

DAFNE

Deep Anatomical Federated Network.

DL

Deep Learning.

FSHD

Fascioscapulohumeral muscular Distrophy.

HD95

Haussdorff Distance 95 percentile.

HV

Healthy Volunteers.

MD1

Myotonic Dystrophy Type 1.

ML

Machine Learning.

MONAI

Medical Open Network for Artificial Intelligence.

MRI

Magnetic Resonance Imaging.

NRM

Nuclear Magnetic Resonance.

Chapter 1

Introduction

1.1 Magnetic Resonance Imaging - MRI

Magnetic Resonance Imaging (MRI) is a non-invasive imaging technique that is deeply used as a diagnostic tool since 1980s, because it is very useful to obtain images of the internal structure of the body with a great contrast among different soft tissues. An important advantage of MRI with respect to other classic techniques such as radiography or computer tomography is that MRI does not use ionizing radiation [1] and this makes this technique particularly indicated for followup of long-term monitoring and for patients with delicate clinical conditions.

1.1.1 Physical Principles

MRI is based on the physical phenomenon of Nuclear Magnetic Resonance (NRM), which arises when there is an interaction between nuclei with non-zero magnetic moment, previously placed in a static magnetic field, and an oscillating external magnetic field that perturbs them [1]. An example of this kind of atoms are hydrogen, that is largely present in water, that is largely present in human body soft tissues: for this reason MRI is very effective in taking images of soft tissues.
The magnetic moment of the nucleus in a static magnetic field B_0 moves as a precession about B_0 at the angular frequency ω_0, that is the Larmor frequency and is proportional to the strength of B_0. Another aspect is that the energy of the interaction between the magnetic nuclear moment and B_0 depends on the direction of magnetic nuclear moments. So when the magnetic nuclear moments are aligned to B_0 there is minimum energy, while when they are perpendicular the energy is maximum.
When a macroscopic sample that contains non-zero magnetic moment nuclei is placed in a static field B_0, each magnetic nuclear moment tends to align to B_0 and

this causes the magnetization of the sample. In case of thermal equilibrium, the magnetization is given by the *Equation 1.1*

$$M = \chi B_0 \tag{1.1}$$

where χ is the nuclear susceptibility.
If this macroscopic sample is, then, exposed to an oscillating magnetic field B_1, perpendicular to B_0 and of frequency ω equals to the Larmor frequency ω_0, produced by an alternating current coil, the NRM phenomenon can be observed: when ω is equal to ω_0 resonance is achieved and the magnetization of the sample can be rotated and placed in the transverse plane, perpendicular to B_0, even by a weak oscillating field pulse. After the end of the excitation pulse, the magnetization of the sample decays because of the presence of B_0 and this induces voltage in the coil (Faraday's Law). The induced voltage can be Fourier transformed in order to obtain the NRM specrum, that contains different contributes from nuclei in different chemical environment, because they have slightly different frequency of precession. So, NRM allows to identify chemically different populations of nuclei and their relative amounts in the sample.
Thanks to F. Bloch, the dynamics of nuclear magnetization can be described with:

- the longitudinal magnetization (the component parallel to B_0), M_{lon},has an exponential relaxation caused by the thermal fluctuations (*Equation 1.2*)

  $$M_{\text{lon}} - \chi B_0 \propto (-t/T_1) \tag{1.2}$$

  where T_1 is a characteristic relaxation time;

- the transverse magnetization (the component perpendicular to B_0), M_{tr}, decays exponentially because of the magnetic interactions between neighboring nuclei (*Equation 1.3*)

  $$M_{\text{tr}} \propto exp(-t/T_2) \tag{1.3}$$

  where T_2 is another characteristic time constant.

According to Lauterbur, the NRM signal acquired as previously described, can be used to obtain MR image, because in presence of magnetic fields gradients the frequency of precession, ω_0 is spacially dependant. Since the NRM signal is composed by different components that come from nuclei at different locations in the sample, and that are characterized by the own unique frequency phase, the image can be obtained following the concepts of spatial encoding and image reconstruction in MRI.
The spatially encoded signals in MRI are obtained by the repetitively excitation of the nuclear magnetization and the following acquisition of NRM signal during a short interval for a set number of time, in presence of an external magnetic field.

For in vivo MRI the best target are Hydrogen nuclei, because they produce the greatest NRM signal among all nuclei present in tissues and they achieve good contrast between different tissues. In particular, a source of contrast in MRI is the dependence of T_1 and T_2 and proton density on the biochemical composition of different tissues. In fact, T_1 and T_2 changes in some cases of malignant tissues, with respect to the correspondent healthy tissues.
MR images can be weighted with $T1$, $T2$ or proton density, according to the purpose: according to the weighting method, there will be different aspects of the tissues that will be highlighted in the image.

1.1.2 Muscle MRI and clinical context

Muscle MRI can be a useful tool in the diagnosis of some pathologies that affect muscles, because it provides information about muscle structure and function of muscles in the body [2]. Moreover, MRI can be also very useful in the followup of the progression of the pathology in patients and in the evaluation of the therapeutic response [3] [4], because it does not use ionizing radiations.
In this work the interest is in the pathological effects on lower limb muscles due to muscles dystrophies. Muscle dystrophies are neuromuscular pathologies whose causes are mostly genetic alterations.
Neuromuscular pathologies may affect in different ways muscles: as can be seen in *Figure 1.1* variation of muscular patters, fat infiltration, asymmetry between left and right size, edema, atrophy and other effects may occur in patient's muscles.
These aspects may be successfully detected by MRI. Furthermore, MRI can help in the selection of a muscles for biopsy in patients without clear muscle weakness or, on the other hand, in patients with a severe degree of muscle atrophy [2]. In some cases, MRI can help in early phase therapeutic trials of patients to demonstrate target engagement in therapies [5].
For these reasons, the first step is to recognize the muscle of interest in MRI volume. The gold standard is the manual segmentation, but it is very time consuming and operator dependent [6], so an alternative automatic method for segmentation is needed.
In this work two muscular pathologies have been considered:

- Fascioscapulohumeral muscular Distrophy (FSHD), one of the most common muscle pathology, that involves muscles of both upper and lower limbs. Usually muscle involvement is asymmetrical between left and right side of the body, especially in lower limbs. Muscles becomes increasingly atrophic with the progression of the disease: in lower limbs there is a strongly involvement of Tibialis Anterior, but also vastii, muscles in the posterior part of thigh and gastrocnemius show visible alterations [2];

- Myotonic Dystrophy Type 1 (MD1) that has different variants and is characterized by a progressive muscle wasting, weakness, myotonia [6] and cognitive dysfunction [7]. Edema-associated process and fat infiltration occur in lower limbs of patient with this disease, in an amount linked to both the progression and the variant of the pathology [7].

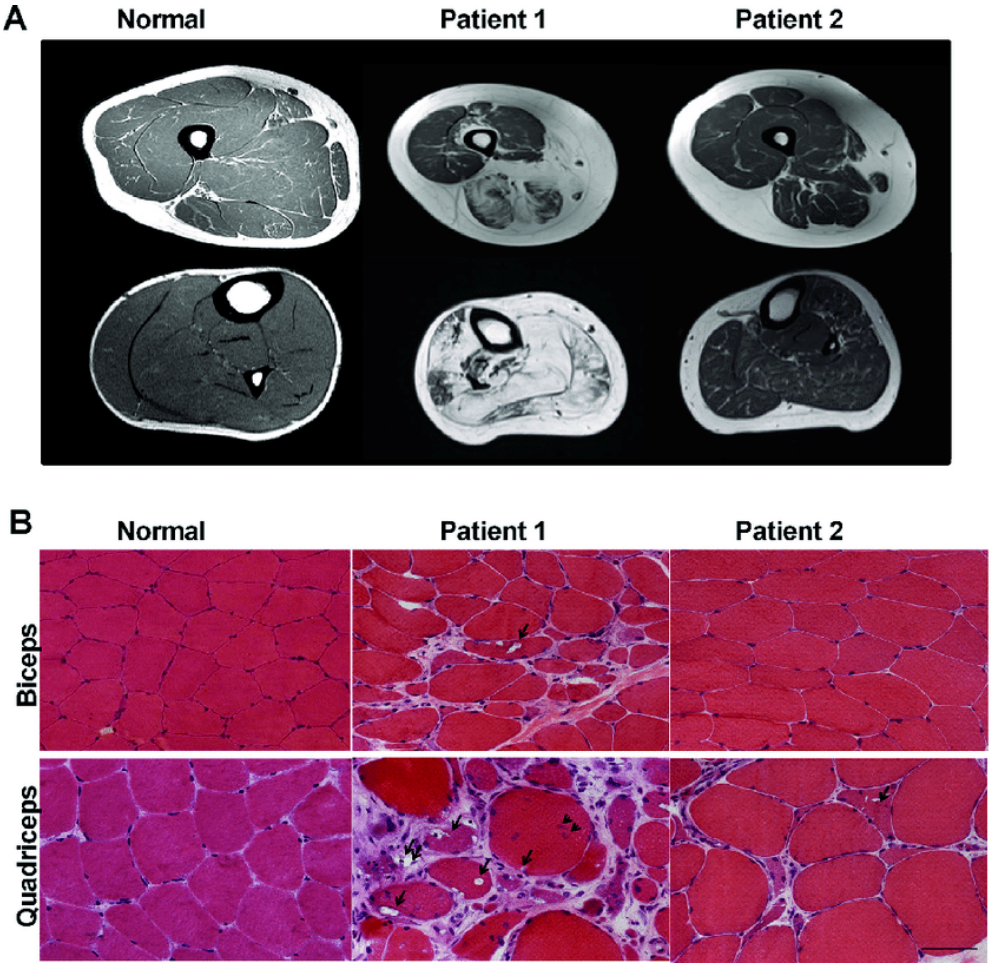

Figure 1.1: Example of pathological effect on patient's muscles. In particular, fat infiltration (white spots) and variation of muscular pattern with respect to the healthy case can be seen [8].

1.2 Deep Learning

1.2.1 Introduction to Deep Learning

Deep Learning (DL) is a cathegory of models and methods whose aim is to solve tasks that require high-level pattern recognition, such as image classification [9]. As shown in *Figure 1.2*, Artificial Neural Networks are a subclass of DL, that is included in Machine Learning (ML), that is part of Artificial Intelligence (AI) field. AI models have a great potential in several clinical and biomedical application, such as risk modeling and stratification, personalized screening, diagnosis, prediction of response to therapy and prognosis [10]. So they are increasingly used in this fields of research.
Data Science includes all of these methods, next to other techniques.

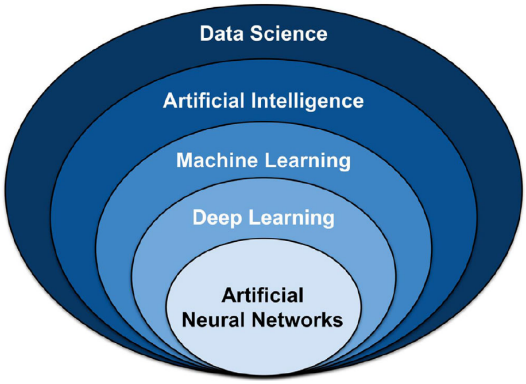

Figure 1.2: Data Science techniques. Relationship between Artificial Neural Networks, DL, ML, AI and Data Science [9].

ML is the part of AI dedicated to learning by developing algorithms. What differs ML with classical programming, is that in the case of classical programming a dataset and an algorithm are the input of a computer, that, thanks to the algorithm, creates the output, while, in ML the input of the computer is the dataset and the associated outputs. So in ML the computers learns and generate an algorithm as output, that describes the relations between the two input and can be used for inference on other datasets [9].
In ML there are four mainly used learning methods:

- supervised learning, where the model infers an algorithm that makes prediction and is informed by the target if it is correct. Dataset is divided into training, validation and test sets and target features are mapped in the training set. In each iteration of the algorithm the model is trained on training set and is tuned on the validation set. The test set is a group of data unknown to the model

that are used to evaluate the performance of the model itself. Regression and classification are the most common supervised learning tasks.

- unsupervised learning, that detects patterns in a dataset and assigns categories according to individual instances in the dataset. The eventual patterns in the dataset are not provided by a target and are determined by the algorithm. Clustering, association and anomaly detection are the most common unsupervised task.

- semisupervised learning, that is a the union between the two ML techniques previously described. The dataset in this case is partially labeled, so, firstly a model is trained with the data and label jet preset, then this model is used to segment the totality of the unlabeled dataset. The resultant totally labeled dataset is use to train another model, the final one.

- reinforcement learning, where there is no single correct answer, but an overall outcome. It is similar to the human learning process: it learns from trials and errors.

DL shares with ML general concepts, such as the ones just explained. It is a specific subclass od ML that includes techniques such as Artificial Neural Networks, that allow the directly processing of the raw data [10].
Artificial Neural Network are algorithms inspires by biological neural network: as neuron, axons and dendrites compose biological neural networks, Artificial Neural Networks contains nodes linked to each other with connections. The connections are weighted according to the ability to provide a desired outcome [9].
An Artificial Neural Network is composed by a series of perceptrons connected to each other: it is a multilayer perceptron algorithm. A perceptron (*Figure 1.3, A*) is a basic machine layer algorithm that, using a sigmoid function, tries to find a line, plane or hyperplane that can separates the classes of the input features and their target.
Artificial Neural network commonly contains a layer of nods for the input, one for the output and a number of hidden layers between these two, usually tens or hundreds. In feedforward neural networks information from the previous layer is transformed and passed to the following layer, while in recurrent neural networks information can also pass between nodes in the same layer and in the previous ones. The information between nodes is transformed by the activation functions that each node contains.
DL can model very complex relationship within large dataset: for this reason it has been largelt used in medical imaging. In particular, the Convolutional Neural Network (CNN) is the most used neural network architecture for medical image processing tasks [10], because it preserves the spacial relationship between pixels in an image: CNN contains convolutional layers (of different kinds according to the

purpose) where the input image is convoluted with a specific kenel function. The specific kernel is called convolutional filter: a small matrix that passes over the original image and performs element-wise matrix moltiplication at each position (*Figure 1.4*). According to the presence of the feature of interest detected by the convolutional filter, a new matrix of output is created. This matrix is the input of the next layer, that will produce a new matrix and so on. The final feature maps are obtained by the combination of each patch and, on these ones, the classification of the image based on the extracted feature is performed.

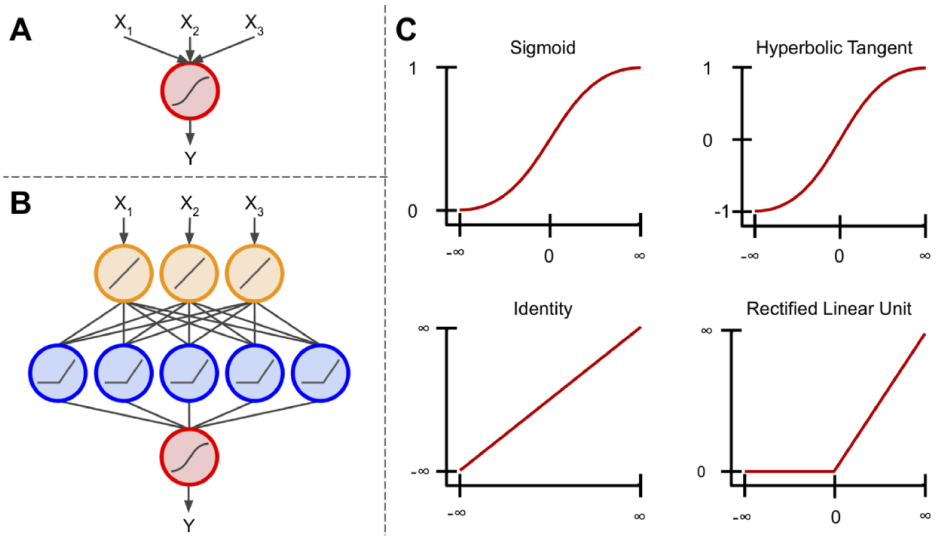

Figure 1.3: Neural Network components. A. Perceptron structure. B. Neural Network structure. C. Examples of four different activation functions [9].

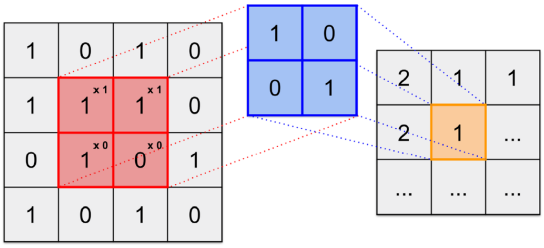

Figure 1.4: Example of a convolutional filter. The feature map (on the right) is obtaining by the combination of each pixel path of the input image (on the left) and the convolutional filter (in the centre of the figure) [9].

DL is very indicated for segmentation task, as well as classification: object can be identified in the image in input.

UNet Neural Network is the most used CNN architectures for medical image segmentation. It is composed by a symmetrical encoder and decoder paths, connected using skip connections. Originally it allowed to segment only 2D images, but it has been modified to manage also 3D volumes [10].
The network parameters are optimized to solve specific task during learning, thanks to several kinds of optimizers: an optimizer is a backpropagation algorithm of error that adjusts the parameters of the neural network, in order to minimize a loss function. The loss function represent the cost function of the network and can be chosen among different types according to the purpose.
The depth of the neural network usually increase with the complexity of the investigated task, but the deeper the neural network is, the more likely it is exposed to the problem of overfitting: there are several techniques that allow to improve the generalizability of the model, such as batch normalization, early stopping and data augmentation [10].

1.2.2 Deep Learning in Muscle MRI of lower limbs

Since the manual segmentation of muscles, the gold standard, is very time consuming operator-dependent and requires expert knowledge, an alternative automatic methods could be very useful to standardize and optimize the process and to reduce the needed time and resources.
Some of the approaches that have been yet proposed in literature to segment lower limbs muscles are the following.
Annasamudram N. V. and co. [11] proposed a multi-method and multi-atlas methodology framework for automated segmentation of the four main functional muscle groups in 3D thigh MRI volumes: gracilis, hamstring, quadriceps femoris and sartorius. They combine the generalizability of multi-atlas segmentation, that is an anatomical mapping from multiple deformable models, and the accuracy of 3D UNet. Their dataset consisted of 21 MRI scans of 15 healthy subject from the age of 20 to 48, of which 12 were males. They evaluated the performances of their framework with DICE coefficient and Haussdorff Distance (95° percentile) (HD95) on these 15 healthy subjects, tested on 4 patients. They reached a DICE of 0.85 and a HD95 of 8.34 all over the muscles, so they obtain quite accurate segmentations of the individual muscle groups without any pre and postprocessing of the volumes.
A different approach was proposed by Piecuch L. and co. [12]: they proposed a framework of automatic segmentation of 18 muscles of the lower limbs, based on a hybrid UNETR architecture that combines convolutional and visual transformer blocks. They used a personalized loss function, that combine the Softdice Cross Entropy loss with prior anatomical information of muscle adjency, extracted by a probabilistic matrix. They implemented the project using MONAI libraries. Their

dataset was composed by 18 MRI volumes of pelvis and thigh from athletes. The chosen metrics were DICE, HD95 and volumetric error.
A fully automated 3D segmentation of leg muscles in MRI volumes was implemented by Guo Z. and co. [13]: they proposed a neighorhood relationship aware Fully Convolutional Network (FCN), based on FilterNet, a variant od 3D UNet, that can atomatically segment 5 leg muscles. The involved leg muscles are tibialis anterior, tibialis posterior, soleo, gastrocnemius and peroneus. They additionally used a kernel based edge detectors on the prediction maps to regularize the eventual voxel level dissimilarities inside neighborhood region defined by the kernel size. They evaluated their method on 40 T1-weighted MRI bilateral volumes of 10 healthy and 30 pathological subjects. In the preprocessing steps they perform a bias field correction method, to reduce the inhomogeneities in the intensity due to the MRI acquisition; then the intensities of the volumes have been normalized, to reduce the differences among the subjects; an Otsu thresholding was performed to remove part of the foreground; finally with a k-means clustering divided the left and right sides. They used as metrics DICE coefficient and the absolute surface-to-surface distance (ASSD), otaining a DICE of 0.88-0.91 and a ASSD of 1.04-1.66 mm for the five 3D muscles considered.
In the work of Gong and co. [14] the target was slightly different: they proposed an ensemble framework, obtained by the combination of two patch based binary class deep convolutional neural network with the same architecture that can segment thigh muscle and leg muscle from whole body MRI volumes. So, this framework is composed by a patch-based binary-class 3D UNet that segments only thigh muscle (considered as a unity) and another patch-based binary-class 3D UNet that segments only leg muscle. This two 3D UNet have been trained separately.
A preprocessing step was performed previously the training of the neural network: the whole body MRI volumes were firstly crop with a plane placed approximately at the end of the two femoral bones, in order to isolate lower limbs from the upper part of the body; the lower limb volumes obtained in this way were split into two partially overlapping regions, one that will be used to train the neural network for thigh and the other for leg separatly. During training step, the input volume was randomly croped into many overlapping regions and used as the input to train the models.
Also a postprocessing step was performed: after the inference of the two binary 3D models, one for thigh and one for leg, the output predictions were stiched together, in order to recompose a whole body prediction map with thigh muscle, leg muscle and background classes; the left and right thigh and leg muscles were determined by evaluating the distance of the segmentation point from the sagittal plane; finally potential false positives were reduced applying connected component analysis, where only the largest connected component in each muscle mask is preserved. The dataset was composed by 43 MRI bilateral volumes of the whole lower limbs.

The chosen metrics were the DICE coefficient and the absolute surface-to-surface distance. The average DICE value was 0.9189 for left leg, 0.8873 for left thigh, 0.9214 for right leg and 0.8893 for right thigh.
For the work described in this thesis, the chosen pipeline was inspired by these frameworks just described.

1.3 Federated Learning

1.3.1 Introduction to Federated Learning

In classical machine learning method of model training, data are stored in a centralized server, where they are also processing and cleaning. Then the model is trained locally. This process has some problems:

- there can be leak of data during their collection and processing;
- the improvement of the regulation for the user privacy protection, for example the General Data Protection Regulation (GDPR), may make harder the collection of the data needed by the model;
- often data are not enough for a proper training of the model.

A possible solution to these problems is Federated Learning [15].
Federated Learning is a setup in which individual users, also far from each other, collaborate to solve common machine learning purposes, coordinated by a central server, where the results are aggregated. Each user trains the own model locally on the own original data (local computing) that are stored locally. Then, only the locally obtained model is shared with the central server, where there is the aggregation between this model and the model yet present (model transmission). The final model is then used by the future users that will join to the project.
Since the only thing that user shares is the trained model, the dataset used for the training remains local and there is a strong protection of privacy and sensible data. Even without sharing the dataset, the final model obtained from the aggregation of each user's model is actually trained on a dataset that virtually increases for each model that is sent to the central server: this means that each single user can access to a well trained model.
In Federated Learning there are mainly three steps (*Figure 1.5*):

- step 1: the central server sends the initial model to the user;
- step 2: the user train the model locally on the own dataset and, then, share the trained model with the central server;

- step 3: the central server aggregate all the local model in only one final global model. This model will be the new one that will be send to the future user as initial model.

And then the cycle start again.

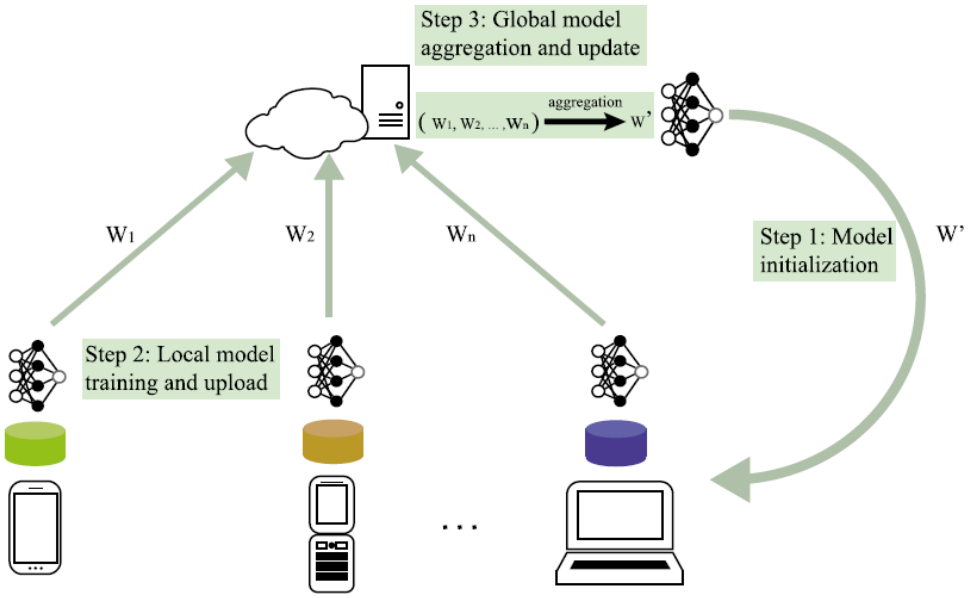

Figure 1.5: Federated learning process [15]. Representation of the three characteristic steps.

Federated Learning systems can be classified according to five different aspects: data partition, privacy mechanism, applicable machine learning model, communication architecture and methods for solving heterogeneity.
According to the data partition method, as shown in *Figure 1.6*, there can be:

- Horizontal Federated learning, that is used when user features of the two datasets overlap a lot, but the users overlap little. In this case the total number of samples for training and the accuracy of the trained model increase. There are possible application in phone model update and in logistic regression;

- Vertical Federated Learning, that is used when user features of the two dataset overlap a little, but the users overlap a lot. In this case the feature dimension of the training data increases and, as a consequence, also the ability of the model. There are possible application in decision tree and neural network;

- Federated Transfer Learning, that is used when both users and user features of the two datasets overlap a little. In this case it can be used to solve problems

of small unilateral data size and small label samples, so the effectiveness of the model can improve. For example, this can be useful in transfer learning, when the purpose is to optimize the performance of a task, but there is not enough related data for training.

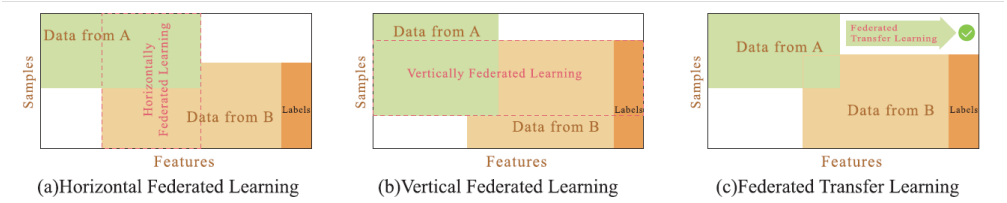

Figure 1.6: Classification of the Federated Learning according to data partition. It can be Horizontal Federated Learning, Vertical Federated Learning or Federated Transfer Learning [15].

According to the privacy protection mechanism, the federated learning can have:

- model aggregation, that is one of the most common, summarize the model parameters from all parties and avoid the transmission of the original data;
- Homomorphic encryption, that encrypts original data, so the users can calculate and process the entrypted data, but no original data will be disclosed in the process;
- Differential privacy, that protect user privacy by adding noise.

Federated Learning mainly supports three types of machine learning models:

- linear model, such as linear regression, ridge regression and lasso regression; it is easy to model and an effective model for federated learning;
- tree model, that is accurate, stable and can map non linear relationships;
- Neural Networks model, that is used for pattern recognition and intelligent control.

The Communication architecture must be designed properly according to the task in order to avoid leak of sensitive information.
There can be four different methods for solving heterogeneity:

- Asynchronous communication, that solves the problem of communication delay

- sampling, that selects the equipment that participates to the training;
- Fault tolerant Mechanism, that can prevent the system from collapsing;
- Model heterogeneity, that solves the corresponding heterogeneous device.

Because of the advantages of Federated Learning in sharing the results, protecting the user privacy and sensible data, systems have been implemented in several field. An example of federated learning system in medical imaging segmentation field is DAFNE.

1.3.2 DAFNE

Deep Anatomical Federated Network (DAFNE) [16] is an open-source multiplatform client/server software system that integrates deep learning segmentation model with an advanced user interface for computer-assistant manual segmentation. DAFNE is proposed as a solution to the problem of muscle segmentation in MRI: MRI volumes may have great variability in acquisition protocols, in particular in the variation of contrast and resolution, even in the case of the same organ, moreover some specific neuromuscular pathologies are rare. So, the problem is that it's difficult to locally collect a dataset for the training of the neural network that could be representative.
DAFNE is a federated learning system that provides to the users deep learning models that are upgraded thanks to the collaboration of the users themselves: so DAFNE models improve their performances with time and can generalize on different data, even non present in the original training dataset.

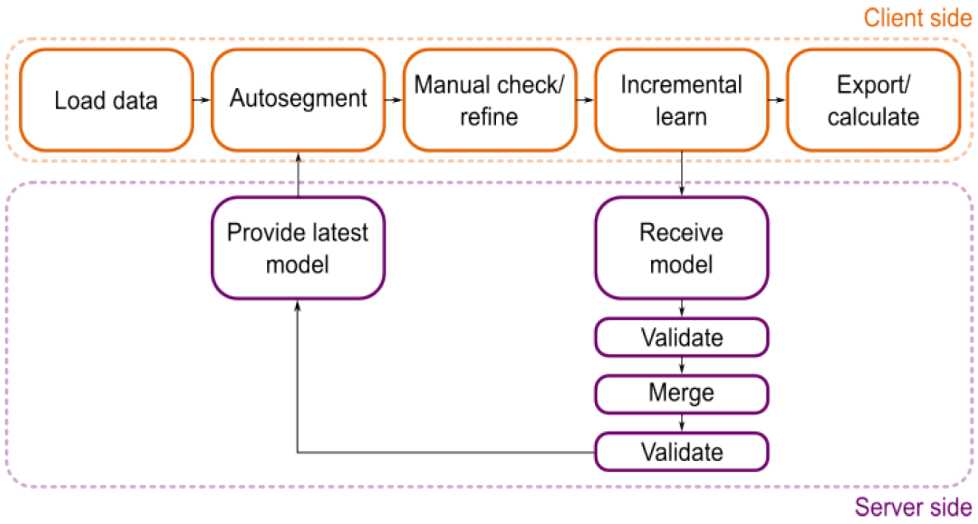

Figure 1.7: DAFNE workflow [16].

As can be seen in the workflow in *Figure 1.7*, the process includes a client side and a server side. On the client side the user:

- uploads the data, that is a MRI volume in NifTi or DICOM format;
- selects the correct region for the segmentation (it can be leg, left leg, right leg, thigh, left thigh, right thigh or none), according to the input volume;
- visualizes the volume slices and adjusts contrast and luminosity;
- with the command *Autosegment* the last version model of the selected body region is downloaded from the server and applied on the selected slices;
- the obtained segmentation can be controlled and manually refined with the available editing tool in the user interface;
- if the user wants, it at least five slices have been segmented, incremental learning can be performed. So, the model is automatically sent to the server, that verify and eventually integrate the model with the central one;
- finally, segmentation can be exported and metrics and features evaluated.

Contemporary, the server:

- provides the available segmentation model to the user;
- receives the upgraded version of the model from the user;
- evaluates the new version model comparing the obtained DICE with the one of the gold standard;
- if the outcome is positive, the new model is merged with the old one.

The models automatically available on the API of DAFNE are:

- leg model, that segments 6 leg muscles (*Figure 1.8*). It is available for left leg, right leg or bilateral volume and can be applied on MRI axial images;
- thigh model, that segment 12 thigh muscles (*Figure 1.9*). It is available for left thigh, right thigh or bilateral volume and can be applied on MRI axial images.

These models have the following characteristics:

- they are customized version of VNet and ResNet;
- the model have been initially pretrained on 44 dataset containing axial proto-density-weighted gradient echo bilateral acquisitions of leg and thigh, with resolution of 1.04x1.04x5.0 mm^3;

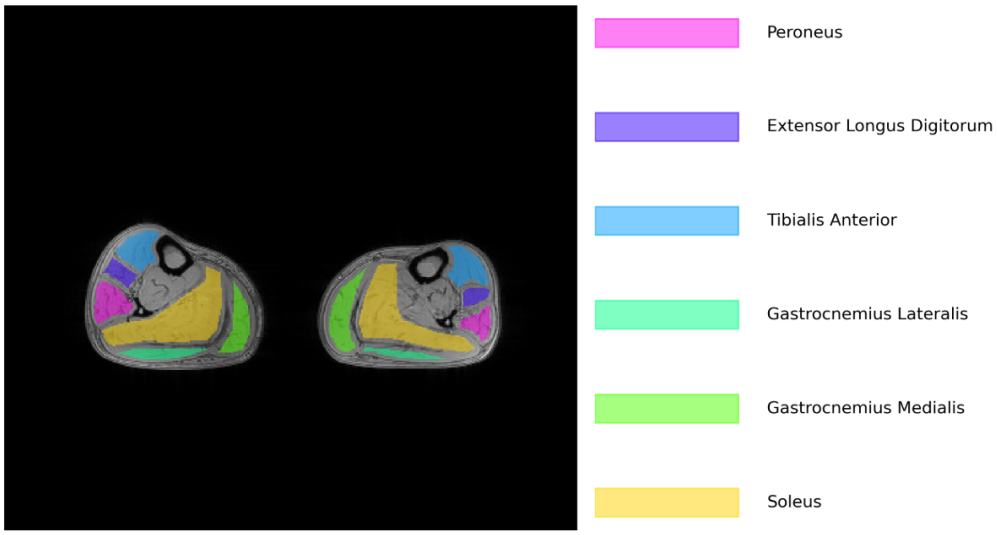

Figure 1.8: DAFNE leg model. It segments 6 leg muscles: Soleus, Gastrocnemius Medialis, Gastrocnemius Lateralis, Tibialis Anterior, Extensor Longus Digitorum and Peroneus [17].

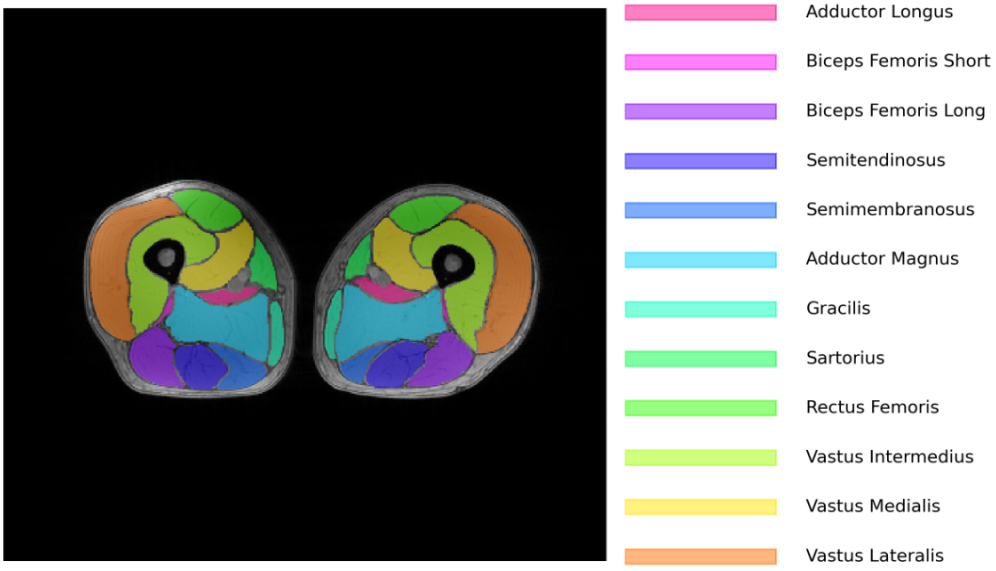

Figure 1.9: DAFNE thigh model. It segments 12 leg muscles: Vastus Lateralis, Vastus Medialis, Vastus Intermedius, Rectus Femoris, Sartorius, Gracilis, Adductor Magnus, Semimembranosus, Semitendinosus, Biceps Femoris Long, Biceps Femoris Short and Adductor Longus. [16].

- Tensorflow and Keras libraries have been used;
- the performances have been tested on 38 datasets of patients with suspected myositis, acquired with the protocol 2D axial T1-weighted turbo spin-echo sequence

On the website an abdominal model that segments 4 abdominal organs and a Lumbar Spine model that segments vertebrae, discs and spinal canal are available and can be downloaded.
The software, written in python, free and open source is available on GitHub [18]. The client is installable by Python Pacage Index (PyPI) and binary files are provided for Microsoft Windows, GNU/Linux and Apple MacOS.
The server is on Google Cloud Virtual Machine.
So, DAFNE provides an intuitive user interface with manual editing tools (user convenience), that allows the user to control and refine the segmentation. Moreover, as other federated learning systems, preserves privacy because data are locally stored by the user and not shared and allows to access to an upgraded model.
Dedicated documentation and more details are available on the website of DAFNE [17].

1.4 Monai

MONAI (Medical Open Network for Artificial Intelligence) is a PyTorch-based open-source framework for deep learning in healthcare imaging [19] [20].
In the work described in this thesis MONAI libraries have been used to implement the designed pipeline.
MONAI libraries provide a series of AI tool in python that are specifically for medical imaging with the purpose of:

- promoting the collaboration between academic, industrial and clinical reserchers developing a community with a common foundation;
- creating a state-of-the-art end-to-end training workflow for healthcare imaging;
- allow the researchers to create and evaluate deep learning models with an optimized and standardized way.

MONAI libraries provides a flexible preprocessing for multi-dimensional medical imaging data, implementations for networks, losses and evaluation metrics specific for the domain, customizable design for different user expertise, multi-GPU multi-node data parallelism support and the possibility to easily integrate in existing workflows thanks to the compositional and portable APIs.
Moreover, on MONAI Model Zoo [21] [22] there is a collection of pretrained medical

imaging models in MONAI Bundle Format. Here researchers and data scientists can share the latest model from the community.
Documentation and more information are available on the dedicated website [19].

Chapter 2

Dataset

2.1 Description of the original dataset

The original dataset is composed by 166 MRI volumes of lower limbs in NifTi format, acquired by Radbound University Medical Center, using the Dixon technique [23]. In the dataset there are 82 MRI volumes of the thigh and 84 MRI volumes of the leg, both of healthy and pathological patients: in particular the most of the volumes are from patients with two different pathologies and a small group is from healthy voluntiers. Some volumes are unilateral, so they represent only one limb of the patient, the right or left one, while other volumes are bilateral, so they represent both limbs.
These are more details about the the distribution of the volumes among the three groups:

- 50 volumes from patients with Fascioscapular muscular Distrophy (FSHD), with these characteristics:
  - 25 volumes of these represent thigh;
  - 25 volumes of these represent leg;
  - bilateral volumes;
  - only one volume for each patient;
  - dimension of (320, 200, 72);
  - spacing of (1.3594, 1.3594, 5.0000);
- 96 volumes from patients with Myotonic Dystrophy (MD1), with these characteristics:
  - 47 volumes of these represent thigh;

  - 49 volumes of these represent leg;
  - unilateral volumes;
  - from one to three volumes for each patient;
  - mostly of these volumes have a dimension of (256, 192, 32);
  - mostly of these volumes have a spacing of (1.0000, 1.0000, 5.0000);

- 20 volumes from healthy patients (HV), with these characteristics:
  - 10 volumes of these represent thigh;
  - 10 volume of these represent leg;
  - unilateral volumes;
  - only one volume for each patient
  - dimension of (256, 192, 32);
  - mostly of these volumes have a spacing of (1.0000, 1.0000, 5.0000), only three volumes have a spacing of (1.09375, 1.09375, 5.00000).

All volumes have already been filtered with N4ITK filter [24], that reduces the Bias Field Effect, a low frequency noise typical of MRI, due to the polarization of the field of the acquisition tools [25].
In the preprocessing step, that will be described with more details later in Chapter 3, each bilateral volume has been divided into two unilateral volumes, one for the right limb and the other for the left one. So, finally, the effective number of volumes that have been used in this work is 216, all of them unilateral.

2.2 Labels

At the beginning the masks of FSHD patient volumes, MD1 patient volumes and HV patient volumes had different combinations of segmented muscles.
To make the dataset uniform, a setup of labels has been chosen for all thigh volumes and another one for all leg volumes.
In particular, in *Figure 2.1* can be seen the chosen setup of thigh muscles and in *Figure 2.2* can be seen the one for leg.

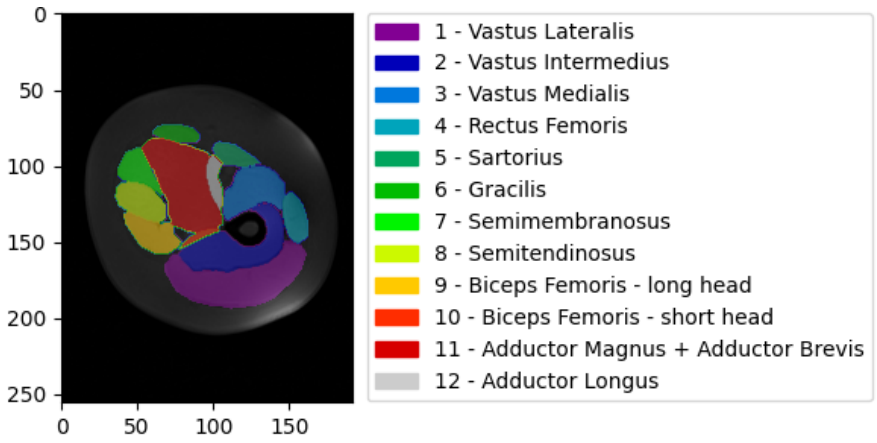

Figure 2.1: Thigh muscle setup. List of the chosen thigh muscles, with the correspondence image in the section of the example thigh volume on the left of the figure.

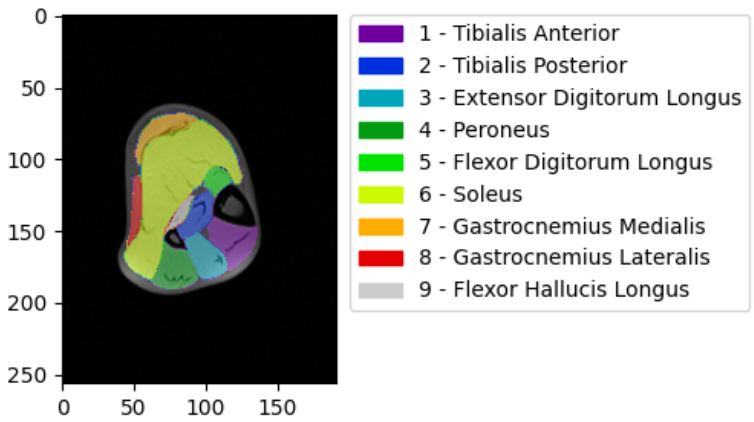

Figure 2.2: Leg muscle setup. List of the chosen leg muscles, with the correspondence image in the section of the example leg volume on the left of the figure.

The masks that were lack of some muscles have been integrated with the absent ones. An example for leg labels is shown by *Figure 2.3, a* and *Figure 2.7, a* where is highlighted the addition of the Flexor Hallucis Longus in a HV volumes and in MD1 volumes respectively. An example for thigh volume is in *Figure 2.4, a* where Semimembranosus, Semitendinosus, Biceps Femoris (long head) and Sartorius are added in HV volumes.
On the other hand, extra muscles with respect to this list have been removed from the masks where they were. This is the case of popliteus in FSHD volumes of leg (*Figure 2.5, c*). Also bone labels have been removed from the volumes that presented them, as can be seen in *Figure 2.5, a* and *Figure 2.6, a*. While,

for the Adductor brevis, since most of volumes presented its labels unified with the one of Adductor magnus, it has been chosen to unify these two labels in all volumes, creating the label "Adductor magnus + Adductor brevis" that includes both muscles, as *Figure 2.6, b* shows.

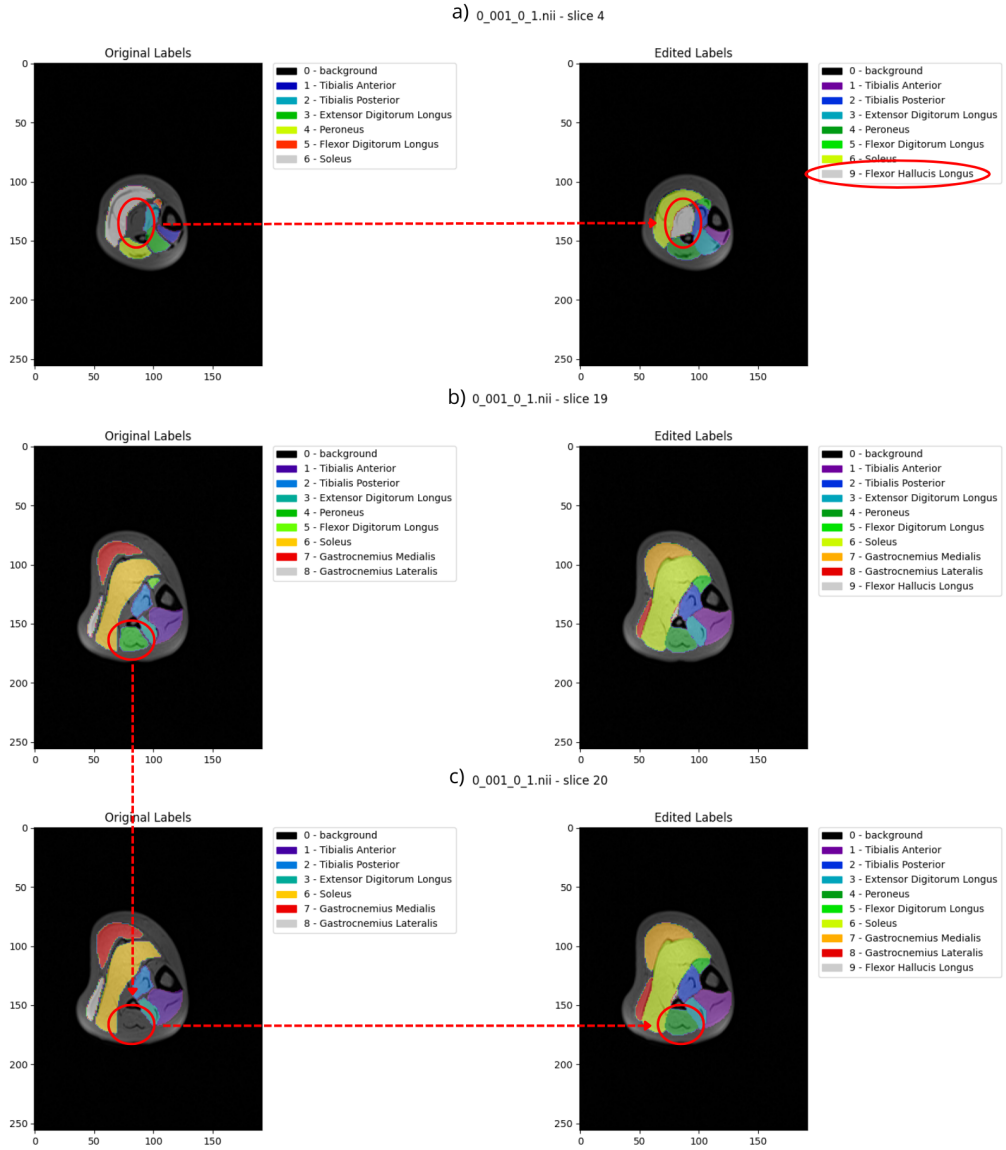

Figure 2.3: Example of some steps of the editing of HV volumes of leg.

In addition to these changes, segmentations have been refined where needed and added to the volume slices where they were absent, in order to obtain, in the end, continuous segmentations along the whole muscles in the region of interest.

Examples can be seen in *Figure 2.3, b and c*, in *Figure 2.4, b and c*, in *Figure 2.7, b and c* and in *Figure 2.8, b and c*. Residual and extra spots of labels have been removed, as *Figure 2.5, a* and *Figure 2.6, a and c* show, refinement of the edge of the labels have been done (*Figure 2.8, a*) and wrong labels have been reassigned (*Figure 2.5, b*).
Finally, muscles labels have been ordered in the same way in each volume of the dataset.

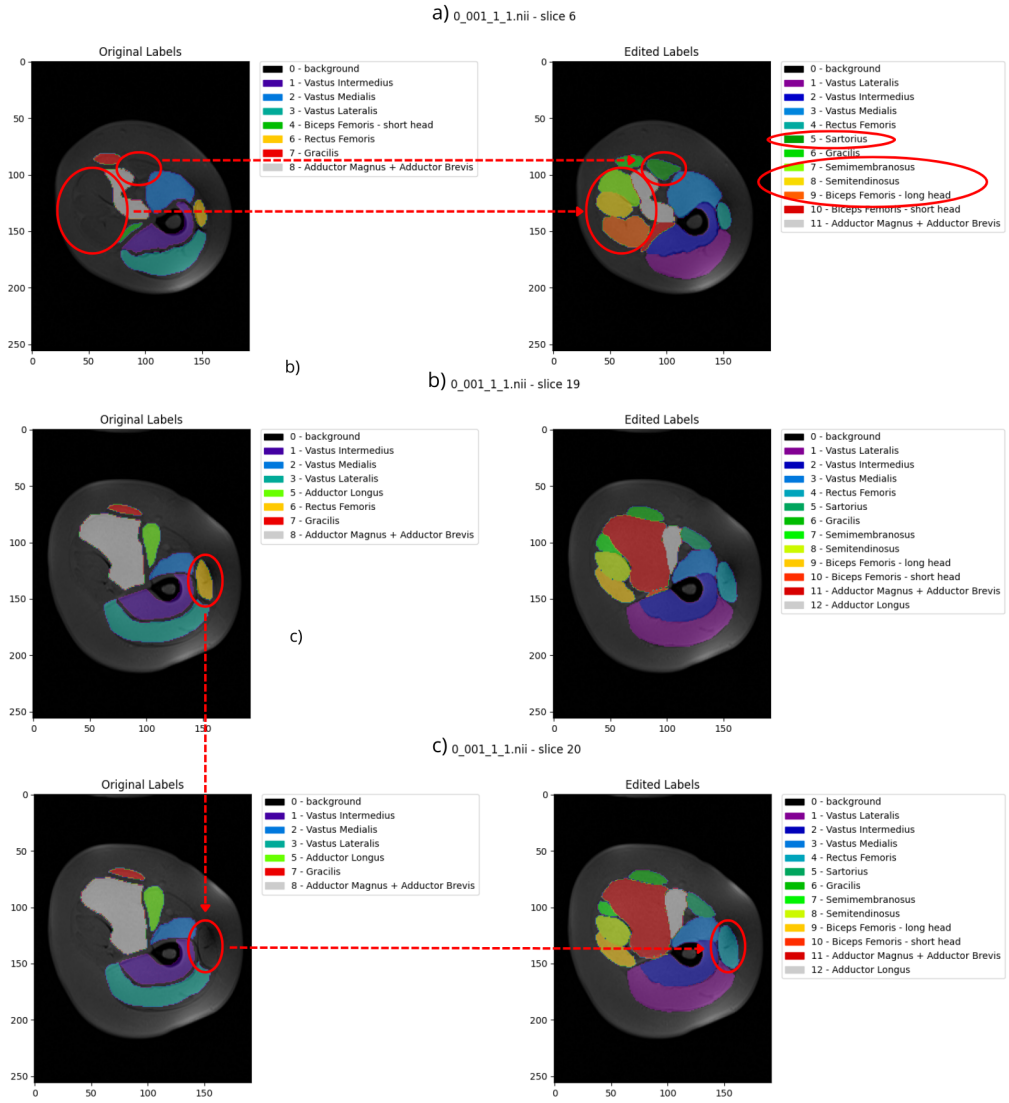

Figure 2.4: Example of some steps of the editing of HV volumes of thigh.

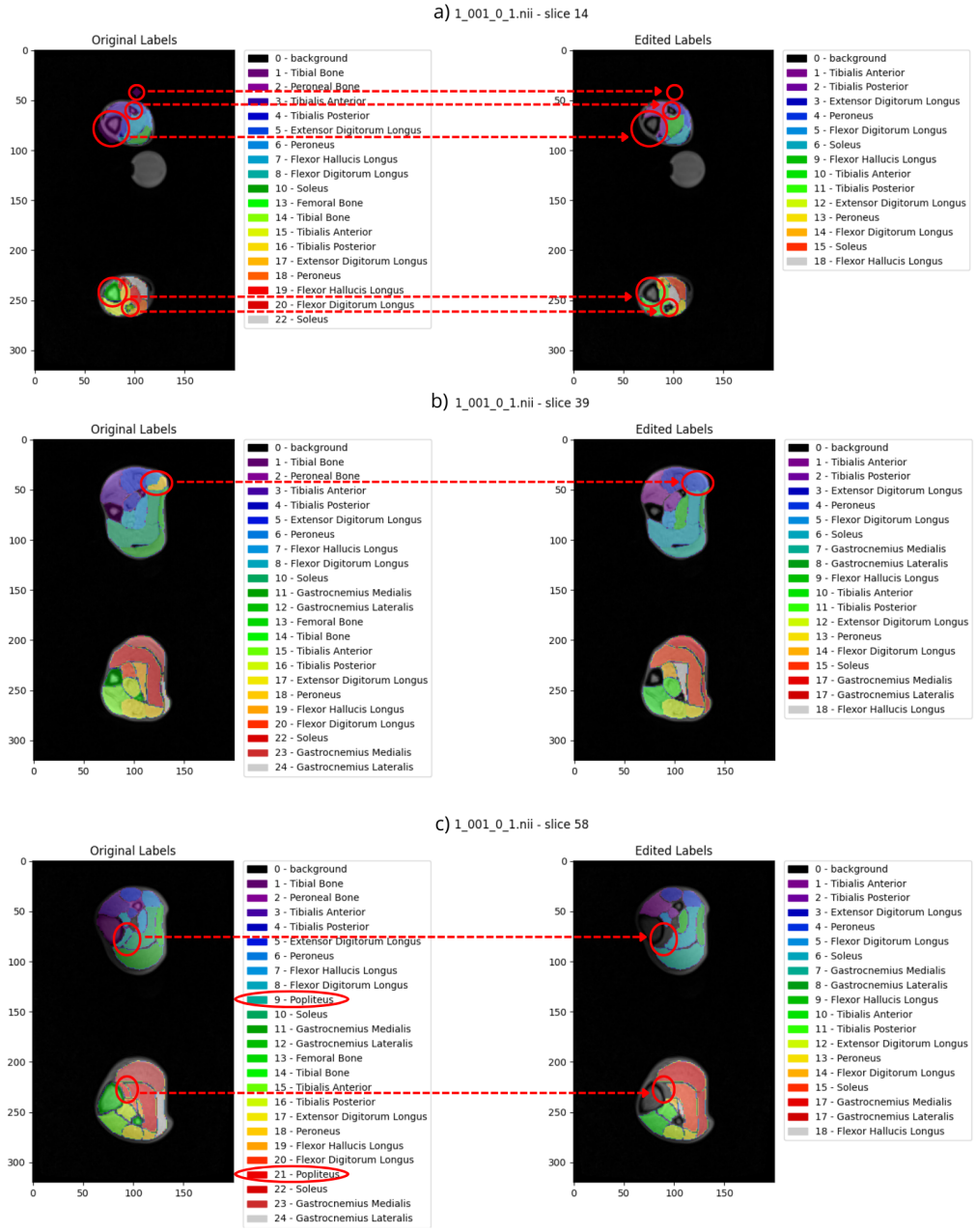

Figure 2.5: Example of some steps of the editing of FSHD volumes of leg.

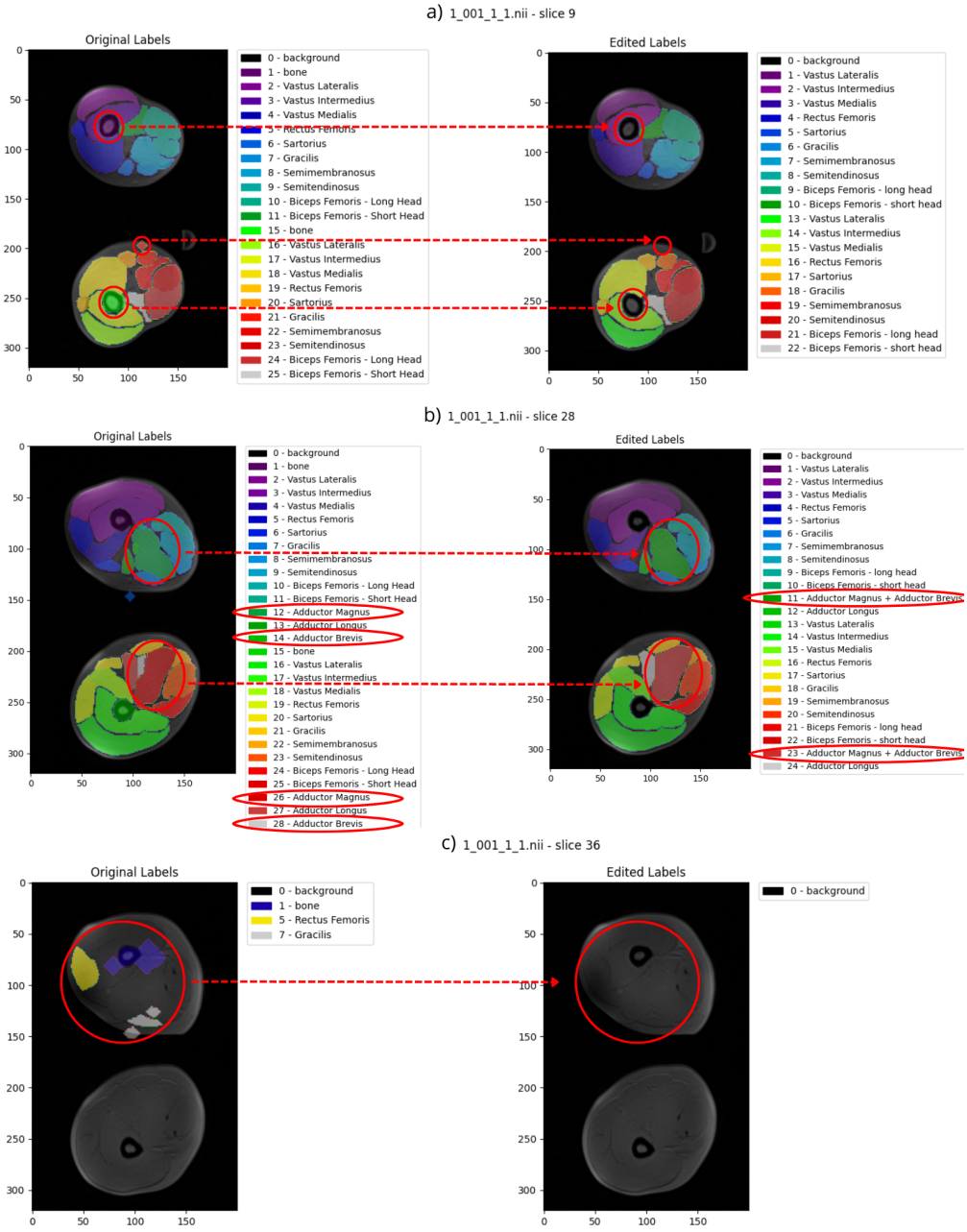

Figure 2.6: Example of some steps of the editing of FSHD volumes of thigh.

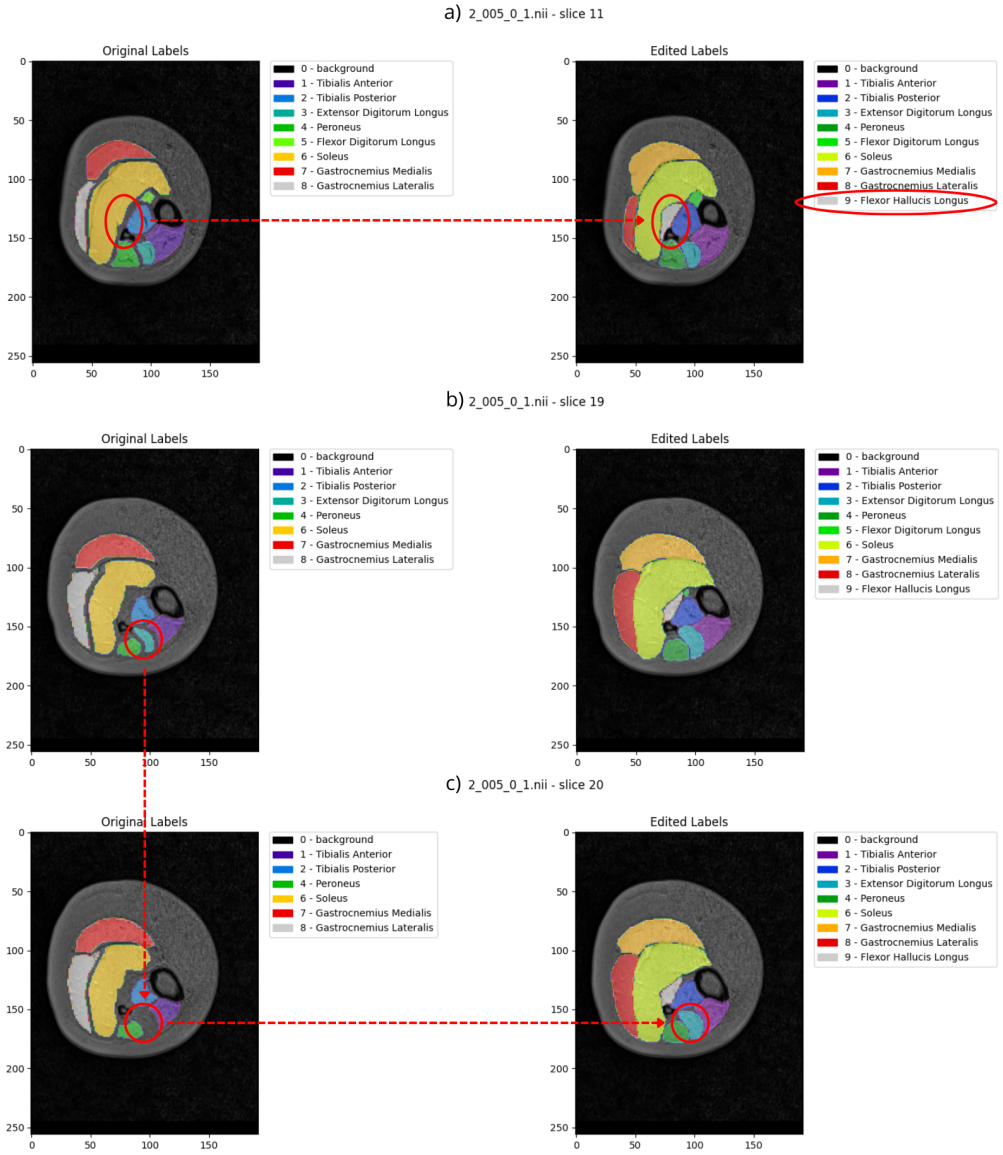

Figure 2.7: Example of some steps of the editing of MD1 volumes of leg.

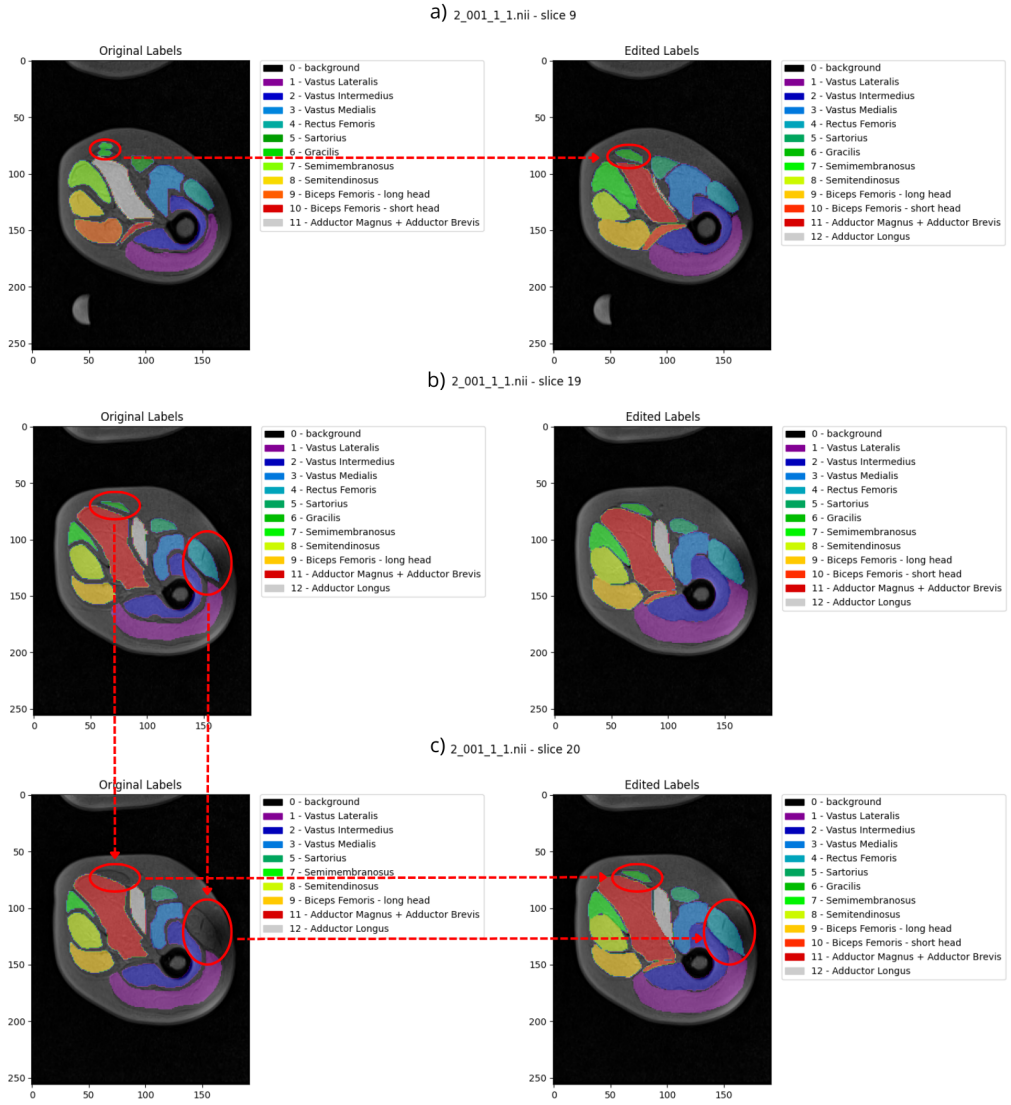

Figure 2.8: Example of some steps of the editing of MD1 volumes of thigh.

The software 3D Slicer has been used to modify and refine segmentations and to adjust the correspondance between label-muscles in the masks according to the chosen lists.
3D Slicer is a free and open source software, useful for visualize, process and segment 3D images, in particular in medical and biomedical fields [26].

2.3 Division in training, validation and test set

Volumes of thighs and volumes of legs have been considered as two separated sub-dataset.
Each sub-dataset has been divided into three groups of volumes, in order to perform all the different steps of the training of a neural network: training, validation and test sets.
Two different division methods have been tested.
The first one randomly extracted volumes from each patient type group, according to specific set percentages. In particular:

- training set: 80% of volumes from each group type have been extracted;
- validation set: 10% of volumes from each group type;
- test set: the remaining 10% of volumes from each group type.

In this way, a training set, a validation set and a test set for the volumes of thigh have been obtained, composed as follows:

- training set: a total of 65 volumes extracted as follows:
  - 20 volumes from FSHD type group;
  - 37 volumes from MD1 type group;
  - 8 volumes from HV type group;
- validation set: a total of 9 volumes extracted as follows:
  - 3 volumes from FSHD type group;
  - 5 volumes from MD1 type group;
  - 1 volumes from HV type group;
- test set: a total of 8 volumes extracted as follows:
  - 2 volumes from FSHD type group;
  - 5 volumes from MD1 type group;
  - 1 volumes from HV type group.

While, the training, validation and test sets obtained for the volumes of leg are composed as follows:

- training set: a total of 67 volumes extracted as follows:
  - 20 volumes from FSHD type group;

  - 39 volumes from MD1 type group;
  - 8 volumes from HV type group;

- validation set: a total of 9 volumes extracted as follows:
  - 3 volumes from FSHD type group;
  - 5 volumes from MD1 type group;
  - and 1 from HV type group;

- test set: a total of 8 volumes extracted as follows:
  - 2 volumes from FSHD type group;
  - 5 volumes from MD1 type group;
  - and 1 from HV type group.

The second method used to separate volumes into sets, randomly extracted volumes for training and validation set only from pathological patient type group according to specific set percentages, while assigned all the healthy ones to test set. In order to balance the final number of volumes for each set to the ones obtained with the previous division method, the chosen percentages for training and validation set was slightly different from the previous case:

- training set: the 85% of the volume of FSHD and MD1 groups;
- validation set: the remaining 15% of the volumes of the two pathological groups.

This time, the training, valitation and test set obtained for thigh volumes where composed as follows:

- training set: a total of 60 volumes extracted as follows:
  - 21 volumes from FSHD type group;
  - 39 volumes from MD1 type group;

- validation set: a total of 12 volumes extracted as follows:
  - 4 volumes from FSHD type group;
  - 8 volumes from MD1 type group;

- test set: a total of 10 volumes, all of them from HV type group.

Volumes of leg were divided into sets as follows:

- training set: a total of 62 volumes extracted as follows:
  - 21 volumes from FSHD type group;
  - 41 volumes from MD1 type group;
- validation set: a total of 12 volumes extracted as follows:
  - 4 volumes from FSHD type group;
  - 8 volumes from MD1 type group;
- test set: a total of 10 volumes, all of them from HV type group.

The *Figure 2.9* summarize the division of thigh and leg volumes according to the first method, while *Figure 2.10* summarize the division according to the second method.

	LEG				THIGH			
	FSHD	MD1	HV	**total**	FSHD	MD1	HV	**total**
Training set	20	39	8	**67**	20	37	8	**65**
Validation set	3	5	1	**9**	3	5	1	**9**
Test set	2	5	1	**8**	2	5	1	**8**

Figure 2.9: Division of the dataset volumes according to the first method. In each cell is indicated the number of volumes.

	LEG				THIGH			
	FSHD	MD1	HV	**total**	FSHD	MD1	HV	**total**
Training set	21	41	0	**62**	21	39	0	**60**
Validation set	4	8	0	**12**	4	8	0	**12**
Test set	0	0	10	**10**	0	0	10	**10**

Figure 2.10: Division of the dataset volumes according to the second method. In each cell is indicated the number of volumes.

These two different divisions of dataset have been used separately: for each of them a neural network has been trained and the corresponding metrics have been calculated. Finally the results have been compared.

Chapter 3

Methods

3.1 Local training

The first aim of this thesis was the implementation of a pipeline to realize a 3D Neural Network that could segment the muscles of interest in 3D MRI volumes of lower limbs.
Python 3.9.19 and MONAI libraries 1.3.0 have been used.
Thigh and leg volumes have been considered separately, so, for each division method of the dataset, two 3D Neural Networks have been trained and tested: one 3D Neural Network can segment the 12 muscles chosen in the thigh and another 3D Neural Network the 9 muscles chosen in the leg, according to the two lists of muscles of interest in Chapter 2.

Figure 3.1: Pipeline - summary diagram

After the division of the dataset into training, validation and test set (considering separately volumes of thigh and volumes of leg) the main steps of this process were the following (*Figure 3.1*):

- preprocessing of the volumes;
- training of the Neural Networks;
- inference;
- testing and evaluation of the results, using the chosen metrics.

MONAI [19] python tools have been used to perform the several steps described in this chapter.

3.1.1 Preprocessing

The preprocessing was the first step done after the fixing of the labels and the division of the dataset, as described in Chapter 2.
Despite the two different portions of lower limb represented, both volumes of thigh and volumes of leg have been preprocessed in the same way.

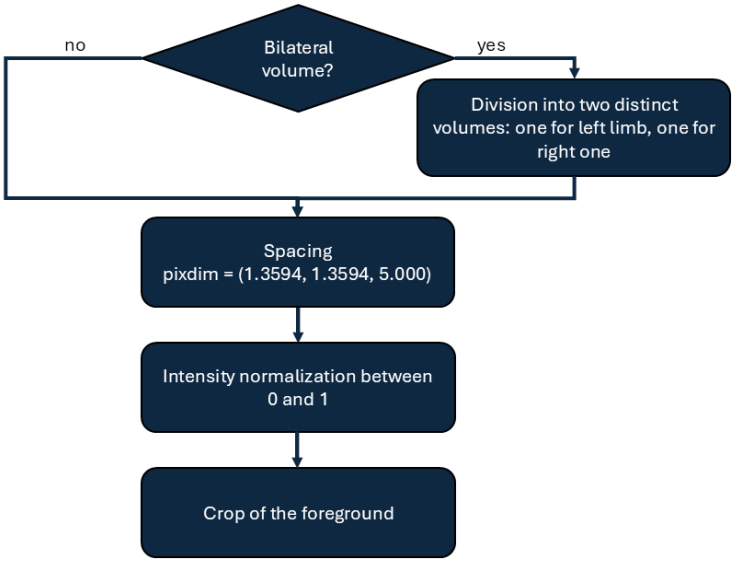

Figure 3.2: Preprocessing workflow. It summarize preprocessing steps.

As summarized in *Figure 3.2*, the preprocessing steps were the following:

- division of the bilateral volumes into two distinct unilateral volumes, one for left limb and one for right one, using an implemented function;
- resize of the voxel at the spacing (1.3594, 1.3594, 5.0000), the bigger voxel dimension among the volumes in the dataset, using MONAI function *Spacing*;
- normalization of the intensity of the pixel between 0 and 1, using MONAI function *ScaleIntendityd*;
- crop of part of the foreground with MONAI function *CropForegroundd*.

At the end of the preprocessing step, volumes have different shape on 3D directions, but the same voxel dimension, intensity between 0 and 1 and a crop of the foreground

all around the volume of interest.
Volumes from FSHD group needed all of these steps of preprocessing, because they were bilateral volumes in the original dataset: so they needed to be firstly divided into left and right limb volumes and then preprocessed. *Figure 3.3* shows an example of a preprocessed bilateral volume from FSHD dataset group.

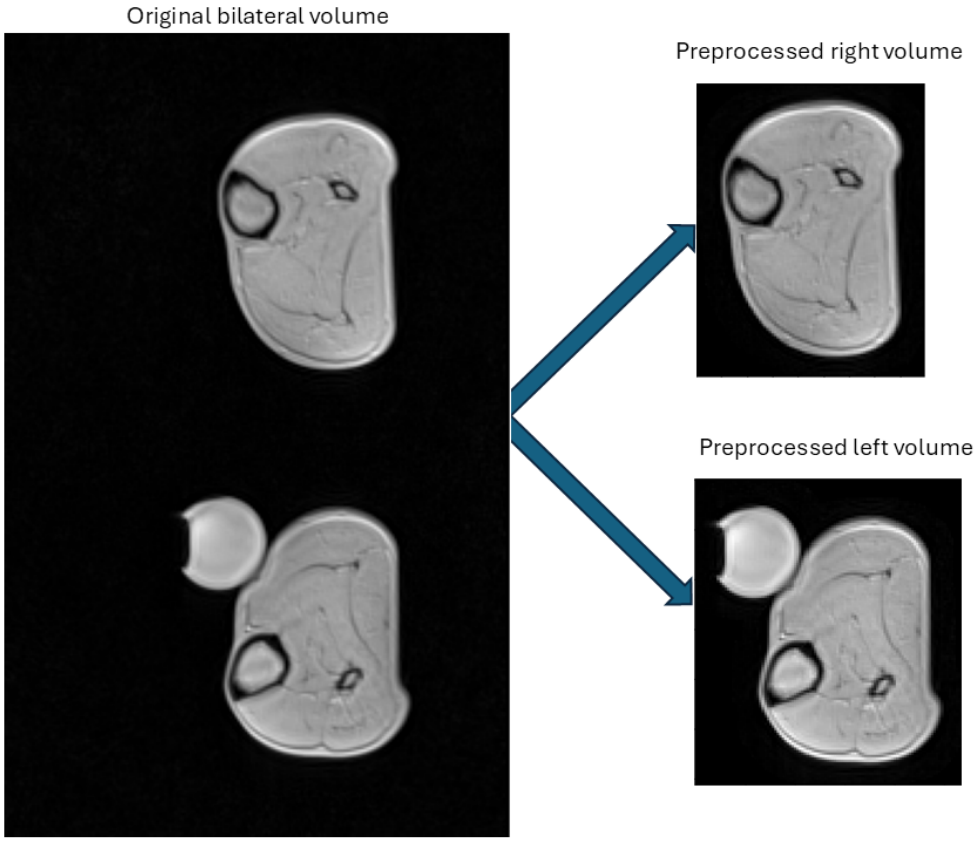

Figure 3.3: Example of a preprocessed bilateral volume of a leg from FSHD dataset group. On the left there is the original bilateral volume, while on the right there are the two extracted and preprocessed unilateral volumes.

While, in *Figure 3.4* can be seen an example of a preprocessed unilateral volume of a thigh. Volumes from MD1 and HV dataset group have been preprocessed in this way.

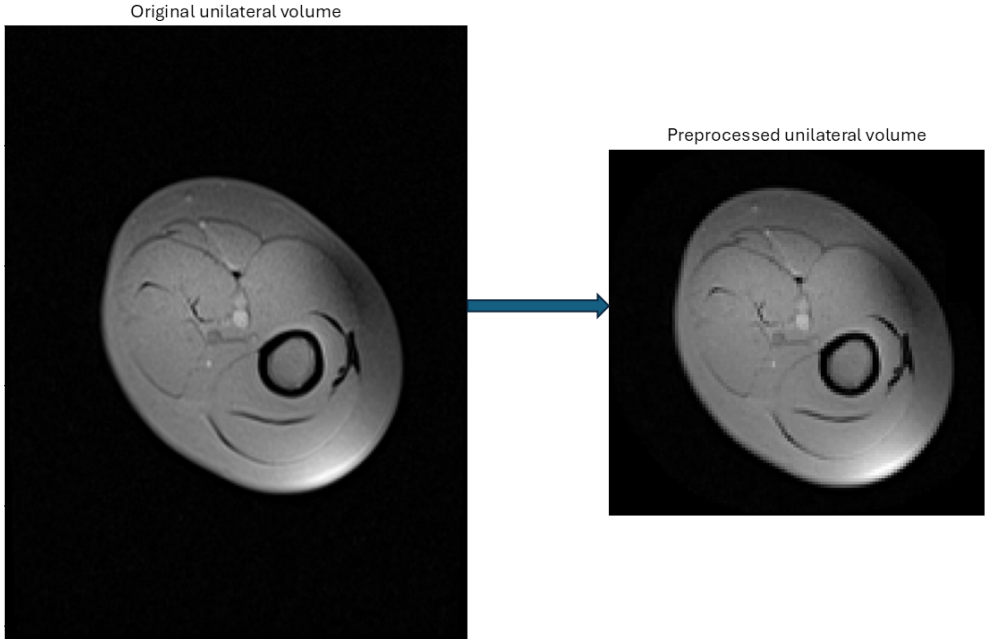

Figure 3.4: Example of a preprocessed unilateral volume of a thigh from HV dataset group. On the left there is the original unilateral volume, while on the right there is the preprocessed one.

3.1.2 Training

The chosen type of 3D Neural Networks trained both for thigh and leg is the SwinUNETR, that usually need larger computational resources and longer training times than traditional UNETR, due to the higher training complexities, but, on the other hand, it should provide better results. As suggestes by Piecuch, L. et al. [12] the first attempts of training have been done with UNETR 3D Neural Network with DiceCE as Loss Function, a combination of Dice Loss and Cross Entropy Loss. To increase the segmentation performances of the Neural Network, the 3D Neural Network type has been changed into the SwinUNETR: this kind of architecture is recenter and more powerful than the UNETR one, even if it is more complex and requires more computational resources and longer time in the training, as written in the article of Piecuch, L. et al. [12].
Five different Loss Functions have been tested for both divisions of dataset for both thigh and leg: a total of 10 3D Neural Networks for thigh and 10 3D Neural Networks for leg have been trained. According to the performances obtained in the segmentation, valuated with the chosen metrics, after the application of the inference and postprocessing steps only two SwinUNETR 3D Neural Network have been chosen: one for thigh and one for leg.

SwinUNETR

In SwingUNETR type of Neural Network there is a swin transformer encoder combined with a CNN-based decoder at multiple resolutions via skip connections [27]. Thanks to Swin transformers the capability of learning multi-scale contextual representations and modeling long-range dependencies increase with respect to the case in which a ViT-based approaches with fixed resolution is used [27].

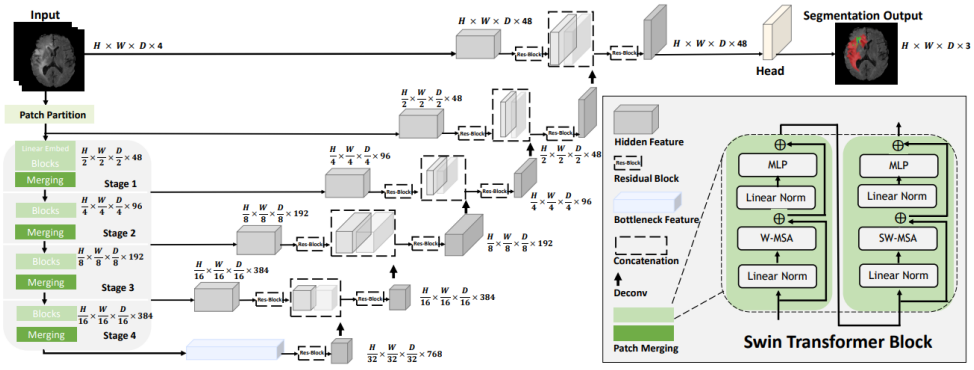

Figure 3.5: SwingUNETR architecture overview [27]

As described in *Figure 3.5*, the input of a SwingUNETR is 3D multi-modal MRI images with 4 channels, so with a dimension of *(H', W', D', S)*. This input pass through a patch partition layer that creates non-overlapping patches of the input volumes that are used to realize windows with desired size. On these windows the self-attention is computed.
The encoder has a patch size of (2, 2, 2) and, in the case of MRI images with 4 channel, the dimension of the feature is (2, 2, 2, 4). There are 8 layers in total in the encoder, because it has 4 stages with 2 transformed block each. There are 4 stages, each of them with a specific resolution, but with the same network design:

- the first step has a resolution of *(H/2, W/2, D/2)*,
- the second stage *(H/4, W/4, D/4)*,
- the third stage *(H/8, W/8, D/8)*,
- and the fourth stage *(H/16, W/16, D/16)*.

The network design, common for each stage is the following.
It starts with a linear embedded layer that create 3D tokens with the same resolution of the specific stage.
In order to maintain the hierarchical structure of the encoder, there is a patch merging layer that decrease the resolution of feature representations by a factor 2

at the end of each stage.
Then, there is a patch merging layer that concatenates patches with a resolution of (2, 2, 2) into a 4C-dimensional feature embedding.
At the end of the stage a linear layer reduces to 2C the feature size of the representations [27].
At each resolution the decoder uses the extracted feature representations of the encoder via skip connections, because the network design of a SwinUNETR as a U-shape.
At the end of each stages the output feature representations are reshaped into the specific resolution size of that stage and go through a residual block, where there are two (3, 3, 3) convolutional layer, that are normalized by instance normalization layers. Then a deconvolutional layer increases the resolution by a factor of 2.
The output features are concatenated with the output ones of the previous stage and the final concatenation goes into a residual block. Finally a convolutional layer of dimension (1, 1, 1) and a sigmoid activation function compute the segmentation outputs obtained in this way [27].

Loss Function

The five different loss functions that have been tested are:

- Dice Cross Entropy Loss function (*DiceCELoss* function in MONAI libraries), a combination of Dice Loss and Cross Entropy Loss functions and it returns the weighted sum of them;
- Generalised Dice Loss function (*GeneralizedDiceLoss* function in MONAI libraries), that has been described by Sudre C. et al [28];
- Focal Loss function (*FocalLoss* function in MONAI libraries), that has been described by Tsung-Yi Lin et al [29];
- Dice Focal Loss function (*DiceFocalLoss* function in MONAI libraries), that combine the Dice Loss and Focal Loss into a weighted sum of them;
- Generalized Dice Focal Loss function (*GeneralizedDiceFocalLoss* function in MONAI libraries), that combine into a weighted average both Generalized Dice Loss and Focal Loss.

All these loss functions are available among MONAI library tools [30].
The best results in therms of metrics for thigh have been obtained with the 3D Neural Network trained with Dice Cross Entropy Loss function, while for leg with Dice Focal Loss function.
More in details, Dice Cross Entropy Loss Function returns a weighted sum of:

- Dice Loss function, described for the first time by Milletari F. et al [31], that is based on the calculation of DICE coefficient between the predicted segmentation and the ground truth. Using Dice Loss function it is no more needed samples re-weighting to establish the right balance between foreground and background voxel, but this is valid if they are strongly unbalanced and this loss function is indicated for binary segmentation task [31];
- Cross Entropy Loss function [32], that is useful with multiclasses tasks. It computes the cross entropy between input logits for each class and the target. It could be used also for unbalances training set.

While, Dice Focal Loss function performs a weighted average between:

- Dice Loss function, previously described [31];
- Focal Loss, that is calculated as in the *equation 3.1*

$$FL(p_t) = -(1 - p_t)^{\gamma} log(p_t) \tag{3.1}$$

where $-\log(p_t)$ is the Cross Entropy Loss. So it adds the factor $(1 - p_t)^{\gamma}$ to the standard Cross Entropy criterion [29].

Training parameters

For both SwinUNETR 3D Neural Network for thigh and SwinUNETR 3D Neural Network for leg the volumes from training and validation set, after the preprocessing step, have been used as input of the training step. During training step, for both sets, using MONAI libraris, the volumes have been:

- loaded with *LoadImaged*;
- added with an initial channel with *EnsureChannelFirstd*;
- normalized according to the previously defined roi size of (96, 96, 32) with *SpacialPadd*.

In addiction to this, for training set 8 samples of the dimension of roi size have been extracted from each volume with *RandCropByPosNegLabeld*. This step acts as a sort of Data Augmentation, that helps in the training of a Neural Network increasing the amount of data.
AdamW has been chosen as optimizer, with a learning rate of 1×10^{-4} and a weight decay of 1×10^{-4}. *Softmax* is the activation function.
The maximum number of epochs has been set to 500. The validation step is done every 5 epochs and, if there is no improvement after 10 validation steps, the early stopping step is performed.

There are two differences between 3D Neural Network for thigh and 3D Neural Network for leg. The first one is in the final chosen loss function: it is Dice Cross Entropy Loss for thigh and Generalized Dice Focal Loss for leg. The second difference is the output channel parameter of SwinUNETR, that is set to 13 for thigh, because there are 12 muscles to be segmented, plus the backgraund, and set to 10 for leg, because there are 9 muscles plus the background.

3.1.3 Inference and Postprocessing

In the inference step the volumes from all the three sets have been processed with the same steps described previously in the preprocessing paragraph. In addiction to those, with *SpatialPadd* the padding of the volumes is performed, according to the setted roi size, that is (96, 96, 32), the dimension of the volumes used to train the 3D Neural Networks.
Then the inference of the 3D Neural Network is applied using a sliding window of the same dimension of the roi size. On the segmentation obtained in this way two different combination of postprocessing adjustments have been tested in order to obtain better results. Each postprocessing kind combines the following two steps:

- smaller object than a set number of pixel have been removed;
- smaller holes than a set number of pixel have been closed.

The first postprocessing removed smaller object than 60 pixel and closed smaller holes than 40 pixel, while the second postprocessing removed smaller object than 30 pixel and closed smaller holes than 20 pixel.
The final segmentation obtained at the end of this step are saved and on these ones the chosen metrics have been calculated to evaluate the performance of the 3D Neural Networks.
This step is the same both for 3D Neural Network for thigh and 3D Neural Network for leg.

3.1.4 Metrics

In order to evaluate the segmentation obtained from the inference of the corresponding Neural Network in the proper data set, two different metrics have been used:

- DICE Coefficient,
- Haussdorff Distance at 95° percentiles.

Both metrics have been calculated with MONAI libraries [33].

DICE Coefficient

DICE coefficient is calculated by the *equation 3.2*:

$$DICE = \frac{2|P \cap G|}{|P| + |G|} \tag{3.2}$$

DICE is a number from 0 to 1 that evaluate the overlap between the segmentations obtained from the model and the reference ones. It is calculated using MONAI function *DiceMetric*. This function computes average Dice score of the segmentation compared to the ground truth and can support multiclasses and multilabel tasks.

Haussdorff Distance

Haussdorff Distance evaluate the similarity between the segmentations obtained from the Neural Network and the reference ones, measuring how the edge of the two segmentations are far from each other. It is in therms of pixel, and, if the resolution, defined as the distance between two adjacent pixels, is known, Haussdorf Distance can be converted into meters.
If the edge of one segmentation is called X and x is point of it, Y is the edge of the other segmentation and y is one point of it, Haussdorff Distance can be calculated with *equation 3.3*

$$HD = max\{sup(d(x, Y),)sup(d(X, y)\}) \tag{3.3}$$

where it is considered equal to the maximum value between the maximum distance between x to Y and the one between y to X.
Haussdorff distance is calculated thanks to the MONAI function *HausdorffDistance-Metric*, that can support multiclasses and multilabel tasks. This function allows to set the percentile of the distance: it has been set at 95° percentile, in order to make the metric robuster and less affectable by residual spot of the segmentation after the postprocessing step.

3.2 Federated Learning segmentation

DAFNE makes available different ways to use federated learning in order to achieve the own purposes in segmentation of muscle of lower limbs.
For example it allows to segment the own dataset using the remote models available on DAFNE repository.
In this work this method was used to generate segmentations, in order to evaluate the corresponding metrics and compare these with the metrics calculated on the segmentations obtained with the local 3D Neural Networks.

Since DAFNE allows to segment both thigh and leg, performances have been calculated and compared for both thigh and leg.
Since DAFNE supports NifTi format for the volumes in input, it was not necessary to convert the dataset into another format. The volumes have been firstly preprocessed with the same steps described in the previous paragraph of this chapter dedicated to the preprocessing for the local 3D Neural Network.
After this step, the preprocessed volumes have been loaded into di DAFNE API, the selected available Neural Network was applied and the resulting segmentation have been saved. The choice of the Neural Network was made according to the kind of each single input volume. In total four different available Neural Network have been used: one for left thigh, one for right thigh, another one for left leg and the last one for right leg.
Two main aspects emerged:

- all the four used Neural Networks available in DAFNE are 2D type, so they segmented one slice of the input volume at a time;
- the original volumes in the starting dataset presented segmentations of the muscles only in the part of interest of the volume, usually in the central part of the longitudinal axes of the limb, even if in the other slices there still were muscles of interest.

To manage these aspects, a step of postprocessing was necessary to balance the segmentation results with the ones obtained from local 3D Neural Network: in the segmentations in output from DAFNE only the slices that contained a muscle segmentation also in the reference segmentation have been conserved, while the others have been made void.
DAFNE available Neural Networks for thigh and leg, both for right and left limb, does not segment all muscles that the local 3D Neural Networks segmented and the labels were in different order. In particular, the muscles of thigh segmented by DAFNE are the following, in the following order:

- label 1: Vastus lateralis
- label 2: vastus medialis
- label 3: vastus intermedius
- label 4: rectus femoris
- label 5: sartorius
- label 6: gracilis
- label 7: adductor magnus
- label 8: semimembranosus

- label 9: semitendinosus
- label 10: biceps femoris - long head
- label 11: biceps femoris - short head
- label 12: adductor longus

While, for the leg the segmentated muscles are the following:

- label 1: soleus
- label 2: gastrocnemius medialis
- label 3: gastrocnemius lateralis
- label 4: tibialis anterior
- label 5: extensor digitorum longus
- label 6: peroneus

In order to compare the results, modifications in the original reference segmentations were needed:

- in the reference segmentations only the label of muscles that were in common with DAFNE segmentations have been kept, the other ones have been erased;
- moreover, the remaining muscle labels have been moved in the same sequence presented in the DAFNE segmentations.

Metrics have been calculated on this segmentations: also in this case DICE and Haussdorff Distance. The results have been evaluated comparing the metrics obtained from these segmentations to the ones obtained form the segmentations in output of the 3D Neural Networks.

Chapter 4

Results

Metrics calculated from the segmentations, both of local SwinUNETR 3D Neural Networks and DAFNE centralized Neural Networks, have been organized in bar graphs to facilitate the reading.

4.1 Results of Local training segmentation

Both for thigh and leg case, the results of each local SwinUNETR 3D Neural Networks that have been testet, have been divided into training, validation and test set. The elaboration of the obtained metric values have led the choice of the final 3D Neural Networks.
DICE and Haussdorff Distance (95° percentile) have been calculated:

- for each whole volume, considering the average value among the segmentation of all muscles. Calculating the average value of metrics on each whole volumes, allows to obtain only one value that can describe in general the performance in segmentation of the Neural Network on each set of volumes. So, it is obtained one average value of the metric of the whole volumes and its standard deviation for the training set, one for the validation set and one for test set;

- for each muscle, calculating the average value of the metrics got for a specific single muscle in each volume of the set. So, for each set, one average value of the metric and its standard deviation are obtained that describe the performances of the Neural Network in segmenting each single muscle.

The following paragraphs report with more details the obtained metrics organized as just described.

4.1.1 SwinUNETR 3D Neural Network for thigh

For each tested 3D Neural Network for thigh, whole volume metric graphs and single muscle metric graphs are reported.
The following graphs show the whole volume metric calculated for each set. Each bar represent a set: the orange bar indicates metrics of training set, the blue bar indicates metrics of validation set and the green bar indicates metrics of test set.

First division of Dataset

In this section graphs obtained from the first division of the dataset are reported, dividing them according to the loss function chosen.

- **1° test: SwinUNETR with Dice Cross Entropy Loss function**
  In *Figure 4.1* graphs of DICE and HD95 obtained with SwinUNETR with Dice Cross Entropy Loss function are represented: mean value and its standard deviation are reported for each set of volumes.
  The *Figure 4.2* graph shows the average DICE value obtained for each muscle with SwinUNETR with Dice Cross Entropy Loss function, for each set of volumes.
  The *Figure 4.3* graph shows the average HD95 value obtained for each muscle with SwinUNETR with Dice Cross Entropy Loss function, for each set of volumes.

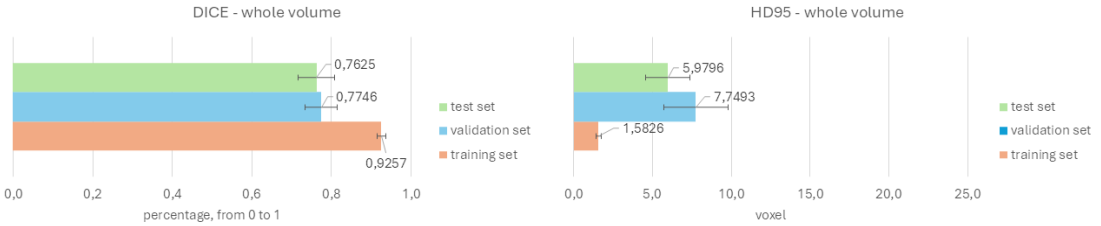

Figure 4.1: Whole volume metrics obtained with SwinUNETR 3D Neural Network with Dice Cross Entropy Loss function for thigh

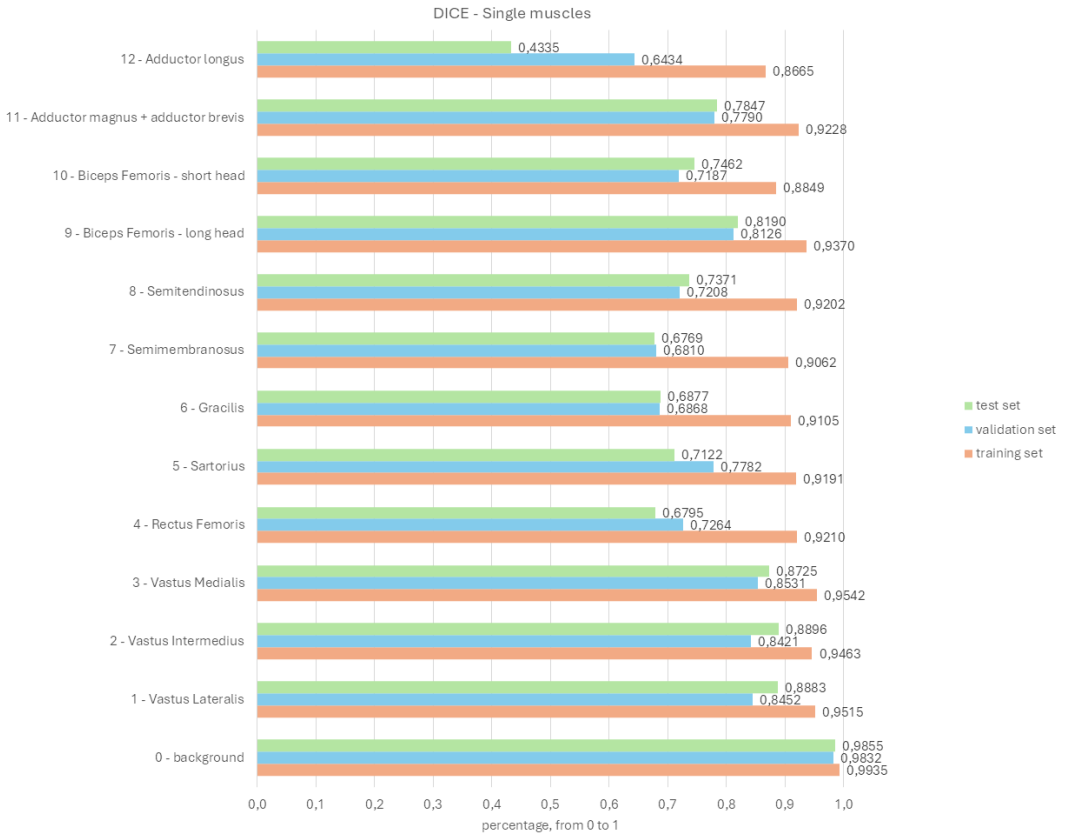

Figure 4.2: DICE of single muscles obtained with SwinUNETR 3D Neural Network with Dice Cross Entropy Loss function for thigh

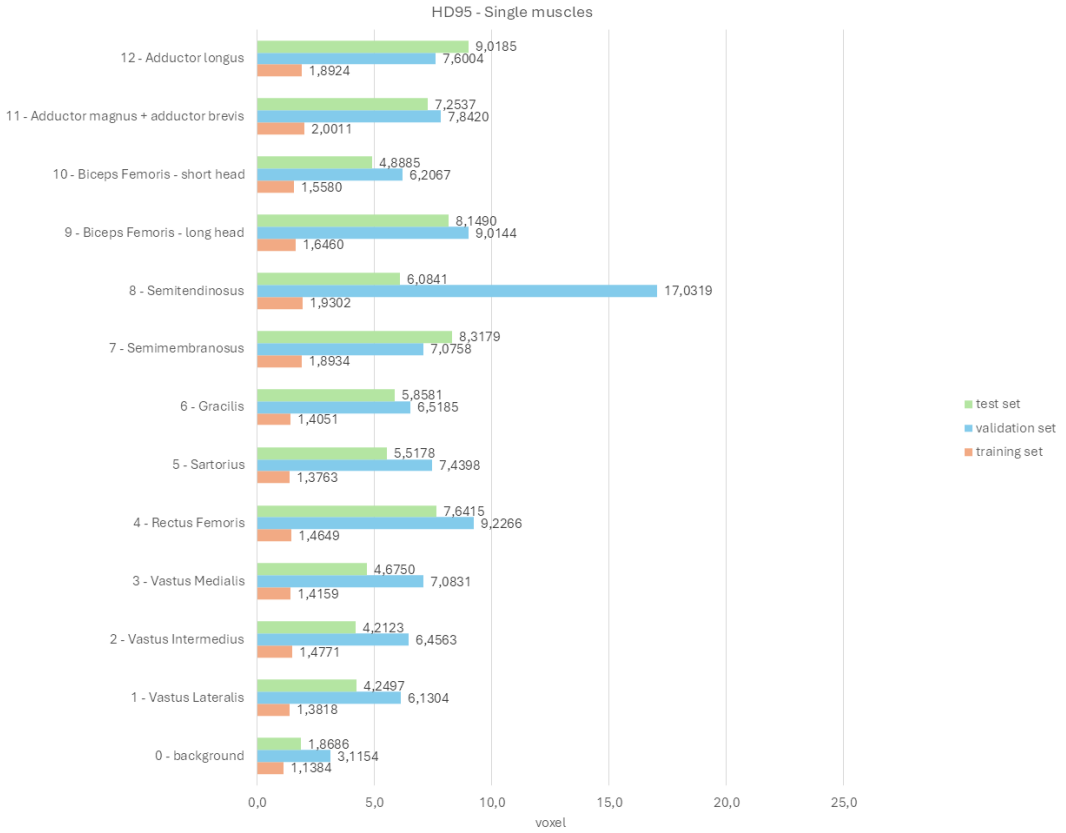

Figure 4.3: HD95 of single muscles obtained with SwinUNETR 3D Neural Network with Dice Cross Entropy Loss function for thigh

- **2° test: SwinUNETR with Generalized Dice Loss function**
  In *Figure 4.4* graphs of DICE and HD95 obtained with SwinUNETR with Generalized Dice Loss function are represented: mean value and its standard deviation are reported for each set of volumes.
  The *Figure 4.5* graph shows the average DICE value obtained for each muscle with SwinUNETR with Generalized Dice Loss function, for each set of volumes.
  The *Figure 4.6* graph shows the average HD95 value obtained for each muscle with SwinUNETR with Generalized Dice Loss function, for each set of volumes.

Figure 4.4: Whole volume metrics obtained with SwinUNETR 3D Neural Network with Generalized Dice Loss function for thigh

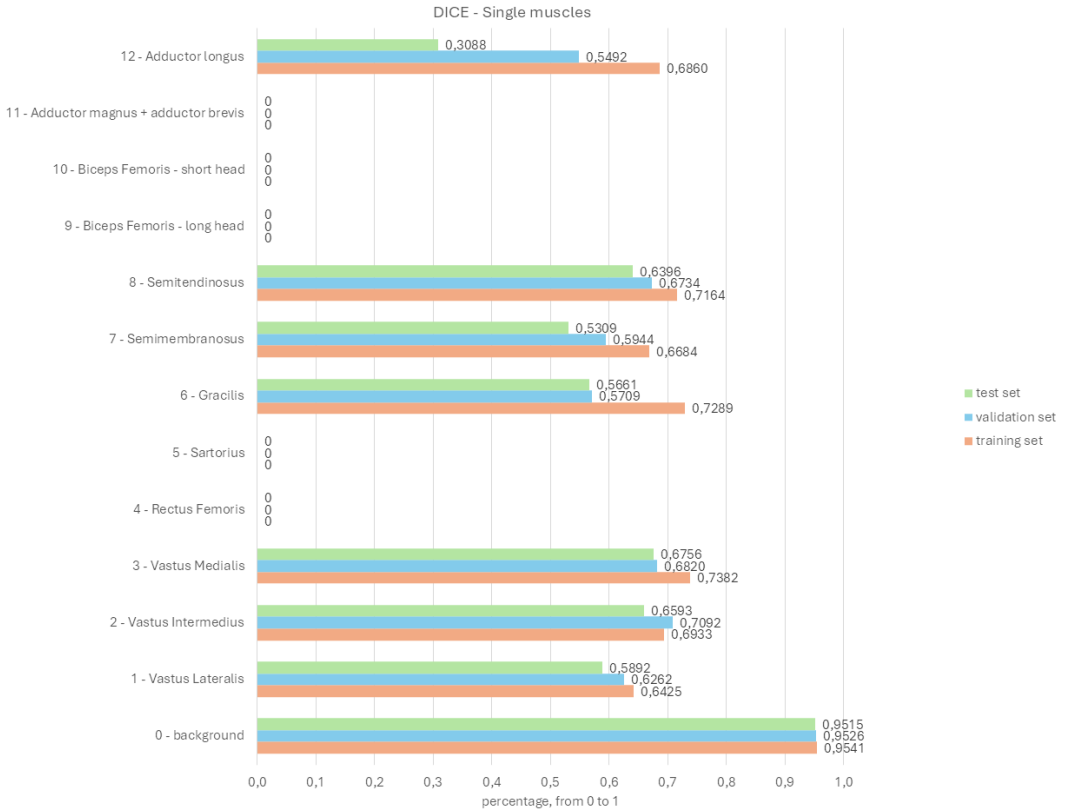

Figure 4.5: DICE of single muscles obtained with SwinUNETR 3D Neural Network with Generalized Dice Loss function for thigh

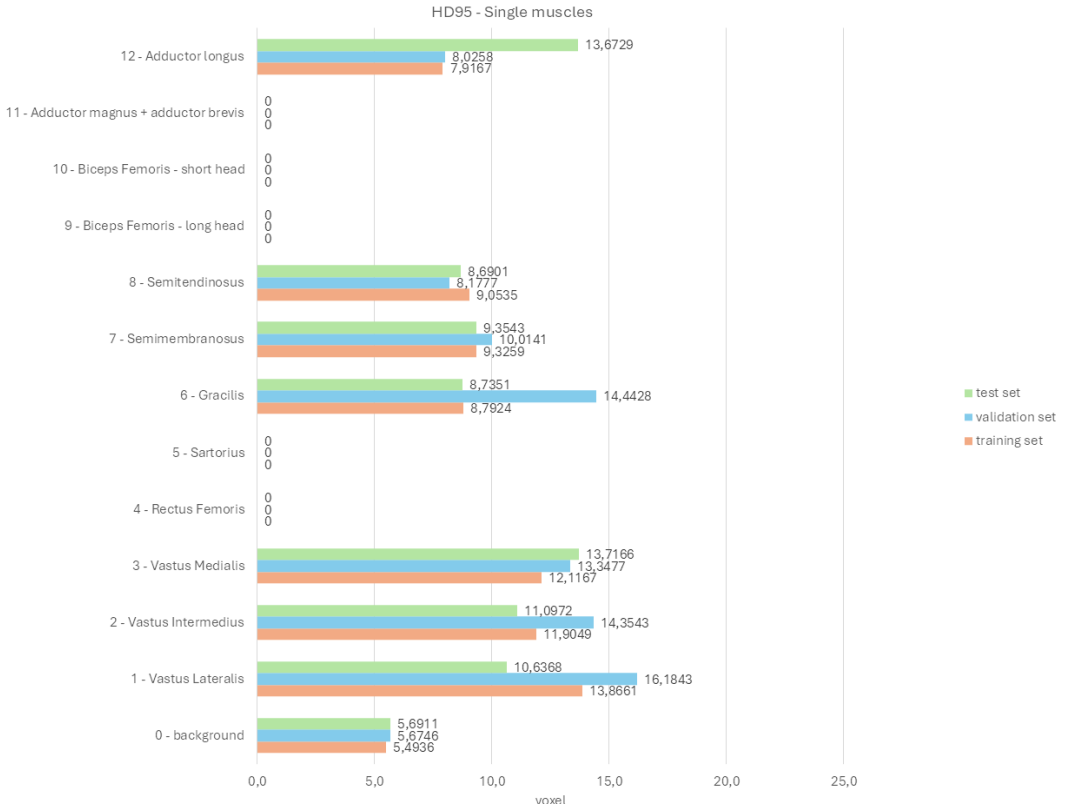

Figure 4.6: HD95 of single muscles obtained with SwinUNETR 3D Neural Network with Generalized Dice Loss function for thigh

- **3° test: SwinUNETR with Focal Loss function**
  In *Figure 4.7* graphs of DICE and HD95 obtained with SwinUNETR with Focal Loss function are represented: mean value and its standard deviation are reported for each set of volumes.
  The *Figure 4.8* graph shows the average DICE value obtained for each muscle with SwinUNETR with Focal Loss function, for each set of volumes.
  The *Figure 4.9* graph shows the average HD95 value obtained for each muscle with SwinUNETR with Focal Loss function, for each set of volumes.

Figure 4.7: Whole volume metrics obtained with SwinUNETR 3D Neural Network with Focal Loss function for thigh

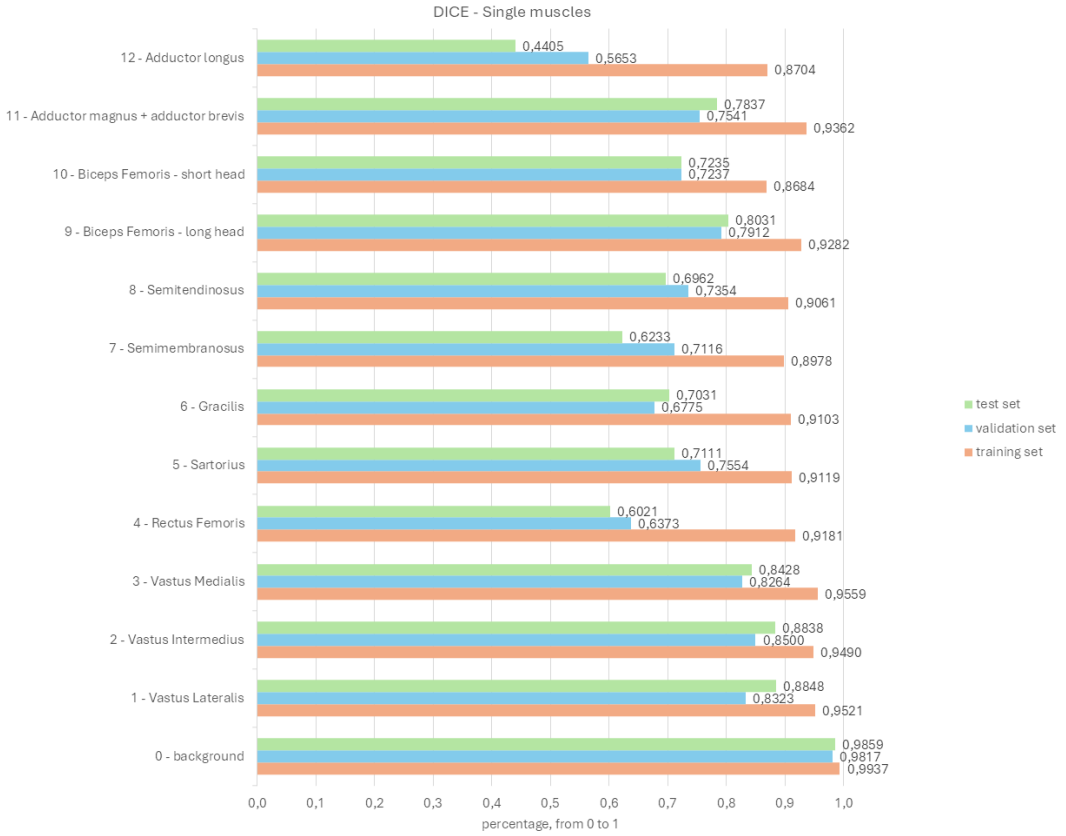

Figure 4.8: DICE of single muscles obtained with SwinUNETR 3D Neural Network with Focal Loss function for thigh

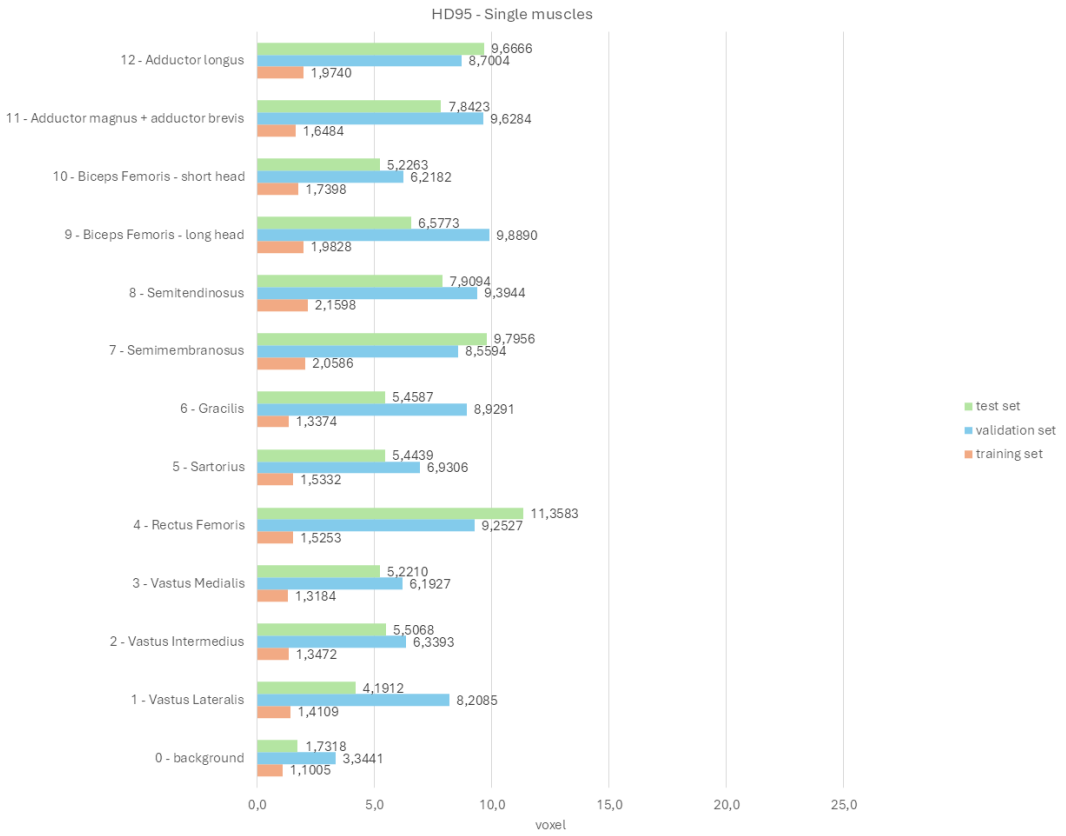

Figure 4.9: HD95 of single muscles obtained with SwinUNETR 3D Neural Network with Focal Loss function for thigh

- **4° test: SwinUNETR with Dice Focal Loss function**
  In *Figure 4.10* graphs of DICE and HD95 obtained with SwinUNETR with Focal Loss function are represented: mean value and its standard deviation are reported for each set of volumes.
  The *Figure 4.11* graph shows the average DICE value obtained for each muscle with SwinUNETR with Dice Focal Loss function, for each set of volumes.
  The *Figure 4.12* graph shows the average HD95 value obtained for each muscle with SwinUNETR with Dice Focal Loss function, for each set of volumes.

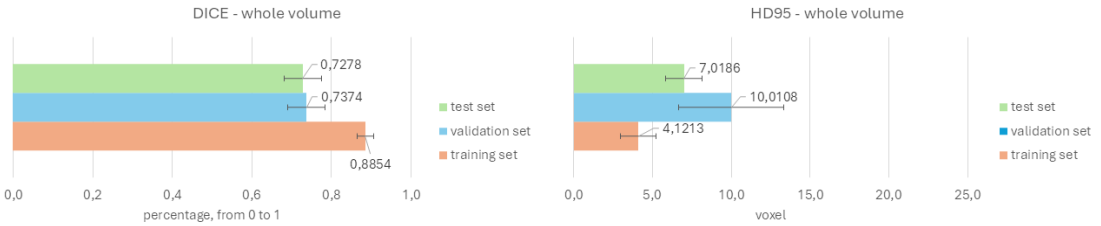

Figure 4.10: Whole volume metrics obtained with SwinUNETR 3D Neural Network with Dice Focal Loss function for thigh

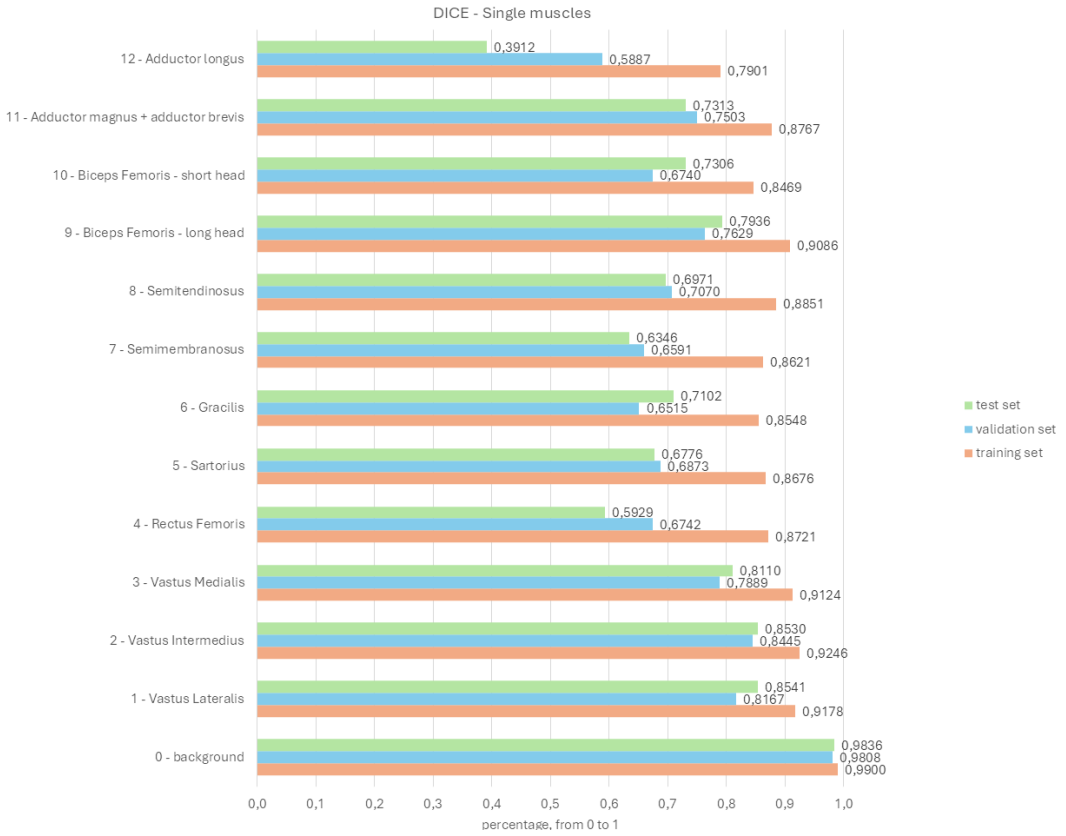

Figure 4.11: DICE of single muscles obtained with SwinUNETR 3D Neural Network with Dice Focal Loss function for thigh

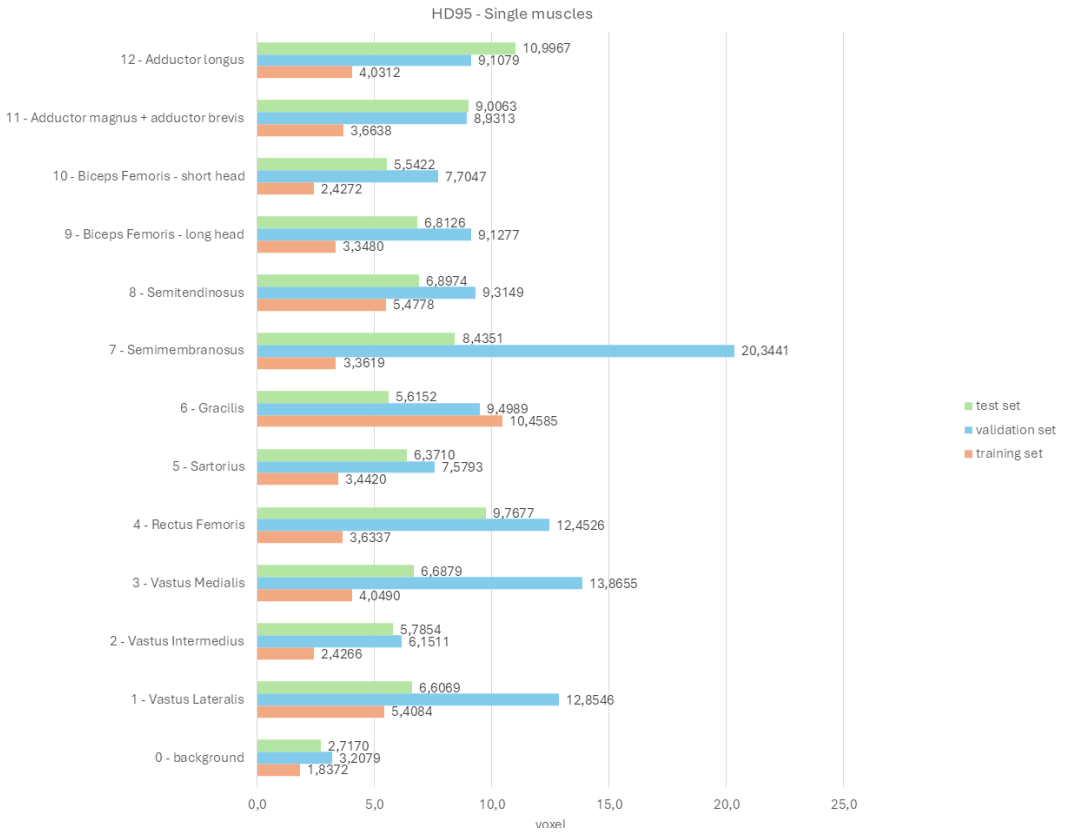

Figure 4.12: HD95 of single muscles obtained with SwinUNETR 3D Neural Network with Dice Focal Loss function for thigh

- **5° test: SwinUNETR with Generalized Dice Focal Loss function**
  In *Figure 4.13* graphs of DICE and HD95 obtained with SwinUNETR with Generalized Dice Focal Loss function are represented: mean value and its standard deviation are reported for each set of volumes.
  The *Figure 4.14* graph shows the average DICE value obtained for each muscle with SwinUNETR with Generalized Dice Focal Loss function, for each set of volumes.
  The *Figure 4.15* graph shows the average HD95 value obtained for each muscle with SwinUNETR with Generalized Dice Focal Loss function, for each set of volumes.

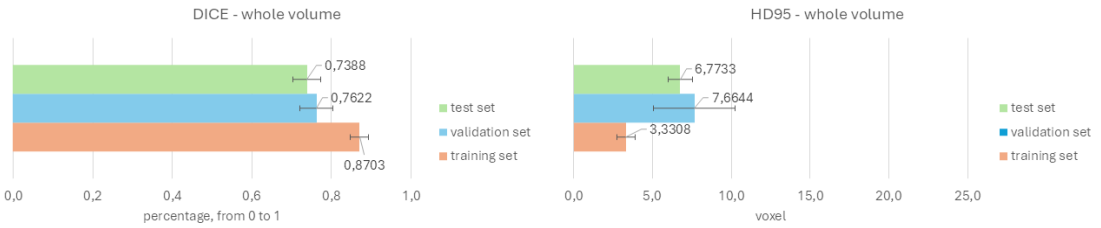

Figure 4.13: Whole volume metrics obtained with SwinUNETR 3D Neural Network with Generalized Dice Focal Loss function for thigh

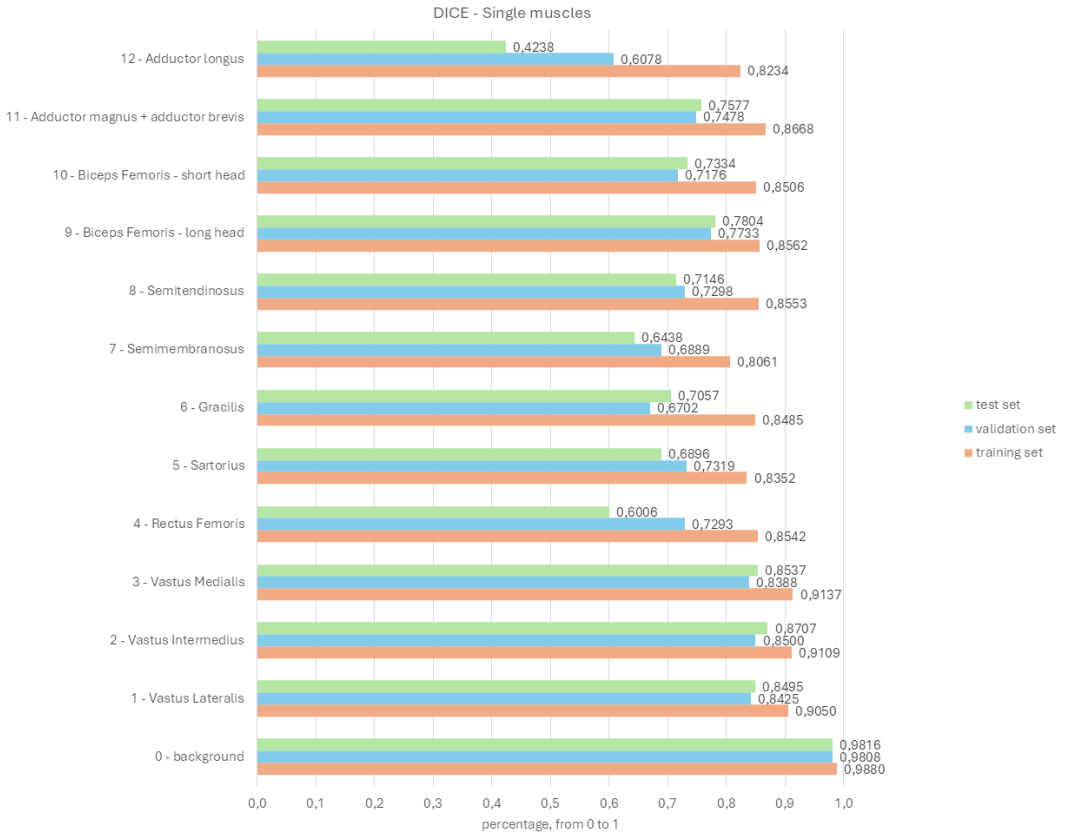

Figure 4.14: DICE of single muscles obtained with SwinUNETR 3D Neural Network with Generalized Dice Focal Loss function for thigh

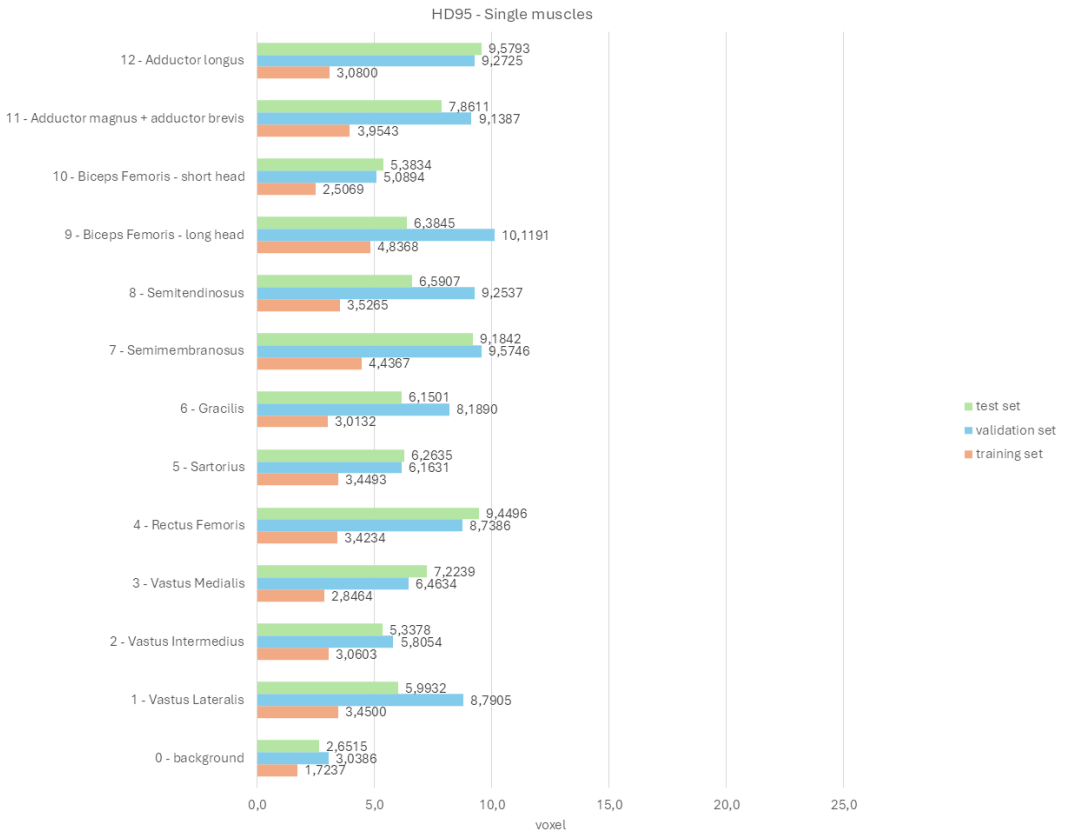

Figure 4.15: HD95 of single muscles obtained with SwinUNETR 3D Neural Network with Generalized Dice Focal Loss function for thigh

Second division of Dataset

In this section, graphs obtained from the second division of the dataset are reported, dividing them according to the loss function chosen.

- **1° test: SwinUNETR with Dice Cross Entropy Loss function**
  In *Figure 4.16* graphs of DICE and HD95 obtained with SwinUNETR with Dice Cross Entropy Loss function are represented: mean value and its standard deviation are reported for each set of volumes.
  The *Figure 4.17* graph shows the average DICE value obtained for each muscle with SwinUNETR with Dice Cross Entropy Loss function, for each set of volumes.
  The *Figure 4.18* graph shows the average HD95 value obtained for each muscle with SwinUNETR with Dice Cross Entropy Loss function, for each set of volumes.

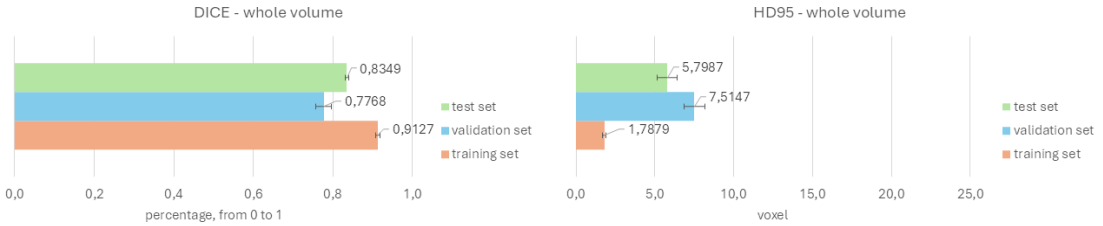

Figure 4.16: Whole volume metrics obtained with SwinUNETR 3D Neural Network with Dice Cross Entropy Loss function for thigh

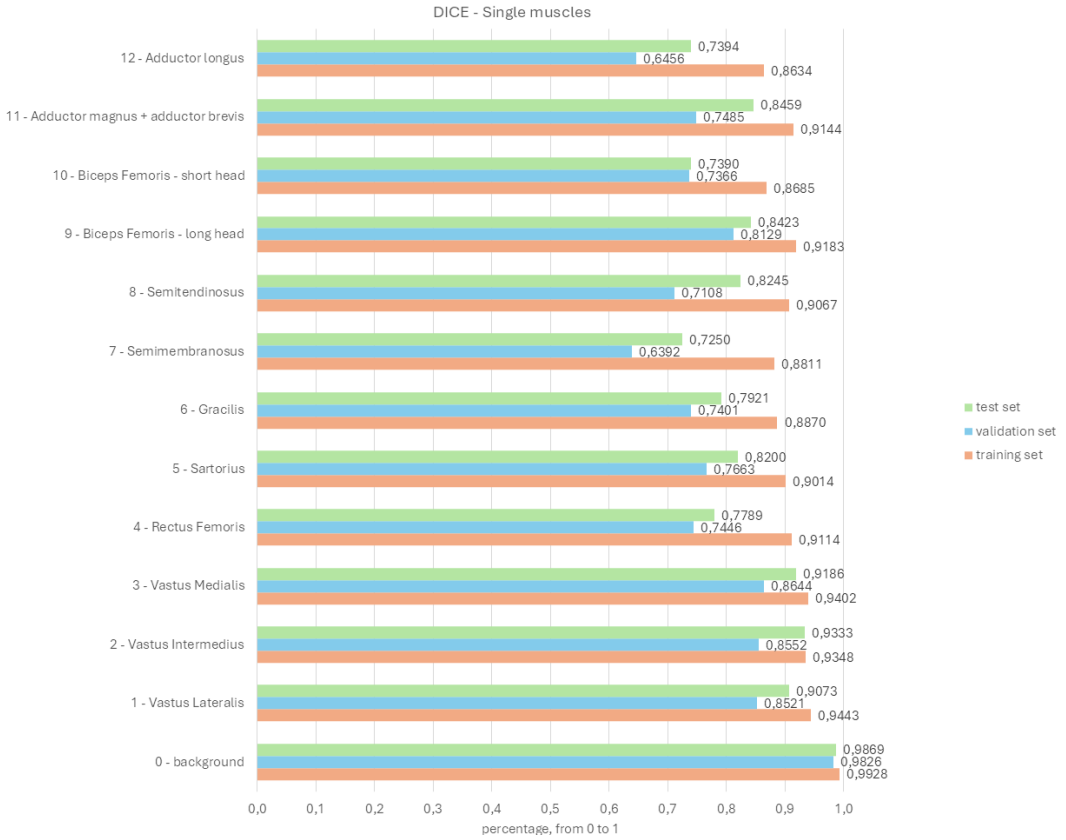

Figure 4.17: DICE of single muscles obtained with SwinUNETR 3D Neural Network with Dice Cross Entropy Loss function for thigh

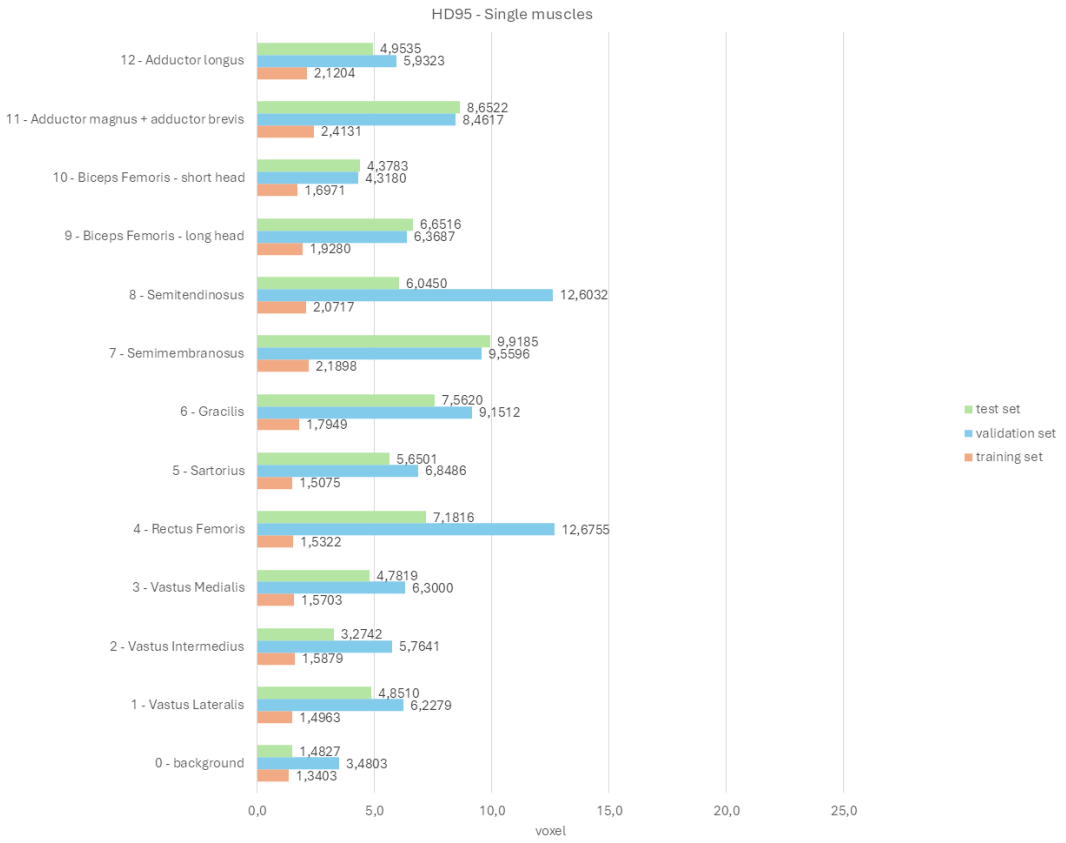

Figure 4.18: HD95 of single muscles obtained with SwinUNETR 3D Neural Network with Dice Cross Entropy Loss function for thigh

- **2° test: SwinUNETR with Generalized Dice Loss function**
  In *Figure 4.19* graphs of DICE and HD95 obtained with SwinUNETR with Generalized Dice Loss function are represented: mean value and its standard deviation are reported for each set of volumes.
  The *Figure 4.20* graph shows the average DICE value obtained for each muscle with SwinUNETR with Generalized Dice Loss function, for each set of volumes.
  The *Figure 4.21* graph shows the average HD95 value obtained for each muscle with SwinUNETR with Generalized Dice Loss function, for each set of volumes.

Figure 4.19: Whole volume metrics obtained with SwinUNETR 3D Neural Network with Generalized Dice Loss function for thigh

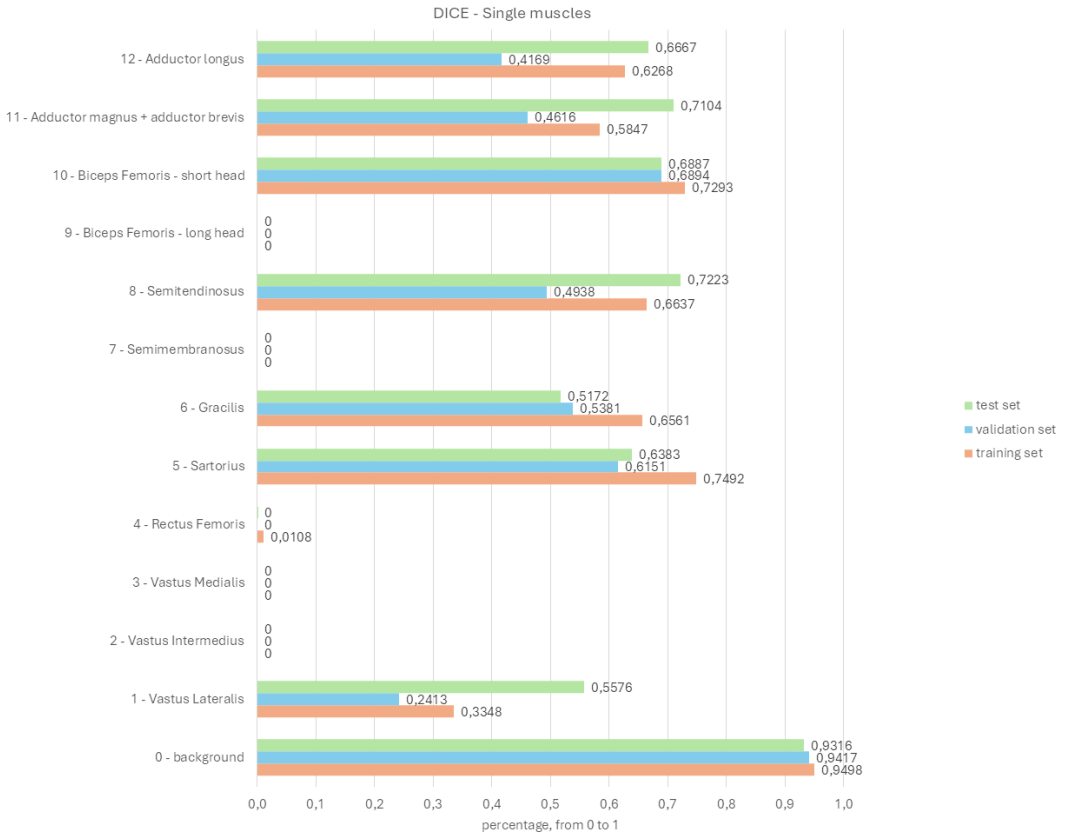

Figure 4.20: DICE of single muscles obtained with SwinUNETR 3D Neural Network with Generalized Dice Loss function for thigh

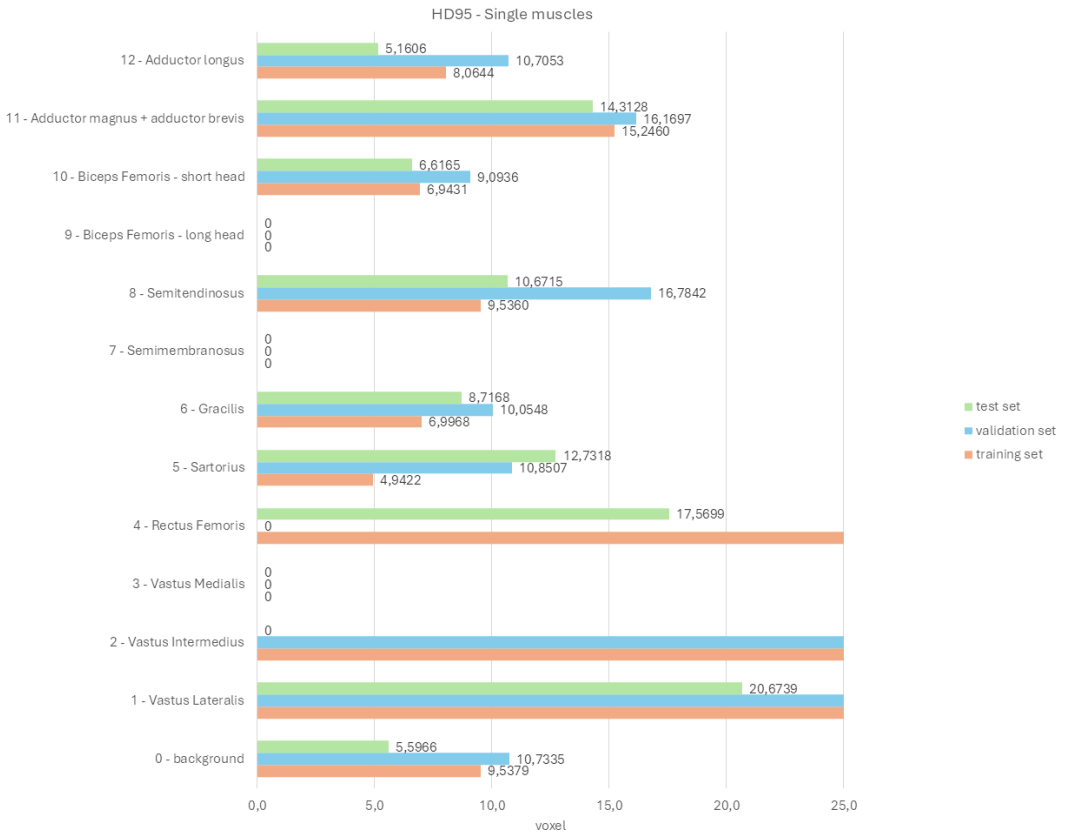

Figure 4.21: HD95 of single muscles obtained with SwinUNETR 3D Neural Network with Generalized Dice Loss function for thigh

- **3° test: SwinUNETR with Focal Loss function**
  In *Figure 4.22* graphs of DICE and HD95 obtained with SwinUNETR with Focal Loss function are represented: mean value and its standard deviation are reported for each set of volumes.
  The *Figure 4.23* graph shows the average DICE value obtained for each muscle with SwinUNETR with Focal Loss function, for each set of volumes.
  The *Figure 4.24* graph shows the average HD95 value obtained for each muscle with SwinUNETR with Focal Loss function, for each set of volumes.

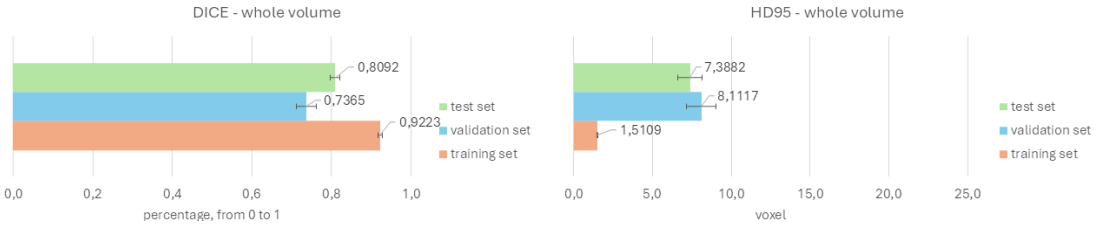

Figure 4.22: Whole volume metrics obtained with SwinUNETR 3D Neural Network with Focal Loss function for thigh

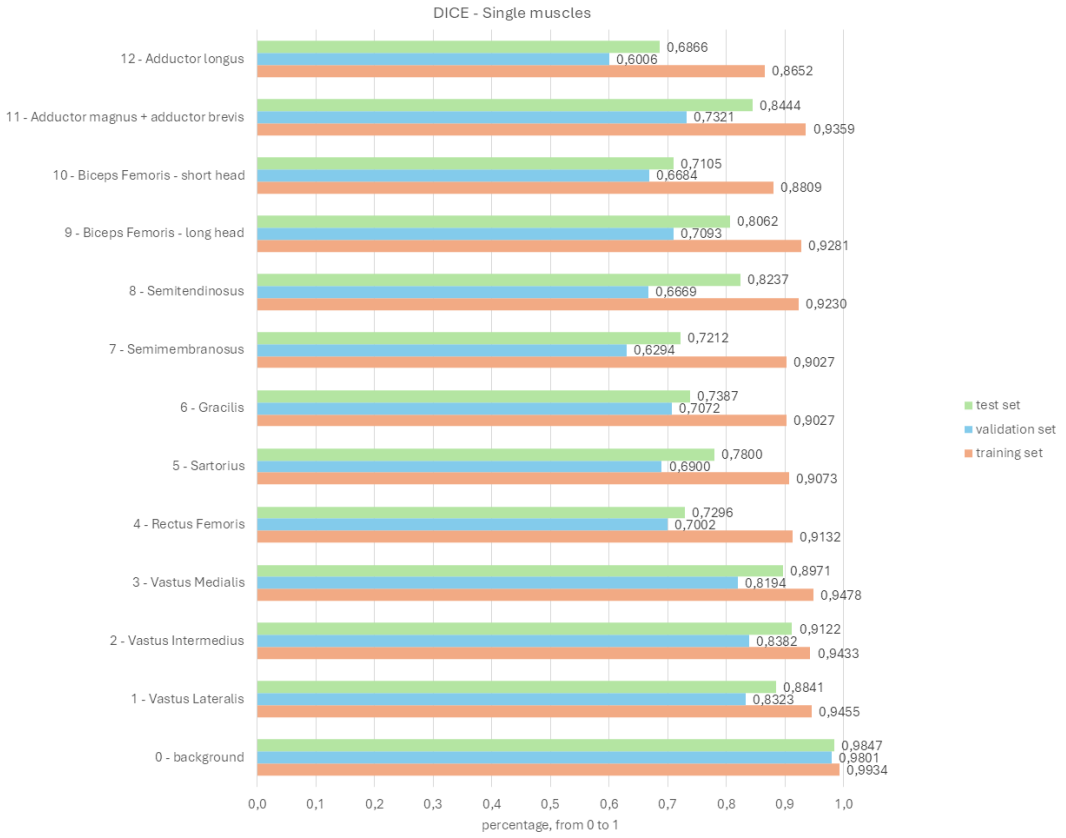

Figure 4.23: DICE of single muscles obtained with SwinUNETR 3D Neural Network with Focal Loss function for thigh

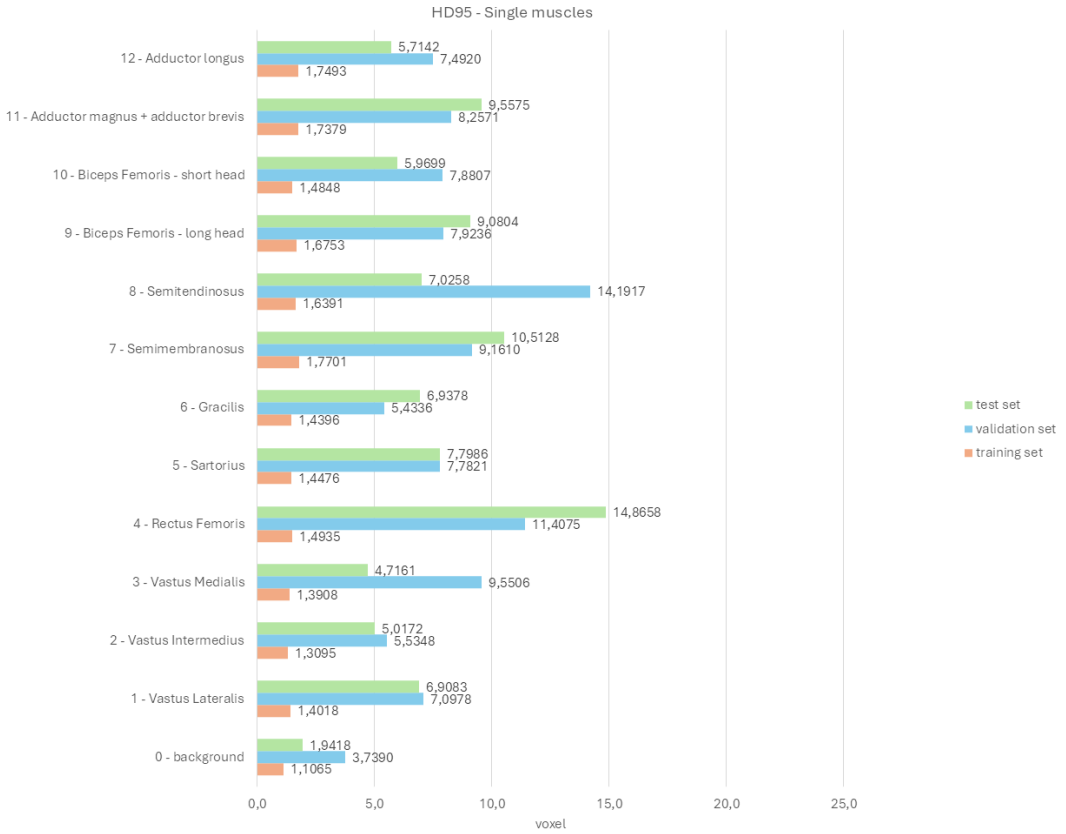

Figure 4.24: HD95 of single muscles obtained with SwinUNETR 3D Neural Network with Focal Loss function for thigh

- **4° test: SwinUNETR with Dice Focal Loss function**
  In *Figure 4.25* graphs of DICE and HD95 obtained with SwinUNETR with Focal Loss function are represented: mean value and its standard deviation are reported for each set of volumes.
  The *Figure 4.26* graph shows the average DICE value obtained for each muscle with SwinUNETR with Dice Focal Loss function, for each set of volumes.
  The *Figure 4.27* graph shows the average HD95 value obtained for each muscle with SwinUNETR with Dice Focal Loss function, for each set of volumes.

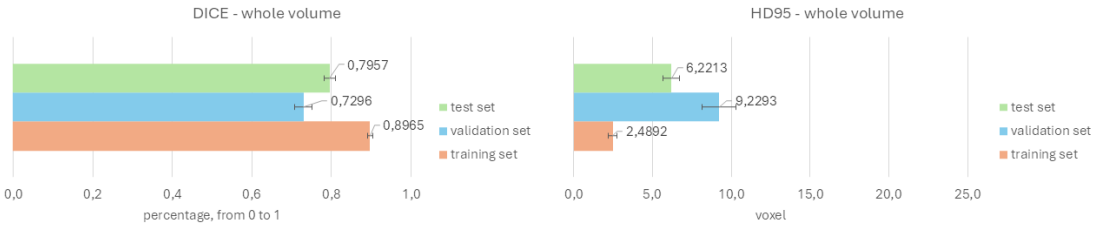

Figure 4.25: Whole volume metrics obtained with SwinUNETR 3D Neural Network with Dice Focal Loss function for thigh

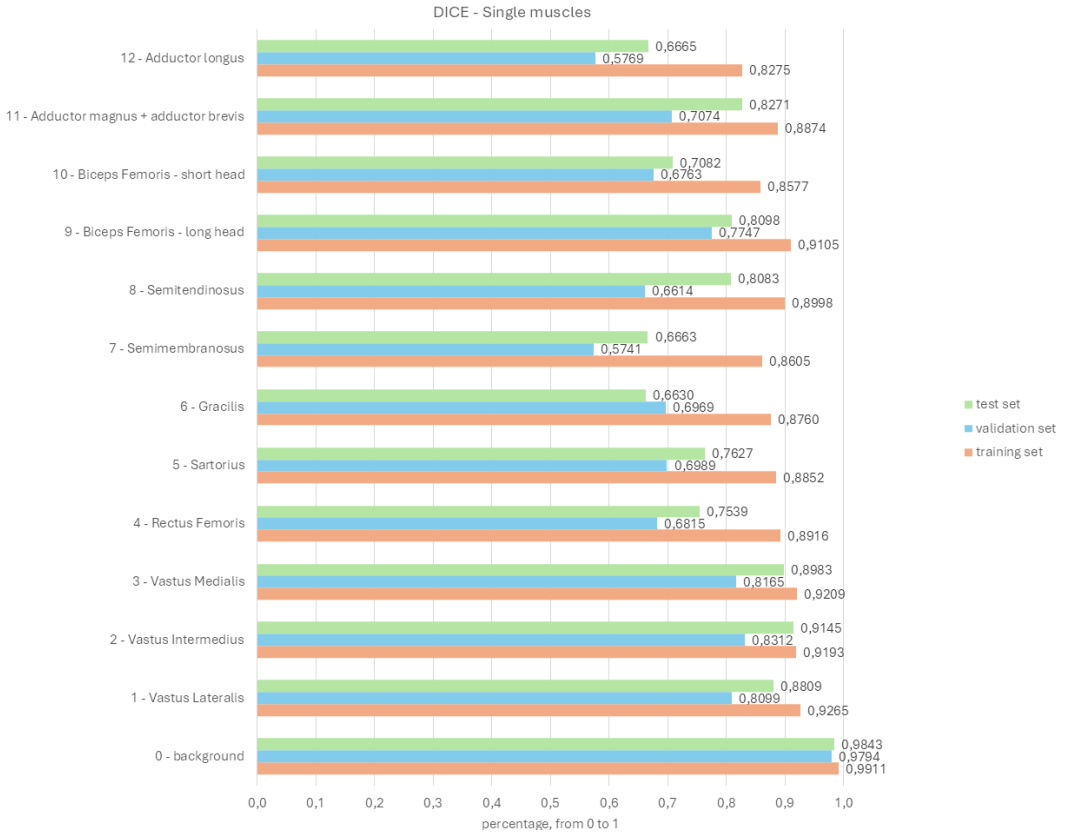

Figure 4.26: DICE of single muscles obtained with SwinUNETR 3D Neural Network with Dice Focal Loss function for thigh

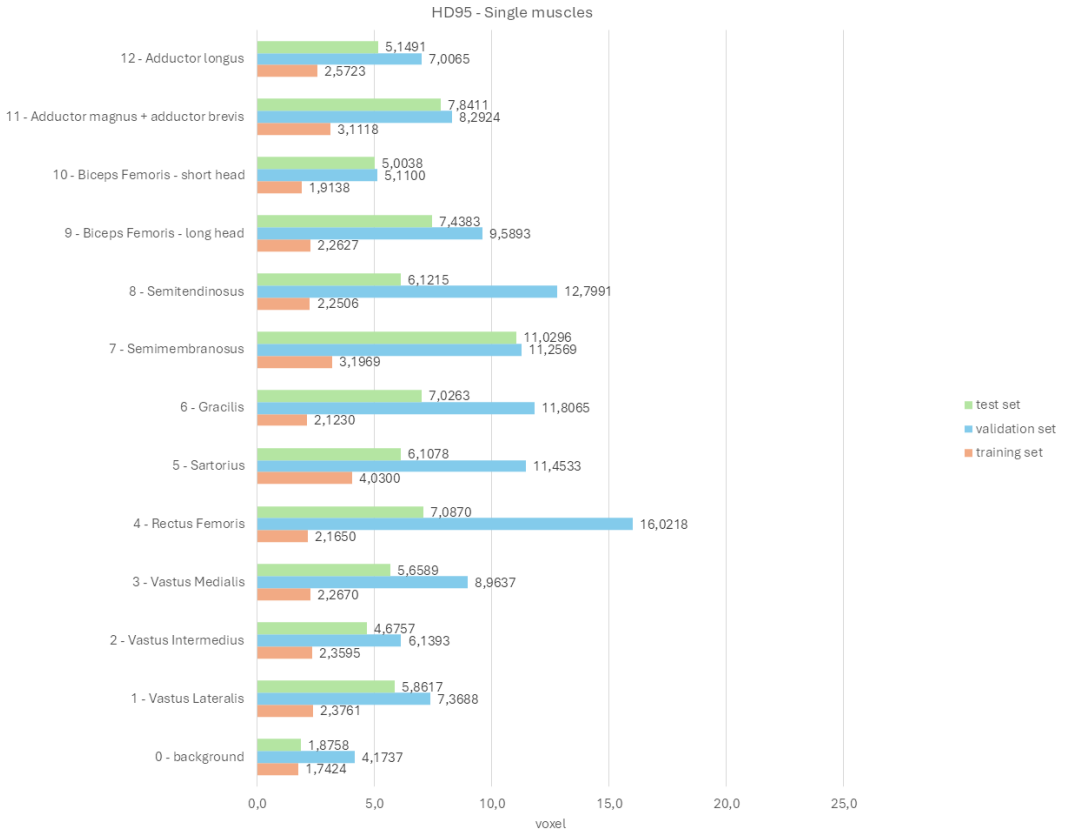

Figure 4.27: HD95 of single muscles obtained with SwinUNETR 3D Neural Network with Dice Focal Loss function for thigh

- **5° test: SwinUNETR with Generalized Dice Focal Loss function**
  In *Figure 4.28* graphs of DICE and HD95 obtained with SwinUNETR with Generalized Dice Focal Loss function are represented: mean value and its standard deviation are reported for each set of volumes.
  The *Figure 4.29* graph shows the average DICE value obtained for each muscle with SwinUNETR with Generalized Dice Focal Loss function, for each set of volumes.
  The *Figure 4.30* graph shows the average HD95 value obtained for each muscle with SwinUNETR with Generalized Dice Focal Loss function, for each set of volumes.

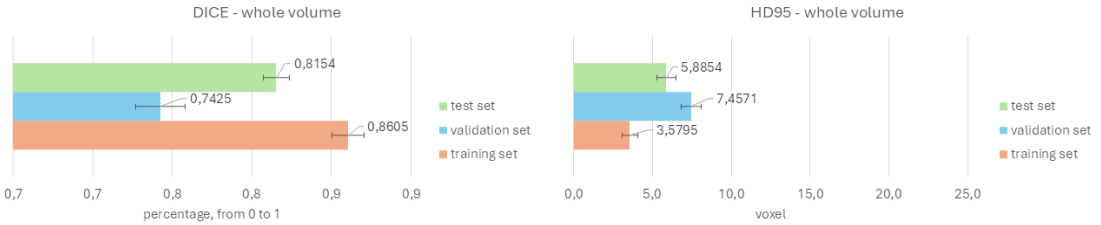

Figure 4.28: Whole volume metrics obtained with SwinUNETR 3D Neural Network with Generalized Dice Focal Loss function for thigh

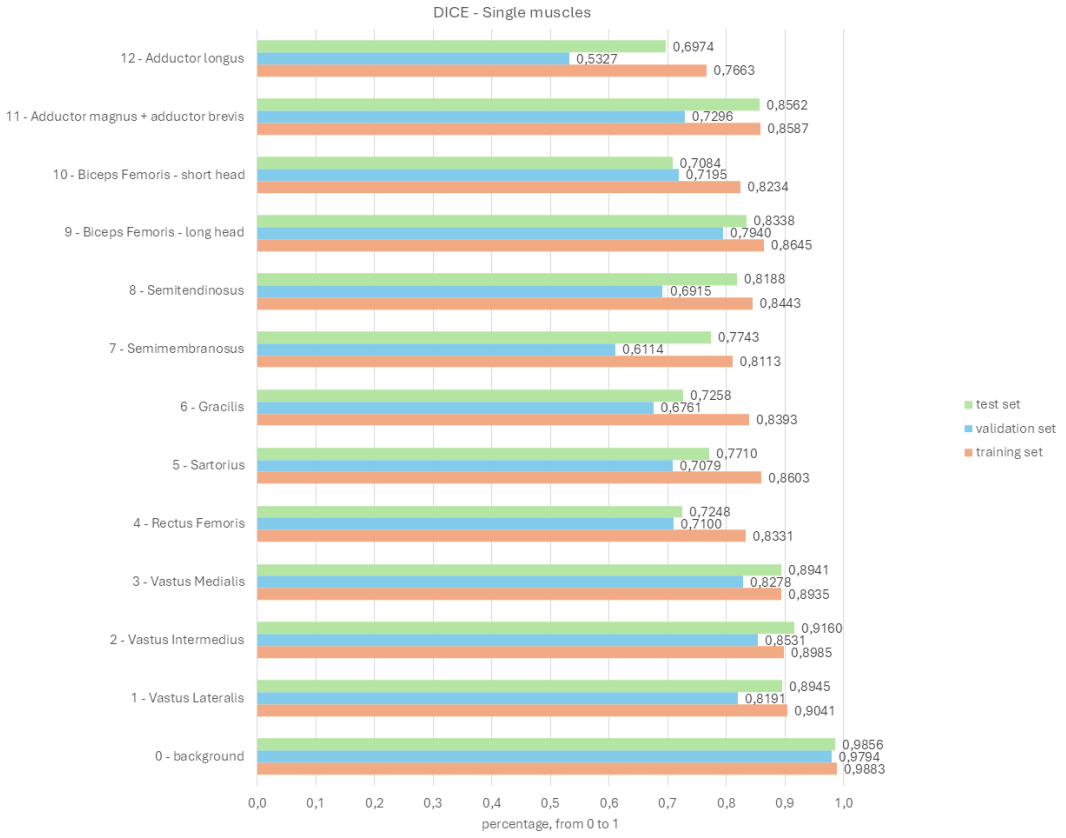

Figure 4.29: DICE of single muscles obtained with SwinUNETR 3D Neural Network with Generalized Dice Focal Loss function for thigh

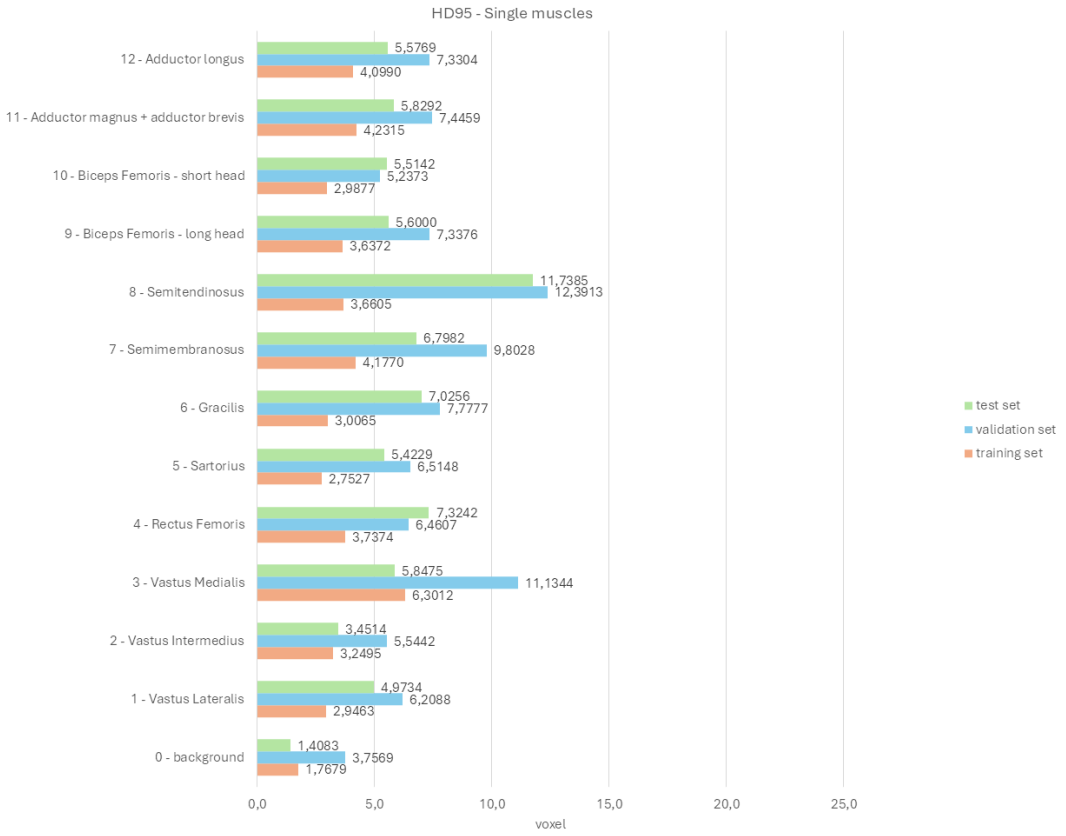

Figure 4.30: HD95 of single muscles obtained with SwinUNETR 3D Neural Network with Generalized Dice Focal Loss function for thigh

4.1.2 SwinUNETR 3D Neural Network for leg

For each 3D Neural Network for leg tested, whole volume metric graphs and single muscle metric graphs are reported.
The following graphs show the whole volume metric calculated for each set. Each bar represent a set: the orange bar indicates metrics of training set, the blue bar indicates metrics of validation set and the green bar indicates metrics of test set.

First division of Dataset

In this section graphs obtained from the first division of the dataset are reported, dividing them according to the loss function chosen.

- **1° test: SwinUNETR with Dice Cross Entropy Loss function**
  In *Figure 4.31* graphs of DICE and HD95 obtained with SwinUNETR with Dice Cross Entropy Loss function are represented: mean value and its standard

deviation are reported for each set of volumes.
The *Figure 4.32* graph shows the average DICE value obtained for each muscle with SwinUNETR with Dice Cross Entropy Loss function, for each set of volumes.
The *Figure 4.33* graph shows the average HD95 value obtained for each muscle with SwinUNETR with Dice Cross Entropy Loss function, for each set of volumes.

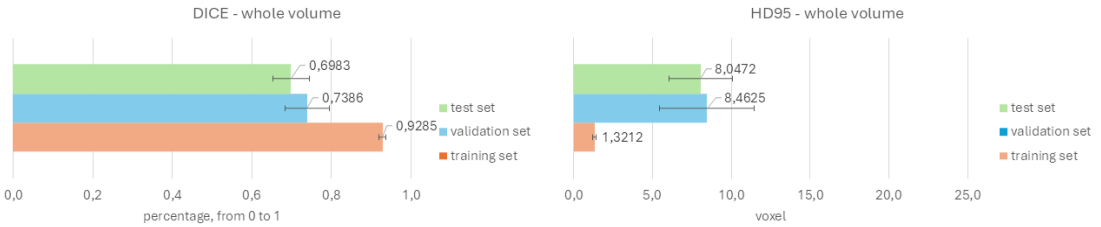

Figure 4.31: Whole volume metrics obtained with SwinUNETR 3D Neural Network with Dice Cross Entropy Loss function for leg

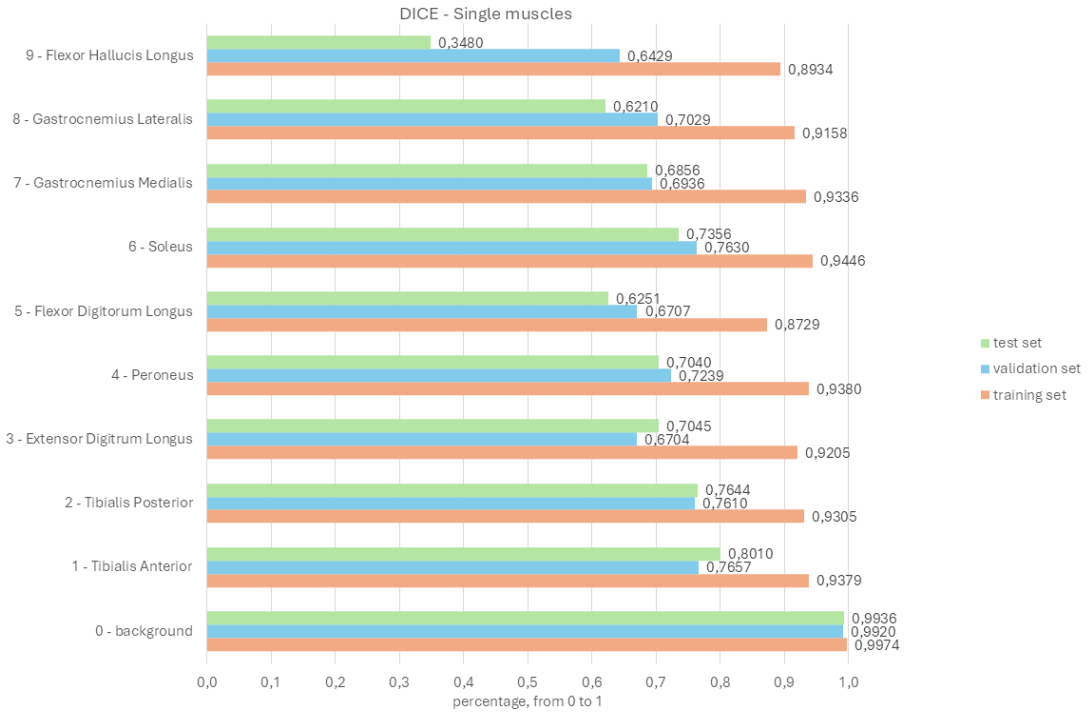

Figure 4.32: DICE of single muscles obtained with SwinUNETR 3D Neural Network with Dice Cross Entropy Loss function for leg

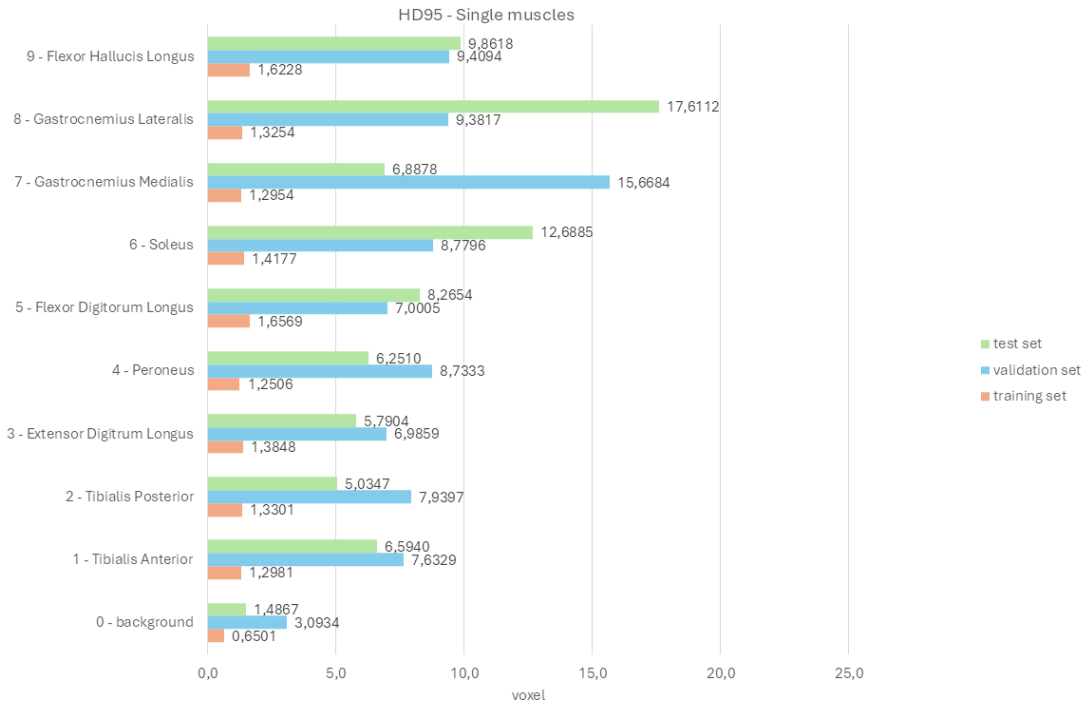

Figure 4.33: HD95 of single muscles obtained with SwinUNETR 3D Neural Network with Dice Cross Entropy Loss function for leg

- **2° test: SwinUNETR with Generalized Dice Loss function**
  In *Figure 4.34* graphs of DICE and HD95 obtained with SwinUNETR with Generalized Dice Loss function are represented: mean value and its standard deviation are reported for each set of volumes.
  The *Figure 4.35* graph shows the average DICE value obtained for each muscle with SwinUNETR with Generalized Dice Loss function, for each set of volumes.
  The *Figure 4.36* graph shows the average HD95 value obtained for each muscle with SwinUNETR with Generalized Dice Loss function, for each set of volumes.

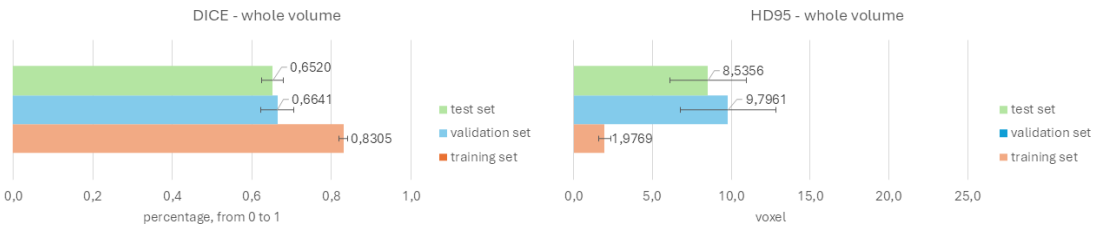

Figure 4.34: Whole volume metrics obtained with SwinUNETR 3D Neural Network with Generalized Dice Loss function for leg

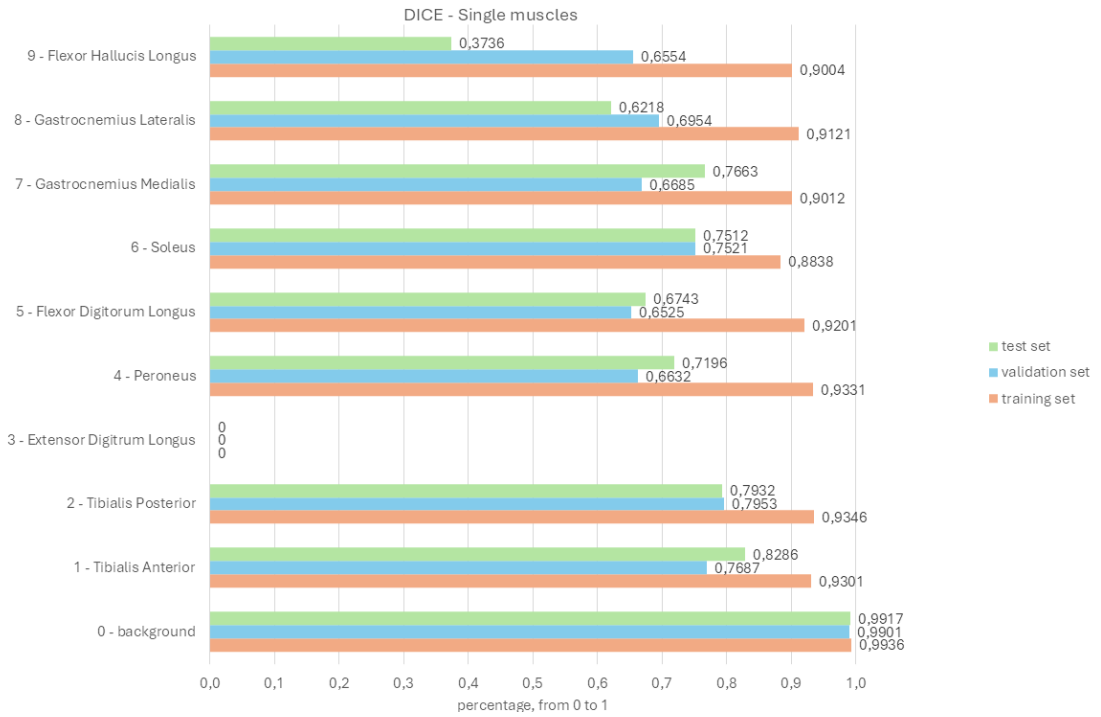

Figure 4.35: DICE of single muscles obtained with SwinUNETR 3D Neural Network with Generalized Dice Loss function for leg

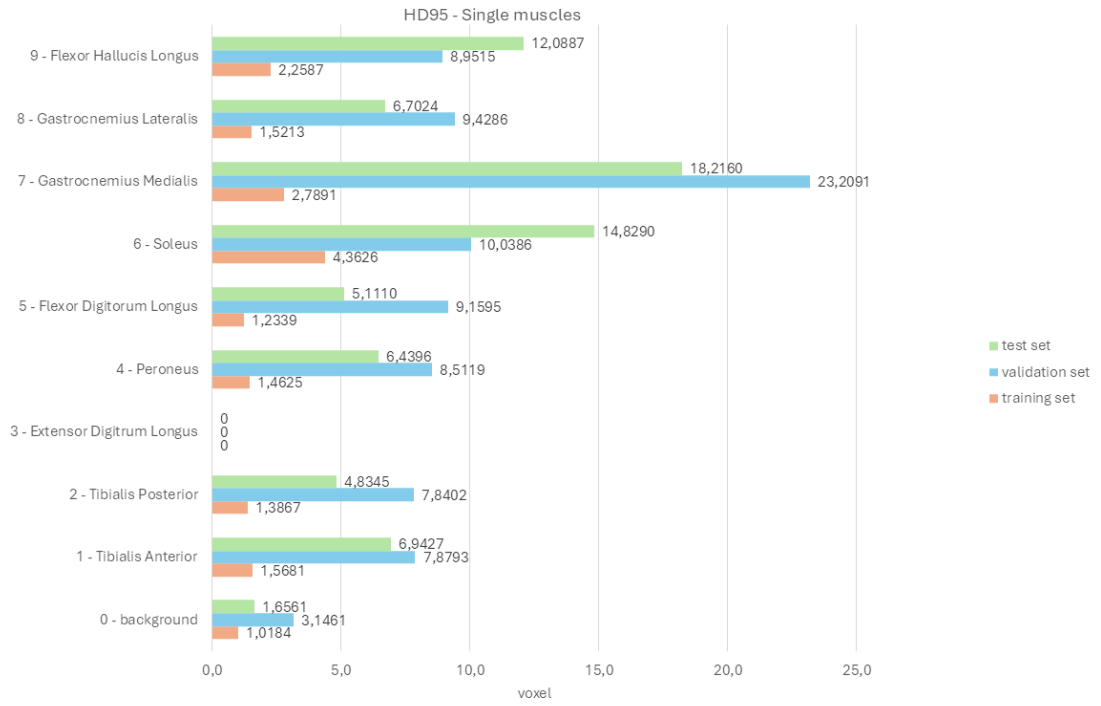

Figure 4.36: HD95 of single muscles obtained with SwinUNETR 3D Neural Network with Generalized Dice Loss function for leg

- **3° test: SwinUNETR with Focal Loss function**
  In *Figure 4.37* graphs of DICE and HD95 obtained with SwinUNETR with Focal Loss function are represented: mean value and its standard deviation are reported for each set of volumes.
  The *Figure 4.38* graph shows the average DICE value obtained for each muscle with SwinUNETR with Focal Loss function, for each set of volumes.
  The *Figure 4.39* graph shows the average HD95 value obtained for each muscle with SwinUNETR with Focal Loss function, for each set of volumes.

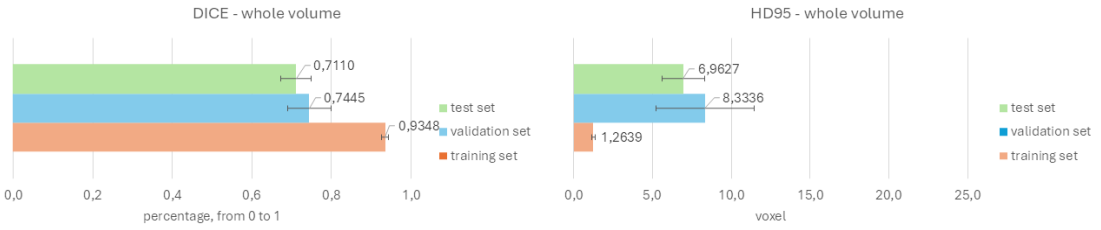

Figure 4.37: Whole volume metrics obtained with SwinUNETR 3D Neural Network with Focal Loss function for leg

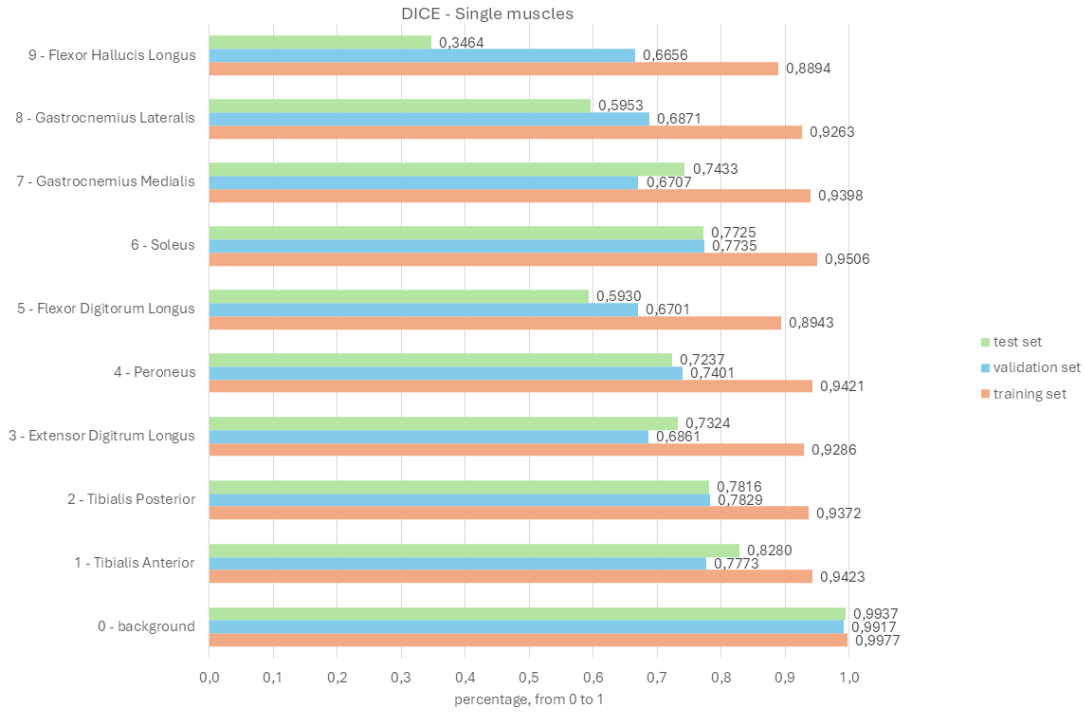

Figure 4.38: DICE of single muscles obtained with SwinUNETR 3D Neural Network with Focal Loss function for leg

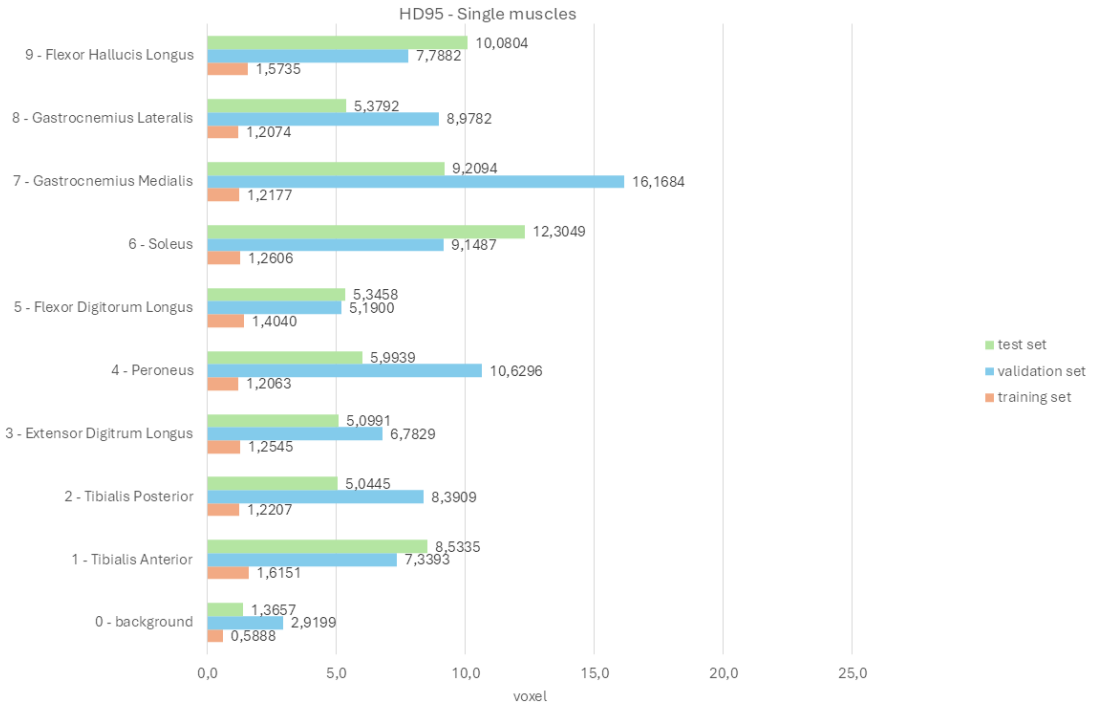

Figure 4.39: HD95 of single muscles obtained with SwinUNETR 3D Neural Network with Focal Loss function for leg

- **4° test: SwinUNETR with Dice Focal Loss function**
  In *Figure 4.40* graphs of DICE and HD95 obtained with SwinUNETR with Focal Loss function are represented: mean value and its standard deviation are reported for each set of volumes.
  The *Figure 4.41* graph shows the average DICE value obtained for each muscle with SwinUNETR with Dice Focal Loss function, for each set of volumes.
  The *Figure 4.42* graph shows the average HD95 value obtained for each muscle with SwinUNETR with Dice Focal Loss function, for each set of volumes.

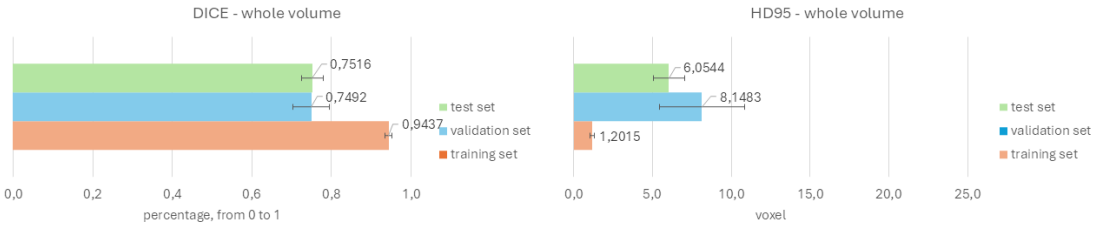

Figure 4.40: Whole volume metrics obtained with SwinUNETR 3D Neural Network with Dice Focal Loss function for leg

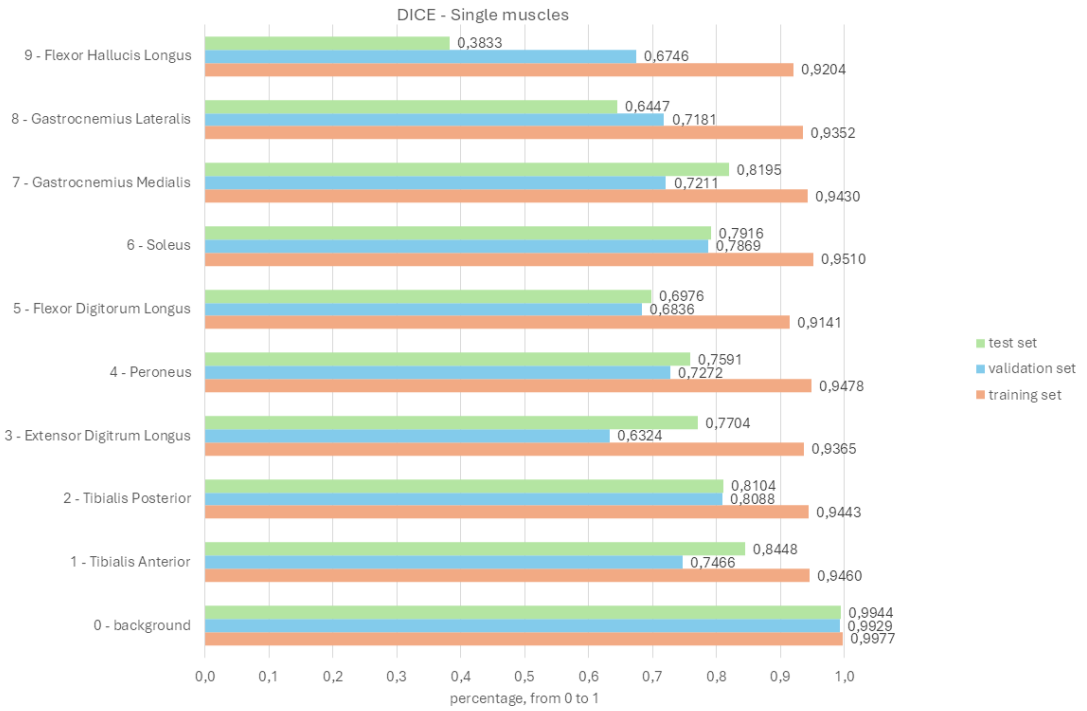

Figure 4.41: DICE of single muscles obtained with SwinUNETR 3D Neural Network with Dice Focal Loss function for leg

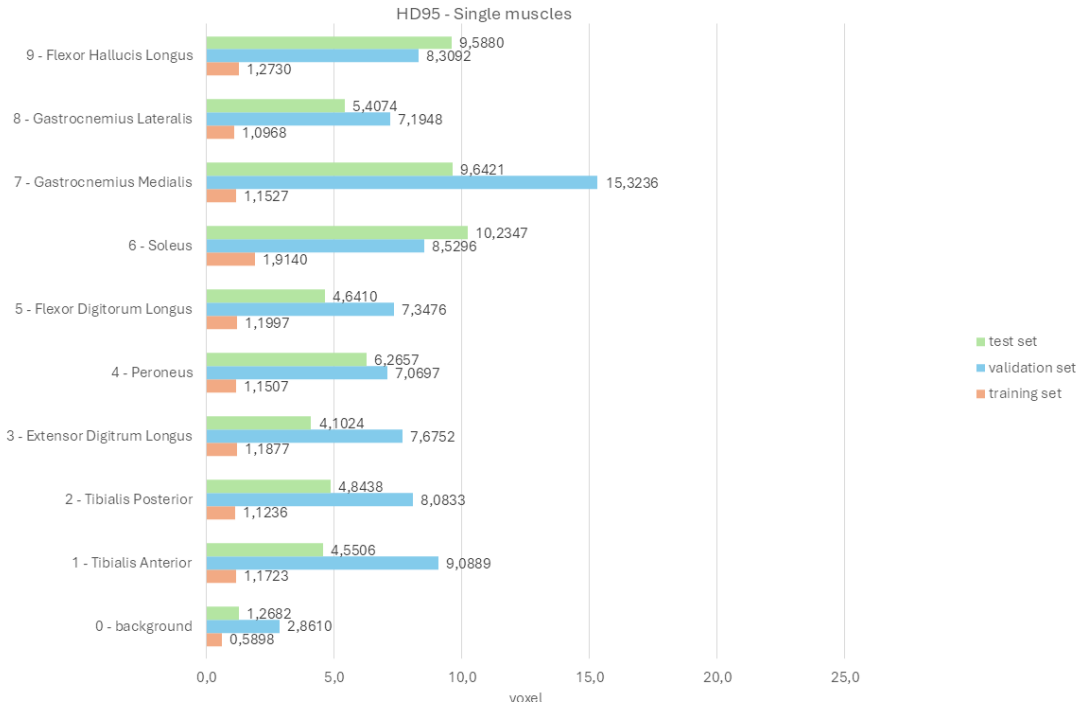

Figure 4.42: HD95 of single muscles obtained with SwinUNETR 3D Neural Network with Dice Focal Loss function for leg

- **5° test: SwinUNETR with Generalized Dice Focal Loss function**
  In *Figure 4.43* graphs of DICE and HD95 obtained with SwinUNETR with Generalized Dice Focal Loss function are represented: mean value and its standard deviation are reported for each set of volumes.
  The *Figure 4.44* graph shows the average DICE value obtained for each muscle with SwinUNETR with Generalized Dice Focal Loss function, for each set of volumes.
  The *Figure 4.45* graph shows the average HD95 value obtained for each muscle with SwinUNETR with Generalized Dice Focal Loss function, for each set of volumes.

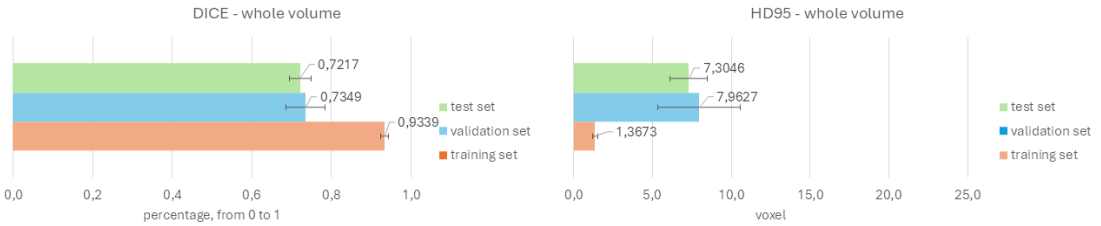

Figure 4.43: Whole volume metrics obtained with SwinUNETR 3D Neural Network with Generalized Dice Focal Loss function for leg

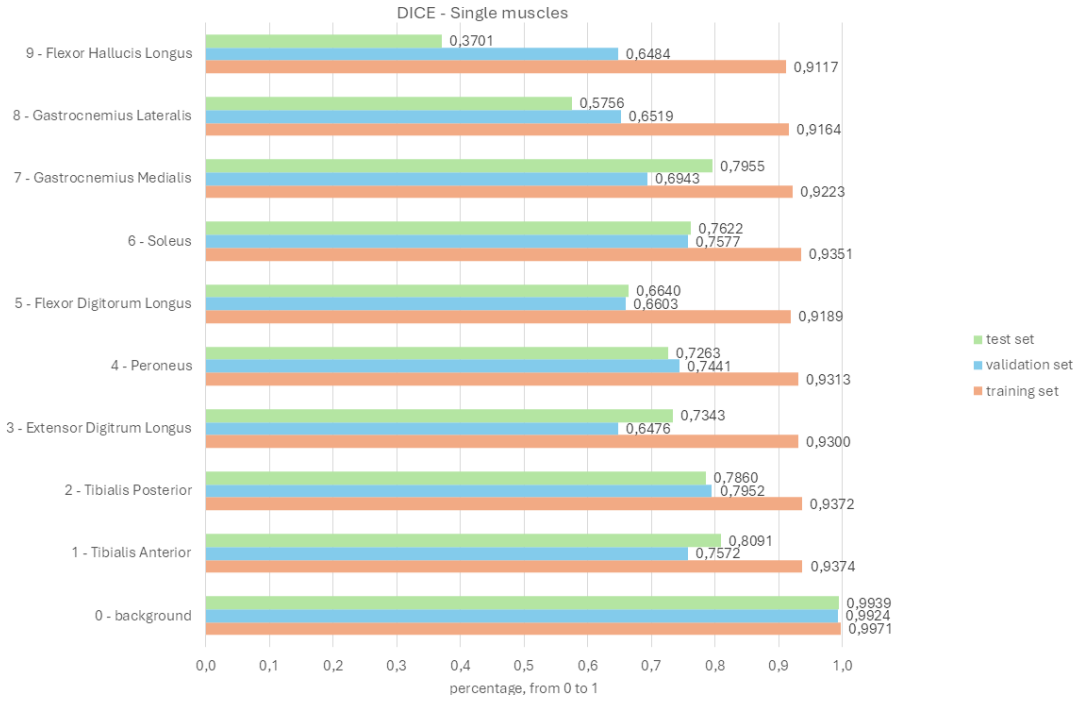

Figure 4.44: DICE of single muscles obtained with SwinUNETR 3D Neural Network with Generalized Dice Focal Loss function for leg

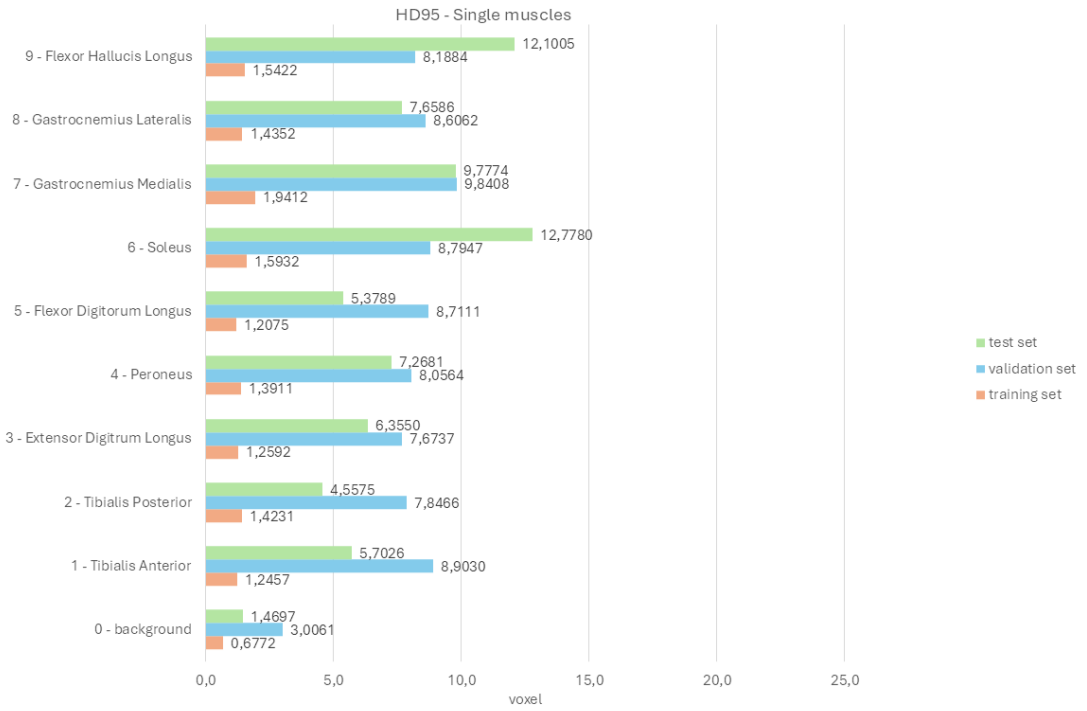

Figure 4.45: HD95 of single muscles obtained with SwinUNETR 3D Neural Network with Generalized Dice Focal Loss function for leg

Second division of Dataset

In this section graphs obtained from the second division of the dataset are reported, dividing them according to the loss function chosen.

- **1° test: SwinUNETR with Dice Cross Entropy Loss function**
  In *Figure 4.46* graphs of DICE and HD95 obtained with SwinUNETR with Dice Cross Entropy Loss function are represented: mean value and its standard deviation are reported for each set of volumes.
  The *Figure 4.47* graph shows the average DICE value obtained for each muscle with SwinUNETR with Dice Cross Entropy Loss function, for each set of volumes.
  The *Figure 4.48* graph shows the average HD95 value obtained for each muscle with SwinUNETR with Dice Cross Entropy Loss function, for each set of volumes.

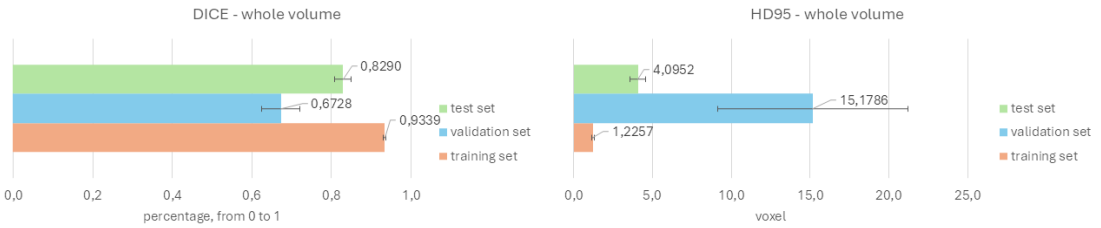

Figure 4.46: Whole volume metrics obtained with SwinUNETR 3D Neural Network with Dice Cross Entropy Loss function for leg

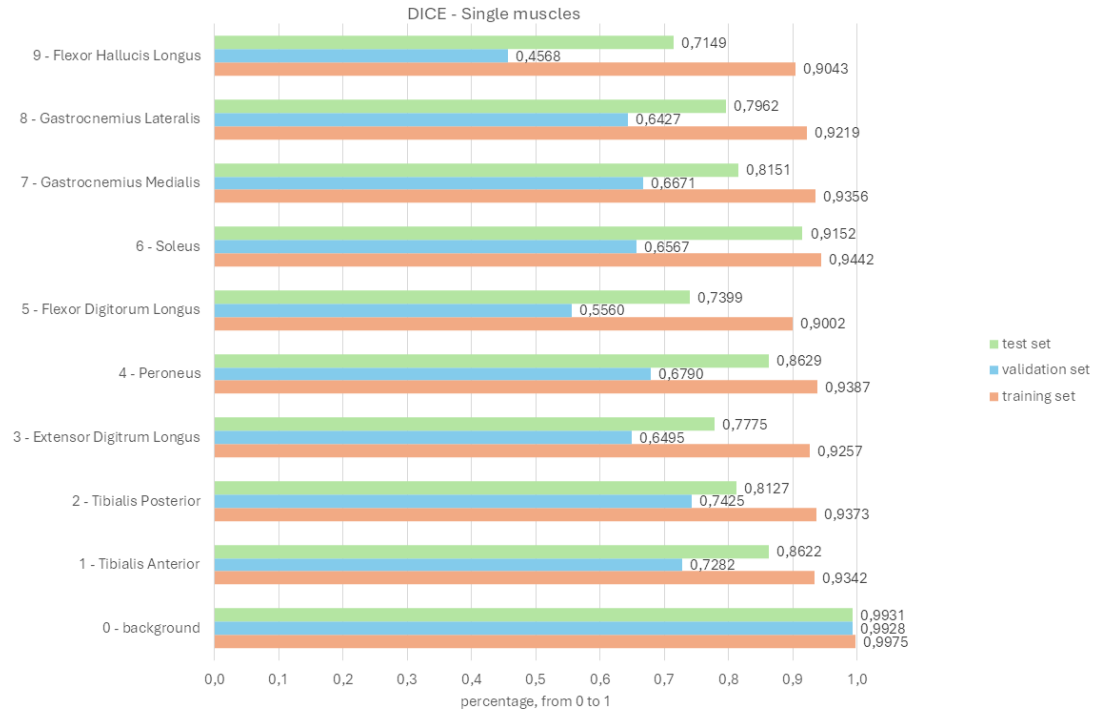

Figure 4.47: DICE of single muscles obtained with SwinUNETR 3D Neural Network with Dice Cross Entropy Loss function for leg

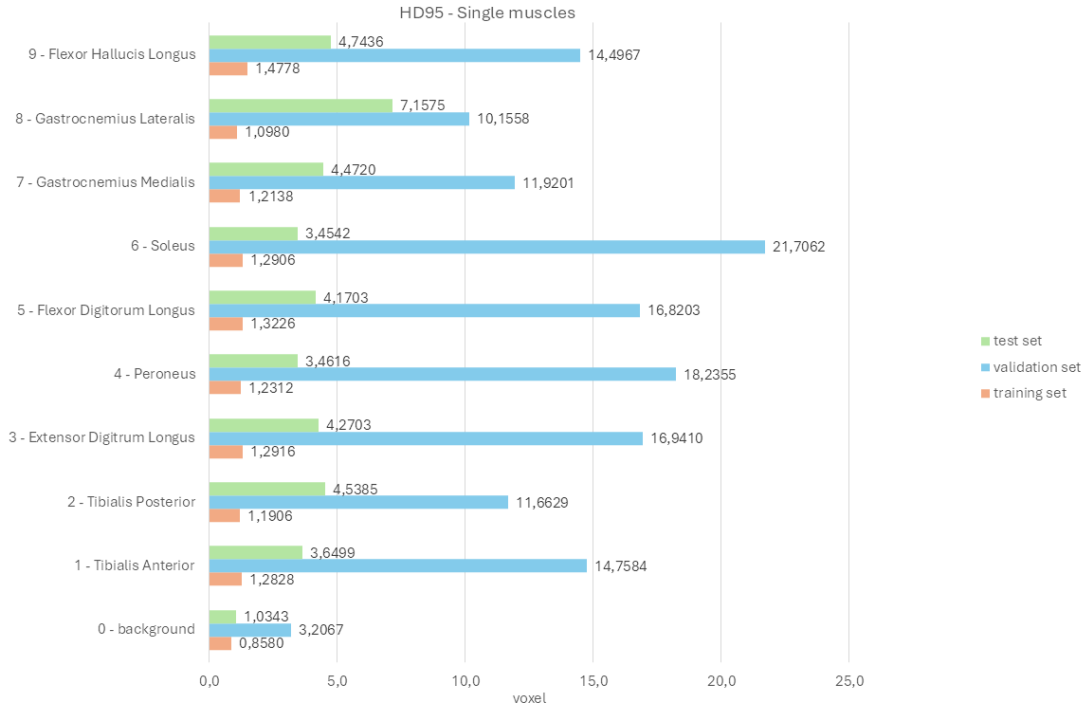

Figure 4.48: HD95 of single muscles obtained with SwinUNETR 3D Neural Network with Dice Cross Entropy Loss function for leg

- **2° test: SwinUNETR with Generalized Dice Loss function**
  In *Figure 4.49* graphs of DICE and HD95 obtained with SwinUNETR with Generalized Dice Loss function are represented: mean value and its standard deviation are reported for each set of volumes.
  The *Figure 4.50* graph shows the average DICE value obtained for each muscle with SwinUNETR with Generalized Dice Loss function, for each set of volumes.
  The *Figure 4.51* graph shows the average HD95 value obtained for each muscle with SwinUNETR with Generalized Dice Loss function, for each set of volumes.

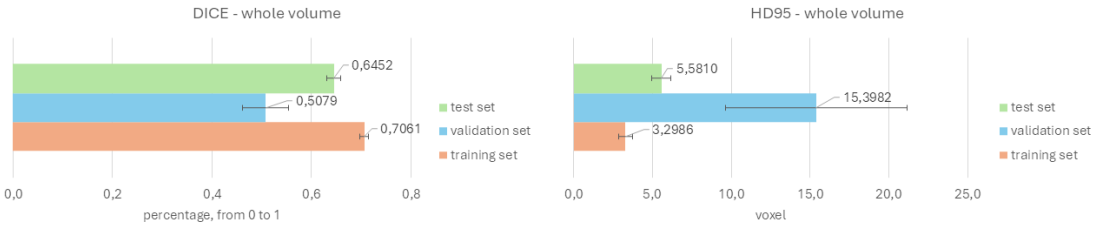

Figure 4.49: Whole volume metrics obtained with SwinUNETR 3D Neural Network with Generalized Dice Loss function for leg

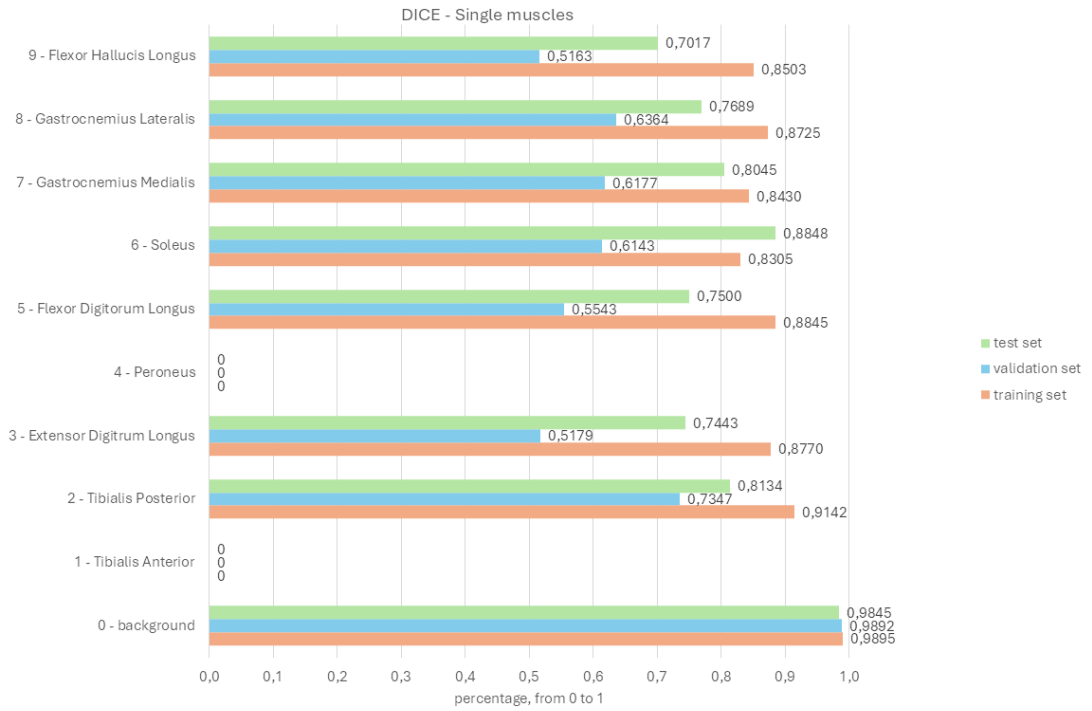

Figure 4.50: DICE of single muscles obtained with SwinUNETR 3D Neural Network with Generalized Dice Loss function for leg

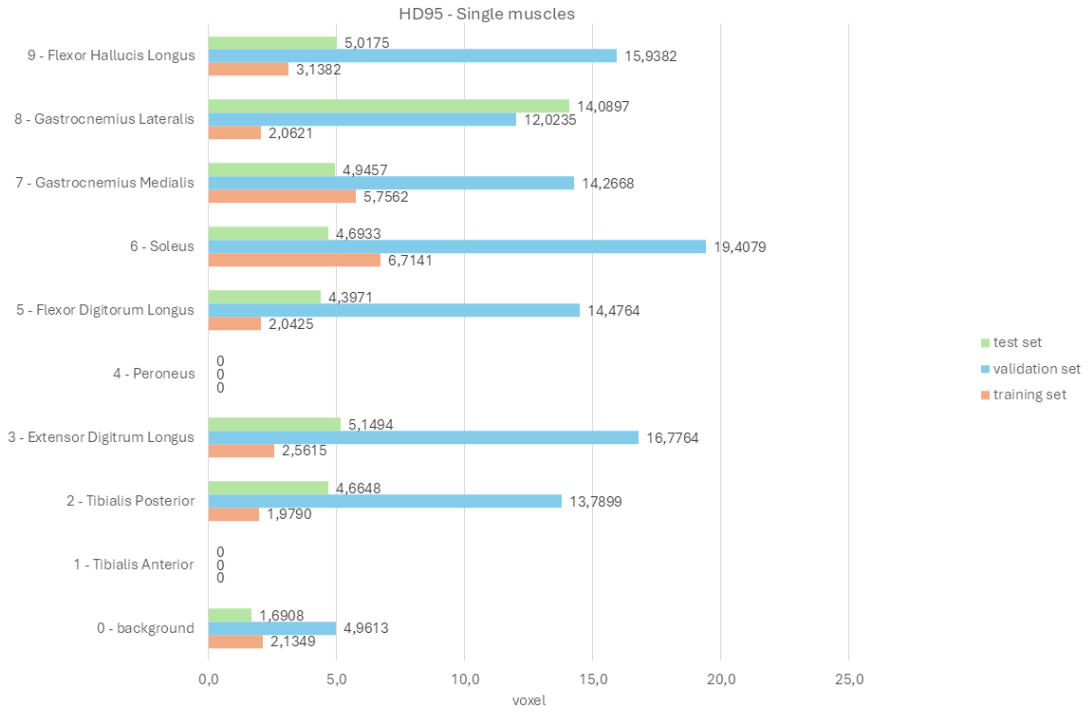

Figure 4.51: HD95 of single muscles obtained with SwinUNETR 3D Neural Network with Generalized Dice Loss function for leg

- **3° test: SwinUNETR with Focal Loss function**
  In *Figure 4.52* graphs of DICE and HD95 obtained with SwinUNETR with Focal Loss function are represented: mean value and its standard deviation are reported for each set of volumes.
  The *Figure 4.53* graph shows the average DICE value obtained for each muscle with SwinUNETR with Focal Loss function, for each set of volumes.
  The *Figure 4.54* graph shows the average HD95 value obtained for each muscle with SwinUNETR with Focal Loss function, for each set of volumes.

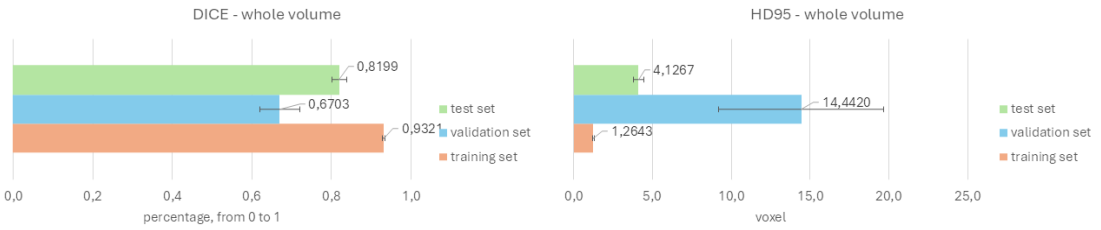

Figure 4.52: Whole volume metrics obtained with SwinUNETR 3D Neural Network with Focal Loss function for leg

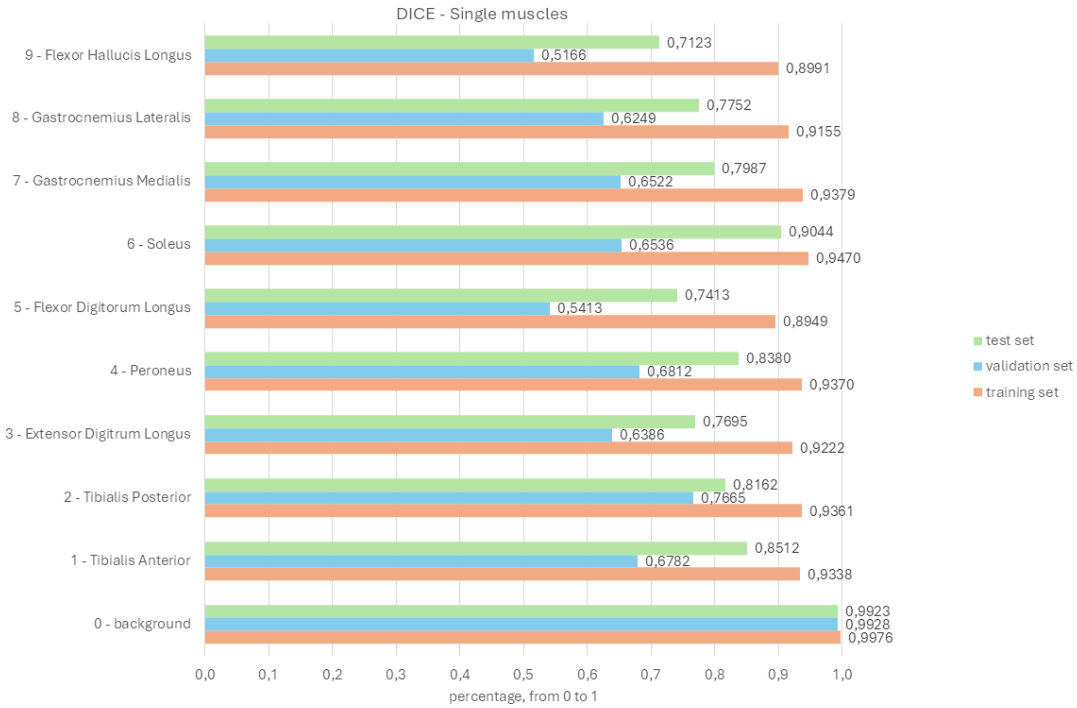

Figure 4.53: DICE of single muscles obtained with SwinUNETR 3D Neural Network with Focal Loss function for leg

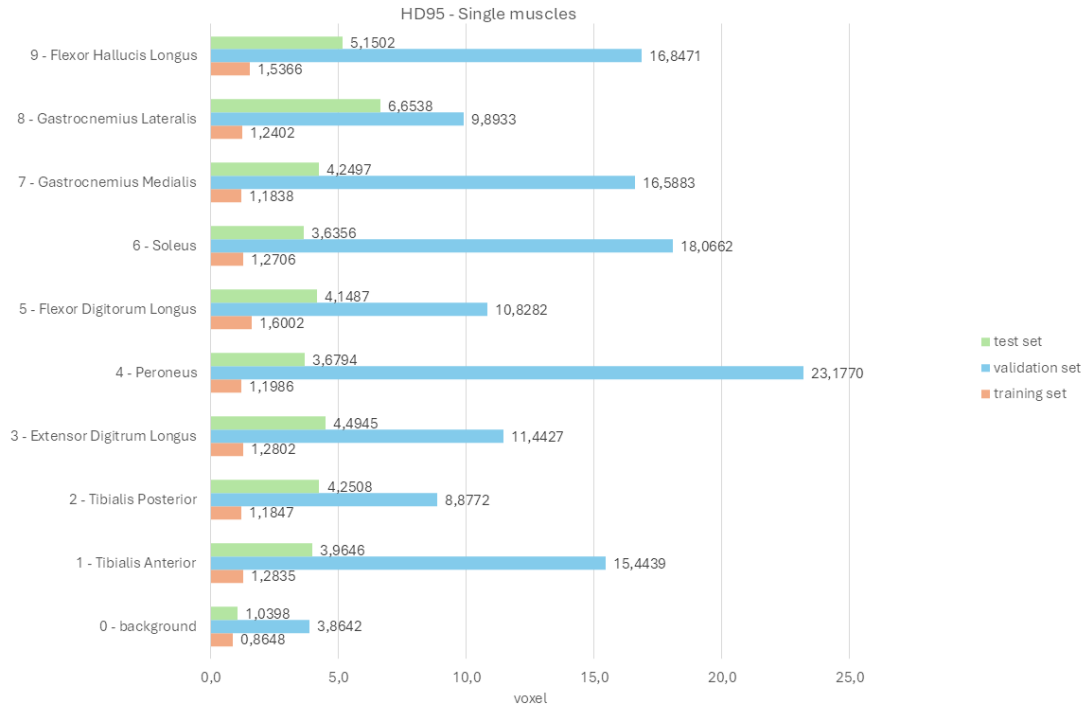

Figure 4.54: HD95 of single muscles obtained with SwinUNETR 3D Neural Network with Focal Loss function for leg

- **4° test: SwinUNETR with Dice Focal Loss function**
  In *Figure 4.55* graphs of DICE and HD95 obtained with SwinUNETR with Focal Loss function are represented: mean value and its standard deviation are reported for each set of volumes.
  The *Figure 4.56* graph shows the average DICE value obtained for each muscle with SwinUNETR with Dice Focal Loss function, for each set of volumes.
  The *Figure 4.57* graph shows the average HD95 value obtained for each muscle with SwinUNETR with Dice Focal Loss function, for each set of volumes.

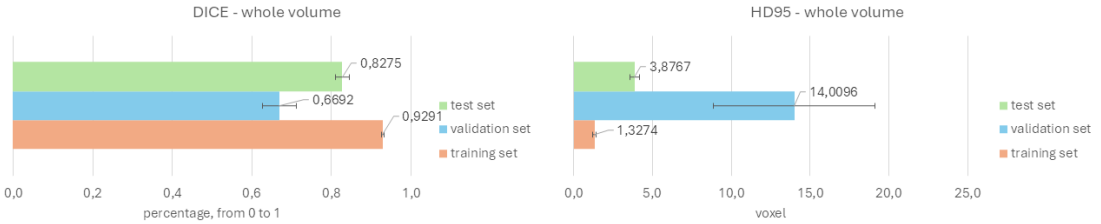

Figure 4.55: Whole volume metrics obtained with SwinUNETR 3D Neural Network with Dice Focal Loss function for leg

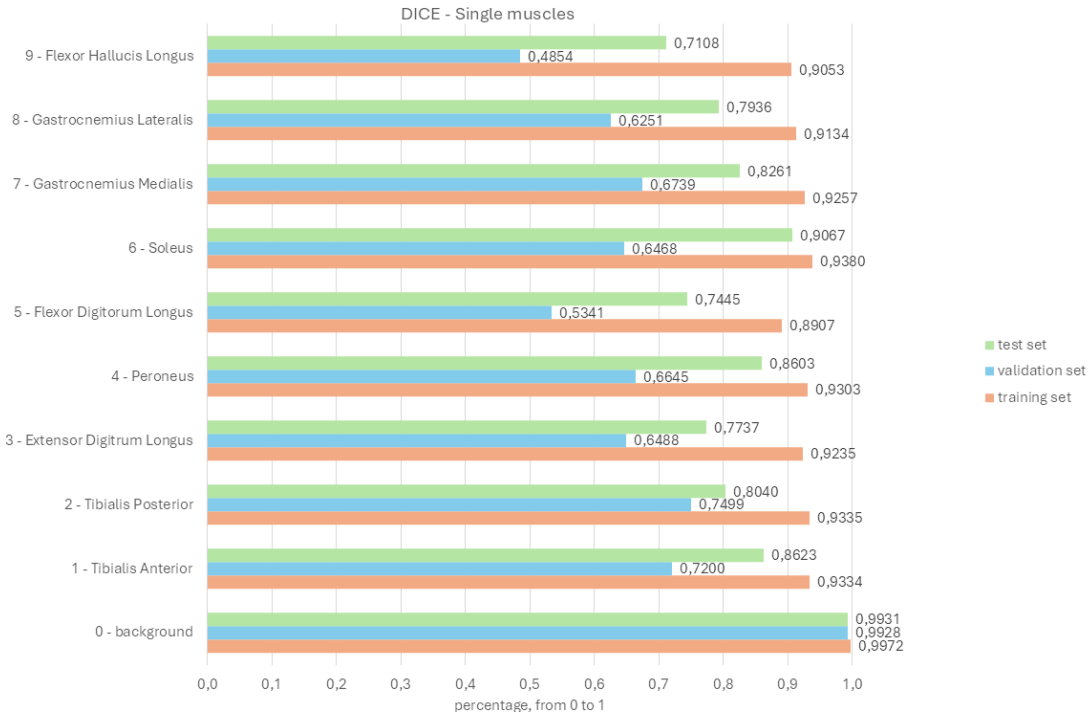

Figure 4.56: DICE of single muscles obtained with SwinUNETR 3D Neural Network with Dice Focal Loss function for leg

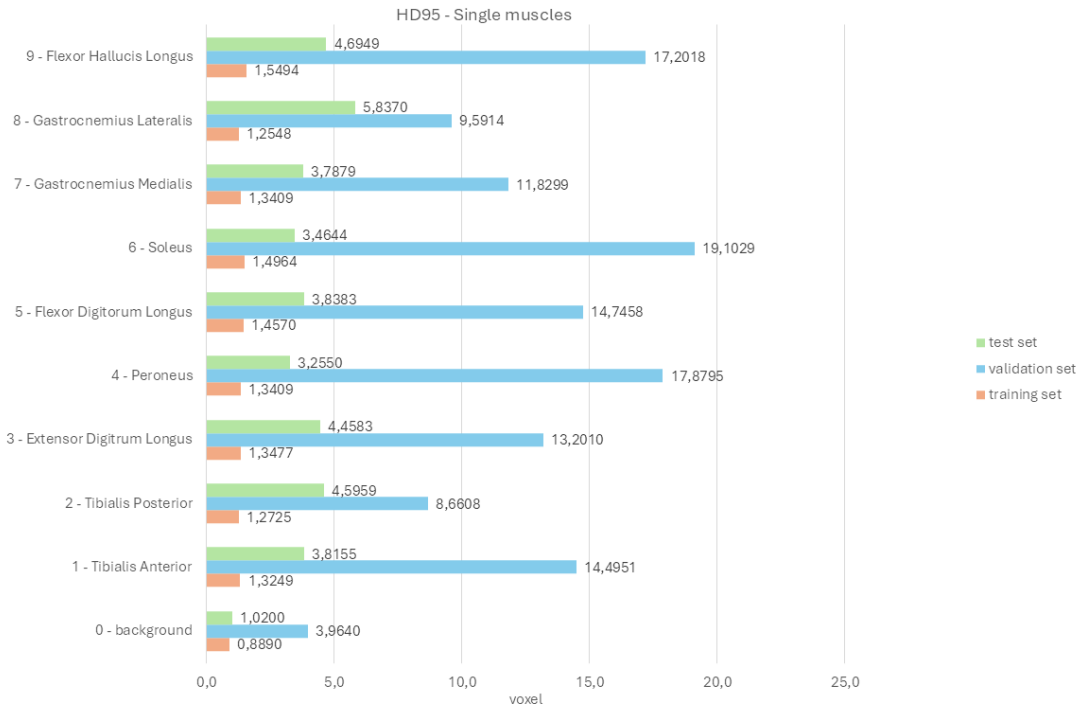

Figure 4.57: HD95 of single muscles obtained with SwinUNETR 3D Neural Network with Dice Focal Loss function for leg

- **5° test: SwinUNETR with Generalized Dice Focal Loss function**
  In *Figure 4.58* graphs of DICE and HD95 obtained with SwinUNETR with Generalized Dice Focal Loss function are represented: mean value and its standard deviation are reported for each set of volumes.
  The *Figure 4.59* graph shows the average DICE value obtained for each muscle with SwinUNETR with Generalized Dice Focal Loss function, for each set of volumes.
  The *Figure 4.60* graph shows the average HD95 value obtained for each muscle with SwinUNETR with Generalized Dice Focal Loss function, for each set of volumes.

Figure 4.58: Whole volume metrics obtained with SwinUNETR 3D Neural Network with Generalized Dice Focal Loss function for leg

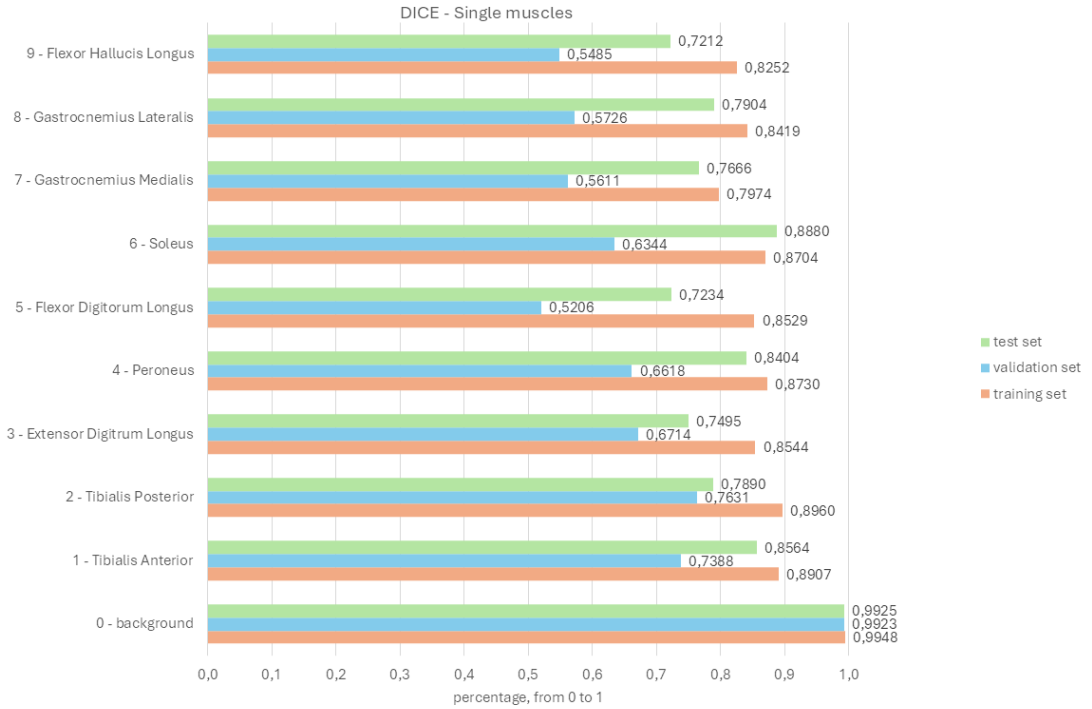

Figure 4.59: DICE of single muscles obtained with SwinUNETR 3D Neural Network with Generalized Dice Focal Loss function for leg

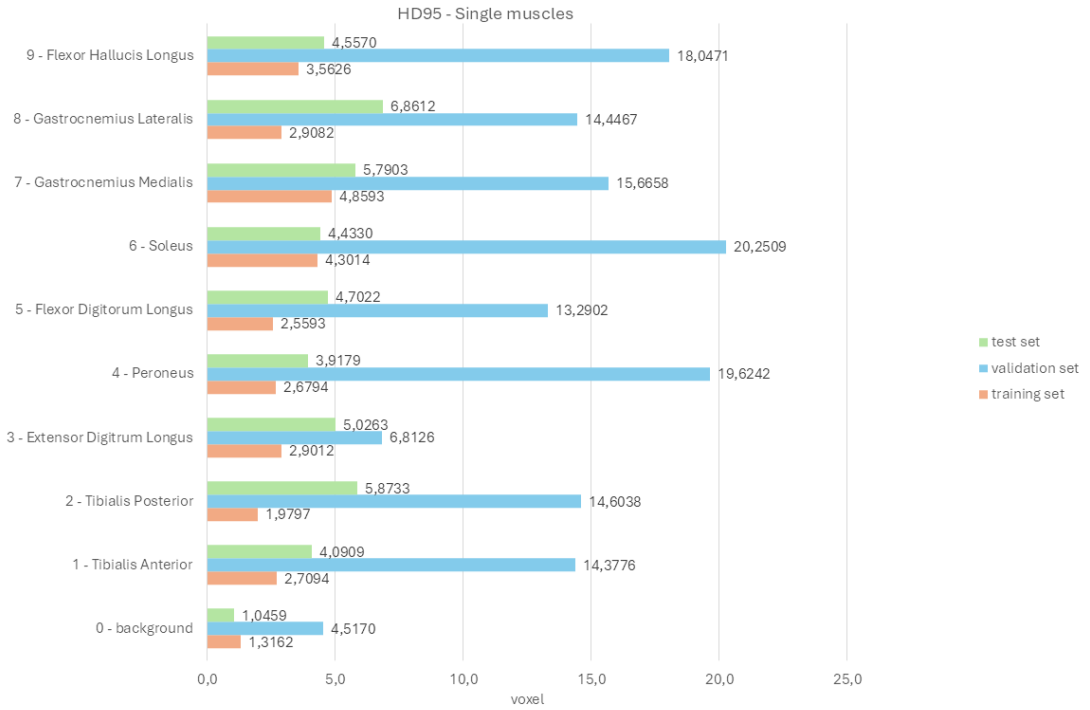

Figure 4.60: HD95 of single muscles obtained with SwinUNETR 3D Neural Network with Generalized Dice Focal Loss function for leg

4.2 Results of Federated Learning segmentation

With the segmentation obtained from DAFNE, for thigh the average value of DICE considering whole volume is 0,2564±0,0264 and the average value of HD95 is 59,033±4,9694 pixel.
For leg segmentation the average metric values obtained are 0,5044±0,0559 for DICE and 29,067±5,0495 pixel for HD95.
The metrics calculated for each single muscle are represented in the following figures: in *Figure 4.61* there are the metrics calculated for thigh, while in *Figure 4.62* there are the ones obtained for leg.

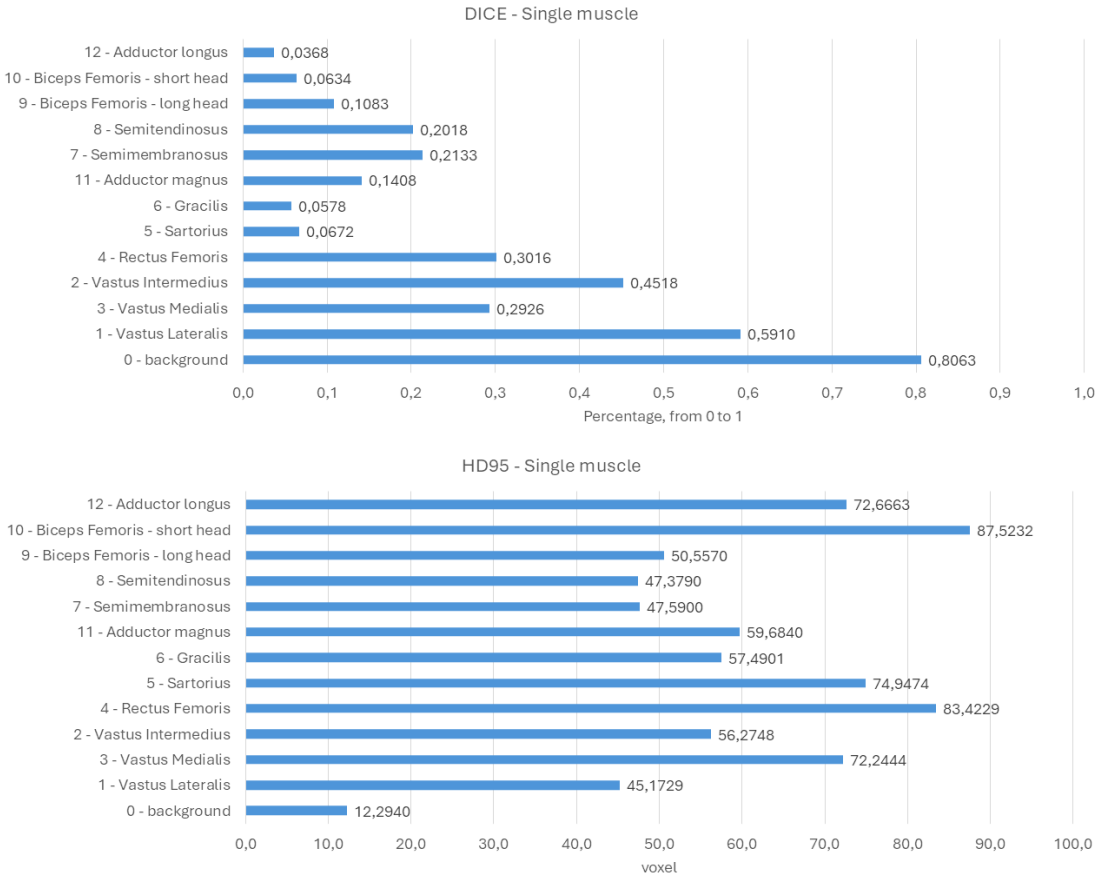

Figure 4.61: Average DICE and HD95 of single muscles obtained with DAFNE segmentation of thigh

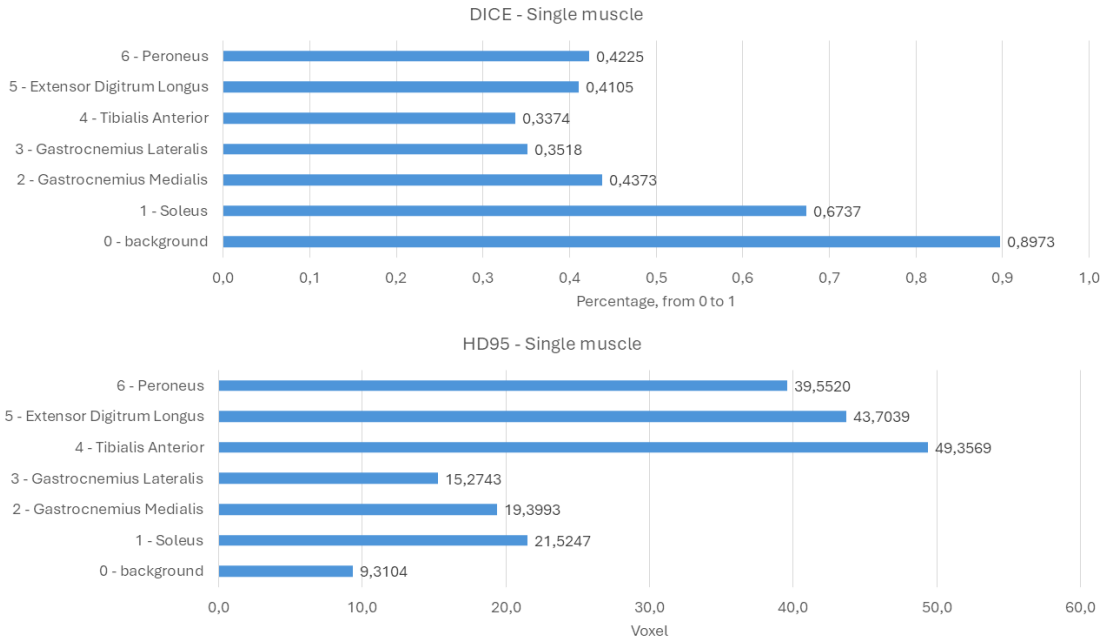

Figure 4.62: Average DICE and HD95 of single muscles obtained with DAFNE segmentation of thigh

Chapter 5

Analysis and discussion

5.1 The chosen 3D SwinUNETR Neural Networks

The final 3D SwinUNETR Neural Networks have been chosen comparing the average values of metrics calculated on the whole volume, both for the first and the second division of the dataset.
In both thigh and leg cases, all 3D SwinUNETR neural networks obtained with the second division of the dataset presented lower performances in the training set than the ones obtained with the first division, and better performances (higher DICE and lower HD95) in the test set than in the validation set, as shown in *Figure 5.1* and in *Figure 5.2*. In these two figures it is represented a comparison between the average metrics calculated in each set and for each tested loss function with the second division of the dataset. The orange bars indicates metrics in training set, the blues bars indicates metrics for validation set and the green bars indicates metrics for test set.
Usually Neural Network with balanced performances between test and validation set are robuster and more reliable in results than Neural Networks with a great difference between calculated metrics in these two sets. For this reason 3D SwinUNETR Neural Networks obtained from the second division have been excluded from the choice.
So the final 3D Neural Networks both for thigh and leg have been chosen among the ones trained with the first division of the dataset.

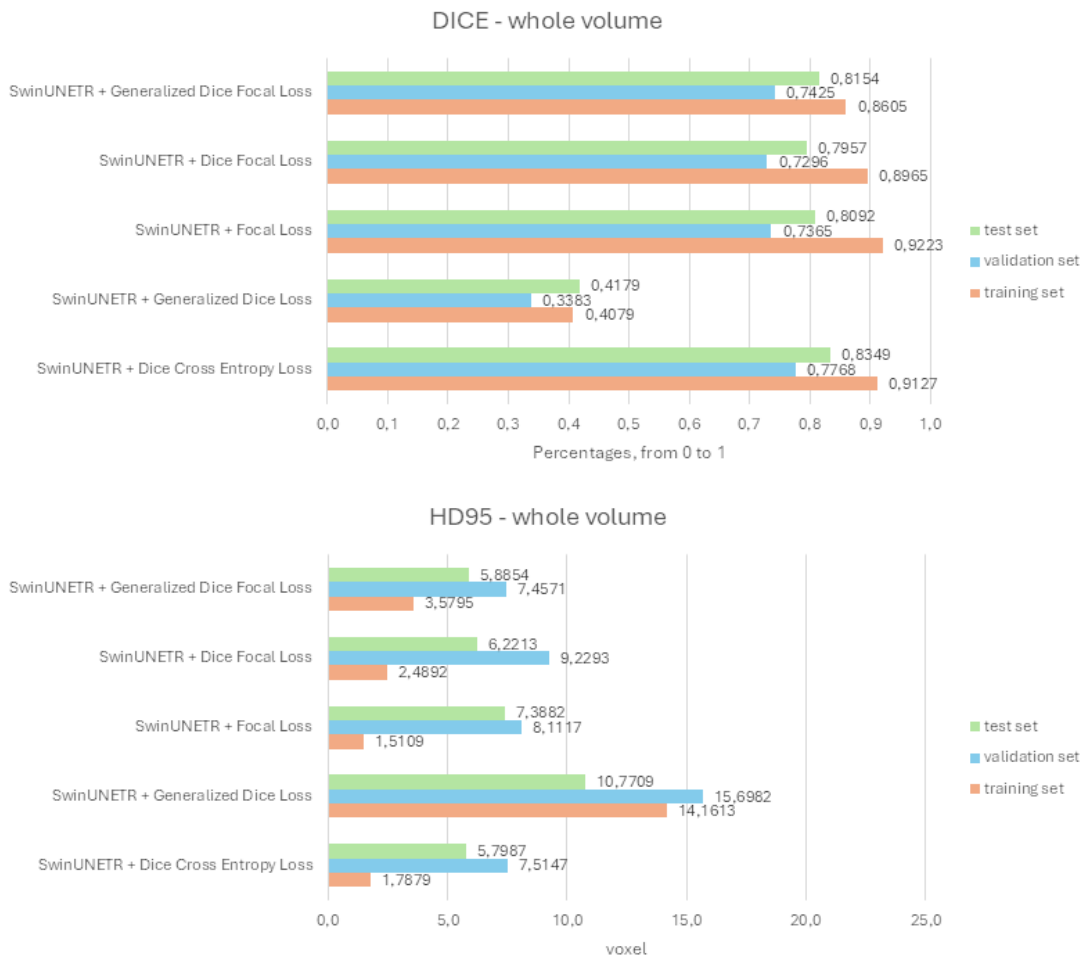

Figure 5.1: Whole volume metrics obtained with SwinUNETR 3D Neural Networks for thigh with the second division of the dataset

Figure 5.2: Whole volume metrics obtained with SwinUNETR 3D Neural Networks for leg with the second division of the dataset

Figure 5.3 shows a summary of all the average metrics obtained from different loss functions and different divisions of the dataset, as indicated in the rows. The green cells indicate the best metric value of the column of that group. Both thigh and leg metrics values are reported. As can be seen, the loss functions that have been chosen in the final 3D local Neural Networks are the ones that presents most of the best average metric values and, at the same time, balanced results in validation and test set. The choice of the final local 3D Neural Networks have been detailed in the following paragraphs.

			TRAINING SET		VALIDATION SET		TEST SET	
		DICE LOSS	DICE	HD95	DICE	HD95	DICE	HD95
THIGH	1° DIVISION	Dice Cross Entropy Loss	0,9257	1,5826	0,7746	7,7493	0,7625	5,9796
		Generalized Dice Loss	0,4481	9,8135	0,4121	11,6687	0,3785	10,1993
		SwinUNETR + Focal Loss	0,9230	1,6253	0,7571	7,9131	0,7449	6,6100
		Dice Focal Loss	0,8854	4,1213	0,7374	10,0108	0,7278	7,0186
		Generalized Dice Focal Loss	0,8703	3,3308	0,7622	7,6644	0,7388	6,7733
	2° DIVISION	Dice Cross Entropy Loss	0,9127	1,7879	0,7768	7,5147	0,8349	5,7987
		Generalized Dice Loss	0,4079	14,1613	0,3383	15,6982	0,4179	10,7709
		SwinUNETR + Focal Loss	0,9223	1,5109	0,7365	8,1117	0,8092	7,3882
		Dice Focal Loss	0,8965	2,4892	0,7296	9,2293	0,7957	6,2213
		Generalized Dice Focal Loss	0,8605	3,5795	0,7425	7,4571	0,8154	5,8854
LEG	1° DIVISION	Dice Cross Entropy Loss	0,9285	1,3212	0,7386	8,4625	0,6983	8,0472
		Generalized Dice Loss	0,8305	1,9769	0,6641	9,7961	0,6520	8,5356
		SwinUNETR + Focal Loss	0,9348	1,2639	0,7445	8,3336	0,7110	6,9627
		Dice Focal Loss	0,9437	1,2015	0,7492	8,1483	0,7516	6,0544
		Generalized Dice Focal Loss	0,9339	1,3673	0,7349	7,9627	0,7217	7,3046
	2° DIVISION	Dice Cross Entropy Loss	0,9339	1,2257	0,6728	15,1786	0,8290	4,0952
		Generalized Dice Loss	0,7061	3,2986	0,5079	15,3982	0,6452	5,5810
		SwinUNETR + Focal Loss	0,9321	1,2643	0,6703	14,4420	0,8199	4,1267
		Dice Focal Loss	0,9291	1,3274	0,6692	14,0096	0,8275	3,8767
		Generalized Dice Focal Loss	0,8697	2,9777	0,6670	15,0035	0,8117	4,6298

Figure 5.3: Summary of the comparison between average metrics calculated with different loss functions and with different divisions of the dataset, both for thigh and leg cases. The circled loss functions are the ones that have been chosen in the final 3D Neural Networks

5.1.1 3D SwinUNETR Neural Network for thigh

The final Neural Network chosen for thigh has been the SwinUNETR with Dice Cross Entropy Loss Function because it obtained the best metrics with respect to the ones calculated in the case of the other Loss functions.
In *Figure 5.4* the average values of the metrics are shown. How it can be seen in these graphs, the average metrics of the combination of the SwinUNETR and the Dice Cross Entropy Loss has given the best results for thigh.
With this Neural Network better results can be observed with bigger muscles: as previously shown in *Figure 4.2*, DICE values bigger than 0.800 is obtained for Vastus Lateralis, Vastus Medialis, Vastus Intermedius and Biceps Femoris in all three sets. On the contrary, the lowest DICE value is obtained for Adductor longus.

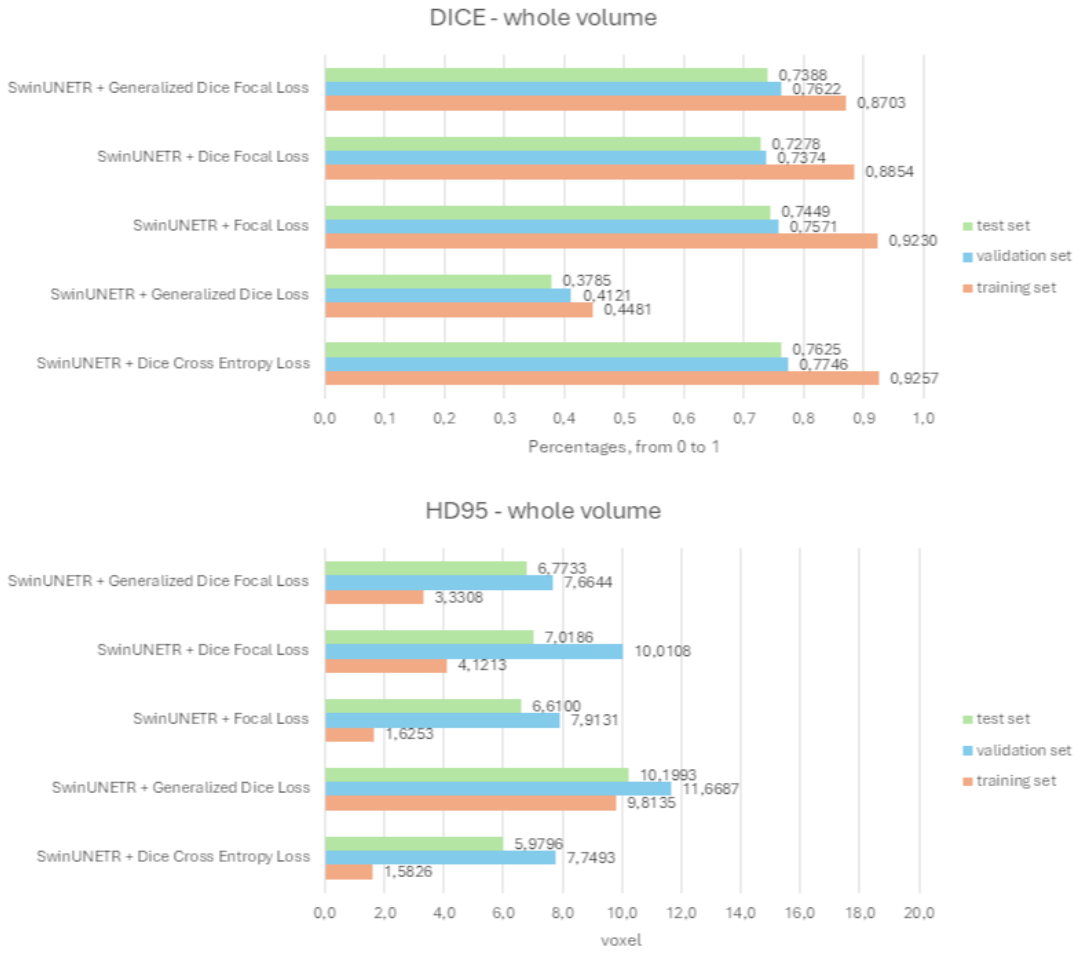

Figure 5.4: Whole volume metrics obtained with SwinUNETR 3D Neural Networks combined with different loss functions for thigh with the first division of the dataset

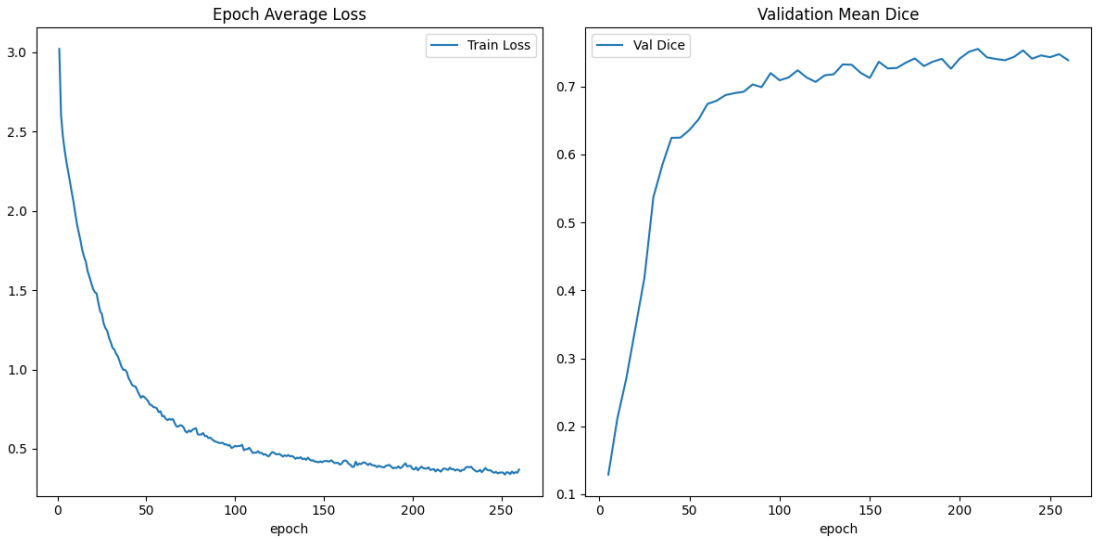

Figure 5.5: Epoch Average Loss graph and Validation mean DICE graph obtained from the training of the SwinUNETR 3D Neural Network with Dice Cross Entropy Loss Function for thigh.

Figure 5.5 shows the Epoch Average Loss graph and Validation Mean DICE graph obtained in the training of the chosen final SwinUNETR 3D Neural Network for the thigh.
The *Figure 5.6* shows the comparison between metrics calculated after the application of the two kinds of postprocessing and without any of them. Each graph refers to one specific set of volumes. Orange bars indicate metrics calculated without the postprocessing step, blue bars indicate metrics obtained applying the first postprocessing and green bars indicate metrics calculated with the second type of postprocessing.
As it can be seen in this figure, both of these two postprocessing have not bring significant improvements in the calculated metrics. For this reason a final pipeline without a postprocessing like these have been chosen. So, in this work, at the end of the inference step no postprocessing is included.

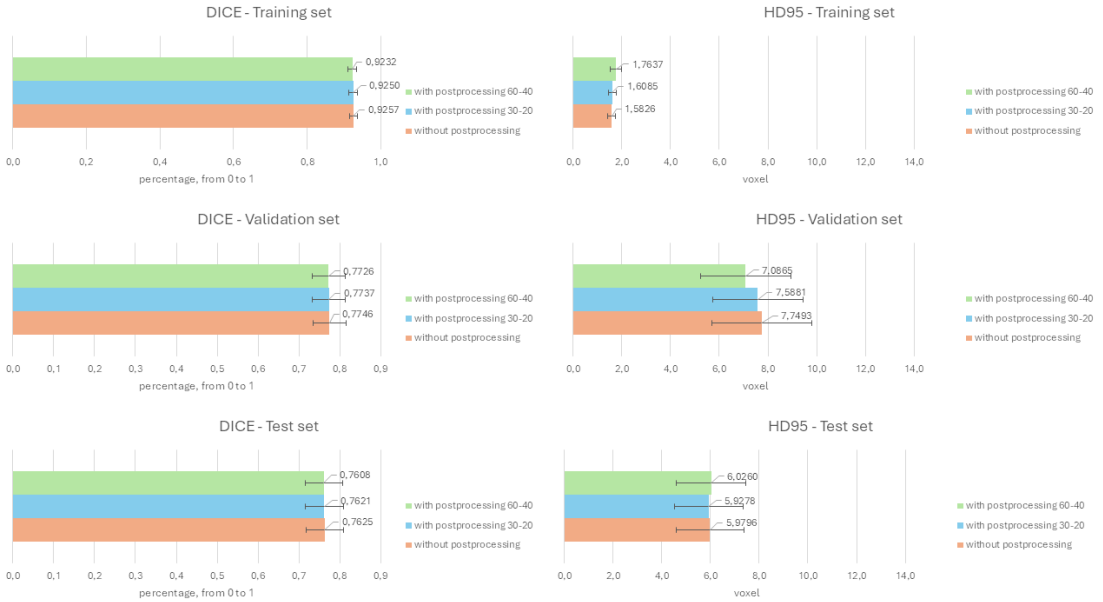

Figure 5.6: Comparison between metrics obtained from segmentation without and with the two considered kinds of postprocessing in the case of thigh. "Postprocessing 30-20" refers to the first of the tested postprocessing, while "Postprocessing 60-40" refers to the second. On the left there are the graphs dedicated to DICE, one for each set, while on the right there are the graphs dedicated to HD95, one for each set.

In *Figure 5.7* there is an example of a prediction obtained with this Local 3D Neural Network. The volume is from the test set of thigh.

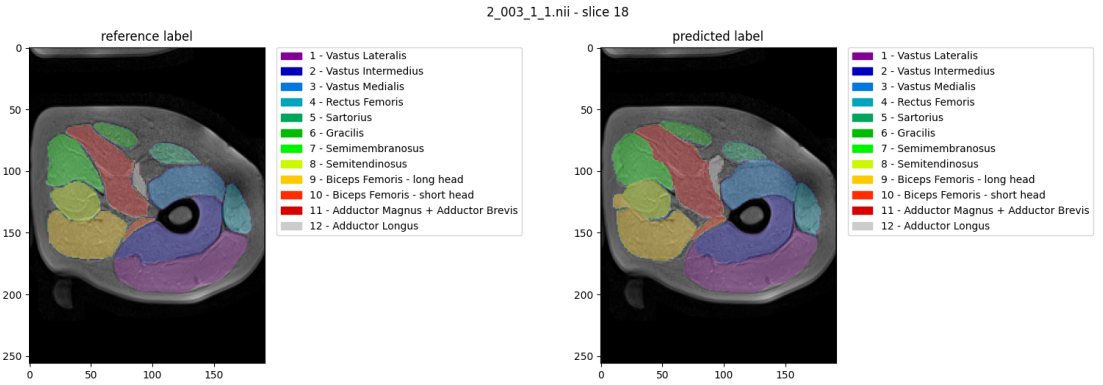

Figure 5.7: Comparison between a predicted and the correspondent reference segmentation of a slice from a volume of thigh test set. On the right there is the reference original mask, while in the left there is the predicted segmentation.

5.1.2 3D SwinUNETR Neural Network for leg

The final Neural Network chosen for leg has been the SwinUNETR with Dice Focal Loss Function.
In *Figure 5.8* the average values of the metrics are shown. How it can be seen in these graphs, the average metrics of the combination of the SwinUNETR and the Dice Focal Loss has given the best results for leg, with respect to the others combinations between SwinUNETR 3D Neural Network and different Loss functions.

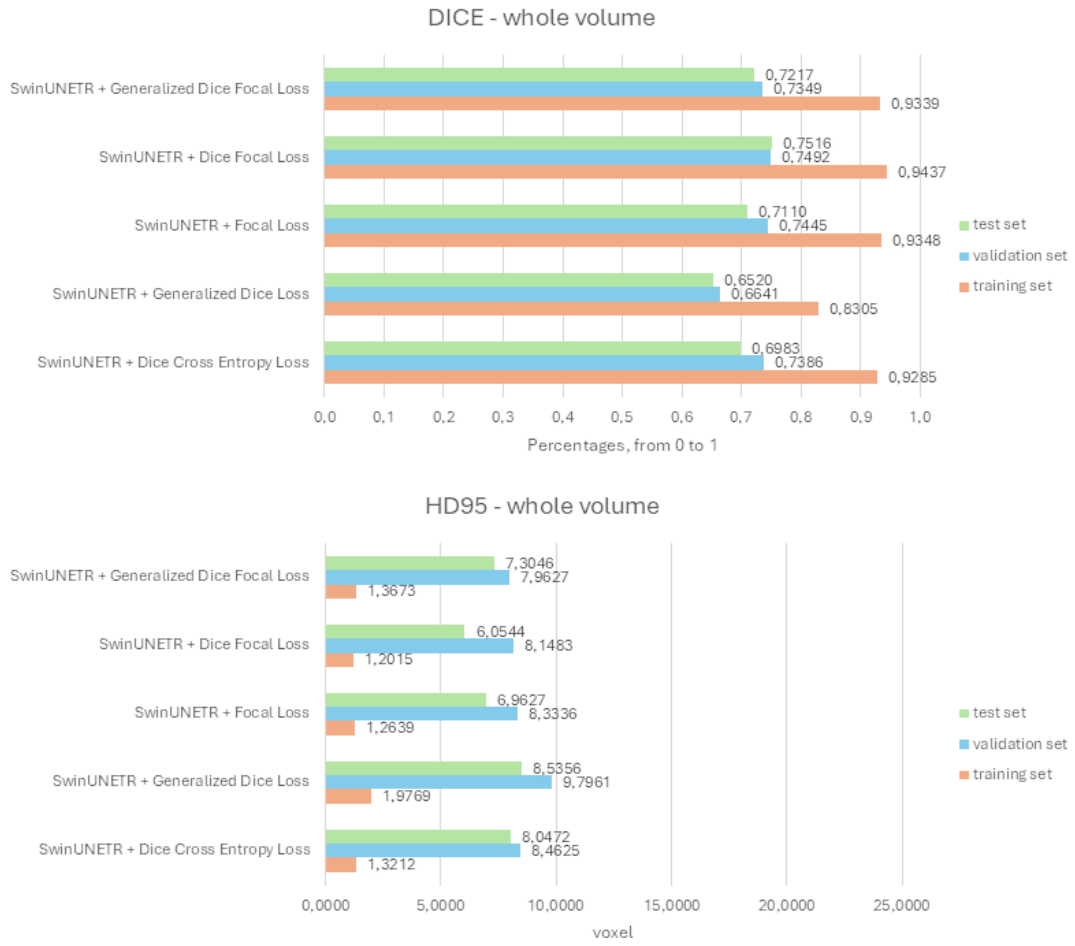

Figure 5.8: Whole volume metrics obtained with SwinUNETR 3D Neural Networks combined with different loss function for leg with the first division of the dataset

As previously shown in *Figure 4.41*, DICE values bigger than 0.700 in all the three sets have been obtained for Gastrocnemius Medialis Soleus, Peroneus, Tibialis

Anterior and Tibialis posterior. On the other hand, the lowest DICE value has been obtained for the Flexor Hallucis Longus.

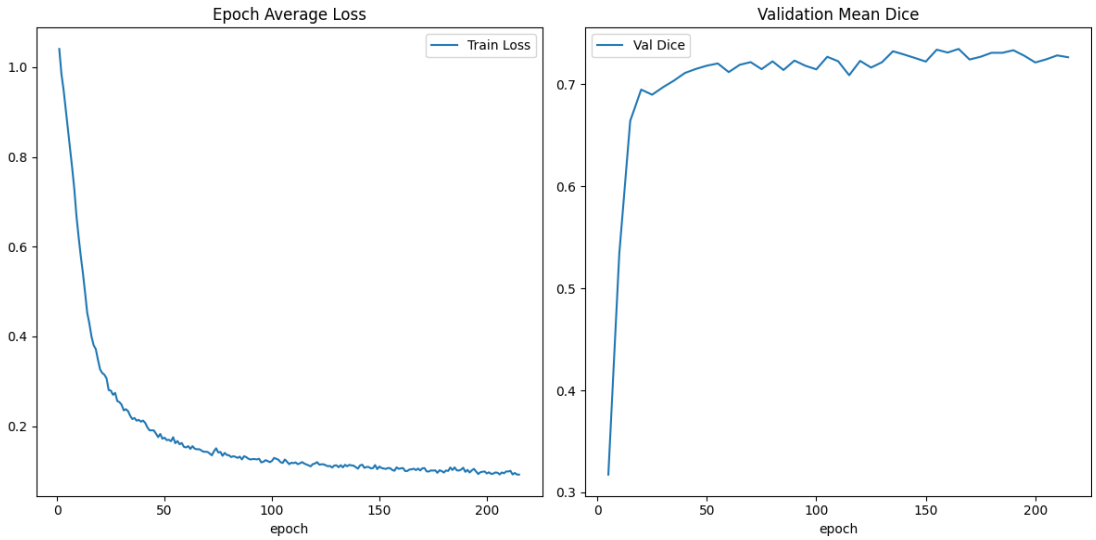

Figure 5.9: Epoch Average Loss graph and Validation mean DICE graph obtained from the training of the SwinUNETR 3D Neural Network with Dice Focal Loss Function for leg.

Figure 5.9 shows the Epoch Average Loss graph and Validation Mean DICE graph obtained in the training of the chosen final SwinUNETR 3D Neural Network for the leg.
Similarly to what has been done in the case of the final SwinUNETR 3D Neural Network chosen for thigh, also in the case of leg the same two combination of postprocessing steps have been tested: in *Figure 5.10* it can be seen a comparison between the average values and standard deviations of metrics calculated on whole volumes. Each graph refers to a different set. Metrics obtained from segmentation without postprocessing are represented with orange bars, metrics obtained with the first considered type of postprocessing are represented with blue bars and metrics obtained with the second considered type of postprocessing are represented with green bars.
Here too, no sensible improvements in performance have been achieved, as can be seen in this picture, so also for leg the final pipeline has no postprocessing step.

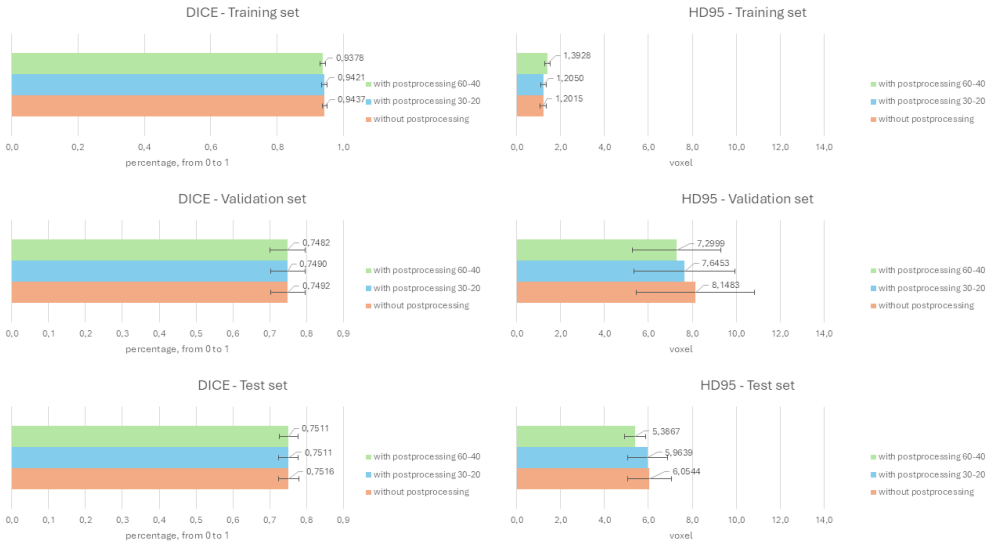

Figure 5.10: Comparison between metrics obtained from segmentation without and with the two considered kinds of postprocessing in the case of leg. "Postprocessing 30-20" refers to the first of the tested postprocessing, while "Postprocessing 60-40" refers to the second. On the left there are the graphs dedicated to DICE, one for each set, while on the right there are the graphs dedicated to HD95, one for each set.

In *Figure 5.11* there is an example of a prediction obtained with this Local 3D Neural Network. The volume is from the test set of leg.

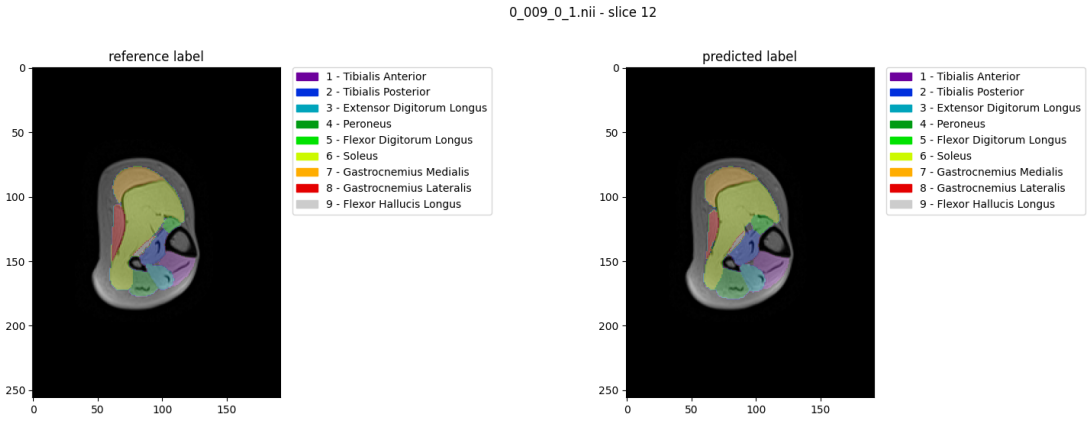

Figure 5.11: Comparison between a predicted and the correspondent reference segmentation of a slice from a volume of leg test set. On the right there is the reference original mask, while in the left there is the predicted segmentation.

5.2 Comparison: local segmentation results vs federated learning segmentation

Local segmentations has provided better results with respect to the segmentations obtained with DAFNE, both on average metrics on whole volume and on single muscle. For the single muscle, DAFNE segmentations get better results in bigger muscles, like vastus lateralis and intermedius in thigh end soleus in leg, with respect to the smaller ones. Otherwise the performances of leg segmentations are better than the ones of thigh segmentation, but still lower than the performances of local neural networks

An important aspect of DAFNE possibilities has not been considered: DAFNE allows to use postprocessing editing tools that allow to refine manually each segmentation on each slice of the volume, but in this work it has been chosen to test the results obtained only using the available centralized neural network, without these tools, in order to get a situation comparable to the one of local SwinUNETR. It is clear that, using these postprocessing tools integrated in the API the results would be better, but less comparable to the one obtained with local SwinUNETR because of the absence of this manual refinement step in local segmentation.

Another important aspect to take into count is that the local SwinUNETR is a 3D Neural Network, that considers the whole volume, while the DAFNE centralized neural networks are 2D, so they have not a global view of the volume as in the case on a 3D neural network.

Finally DAFNE provides separated neural network for left limbs and for right limbs: for each loaded volume the right or left neural network was selected, according to the side of the limb in the volume itself, so the possibility of a human error is not null. While the locally trained 3D SwinUNETR Neural Networks are able to segment both right and left limbs: so there is only one local neural network both for right and left thigh and only another one for right and left leg. This aspect nullify the possible human error in the chose of the correct neural network.

Chapter 6

Conclusion and future developments

In conclusion, the best results in automatic segmentation have been obtained in this work with Local 3D Neural Networks of the SwinUNNETR type, combined with a Dice Cross Entropy Loss Function in the case of thigh volumes, and with Dice Focal Loss Function in the case of leg volumes.
Despite the good results, these local Neural Networks keep the disadvantages of every other kinds of local learning: one of the most important in the final performance is that the dataset is strongly limited to the local stored volumes. This strongly limits the performances that a neural network could achieve.
DAFNE, as well as Federated Learning, provides undeniable advantages in therms of privacy and data protection, collaboration between users and sharing of models yet trained on a virtually large dataset, but, according to the obtained results, at the moment a locally trained 3D Neural Network, with the characteristics previously described, is the best choice for thigh and leg muscle segmentation in the provided dataset.
It is important to highlight that only the basic segmentation functions of DAFNE have been tested. It could be very interesting testing other functionalities that DAFNE provides such as the postprocessing editig tool: using them properly and, then, performing the incremental learning, the model given by DAFNE can be adapted to the own dataset and the following segmentation should get better results.
Another interesting aspect is the possibilities to segment in a "mixed" mode on DAFNE: the system gives the possibilities to merge a local Neural Network to the provided one. With the resulting new Neural Network the segmentation is performed. This option has been excluded in this work because of the incompatibility between the expected input data in the two cases: in the cases of local Neural Network the

input data are 3D volumes, while the Neural Network provided by DAFNE works on 2D input.
Finally, a proper postprocessing for the local Neural Network could bring further improvement in the results.

Bibliography

[1] Vadim. Kuperman. *Magnetic resonance imaging : physical principles and applications*. eng. Electromagnetism. San Diego: Academic Press, 2000. ISBN: 1-281-05699-5 (cit. on p. 1).

[2] Jordi Díaz-Manera, Jaume Llauger, Eduard Gallardo, and Isabel Illa. «Muscle MRI in muscular dystrophies». eng. In: *Acta myologica* 34.2-3 (2015), pp. 95–108. ISSN: 1128-2460 (cit. on p. 3).

[3] Lara Riem et al. «AI driven analysis of MRI to measure health and disease progression in FSHD». eng. In: *Scientific reports* 14.1 (2024), pp. 15462–15. ISSN: 2045-2322 (cit. on p. 3).

[4] Sanne C.C. Vincenten, Nicol C. Voermans, Donnie Cameron, Baziel G.M. van Engelen, Nens van Alfen, and Karlien Mul. «The complementary use of muscle ultrasound and MRI in FSHD: Early versus later disease stage follow-up». In: *Clinical Neurophysiology* (2024). ISSN: 1388-2457. DOI: https://doi.org/10.1016/j.clinph.2024.02.036. URL: https://www.sciencedirect.com/science/article/pii/S1388245724000646 (cit. on p. 3).

[5] Leo H Wang et al. «MRI-informed muscle biopsies correlate MRI with pathology and DUX4 target gene expression in FSHD». eng. In: *Human molecular genetics* 28.3 (2019), pp. 476–486. ISSN: 0964-6906 (cit. on p. 3).

[6] Ellen Plas, Laurie Gutmann, Dan Thedens, Richard K. Shields, Kathleen Langbehn, Zhihui Guo, Milan Sonka, and Peggy Nopoulos. «Quantitative muscle MRI as a sensitive marker of early muscle pathology in myotonic dystrophy type 1». eng. In: *Muscle & nerve* 63.4 (2021), pp. 553–562. ISSN: 0148-639X (cit. on pp. 3, 4).

[7] Linda Heskamp, Marlies van Nimwegen, Marieke J. Ploegmakers, Guillaume Bassez, Jean-Francois Deux, Sarah A. Cumming, Darren G. Monckton, Baziel G.M. van Engelen, and Arend Heerschap. «Lower extremity muscle pathology in myotonic dystrophy type 1 assessed by quantitative MRI». eng. In: *Neurology* 92.24 (2019), e2803–e2814. ISSN: 0028-3878 (cit. on p. 4).

[8] Jennifer Garland et al. «Identification of an Alu element-mediated deletion in the promoter region of GNE in siblings with GNE myopathy». In: *Molecular Genetics and Genomic Medicine* 5 (June 2017). DOI: 10.1002/mgg3.300 (cit. on p. 4).

[9] Rene Y Choi, Aaron S Coyner, Jayashree Kalpathy-Cramer, Michael F Chiang, and J Peter Campbell. «Introduction to Machine Learning, Neural Networks, and Deep Learning». eng. In: *Translational vision science & technology* 9.2 (2020), pp. 14–14. ISSN: 2164-2591 (cit. on pp. 5–7).

[10] Isabella Castiglioni et al. «AI applications to medical images: From machine learning to deep learning». eng. In: *Physica medica* 83 (2021), pp. 9–24. ISSN: 1120-1797 (cit. on pp. 5, 6, 8).

[11] Nagasoujanya V. Annasamudram, Azubuike M. Okorie, Richard G. Spencer, Rita R. Kalyani, Qi Yang, Bennett A. Landman, Luigi Ferrucci, and Sokratis Makrogiannis. «Deep network and multi-atlas segmentation fusion for delineation of thigh muscle groups in three-dimensional water–fat separated MRI». eng. In: *Journal of medical imaging (Bellingham, Wash.)* 11.5 (2024), pp. 054003–054003. ISSN: 2329-4302 (cit. on p. 8).

[12] Louise Piecuch et al. «Muscle Volume Quantification: Guiding Transformers with Anatomical Priors». eng. In: *Lecture Notes in Computer Science*. Vol. 14350. Lecture Notes in Computer Science. Switzerland: Springer, 2023, pp. 173–187. ISBN: 3031469135 (cit. on pp. 8, 33).

[13] Zhihui Guo, Honghai Zhang, Zhi Chen, Ellen van der Plas, Laurie Gutmann, Daniel Thedens, Peggy Nopoulos, and Milan Sonka. «Fully automated 3D segmentation of MR-imaged calf muscle compartments: Neighborhood relationship enhanced fully convolutional network». eng. In: *Computerized medical imaging and graphics* 87 (2021), pp. 101835–101835. ISSN: 0895-6111 (cit. on p. 9).

[14] Zhendi Gong, Rosemary Nicholas, Susan T. Francis, Xin Chen, Angelica Aviles-Rivero, Carola-Bibiane Schönlieb, Guang Yang, and Michael Roberts. «Thigh and Calf Muscles Segmentation Using Ensemble of Patch-Based Deep Convolutional Neural Network on Whole-Body Water-Fat MRI». eng. In: *Medical Image Understanding and Analysis*. Vol. 13413. Lecture Notes in Computer Science. Switzerland: Springer International Publishing AG, 2022, pp. 262–270. ISBN: 9783031120527 (cit. on p. 9).

[15] Chen Zhang, Yu Xie, Hang Bai, Bin Yu, Weihong Li, and Yuan Gao. «A survey on federated learning». eng. In: *Knowledge-based systems* 216 (2021), pp. 106775–. ISSN: 0950-7051 (cit. on pp. 10–12).

[16] Francesco Santini et al. «Deep Anatomical Federated Network (Dafne): an open client/server framework for the continuous collaborative improvement of deep-learning-based medical image segmentation». eng. In: (2023) (cit. on pp. 13, 15).

[17] *Dafne*. URL: https://dafne.network/ (cit. on pp. 15, 16).

[18] *Dafne Repository*. URL: https://github.com/dafne-imaging (cit. on p. 16).

[19] *MONAI - Medical Open Network for Artificial Intelligence*. URL: https://monai.io/ (cit. on pp. 16, 17, 31).

[20] *Project MONAI*. URL: https://docs.monai.io/en/stable/ (cit. on p. 16).

[21] *MONAI Model Zoo*. URL: https://monai.io/model-zoo.html (cit. on p. 16).

[22] *MONAI Model Zoo Repository*. URL: https://github.com/Project-MONAI/model-zoo (cit. on p. 16).

[23] Evelin Omobono. *Segmentazione e caratterizzazione di volumi MRI di arto inferiore tramite Deep Learning e Texture Analysis per il riconoscimento di distrofie muscolari = Segmentation and characterization of lower limb MRI volumes using Deep Learning and Texture Analysis to recognize musclular dystrophies*. Meiburger, Kristen Mariko, 2023 (cit. on p. 18).

[24] Nicholas J. Tustison, Brian B. Avants, Philip A. Cook, Yuanjie Zheng, Alexander Egan, Paul A. Yushkevich, and James C. Gee. «N4ITK: Improved N3 Bias Correction». In: *IEEE Transactions on Medical Imaging* 29.6 (2010), pp. 1310–1320. DOI: 10.1109/TMI.2010.2046908 (cit. on p. 19).

[25] Jaber Juntu, Jan Sijbers, Dirk Van Dyck, and Jan Gielen. «Bias Field Correction for MRI Images». In: *Computer Recognition Systems*. Ed. by Marek Kurzyński, Edward Puchała, Michał Woźniak, and Andrzej żołnierek. Berlin, Heidelberg: Springer Berlin Heidelberg, 2005, pp. 543–551. ISBN: 978-3-540-32390-7 (cit. on p. 19).

[26] *3DSlicer*. URL: https://www.slicer.org/ (cit. on p. 26).

[27] Ali Hatamizadeh, Vishwesh Nath, Yucheng Tang, Dong Yang, Holger Roth, and Daguang Xu. *Swin UNETR: Swin Transformers for Semantic Segmentation of Brain Tumors in MRI Images*. 2022. arXiv: 2201.01266 [eess.IV]. URL: https://arxiv.org/abs/2201.01266 (cit. on pp. 34, 35).

[28] Carole H. Sudre, Wenqi Li, Tom Vercauteren, Sebastien Ourselin, and M. Jorge Cardoso. «Generalised Dice Overlap as a Deep Learning Loss Function for Highly Unbalanced Segmentations». In: *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*. Ed. by M. Jorge Cardoso et al. Cham: Springer International Publishing, 2017, pp. 240–248. ISBN: 978-3-319-67558-9 (cit. on p. 35).

[29] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollar. «Focal Loss for Dense Object Detection». In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*. Oct. 2017 (cit. on pp. 35, 36).

[30] *Cross Entropy Loss*. URL: `https://docs.monai.io/en/stable/losses.html` (cit. on p. 35).

[31] 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)*. 2016, pp. 565–571. DOI: `10.1109/3DV.2016.79` (cit. on p. 36).

[32] *Cross Entropy Loss*. URL: `https://pytorch.org/docs/stable/generated/torch.nn.CrossEntropyLoss.html` (cit. on p. 36).

[33] *Metrics*. URL: `https://docs.monai.io/en/stable/metrics.html` (cit. on p. 37).