UNIVERSIDADE CATÓLICA DE PETRÓPOLIS

CENTRO DE ENGENHARIA E COMPUTAÇÃO

MESTRADO PROFISSIONAL EM GESTÃO DE SISTEMAS DE ENGENHARIA

A Deep Learning Model for Semantic Segmentation of Intravascular Ultrasound Images

Daniel Cunha de Araujo Junior

Petrópolis

2023

UNIVERSIDADE CATÓLICA DE PETRÓPOLIS

CENTRO DE ENGENHARIA E COMPUTAÇÃO

MESTRADO PROFISSIONAL EM GESTÃO DE SISTEMAS DE ENGENHARIA

A Deep Learning Model for Semantic Segmentation of Intravascular Ultrasound Images

Dissertação apresentada ao Centro de Engenharia e Computação da Universidade Católica de Petrópolis como requisito parcial para conclusão do Curso de Mestrado Profissional em Gestão de Sistemas de Engenharia.

Professor Orientador

Pablo Javier Blanco

Daniel Cunha de Araujo Junior

Petrópolis

2023

CIP – Catalogação na Publicação

A663d Araujo Junior, Daniel Cunha de.
A Deep Learning Model for semantic segmentation of intravascular ultrasound images / Daniel Cunha de Araujo Junior. – 2023.
93 f. : il.

Dissertação (Mestrado Profissional em Gestão de Sistemas de Engenharia) – Universidade Católica de Petrópolis, 2023.
Orientação: Prof. Dr. Pablo Javier Blanco.
Linha de pesquisa: Modelagem Computacional

1. Deep learning. 2. Computer vision. 3 Medical imaging. I. Javier Blanco, Pablo (Orient.). II. Título.
CDD: 006.37

Universidade Católica de Petrópolis (UCP)
Bibliotecária responsável: Marlena H. Pereira – CRB7: 5075

UNIVERSIDADE CATÓLICA DE PETRÓPOLIS

Centro de Engenharia e Computação

Programa de Mestrado Profissional em Gestão de Sistemas de Engenharia

"A DEEP LEARNING MODEL FOR SEMANTIC SEGMENTATION OF INTRAVASCULAR ULTRASOUND IMAGES"

Mestrando: **Daniel Cunha de Araújo Júnior**

Orientador: **Pablo Javier Blanco**

Petrópolis, 14 de dezembro de 2022.

Banca Examinadora:

Prof. Dr. Pablo Javier Blanco, UCP.

Prof. Dr. Giovane Quadrelli, UCP.

Prof. Dr. Vitor Hugo Ferreira, UFF.

Este trabalho é dedicado a minha mãe,
que sempre ansiou pela medicina.

Agradecimentos

Agradeço principalmente à Deus, autor da minha salvação, a meus pais, por terem investido tanto em mim, e aos professores que realmente se importaram com a minha aprendizagem.

Agradecimentos também ao grupo HeMoLab pelas sugestões e especialmente ao meu orientador Pablo Javier Blanco pela paciência nos momentos difíceis. Agradeço ao ainda ao grupo HeMoLab e também ao LNCC por disponibilizarem os recursos computacionais necessários para a execução deste trabalho, sem os quais tal não seria possível.

E vós também, pondo nisto mesmo toda a diligência,
acrescentai à vossa fé a virtude, e à virtude a ciência,
E à ciência a temperança, e à temperança a paciência,
e à paciência a piedade, e à piedade o amor fraternal,
e ao amor fraternal a caridade.
(Bíblia, 2 Pedro 1:5-7)

Resumo

Imagiologia médica, que é a geração de imagens para auxiliar no processo de diagnóstico e sequente tratamento, tem se mostrado um campo fértil para a aplicação de tecnologias de Deep Learning: com o uso de Deep Learning se torna possível a um computador executar tarefas que antes só eram possíveis de serem executadas com recursos humanos, muitas vezes com maior acurácia e eficiência, tendo assim ganhado interesse por parte da comunidade médica. Este trabalho, feito em parceria entre a UCP e o HeMoLab (LNCC), visou estudar a aplicação de técnicas de Deep Learning em imagiologia médica, especificamente a segmentação das regiões de lumen e placa em imagens de Ultrassonografia Intravascular (IVUS), um exame conduzido para investigar doenças de natureza coronárias. Durante a execução deste trabalho, mais de 80 treinamentos com mais de 40 redes neurais complexas foram executados com um dataset de 160 pullbacks disponibilizado pelo HeMoLab, totalizando cerca de 287 mil imagens, com cada treinamento levando cerca de uma semana para completar utilizando o supercomputador Santos Dumont. Os melhores resultados foram encontrados em uma rede neural convolucional (CNN) com 9.01×10^8 parâmetros, que alcançou um IoU de lumen mediano de 0.9076, um IoU de placa mediano de 0.7392 e um IoU de vaso mediano de 0.9331. A solução está considerada pronta para ser utilizada como assistente de diagnóstico, com potenciais ganhos em confiabilidade e aceleração do fluxo de trabalho típico de um exame, reduzindo custos e assegurando segurança.

Palavras-chave: Aprendizagem Profunda, Computação Visual, Imagiologia Médica.

Abstract

Medical Imaging, which is the generation of images to assist in medical analysis and treatment, has shown to be a thriving field for the application of Deep Learning techniques: with the use of Deep Learning it is possible for a computer to perform tasks previously thought accomplishable only by trained technicians, often with increased accuracy and efficiency, thus being of great interest for the medical community. This work, done in a partnership between UCP and HeMoLab (LNCC), aimed to study the application of Deep Learning techniques in medical imaging, specifically the segmentation of Lumen and Plaque regions in IVUS imaging, an exam conducted in order to investigate coronary diseases. During this work's execution, over 80 trainings with more than 40 different resource intensive networks were performed on HeMoLab's dataset of 160 pullbacks, totalling 287 thousand frames, with each training taking around one week to complete on Santos Dumont supercomputer. The best results were found in a convolutional neural network with 9.01×10^8 parameters, which achieved a median lumen IoU of 0.9153, a median plaque IoU of 0.7679 and a median vessel IoU of 0.9365. The best results were found in a convolutional neural network with 9.01×10^8 parameters, which achieved a median lumen IoU of 0.9076, a median plaque IoU of 0.7392 and a median vessel IoU of 0.9331. The solution is considered ready to be used as an AI assistant for medical diagnosis, with potential gains in reliability and acceleration of the typical exam workflow, reducing costs while assuring safety.

Keywords: Deep Learning, Computer Vision, Medical Imaging.

List of Figures

Table of Contents

1 Introduction

Deep Learning techniques, a branch of Machine Learning, allow a computer to learn subtle relationships between an input x and a desired output y, often hard or infeasible to be explicitly programmed. Since the advent of modern GPUs, the field has grown to include applications in several tasks, including for instance demand prediction, fraud detection, natural language processing, and as in the case of this work, computer vision.

Medical Imaging, which is the generation of images to assist in medical analysis and treatment (BUSHBERG, 2012), has shown to be a thriving field for the application of Deep Learning techniques: with the use of Deep Learning it is possible for a computer to perform tasks previously thought accomplishable only by trained technicians, often with increased accuracy and efficiency, thus being of great interest for the medical community.

Table 1 – Deep Learning in Medical Imaging Examples

Segmentation Target	Examples
Prostate Cancer	(FELDMAN et al., 2019);
Breast Cancer	(ALMBERG et al., 2022); (BUELENS et al., 2022);
Brain Tumor	(CHEN et al., 2019);
Lymph Node	(TAKU et al., 2022);
Colon Cancer	(KASSANI et al., 2022);
Heart	(CHEN et al., 2021);
Abdominal Organs	(CHEN et al., 2021);
Artery Lumen	(ZIEMER et al., 2020);
Lungs	(GORDIENKO et al., 2019); (ISLAM; ZHANG, 2018); (SKOURT; HASSANI; MAJDA, 2018); (GORDIENKO et al., 2019);

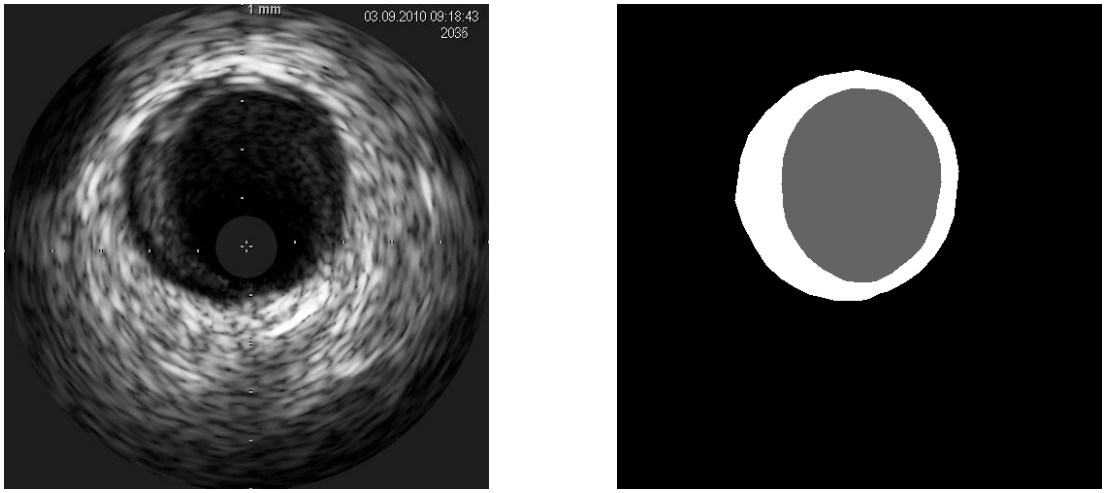
Source: Author

Among applications of deep learning in medical imaging, image segmentation stands out as one of the most interesting and promising research prospects: a neural network could, for instance, be tasked with detecting a brain tumor (CHEN et al., 2019), receiving a CT scan and returning the same scan annotated with the tumor highlighted. The process could be automated further: with tumor's segmentation available its properties, such as size and shape, could be assessed immediately and returned all at once. Table 1 exemplifies some recent applications of Deep Learning for segmentation tasks in medical imaging.

As aforementioned, the use of deep learning in medical imaging can lead to increased reliability, automation and acceleration of the typical exam workflow, as the desired segmented images would otherwise have to be manually segmented by a trained technician. This manual segmentation process is particularly onerous in the case of the Intravascular Ultrasonography (IVUS) exam, an important exam typically used in the assessment of a patient's coronary artery disease, where the technician has to manually annotate lumen and vessel regions on thousands of different images, becoming thus one of the potential applications of Deep Learning. While the technology might not be advanced enough for full automation, it could be used to provide assistance, for instance, such that a technician doesn't need to manually segment every frame, but only correct the bad ones.

For such application, Figure 1 exemplifies a typical input and output pair used in training and evaluation of a model:

Figure 1 – Input (left) and desired output (right). IVUS scan and corresponding segmentation of lumen and vessel contours



Source: LNCC

Deep learning state of the art applications in medical imaging usually rely on state-of-the-art convolutional neural networks (YAO et al., 2020), such as the U-Net (RONNEBERGER; FISCHER; BROX, 2015) or a derivative, such as the U-Net++ (ZHOU et al., 2019) or the TransUNet (CHEN et al., 2021).

1.1 Objective

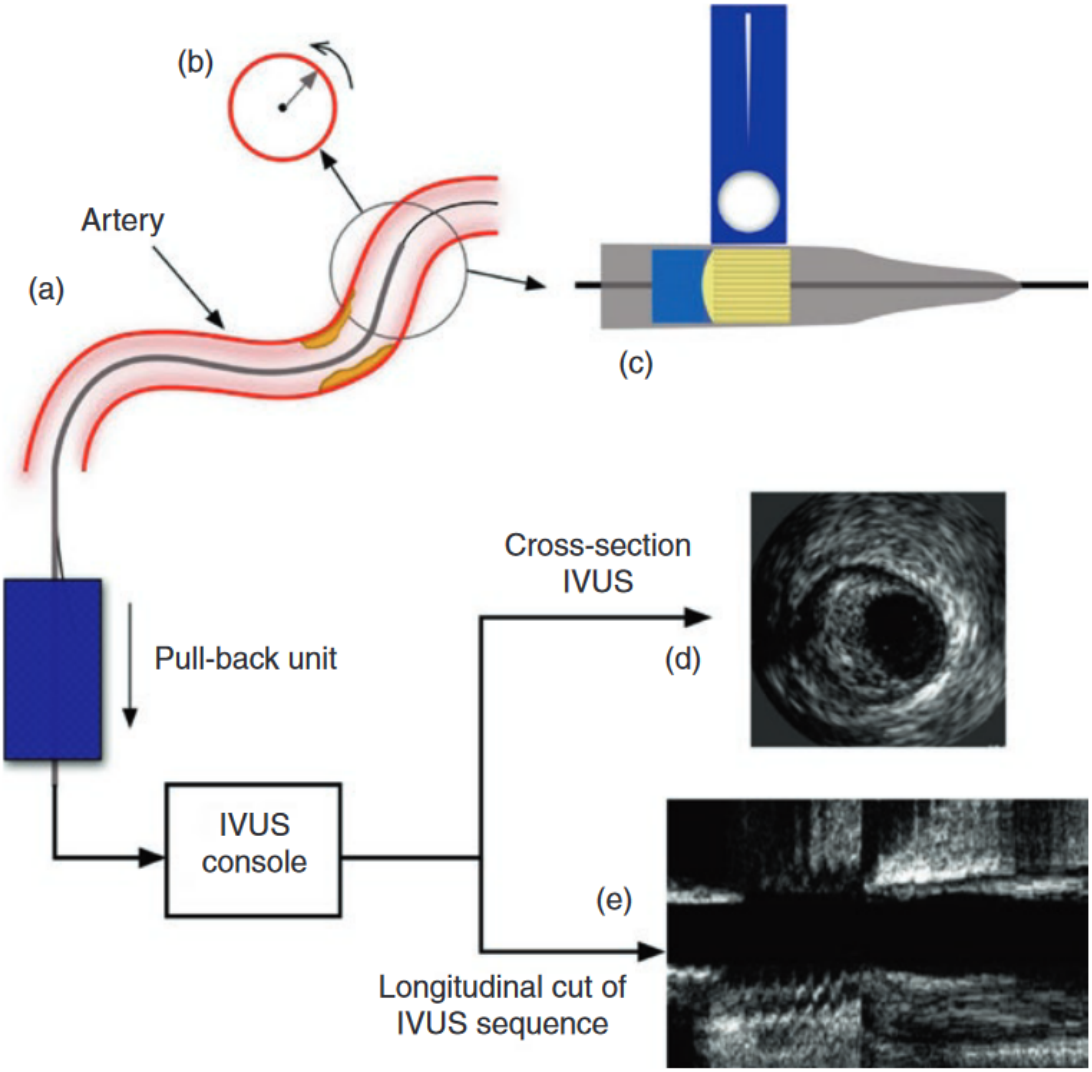
This work, done in a partnership between UCP and HeMoLab (LNCC), aims to study the application of Deep Learning techniques in medical imaging, specifically the segmentation of Lumen and Plaque regions in IVUS imaging, an exam conducted in order to investigate coronary diseases. It is motivated by the author's desire to learn the associated Deep Learning techniques and contribute to state-of-the-art medical research, a desire shared by HeMoLab.

1.2 Intravascular Ultrasonography - IVUS

The Intravascular Ultrasonography (IVUS) is a diagnostic procedure that, through the use of a catheter alongside ultrasound technology, allows the visualization of an artery's lumen, deposits and other correlated structures. It is often used in order to measure the artery's stenosis/plaque burden (how abnormally narrow an artery is) (ZIPES et al., 2019).

Figure 2 showcases the IVUS procedure with a mechanical catheter: after being manually inserted inside the artery, the catheter is pulled back (extracted) by a pull-back unit at a constant linear speed. In the case of the mechanical catheter, rotating ultrasound sources are used at constant angular velocity. Alternatively, phased array catheters would use sequentially flashing ultrasound sources. Finally the retrieved data is then processed by a console into the desired grayscale images. (HONG, 2018)

Figure 2 – IVUS Diagram

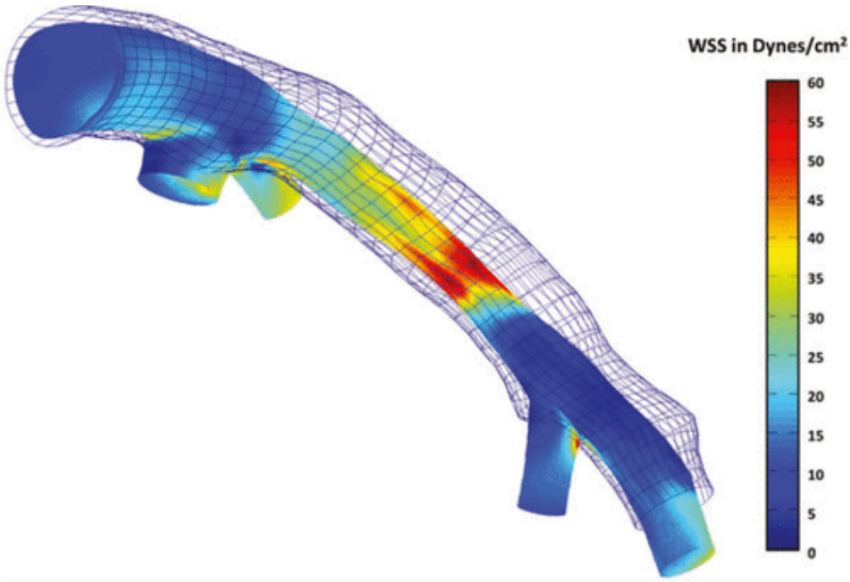


Source: HONG, 2018

These grayscale images, once appropriately retrieved, are analyzed by a trained technician, who will annotate the lumen and vessel contours. Afterwards, with the appropriate processing, metrics such as the plaque burden are retrieved: the plaque burden is calculated as the plaque area divided by the area encapsulated by the external elastic membrane (EEM, referred to as "vessel"in this work). (MCDANIEL et al., 2011; ESHTEHARDI et al., 2012)

Naturally, with the appropriate lumen and vessel segmentations of the artery at disposal, other, more complex studies can be performed, such as, for instance, a computational fluid dynamics (CFD) simulation to estimate wall shear stress (WSS), as exemplified in Figure 3.

Figure 3 – CFD Simulation of Wall Shear Stress in an Artery



Source: ESHTEHARDI et al., 2012

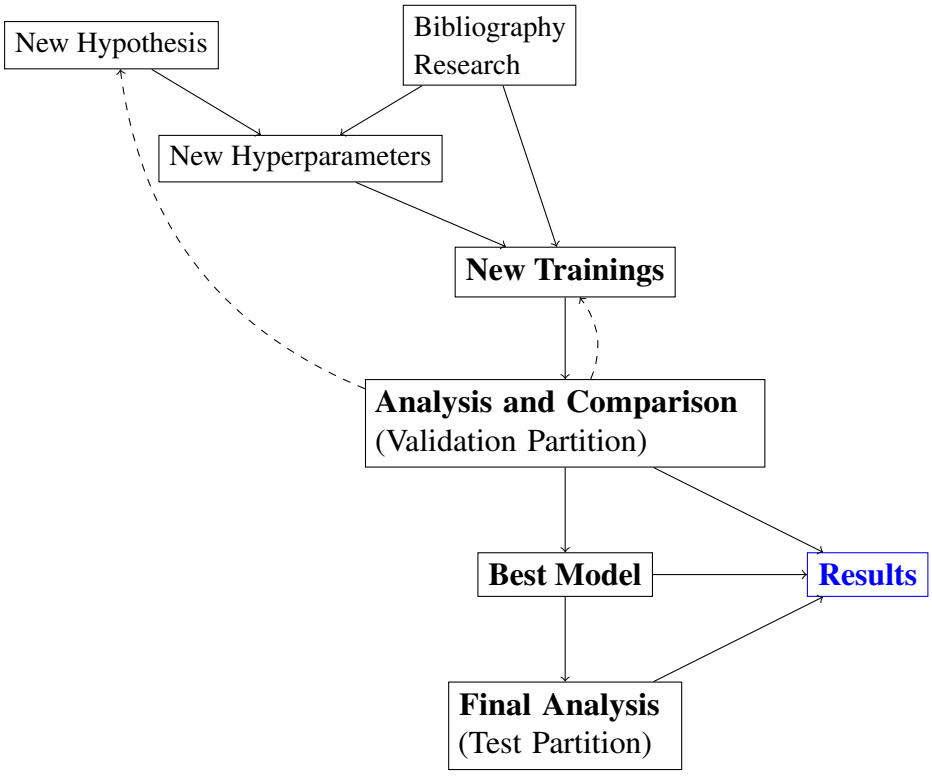
2 Methodology

In order to attempt to improve the state of the art IVUS segmentation technology and to better understand possible paths for further improvement, a quantitative, practical and bibliographical research was applied. Figure 4 illustrates the applied methodology. Once a possible improvement was identified, the implemented code was parameterized as necessary (creating thus new hyperparameters), a test training was made, and the results reported. Each training was given an ID number for easy reference and reported in the appendix A. Naturally, due to time and resource constraints not every promising path could be tested (or fully explored), limiting the set of searched hyperparameters.

One of the main objectives of the bibliographical aspect of this work is to find possible alternatives in order to improve current results, which by consequence affected the multiple hyperparameter combinations that were tested. Understanding said hyperparameter combinations as a grid search, a practical way of understanding this work's methodology is by understanding it as a months-long grid search where the grid itself was the output of a (subjective) Delphi survey. In order to infer the effect of the parameter in the segmentation's quality, hyperparameters were mostly changed one at a time, while keeping all others constant.

Trainings were performed at a fixed number of epochs, and then extended, also by a fixed

Figure 4 – Methodology



amount of epochs, where deemed necessary. While the main motivation behind this strategy was not having to wait a long time in order to judge if a specific hyperparameter combination was working as intended (which extends to bug-catching), it also allowed some neural networks to be analyzed at multiple epoch values and minimized the amount computational resources and time needed for the completion of this work. In order to address the risk of the comparisons becoming skewed towards fast converging networks, the training curve was also analyzed during the decision making of whether or not to extend training time. Nevertheless, due to the aforementioned time constraint not every possible extended training was executed.

It should be noted that in order to compare different network structures, some studies attempt to track and limit the network's number of parameters variation. Such approach is not used in this work. Given that there are enough computational resources and the total training time is reasonable (1-2 weeks maximum per ID), the only concern was increasing the quality of the segmentation.

Finally, although training until convergence tends to minimize the models' quality metrics' variance, the results presented in this work should still be treated as samples of a distribution, as opposed to standalone "definitive" values. Unfortunately, the amount of computational resources needed to draw multiple samples of such a distribution (each "sample" requiring a complete training of a network) alongside the time constraints of this work make deeper statistical studies unfeasible.

It should be noted that some requirements and orientations from HeMoLab's part specified this work's scope further. For instance, it was desired that the provided dataset and its predefined partitions were used as is (further explanation of the dataset in section 2.2) and that frames were output at a similar resolution than the one observed in the dataset, approximately 500×500 pixels. These requirements occasionally led to some developments or workarounds; the resolution requirement, for example, led to the development of the Downsizing strategy (see section 2.4.6).

2.1 Deep Learning Technologies

This section is dedicated to presenting and explaining some of the main deep learning techniques to be used within this work.

2.1.1 Convolutional Networks in Semantic Segmentation

The application of semantic segmentation means that the network's task is to detect certain shapes in the input data, as exemplified in Figure 5. This is done by making the network run a pixel-wise classification of the input image: the network decides to which class each pixel

belongs to, outputting for each pixel, a vector of probabilities regarding each class.

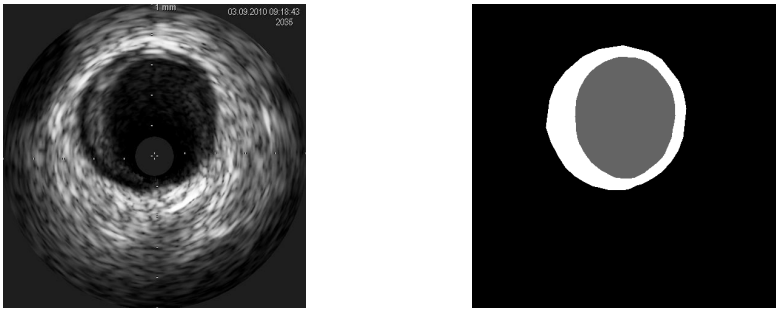
Figure 5 – Example Input and Output Segmentation from the VOC2012 Challenge (PASCAL 2, 2012)

(a) Input Image (b) Output Segmentation

Source: PASCAL 2, 2012

As previously exemplified by Figure 1 and shown again below in Figure 6 for convenience, in the context of this work, it means that the network will (attempt) to classify each pixel in one of the three classes shown in Figure 6b: *External* (in black), *Lumen* (gray) and *Plaque* (white).

Figure 6 – Example input and output segmentation



(a) Input Image (b) Output Segmentation

Source: LNCC

One way of understanding (convolutional) neural networks is seeing them as a pipeline of mathematical operations to be applied on an input, transforming it into a desired output (GOODFELLOW; BENGIO; COURVILLE, 2016). They could also be understood as functions that translate an input from one domain to another, similar to how a function f takes an input value x and returns a value y, such that $y = f(x)$.

Given that many of these operations have parameters that must be specified in order to apply the transformation (for instance, the weights in a weighted sum), these are automatically "learned" (optimized) through an optimization routine (KINGMA; BA, 2017) in order to obtain best results.

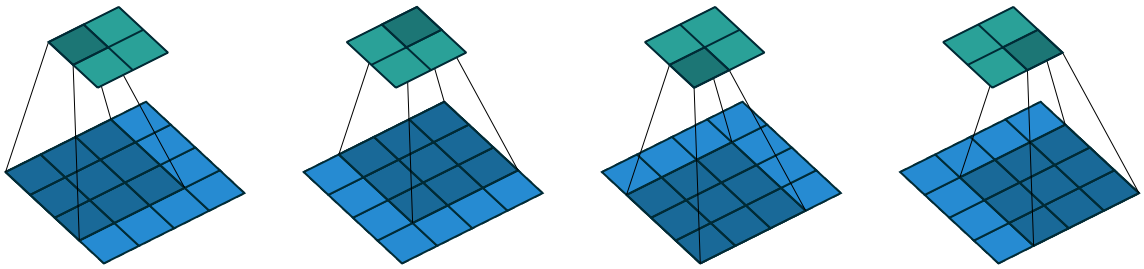
Each operation (also called layer) fulfills a specific need in the domain translation process, be it the detection of low or high level features, the introduction of nonlinearities (so

that the network might be able to learn complex relationships), the decrease of the dimensional complexity (so that the network can be ran with limited computational resources), the rescaling of a vector of values in such way that they might be understood as probabilities, among others. (GOODFELLOW; BENGIO; COURVILLE, 2016) That said, the sections below present some of the main layers commonly used for this class of problems.

2.1.1.1 Convolution Layer

Within the context of Deep Learning or Convolutional Networks, a convolution operation is an operation where a kernel, also called filter, walks along a vector/matrix/tensor's dimensions, processing a local neighbourhood and producing an output value for each position it has passed (Figure 7). It is capable of processing an image (which can be interpreted as a matrix or tensor of pixel values) while inherently taking spatial information in consideration (GOODFELLOW; BENGIO; COURVILLE, 2016), as can be seen in Figure 7:

Figure 7 – A Convolution Operation



Source: DUMOULIN; VISIN, 2016

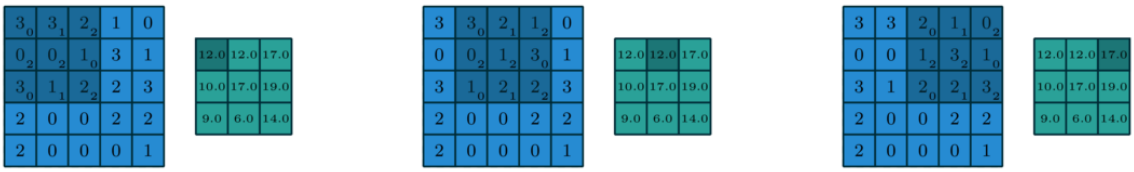
A kernel is essentially a matrix or a tensor of weights and a bias term (GOODFELLOW; BENGIO; COURVILLE, 2016). For instance, let k_1 be a 3×3 kernel to be applied over an arbitrary grayscale image (a matrix) A, and k_1's bias term be equal to 0:

$$k_1 = \begin{bmatrix} 0 & 1 & 2 \\ 2 & 2 & 0 \\ 0 & 1 & 2 \end{bmatrix} \tag{2.1}$$

With the kernel's weights and bias defined, the image can then be processed. With B as the output matrix:

Note that in its simplest form, a convolution's output will have reduced dimensionality, which from a naive point of view might be of interest, but in practice leads to problems, due to the often need of performing operations such as concatenating or adding tensors. Because of such, the input image is often padded with rows and columns of zeros in such way that the output image dimensions are equal to the input's original dimensions:

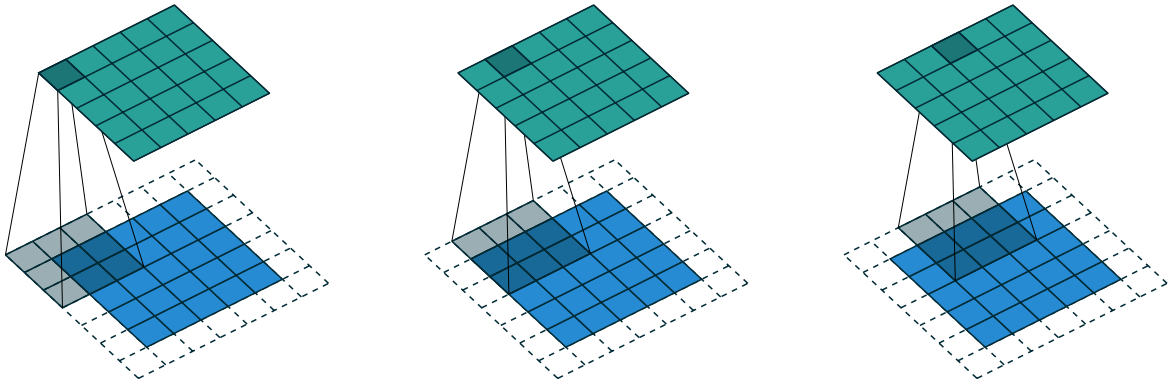
Figure 8 – Convolution Operations Performed in an Image



(a) $b_{11} = 12.0$ (b) $b_{12} = 12.0$ (c) $b_{13} = 17.0$

Source: DUMOULIN; VISIN, 2016

Figure 9 – Padded Convolution Operations Performed in an Image



Source: DUMOULIN; VISIN, 2016

2.1.1.2 Rectified Linear Unit (ReLU) and Parametric ReLU (PReLU) Activation Layers

The Rectified Linear Unit (ReLU) is an activation function commonly used in state-of-the-art deep neural network applications, consisting of the positive part of the input (see Eq. 2.2) (XU et al., 2015):

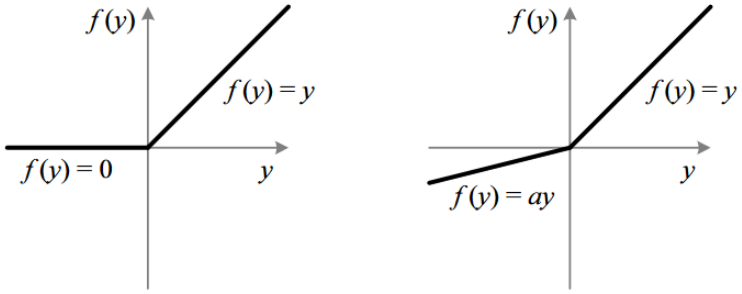
$$f(y) = max(0, y) \tag{2.2}$$

An alternative to the ReLU, called Parametric Rectified Linear Unit (PReLU), is proposed in order to improve results (see Eq. 2.3). (HE et al., 2015)

$$f(y) = \begin{cases} ay, & \text{if } y \leq 0 \\ y, & \text{if } y > 0 \end{cases} \tag{2.3}$$

While in the ReLU activation when $y < 0$ the value of $f(y)$ is equal to 0, in the PReLU activation the value of $f(y)$ is such that $f(y) = ay$, where a is a trainable parameter. Fig. 10 illustrates the difference between ReLU and PReLU approaches.

Figure 10 – ReLU vs. PReLU

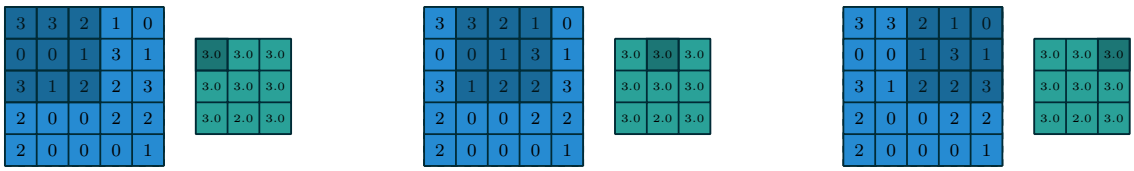


Source: HE et al., 2015

2.1.1.3 Max Pooling and Up-Sampling Layers

Max Pooling is a technique used in segmentation tasks to decrease the dimensional complexity of a tensor, by reducing the size of the feature maps (usually by half per layer), while also summarizing a local neighbourhood (DUMOULIN; VISIN, 2016). It follows the same idea of the tradicional convolution mentioned in section 2.1.1.1. However, instead of performing a linear combination it simply returns the maximum value of a neighbourhood, as exemplified by Figure 11:

Figure 11 – Max Pooling Operations Performed in an Image



Source: DUMOULIN; VISIN, 2016

On the other hand, an upsampling layer will increase the size of the feature maps (usually two-fold). In its simplest form, it merely repeats a given item in every direction, although more complex algorithms could be used, such as a bilinear interpolation (Keras Team, 2022f). For instance, let A be a 2×2 matrix. Performing a typical upsampling operation on A will yield:

$$U(A) = U\left(\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}\right) = \begin{bmatrix} a_{11} & a_{11} & a_{12} & a_{12} \\ a_{11} & a_{11} & a_{12} & a_{12} \\ a_{21} & a_{21} & a_{22} & a_{22} \\ a_{21} & a_{21} & a_{22} & a_{22} \end{bmatrix} \tag{2.4}$$

2.1.1.4 Softmax and Sigmoid Layers

The Softmax function is often used to transform the numerical output of a neural network, allowing them to be interpreted as probabilities (MURPHY, 2012). In essence, let X be an input vector $\{x_1, x_2, x_3, \cdots, x_n\}$ to be converted to a vector of class probabilities for a pixel. The output of a softmax function is given by:

$$\sigma(X) = \left\{ \frac{e^{x_i}}{\sum_{j=1}^{n} e^{x_j}} \quad \forall \quad i = 1, 2, 3, \cdots, n \right\} \tag{2.5}$$

In the case of a binary classification (that is, $|X| = 1$ and $X = \{x_1\}$), another function is used with similar purpose: the sigmoid function (GéRON, 2019). Referring to x_i as simply x, it is given by:

$$S(X) = \left\{ \frac{1}{1 + e^{-x}} \right\} \tag{2.6}$$

2.1.1.5 U-Net

The U-Net, named after its shape (see Fig. 12), is a convolutional neural network developed for medical image segmentation (RONNEBERGER; FISCHER; BROX, 2015), being based on the Fully Convolutional Network (SHELHAMER; LONG; DARRELL, 2017). It has been used for various segmentations tasks, including the application being studied in this work, the segmentation of lumen and vessel's contour from IVUS scans (ZIEMER et al., 2020).

The main idea being implemented in the U-Net is to append to the encoder (the left side of the U-shape in Fig. 12) a expansive path, where instead of successive max pooling operations upsampling operations are used. High resolution features are then concatenated with the upsampled features, giving the network the ability to produce detailed segmentations (RONNEBERGER; FISCHER; BROX, 2015).

Figure 12 – U-Net

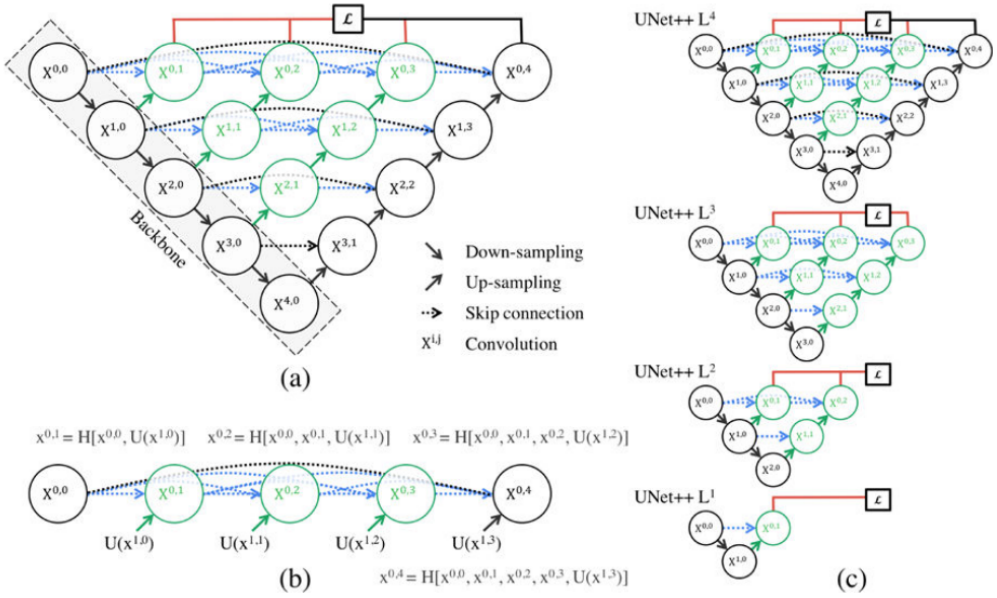
Source: RONNEBERGER; FISCHER; BROX, 2015

2.1.1.6 U-Net++

The U-Net++ (ZHOU et al., 2018a) is a neural network that builds on the idea of the U-Net. It can be understood as multiple U-Nets nested together.

It's main hypothesis is that by altering the connectivity between the encoder and decoder sub-networks the training process can be improved: by switching the U-Net's "direct" skip connections for dense convolution blocks (see Fig. 13) adjacent feature maps will be semantically similar, and thus the optimizer will have an easier optimization problem, which could lead to better results (ZHOU et al., 2018a).

Figure 13 – U-Net++ Structure



Source: ZHOU et al., 2018b

2.1.2 Neural Network Training

It was previously mentioned that a neural network will usually have many parameters that need to be optimized in order to perform the desired task. This is done by first feeding batches of inputs to the network and them comparing the network's outputs to the desired outputs through the lens of a loss function. Afterwards, through a technique called backpropagation, an optimizer (usually Adam (KINGMA; BA, 2017)) will weight each parameter's contribution to the total loss, altering the first in order to optimize the latter. Such optimization is performed by steps, with the learning rate (that is, how large each step is) being controlled by the optimizer's algorithm. (GOODFELLOW; BENGIO; COURVILLE, 2016)

Considering a practical application, where the set of available data is limited, samples will often be reused to train the network; in that context each full pass through the dataset is called

an epoch. However, before using the data, it must be split into sensible partitions (usually one partition for training, one for validation and one for testing) due to issues related to the limited data available. From the perspective of the aforementioned domain transformation function interpretation of neural networks, only a discrete fraction of the input's and output's domains are available to be seen and analyzed. A network could, for instance, fail to be able to generalize to previously unseen samples. (GOODFELLOW; BENGIO; COURVILLE, 2016)

As an analogy, consider a student that in order to prepare for a test, memorizes every exercise in the textbook instead of learning the logic behind the correct answers. When the test comes, they can't answer a single question, since it was not previously seen in the textbook.

In the process of developing a deep learning-based solution, many parameters need to be decided *before* the previously mentioned optimization (training) routine (also called "training") is executed. For instance, the batch size, which refers to how many training samples will be fed to the network at a time during training, needs to be decided before the training starts. These types of parameters are usually called *hyperparameters* to differentiate from the ones that will be optimized during training. That said, the following sections present a brief explanation of some of the loss functions and metrics used.

2.1.3 Metrics and Loss

2.1.3.1 Cross-Entropy

Cross-Entropy is commonly used as a loss in Deep Learning applications. It essentially measures the difference between two probability distributions. It is built on the concept of entropy, which relates to how uncertain a random variable is. (MURPHY, 2012)

With n as the number of classes, t_i and p_i as the probabilities of the i-th class given by the ground truth (the ideal output) and the model's prediction, respectively, the Cross-Entropy of a *pixel* is given by:

$$CE = -\sum_{i=1}^{n} t_i \log p_i \tag{2.7}$$

In practical implementations, the average Cross-Entropy across all pixels is used. That is, with M as the set of pixels in an image:

$$CE = -\frac{1}{|M|} \sum_{m \in M} \sum_{i=1}^{n} t_{im} \log p_{im} \tag{2.8}$$

Where $|M|$ is the size of the set M.

2.1.3.2 IoU - Jaccard Index

Also known as the Jaccard Index (JACCARD, 1912), the IoU metric is defined as the area of intersection over the area of union. If A and B are the models' segmentation and the ground truth's segmentation, then the IoU is given by:

$$IoU = \frac{|A \cap B|}{|A \cup B|} \tag{2.9}$$

To use the concept of IoU as a loss for neural network training, an approximation, referred to as IoU* (BEERS et al., 2019), can be deduced by treating A and B as pixel-wise probabilities:

$$IoU* = \frac{A * B}{A + B - (A * B)} \tag{2.10}$$

This adaptation is necessary to ensure that the loss calculation routine can appropriately calculate the derivative. Thus, IoU loss can be defined as:

$$IoU_{Loss} = 1 - IoU* \tag{2.11}$$

2.1.3.3 Mean IoU

A readily available metric (Keras Team, 2022c), it is simply the mean of the IoUs of the set comprised of all classes plus each class' complement. That is, for each class c, the corresponding class not-c, or c'. Taking a set C of classes and their IoU (eq. 2.9), the Mean IoU is given by Equation 2.12.

$$mIoU = \frac{1}{2|C|} \sum_{c \in C} IoU_c + IoU_{c'} \tag{2.12}$$

2.1.3.4 DICE - Sørensen-Dice Index

Also known as the Sørensen-Dice Index (SøRENSEN, 1948; DICE, 1945), the DICE metric is of similar nature to the IoU, being often used to access a models' segmentation's quality:

$$DICE = \frac{2|A \cap B|}{|A| + |B|} \tag{2.13}$$

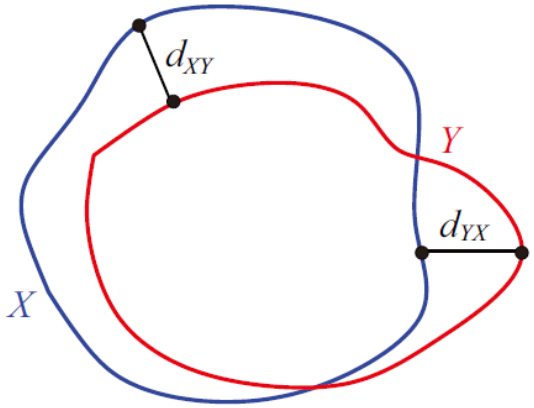
2.1.3.5 Hausdorff Distance

The metric known as Hausdorff distance (ROCKAFELLAR; WETS, 2004; CERAGIOLI, 2006) refers to the maximum distance of a set to the nearest point in the other set. That is, let X and Y refer to two contours. Hausdorff distance is given by:

$$HD = \max(d_{XY}, d_{YX}) = \max\left(\max_{x \in X}\left(\min_{y \in Y} d(x,y)\right), \max_{y \in Y}\left(\min_{x \in X} d(x,y)\right)\right) \tag{2.14}$$

Where $d(x,y)$ refers to the distance between two points x and y. In a perhaps friendlier definition, Hausdorff distance could be understood as a measure of how much two contours diverge in the worst case scenario. Figure 14 illustrates this:

Figure 14 – Hausdorff Distance Between Two Sets



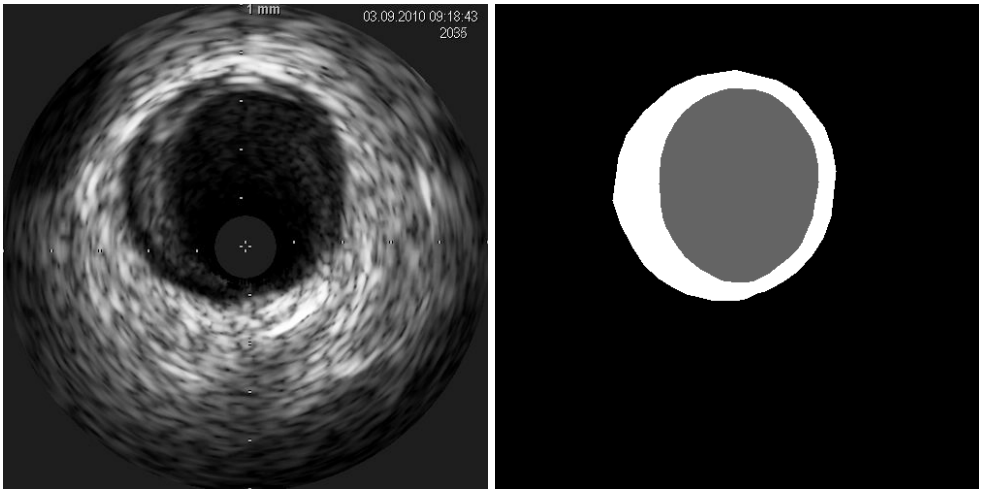
Source: Radboud University Medical Center, 2019

2.2 IVUS Dataset

2.2.1 Available Dataset

The dataset provided by LNCC is comprised of 160 pullbacks of varying frame amounts and already came with pre-defined training (60%), validation (20%) and testing (20%) partitions done at pullback level, being split in such way that both the resulting total amount of frames and the resulting total amount of pullbacks in each partition approximated a 60-20-20 split. The figure below exemplifies the dataset's typical input frame and desired segmentation mask:

Figure 15 – Raw input image (left) and Segmentation Mask (right)



Source: LNCC

2.2.2 Data Augmentation

Given that an artery i has n_i frames (images) to be extracted, let A_i denote the ideal set of ultrasound frames, to be extracted by the pullback unit. Due to problems during the extraction, such as blurring caused by heart movement, only a proper subset R_i of A_i is actually available, as shown by equation 2.15:

$$R_i \subsetneq A_i \,, \qquad A_i = \{X_j \quad | \quad j = 1, 2, 3, \cdots, n_i\} \tag{2.15}$$

The R_i set of available frames is naturally corresponded by a G_i set of available ground truths, which are the manual segmentations done by the technicians. Equation 2.16 illustrates the relationship between both sets.

$$G_i = \{Y_j \quad | \quad X_j \in R_i\} \tag{2.16}$$

With both sets available, a data augmentation technique was applied with the intention of increasing the available amount of data. It was also desired to find an approximation to the frames that were not available due to the aforementioned issues in extraction ($A_i \backslash R_i$). This led to the development of a hybrid set of ultrasound frames R'_i, constructed by adding pixel-wise linear combinations of nearby frames, as surrogates for the ones missing. The corresponding set of hybrid ground truths G'_i was created by interpolating the contours of nearby frames. Tables 2 and 3 showcase some descriptive statistics regarding the distribution of hybrid and native frames from a partition-level perspective and an artery-level perspective, respectively.

As shown in Tables 2, a grand total of around 288 thousand frames are available in the full dataset. Of those, 13.5 thousand (around 5%) are native. Similar hybrid/native ratios are observed for all partitions, at approximately 21 hybrid frames for each native frame.

Table 2 – Hybrid and Native Frames' per Partition Statistics

Partition	Hybrid	Native	Hybrid/Native Ratio	Native Percent in Hybrid
Train	180985	8402	21.54	4.64%
Validation	54310	2583	21.03	4.76%
Test	52606	2548	20.65	4.84%
Total	287901	13533	21.27	4.70%

Source: Author

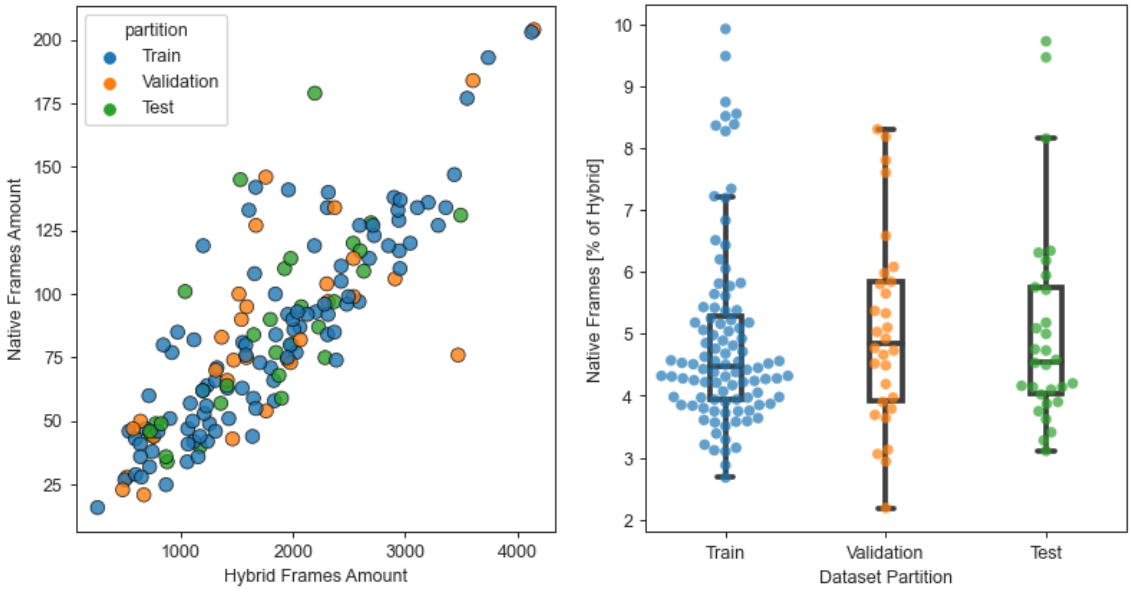
Table 3 – Hybrid and Native Frames' per Artery Statistics

	Hybrid	Native	Hybrid/Native Ratio	Native Percent In Hybrid
Mean	1799.38	84.58	21.87	4.95%
Std	840.89	39.07	5.83	1.52%
Min	258.00	16.00	10.08	2.19%
25%	1167.75	51.00	18.27	3.94%
50%	1732.50	80.00	22.07	4.53%
75%	2329.25	110.25	25.37	5.47%
Max	4149.00	204.00	45.70	9.92%

Source: Author

As shown in Table 3, each artery has an average of around 1800 hybrid frames, of which around 85 are native. Comparing with the per partition analysis, a larger variation on the percentage of native frames is evident, with some arteries going as low as 2.19% percent of natives. Figure 16 further illustrates the observations.

Figure 16 – Native and Hybrid Frames Distribution



Source: Author

2.3 Implementation and Environment

Code implementation was written by the author in Python 3.7 with the use of Keras (CHOLLET; others, 2015) and TensorFlow APIs (TensorFlow Developers, 2022b), version 2.4.1. Other libraries, such as Scikit-Image, Pandas and Seaborn were used for utilitarian functions and presenting results.

Trainings were processed in the Santos Dumont, the supercomputer present in Laboratório Nacional de Computação Científica (LNCC, 2022). Specifically, the supercomputer's BullSequana nodes were used (one node was used per training), with each node consisting of:

- 2x Intel Xeon CPUs
- 48 cores (24 cores per CPU)
- 384Gb RAM memory
- 4x NVIDIA Volta V100 GPU

2.4 Hyperparameters Explanation

Table 4 summarizes the range of parameters considered in the present work:

It should be noted that some parameter choices can impose limitations on other parameters, whether by inherent logic in the network construction or by raw consumption of

Table 4 – Hyperparameters Values Considered

		Searched Parameters
Fit Config	Batch Size	8, 32, 4, 16, 12
	Dataset	Hybrid, Native
	Epochs	30, 50, 70, 90, 110, 641, 107
	Learning Rate	0.001, 0.0003, 0.0001
	Loss	Crossentropy, IoU, IoU + Crossentropy
	Optimizer	Adam
Neural Network	Base Filters	32, 64, 16, 8, 128
	Depth	6, 7, 5, 8, 9, 3, 4
	Downsizing	1, 2, 4
	Input Normalization	False, True
	Kernel Size	3, 5, 7, [9, 5], [9, 3]
	Macro Structure	U-Net++, U-Net, S-Net++
	Multi-Output	False, True
	Multi-channel	[1, 1], [3, 1], [5, 1], [7, 1], [9, 1], [11, 1]
	Node Structure	4, 0, 5, 1, 7, 10, 8
	Pool Factor	2, 4
	T. Conv. Filters	None
	Upsampling	Bilinear, T. Conv.
Tensorflow	Auto-Clustering	True
	Mixed Precision	False, True
	Multi-GPU	False, True

Source: Author

computational resources. For instance, let D be the maximum depth a network can have. If the original resolution of 512×512 is used, then:

$$\frac{512}{2^{D-1}} = 1 \quad \therefore \quad D = 10 \tag{2.17}$$

However, if a downsizing value of 2 is used, the base resolution that the encoder sees will be reduced to 256×256. Therefore the maximum depth possible will also be reduced:

$$\frac{256}{2^{D-1}} = 1 \quad \therefore \quad D = 9 \tag{2.18}$$

The following subsections contain, where needed, a brief explanation of each parameter.

2.4.1 Epochs and Base ID

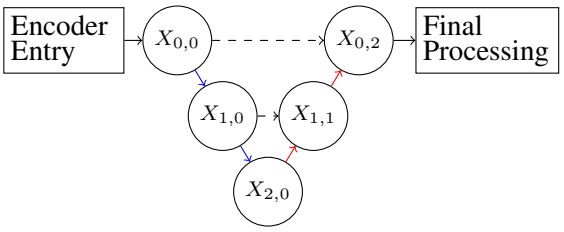
As aforementioned in Section 2, trainings were performed at a fixed number of epochs (usually 30), and where necessary, extended by also a fixed number of epochs (20). This means that for the scope of this work the *Epochs* parameter refers to for how many epochs the network has been trained, including previous trainings.

In the case of extended trainings, a *Base ID* is also reported, which refers to the original training upon which a new one is being made.

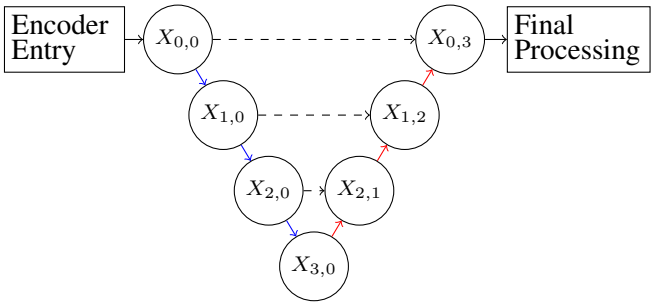
2.4.2 Depth

The $Depth$ parameter refers to how deep the network is. Figure 17 shows the resulting U-net for the depth values of 3, 4 and 5, respectively. Blue arrows indicate Max Pooling operations, red arrows indicate upsampling operations and dashed arrows indicate skip connections (a node's output is simply passed as another's input). When a node's input is comprised of multiple outputs, such as node $X_{0,2}$ in Figure 17a, they are concatenated before being processed.

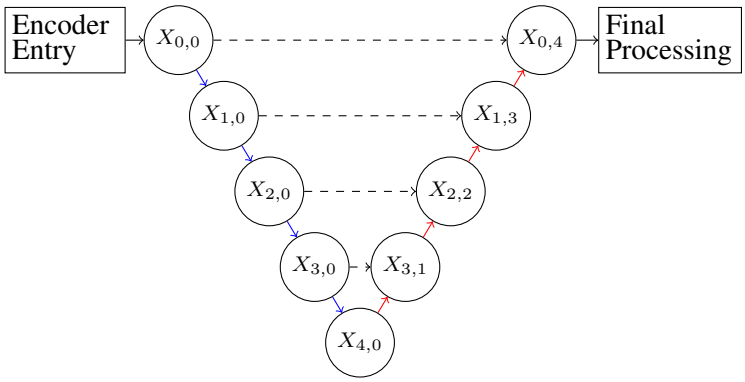
Figure 17 – Resulting U-Net for Varying Depths



(a) Depth value of 3



(b) Depth value of 4



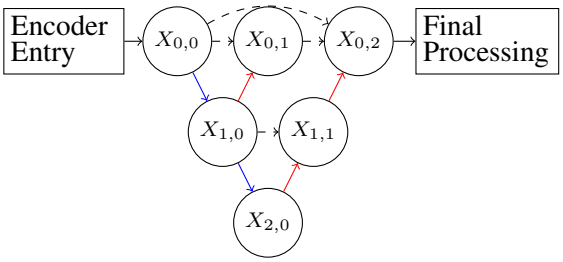
(c) Depth value of 5

Source: Author

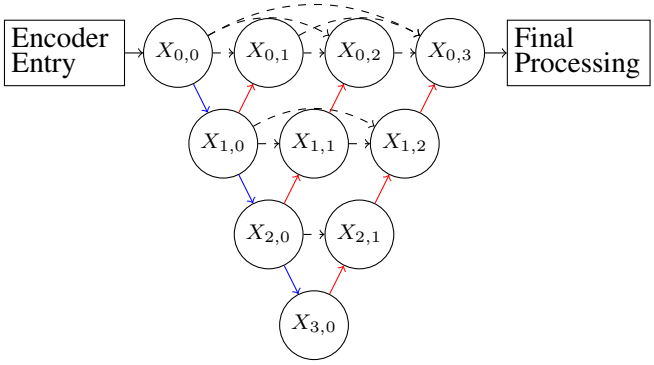
In the case of the U-Net++, the value of $Depth$ will also impact the amount of nodes in the skip-connections. Figure 18 shows the resulting neural network for depth values of 3 and 4,

respectively.

Figure 18 – Resulting U-Net for Varying Depths



(a) Depth value of 3



(b) Depth value of 4

Source: Author

2.4.3 Network Macro Structure

Network Macro Structure or simply *Macro Structure* refers to the general structure of the network. That is, how the network's nodes (convolution blocks) are interconnected. The parameter usually refers to the U-Net or the U-Net++, indicating the presence of the U-Net++ characteristic skip connections. See Figures 17 and 18 for examples.

An alternative called S-Net++ was also tested, being the main difference between it and the U-Net++ the use of strided convolutions instead of max-pooling and strided transposed convolutions instead of bilinear interpolation upsampling. In the case of strided transposed convolutions, kernel size and filters amount refer to the kernel size and filters amount in the strided convolution's output depth level.

2.4.4 Base Filters

The number of filters that a convolution layer will have can be calculated by Equation 2.19, where $BaseFilters$ is the specified parameter, and i is depth of the convolution layer:

$$\text{Number of Filters} = \text{Base Filters} \times 2^i \tag{2.19}$$

Note that in the first depth level i equals 0, not 1. For example, a network with the $BaseFilters$ parameter set to 8 will have 8 filters on first level convolutions, 16 filters on second level convolutions, 32 filters on the third etc.

2.4.5 Upsampling

Upsampling refers to the upsampling strategy utilized in the decoder, among which two were tested: bilinear interpolation and transposed convolutions.

In the case of transposed convolutions, the number of filters used equals the number of filters in depth level of the output. That is, in a transposed convolution operation bringing an input tensor from depth level i to depth level $i - 1$, the number of filters in the operation equals to:

$$\text{Number of Filters} = \text{Base Filters} \times 2^{i-1} \tag{2.20}$$

2.4.6 Downsizing

One possible strategy that might have potential is downsizing the input and output images from the perspective of the neural network: before entering the proper encoder/decoder structure of the network (such as the U-Net or the U-Net++, for instance), the x sample is downsized from the resolution of 512×512 to the resolution of 256×256, and upsampled back to the original resolution immediately before the last activation layer (usually softmax of sigmoid) is applied; before the sample is compared to the y ground truth. Figure 19 illustrates this approach:

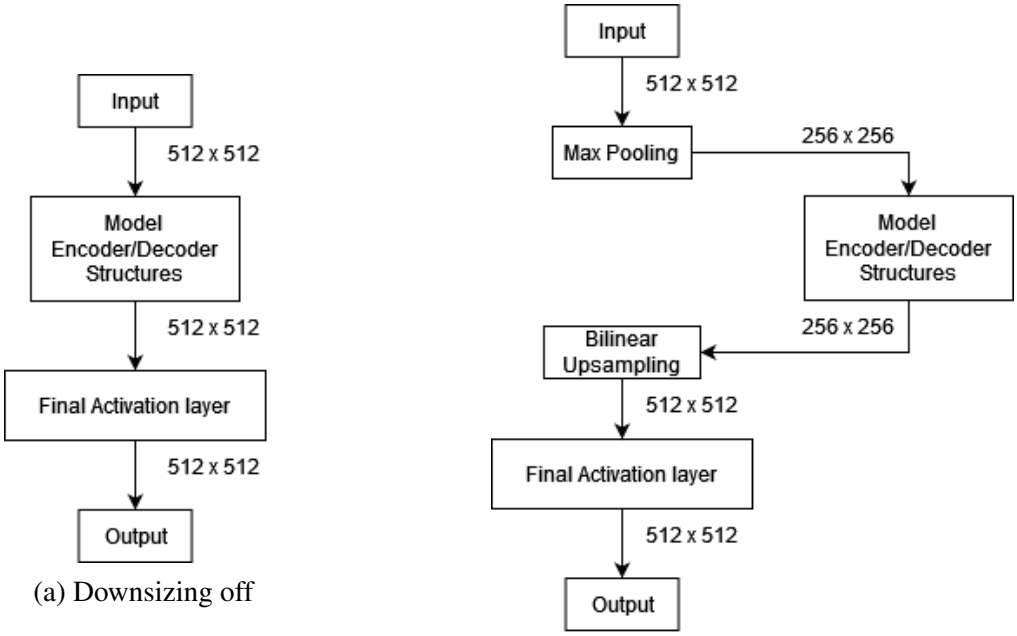
In other words, instead of resizing the input image as a preprocessing step, it's done internally by the network.

While such strategy might cause loss of information due to the decrease of resolution, there are three improvements that might be worth noting:

First, it would severely reduce the computational load (approximately by a factor of 4, in the case of the suggested resolutions above being used), which could be redirected towards other improvements. Second, the complexity of the problem will be reduced. Third, it would artificially increase the kernel size (the kernel field of view would be larger). The area percentage scanned by a 3×3 kernel in a image of 512×512 is given by Equation 2.21:

$$\text{Area Percentage} = 100 \times \frac{\text{Kernel Area}}{\text{Image Area}} = 100 \times \frac{3^2}{512^2} = 0.0034\% \tag{2.21}$$

Figure 19 – Downsizing strategy



(a) Downsizing off

(b) Downsizing factor of 2

Source: Author

However, by downsizing the input resolution to 256×256 the area percentage will increase to (see Eq. 2.22):

$$\text{Area Percentage} = 100 \times \frac{\text{Kernel Area}}{\text{Image Area}} = 100 \times \frac{3^2}{256^2} = 0.0137\% \tag{2.22}$$

Which is 4 times bigger, and thus might improve the overall prediction quality.

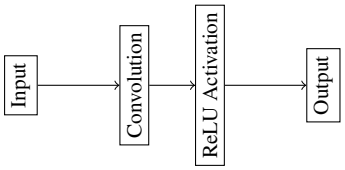
2.4.7 Node Structure

While the overall network has a structure primarily defined by *Network Macro Structure* (see Section 2.4.3) alongside other hyperparameters such as *Multi-Output* (see Section 2.4.9), a choice can still be made regarding the base node structure: That is, there is room for decision making regarding what happens in each $X^{i,j}$ node in Figure 13. Therefore, various node structures can be proposed, as shown in the sections below. Note that some node types implemented were not used in this work (or were not reported), hence the non-continuous enumeration.

2.4.7.1 Node Type 0

Shown by Figure 20, node type 0 is the most basic node structure, referring to the node used in the original U-Net++ (ZHOU et al., 2019).

Figure 20 – Node Type 0: Convolution followed by ReLU Activation

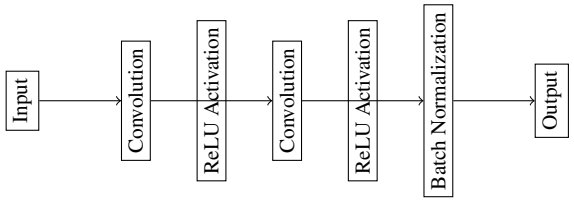


Source: Author

2.4.7.2 Node Type 1

Shown by Figure 21, node type 1 builds on node type 0 by applying the another sequence of convolution-ReLU operations and adding a batch normalization layer at the output.

Figure 21 – Node Type 1: 2x Node Type 0 followed by a Batch Normalization



Source: Author

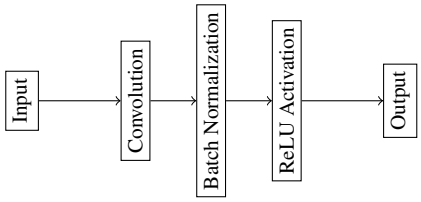
2.4.7.3 Node Type 2

2.4.7.4 Node Type 3

2.4.7.5 Node Type 4

Shown by Figure 22, node type 4 inserts a batch normalization layer between the convolution and ReLU activation proposed in node type 0 (Figure 20). This development was partially inspired by the SegNet's node structure (BADRINARAYANAN; KENDALL; CIPOLLA, 2016).

Figure 22 – Node Type 4: Convolution followed by Batch Normalization and ReLU Activation

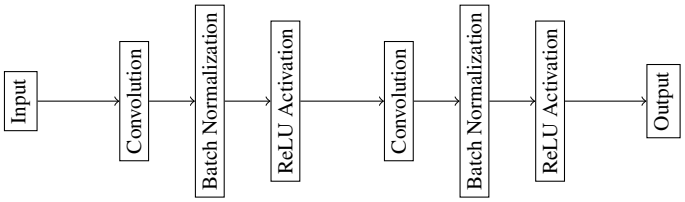


Source: Author

2.4.7.6 Node Type 5

Node type 5 (Figure 23) builds on node type 4 (Figure22) by applying the proposed operation twice, in a sequential manner.

Figure 23 – Node Type 5: 2x Node Type 4 (Sequential)

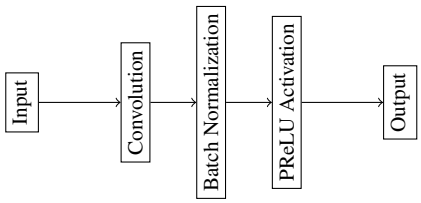


Source: Author

2.4.7.7 Node Type 7

Node type 7 (Figure 24) swaps node type 4's ReLU activation (Figure22) by a PReLU activation (HE et al., 2015).

Figure 24 – Node Type 7: Convolution followed by Batch Normalization and PReLU Activation

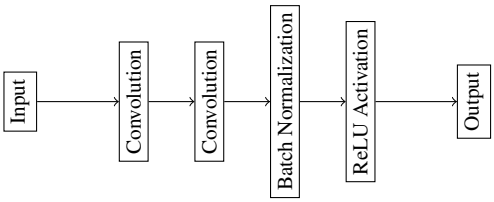


Source: Author

2.4.7.8 Node Type 8

Node type 8 (Figure 25) builds on node type 4's structure (Figure22) by adding a second convolution before proceeding to the batch normalization and ReLU activation layers.

Figure 25 – Node Type 8: Two Convolutions followed by Batch Normalization and ReLU Activation

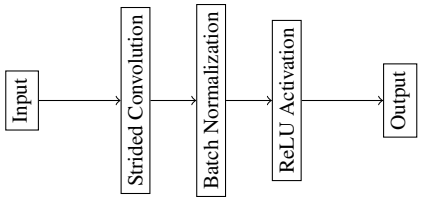


Source: Author

2.4.7.9 Node Type 10

Node type 8 (Figure 25) swaps node type 4's convolution (Figure22) by a strided convolution, to act in place of the traditional max-pooling operations. Note that this node only applies to nodes in the neural network's backbone.

Figure 26 – Node Type 10: Strided Convolution followed by Batch Normalization and PReLU Activation



Source: Author

2.4.8 Kernel Size

The *Kernel Size* parameter refers to the dimensions of the convolution kernels, being thus one of the most resource intensive parameters. Generally, for the scope of this work, a single integer defines the kernel dimensions of every kernel of every convolution in the neural network. That is, a *Kernel Size* of 3 means every kernel is a 3×3 kernel, a *Kernel Size* of 5 refers to a 5×5 kernel, etc.

However, due to the aforementioned resource intensive nature of the parameter together with the desire to increase the kernels' field of view, *Kernel Size* can also be defined as function of depth level with the following notation:

$$KernelSize = [k_0, k_1, k_2 \cdots k_n] \tag{2.23}$$

Where k_0 is the kernel size of all kernels at depth level 0, k_1 is the kernel size of all kernels at depth level 1, and so on. Finally, k_n is the kernel size of all kernels at depth n and of all kernels at a arbitrary depth m such that $m > n$.

For instance, a neural network Y that has a kernel size of 9 in the convolutions of the first depth level, kernel size of 5 in the second and third depth levels and kernel size of 3 in every subsequent depth level will have a *Kernel Size* such that:

$$KernelSize_Y = [9, 5, 5, 3] \tag{2.24}$$

2.4.9 Multi-Output

For the given problem a typical network will perform a per-pixel multi-class classification of the output image in the following strategy: first, at the decoder's end a convolution operation is performed with n filters, where n is the number of classes any pixel can belong to (in the given problem's case, these are *External*, *Plaque* and *Lumen*). Afterwards a softmax activation is performed in order to convert values to a desired range. Finally, in a post-processing step, the predictions for each pixel are converted into proper class labels.

Instead of doing multi-class segmentation, a multi-output strategy is proposed: first, at the decoder's end n convolution operations are performed in parallel (one for each class), each with one filter. Afterwards a *sigmoid* activation is performed. Finally, in a post-processing step, the output of every activation is appropriately processed into the final output, with the appropriate labels.

While a single image is used to represent *Plaque* and *Lumen* classes, there is a "third" class, *Vessel*, from which results are also desired. The *Vessel* class can be defined simply as:

$$Vessel = Plaque \cup Lumen \tag{2.25}$$

Using a Multi-Output strategy, this interesting property of the given problem can be used so that instead of segmenting the standard *External*, *Plaque* and *Lumen* classes, the *Lumen* and *Vessel* classes are segmented. In post-processing the other classes are deduced by the following relationships, which are:

One, the *Plaque* can be defined has the *Vessel* minus the *Lumen*:

$$Plaque = Vessel - Lumen \tag{2.26}$$

And two, the *External* can be defined as the *Vessel*'s complement (in other words, the "not-*Vessel*"):

$$External = Vessel' \tag{2.27}$$

One important aspect of considering this approach is that since the *Plaque* class is geometrically more complex than *Lumen* and *Vessel*, the neural network might have an easier time segmenting the latter, which are, generally speaking, simple circuloid shapes.

2.4.10 Multichannel Input - 2.5D Training

While traditional network implementations utilize a single input frame x_i and a single output frame y_i for segmentation, one option for increasing segmentation quality is to provide

the network with a stack of neighbouring frames instead, allowing the network more information before attempting to segment the desired labels. While referred as *Multichannel* in this work, it is also known as *2.5D Training* in some works (ZIABARI et al., 2018). Thus, instead of modeling the behaviour of a function f such that:

$$y_i = f(x_i) \tag{2.28}$$

Let $2n$ be the number of adjacent frames stacked symmetrically to the right and left of frame i. The proposed approach will be such that f becomes:

$$y_i = f(X_i) \mid X_i = \{x_j \mid j = i - n, \cdots, i + n\} \tag{2.29}$$

The concept can be developed further by optionally spacing the adjacent frames. Let s be the distance (in terms of frames) between to adjacent frames. Thus:

$$X_i = \{x_j \mid j = i - s \cdot n, i - s(n-1), i - s(n-2), \cdots, i + s(n-1), i + s(n-2), i + s \cdot n\} \tag{2.30}$$

With n and s as predefined hyperparameters representing the number of input channels and the stride between each channel pair, respectively, the multichannel strategy MC of a neural network is given by the following notation:

$$MC = [n, s] \tag{2.31}$$

3 Results

In order to compare the results of different trainings, multiple metrics were taken from the statistics partition of the dataset, which refers to the native frames from the validation partition. Finally, in section 3.2.2 the best model is analyzed with regards to the test partition.

The set of reported metrics include Jaccard Index (IoU), Sørensen-Dice Index (DICE) and Hausdorff Distance for all relevant classes, model/ground truth ratios for Plaque Burden (stenosis) and finally, model/ground truth ratios for the area of lumen, plaque and vessel segmentations. From each metric an array of statistical estimators was reported, including mean, minimum, maximum, median, 25-th and 75-th percentiles, standard deviation and interquartile range (IQR).

For the IoU and DICE metrics an "Average" score was defined as the average value of Plaque and Lumen scores, with the justification for excluding the Vessel score being that Plaque and Lumen's segmentations already define the Vessel's segmentation (see section 2.4.9). Thus:

$$IoU_{Average} = \frac{IoU_{Plaque} + IoU_{Lumen}}{2} \tag{3.1}$$

For the Hausdorff distance metric the Plaque value was defined as the maximum value of lumen's and vessel's Hausdorff distance:

$$HD_{Plaque} = max(HD_{Lumen}, HD_{Vessel}) \tag{3.2}$$

Nevertheless, the main estimator chosen by the author to compare the models' segmentation quality was the median value of the Average IoU. Appendix A reports the main metrics and minor specifications of each training performed.

3.1 Parameter Comparisons

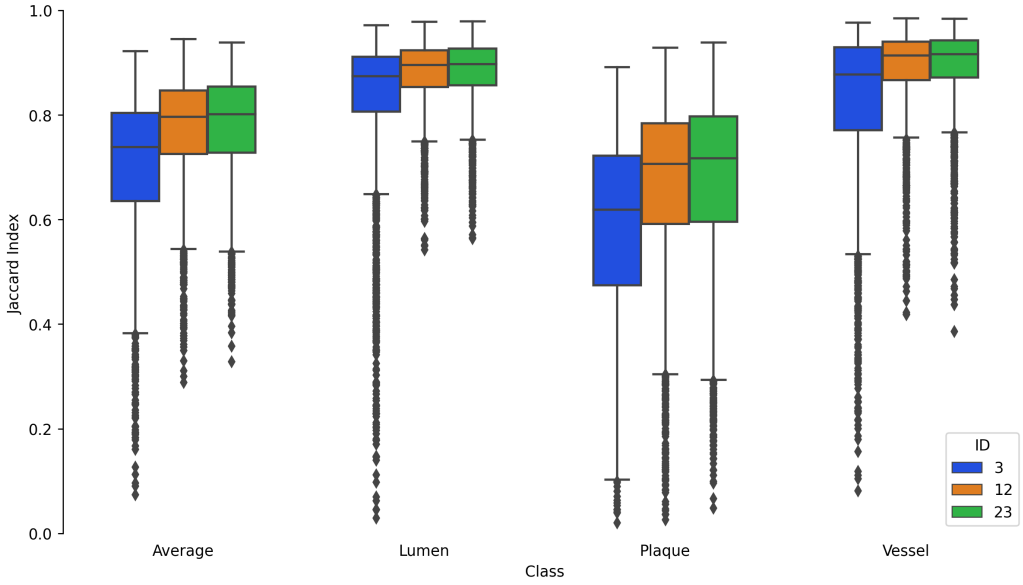
Given the number of trainings performed, which is over 80, there are more than 5000 graphs, plots, and tables that could be presented in order to show comparative results. Naturally, this will not be done; only the hyperparameter comparisons of greater interest will be discussed in the following sections. The full results for each training are available in Appendix A.

It has been observed during this work that a specific parameter's influence on the model's quality is often dependent on the parameters kept *constant*, as demonstrated in Section 3.1.1. With such nonlinearity in mind, the following subsections provide in depth details of comparisons made with specific parameters in mind.

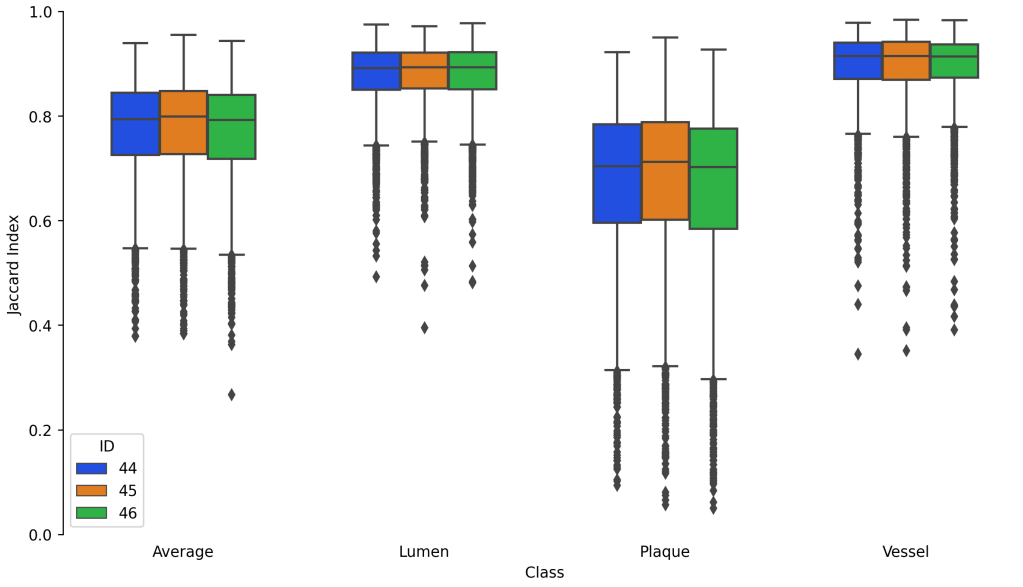
3.1.1 Batch Size and Depth

At a batch size of 8 increasing the network's depth has demonstrated tremendous improvement on the model's quality, while the effect is unfortunately not seen when performing the same increase at a batch size of 16. In fact, a small decrease is observed, possibly due to stochastic nature of the trained network (see Figure 27).

Figure 27 – Jaccard Index at Different Depths with Multiple Batch Sizes



(a) Depth 6 (ID 1), 7 (ID 12) and 8 (ID 23) at a batch size of 8



(b) Depth 6 (ID 44), 7 (ID 45) and 8 (ID 46) at a batch size of 16

Source: Author

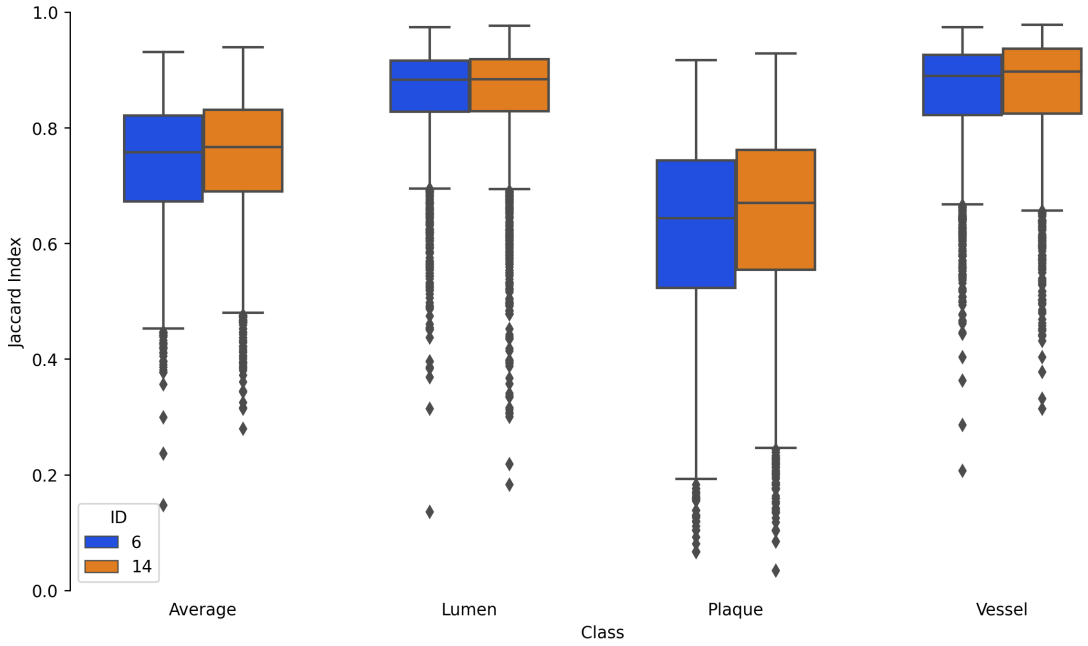
3.1.2 Loss

On total 3 loss functions were evaluated, that is, the IoU loss (Equation 2.11), the Cross Entropy loss (Equation 2.8), and the combination of both losses, which is the sum of both (Equation 3.3).

$$Total_{Loss} = IoU_{Loss} + Crossentropy_{Loss} \tag{3.3}$$

Figure 28 shows the comparison of Crossentropy and IoU losses, on trainings performed at depth level 6, kernel size of 3 and batch sizes of 32. Results show that the IoU loss might, generally speaking, have an edge against the Crossentropy, as shown on Figure 28.

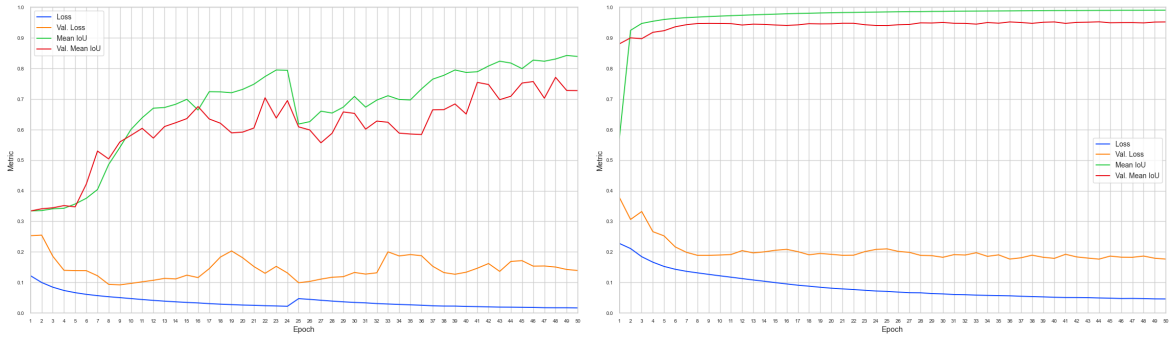
Figure 28 – Crossentropy (ID 6) vs. IoU loss (ID 14)



Source: Author

Inspection of the training curves (Figure 29) further supports the hypothesis, showing that the IoU loss improves the Mean IoU much faster than the much more unstable Crossentropy training curve.

Figure 29 – Crossentropy and IoU Loss Training Curve Comparison



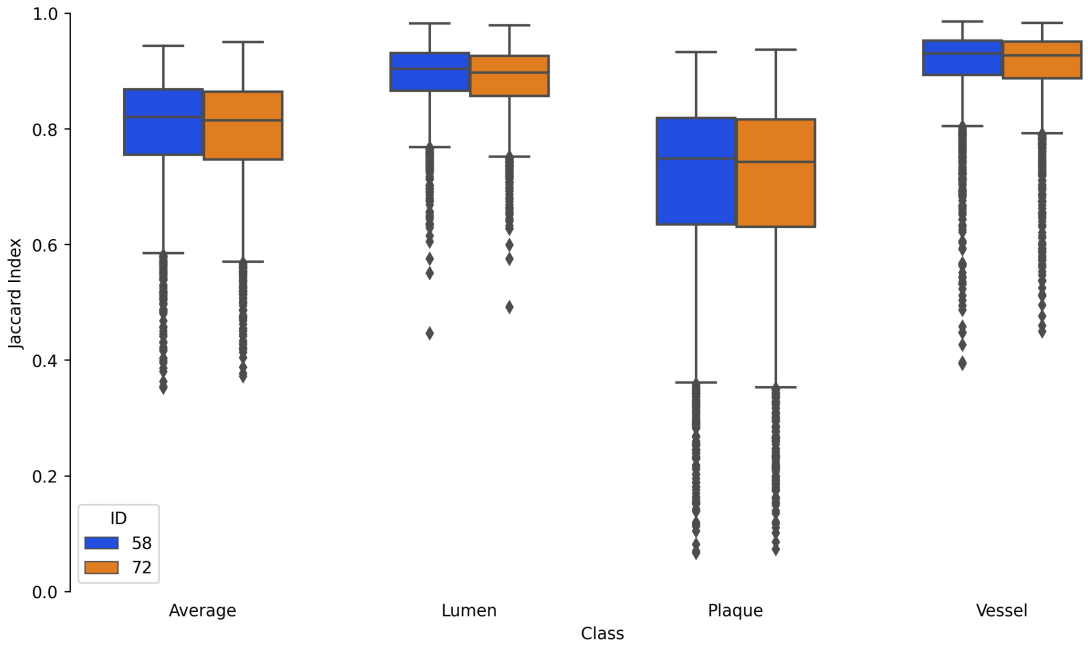
(a) Crossentropy (b) IoU

Source: Author

Repeating the comparison again with a more complex (more parameters) network (Figures 30 and 31), again a similar improvement can be observed, however less pronounced. The trainings in question (ID 72 and ID 58) were performed at depth 8, kernel size of 5 and batch

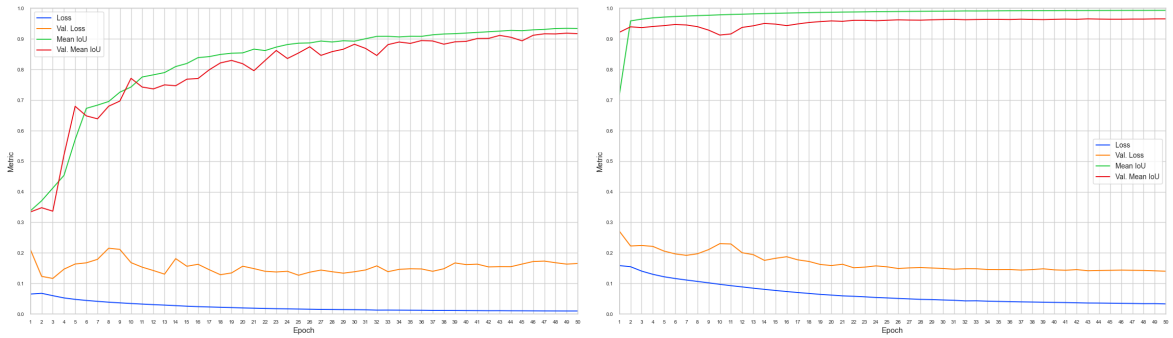
size 8. Furthermore, section 3.1.3 shows some details on trainings with the combined IoU + Crossentropy loss.

Figure 30 – IoU (ID 58) vs. Crossentropy loss (ID 72)



Source: Author

Figure 31 – Second Crossentropy and IoU Loss Training Curve Comparison



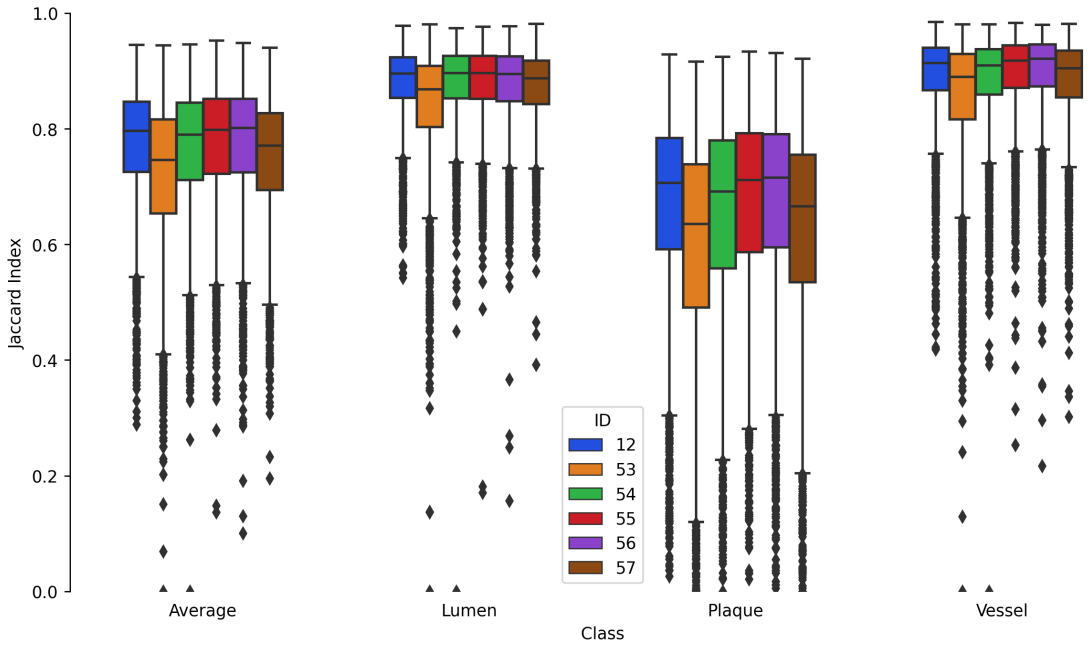
(a) Crossentropy (b) IoU

Source: Author

3.1.3 2.5D Training

In order to assess the effects of using 2.5D training, multiple trainings were executed at nearly equal parameters (learning rate had to be decreased from 0.001 to 0.0003 for large multi-channel values (IDs 54 to 57) due to issues with numerical errors during training). Figure 32 shows the resulting IoU distributions for each training, executed with a batch size of 8, depth 7, kernel size 3, base filters equal to 32 and IoU loss.

Figure 32 – 2.5D Training with 1, 3, 5, 7, 9 and 11 channels



Source: Author

While increasing the number of channels clearly provided an improvement (see the plaque's IoU trend in IDs 54, 55 and 56), it could be speculated that the noticeable dip in net performance when adding neighbouring frames (IDs 12 to 53 and 56 to 57) is due to the network not having enough filters to properly process the newly added information.

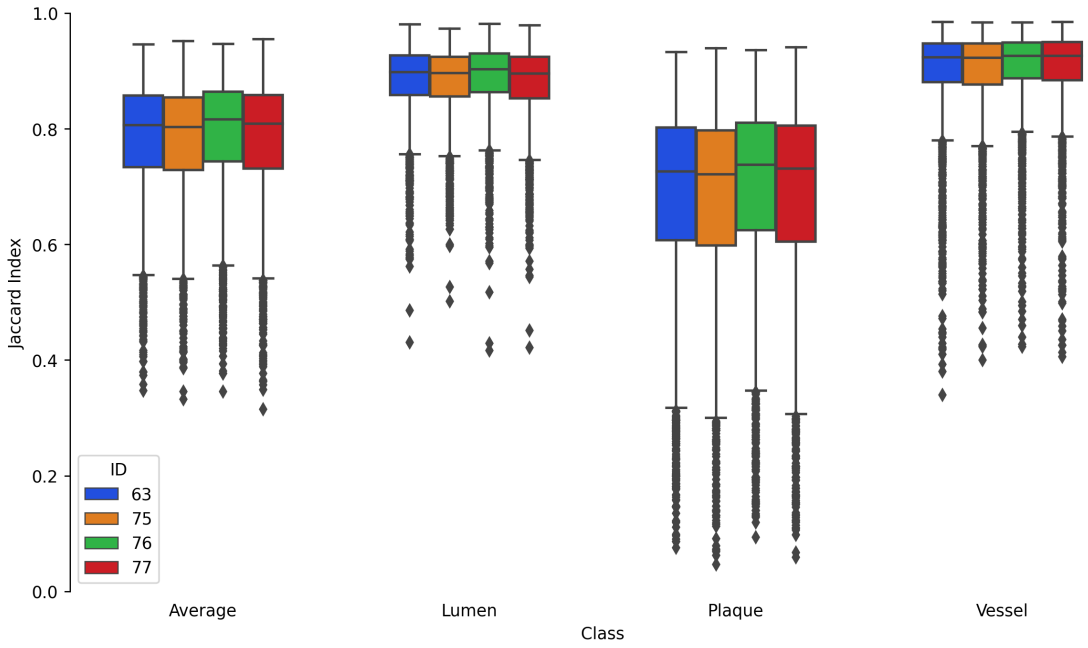
That said, another test was performed, this time increasing the base filters from 32 to 64, kernel size to $[9, 5]$ and batch size to 16. IoU + Crossentropy (Equation 3.3) was used as loss, and a Downsizing factor of 2 was added so that the training could run on limited computational resources. Due to time constraints, the test was not performed on multichannel values of 1 and 3. Jaccard Index for the result trainings are presented in Figure 33.

As Figure 33 shows, the best results were found on ID 76 (7 channels). Regardless of the Average IoU score (eq. 3.1), which is also conveniently higher than all others in this comparison, special attention should be given to the fact that using specifically 7 channels led to a much higher performance at the 25th-percentile, specially so for the plaque segmentation, sitting at 0.6252. Meanwhile the second best 25th-percentile for the plaque segmentation came from using 11 channels (ID 63), at 0.6073.

Selecting ID 76 as the "winner" and extending it by 20 more epochs showed improved results (Figure 34). Particularly, improvements were observed in the plaque segmentation, probably due to improvements in the vessel segmentation, as the lumen quality remained more or less constant.

Given the improvements observed in extending the ID 76's training, these were extended again, this time with a lower learning rate (0.0001), as analysis of the training curve suggested

Figure 33 – 2.5D with 11 (ID 63), 5 (ID 75), 7 (ID 76) and 9 (ID 77) channels



Source: Author

that the network was very close to convergence. Figure 35 shows the resulting IoU distributions.

Extending ID 82 by 20 more epochs, thus creating ID 85, with a total of 70 epochs of training, provided virtually no improvement (in fact, a small decrease in quality can be observed in the 75th-percentile of lumen, plaque and vessel classes). Nevertheless a slight improvement was observed in the Hausdorff Distance metric, as shown by Figure 36 and further illustrated by Table 5.

Table 5 – IDs 76 (30 epochs), 82 (50 epochs) and 85 (70 epochs) Hausdorff Distance Scores' Distribution Statistics

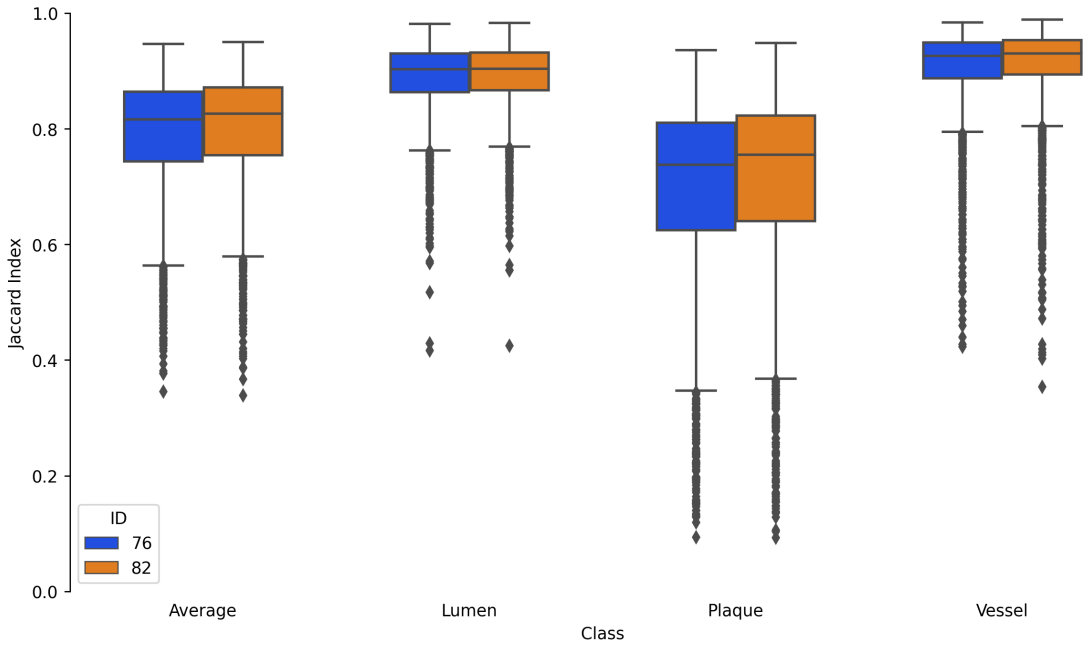
	Mean			Median			IQR		
ID	Lumen	Plaque	Vessel	Lumen	Plaque	Vessel	Lumen	Plaque	Vessel
76	0.3038	0.4008	0.3060	0.2148	0.2776	0.2227	0.1624	0.2379	0.2098
82	0.2864	0.4053	0.3196	0.2072	0.2657	0.2072	0.1513	0.2264	0.1923
85	0.2751	0.3692	0.2877	0.2011	0.2547	0.1992	0.1442	0.2221	0.1774

Source: Author

In a similar manner, ID 63 (11 channels) was also extended by a total of 90 epochs, from which the comparisons are presented in Figures 37 and 38 and Table 6. Both from a Jaccard-Index and from a Hausdorff distance perspective, the segmentation quality stabilizes in ID 90 (110 epochs), which the validation loss curve seems to corroborate, as seen in Figure 39.

With both original networks (IDs 76 and 63) further trained until their segmentation quality was found relatively stable (IDs 85 and 90, respectively), from the perspective of both

Figure 34 – ID 76 (30 epochs) and ID 82 (50 epochs)



Source: Author

Table 6 – IDs 63 (30 epochs), 78 (50 epochs), 83 (70 epochs), 89 (90 epochs) and ID 90 (110 epochs) Hausdorff Distance Scores' Distribution Statistics

	Mean			Median			IQR		
ID	Lumen	Plaque	Vessel	Lumen	Plaque	Vessel	Lumen	Plaque	Vessel
63	0.3209	0.4970	0.3973	0.2227	0.2975	0.2348	0.1663	0.3041	0.2720
78	0.2592	0.3646	0.3024	0.2104	0.2628	0.2039	0.1470	0.2290	0.1852
83	0.2532	0.3844	0.3238	0.2011	0.2606	0.2011	0.1487	0.2263	0.1939
89	0.2569	0.3692	0.3055	0.2011	0.2539	0.2011	0.1442	0.2325	0.1907
90	0.2575	0.3731	0.3057	0.1992	0.2539	0.1992	0.1470	0.2325	0.1836

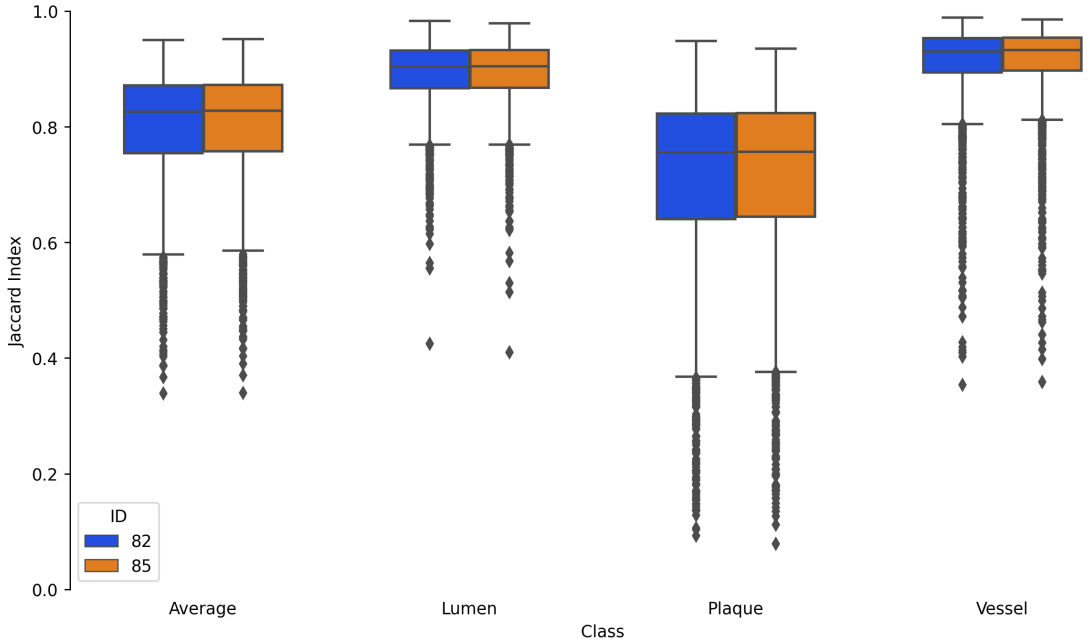
Source: Author

IoU scores and validation loss curves, comparisons of both IDs can be made. Figure 39 show the networks' validation loss curves, while Figures 40 and 41 respectively compare Jaccard-Index and Hausdorff Distance performances.

As observed in the figures, from the Jaccard-Index perspective IDs 85 and 90 have nearly indistinguishable performance, while the Hausdorff Distance perspective suggests better performance from ID 85, specially considering the difference in the plaque class' 75th-percentile.

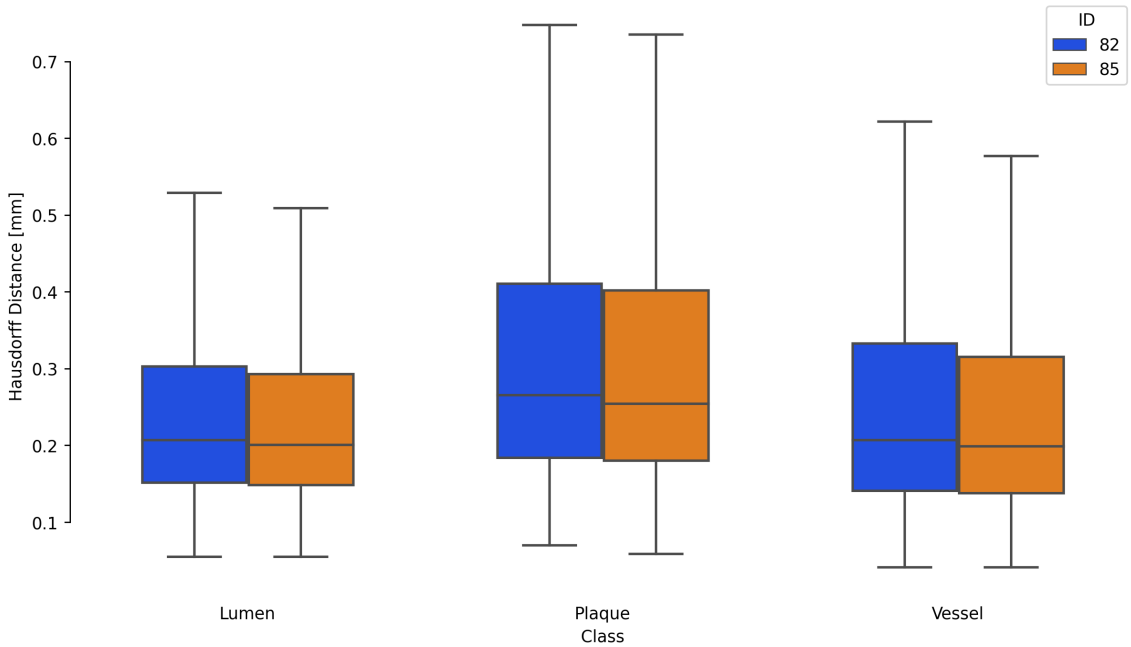
Considering that both trainings have nearly identical performance with regards to the established Average IoU metric, and also that some variation is present due to the aforementioned sampling issues, ID 85 should be considered the superior model, since it's the simpler one, using less channels, and also required less training, needing only 70 epochs as opposed to 110.

Figure 35 – ID 82 (50 epochs) and ID 85 (70 epochs) IoU Scores



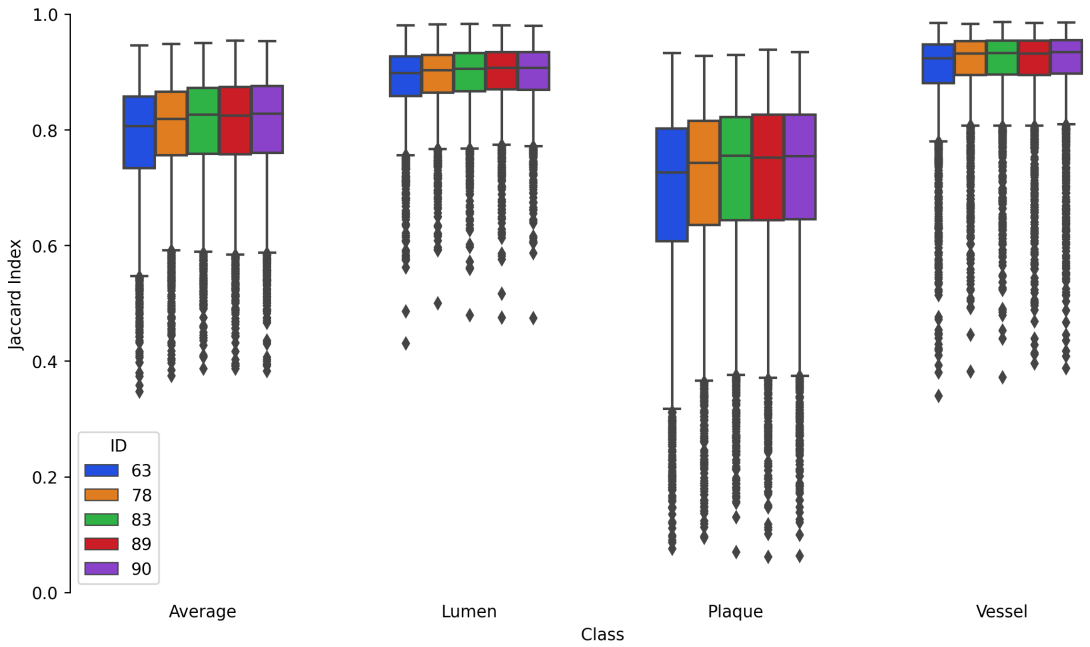
Source: Author

Figure 36 – ID 82 (50 epochs) and ID 85 (70 epochs) Hausdorff Distance Scores



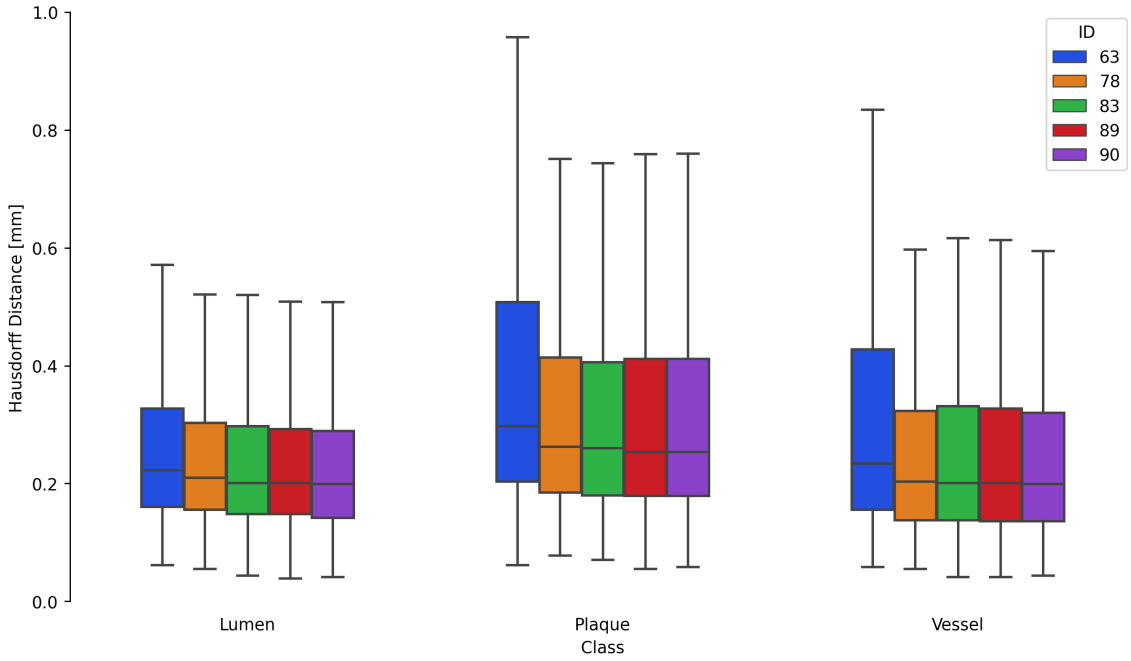
Source: Author

Figure 37 – IDs 63 (30 epochs), 78 (50 epochs), 83 (70 epochs), 89 (90 epochs) and ID 90 (110 epochs) IoU Scores



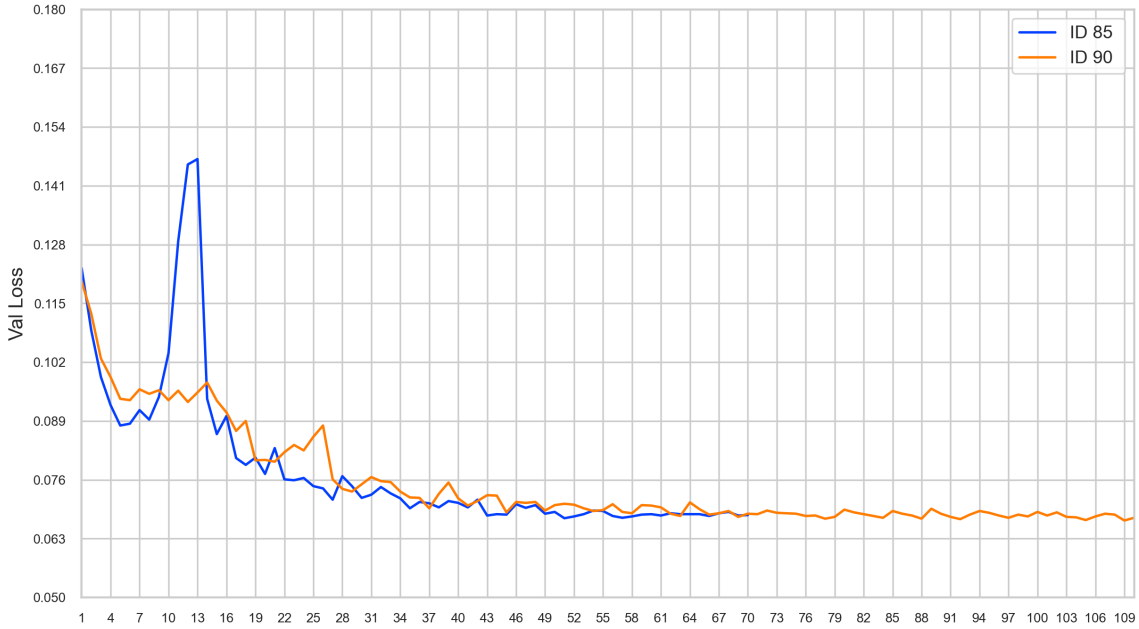
Source: Author

Figure 38 – IDs 63 (30 epochs), 78 (50 epochs), 83 (70 epochs), 89 (90 epochs) and ID 90 (110 epochs) Hausdorff Distance Scores



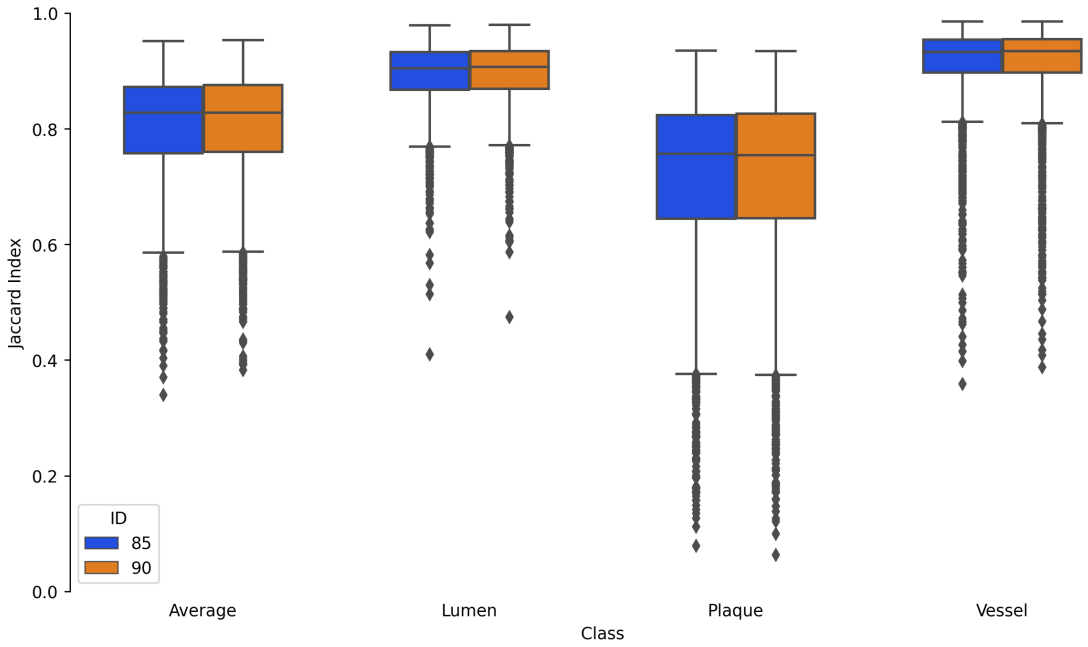
Source: Author

Figure 39 – IDs 85 and 90 Training Curves



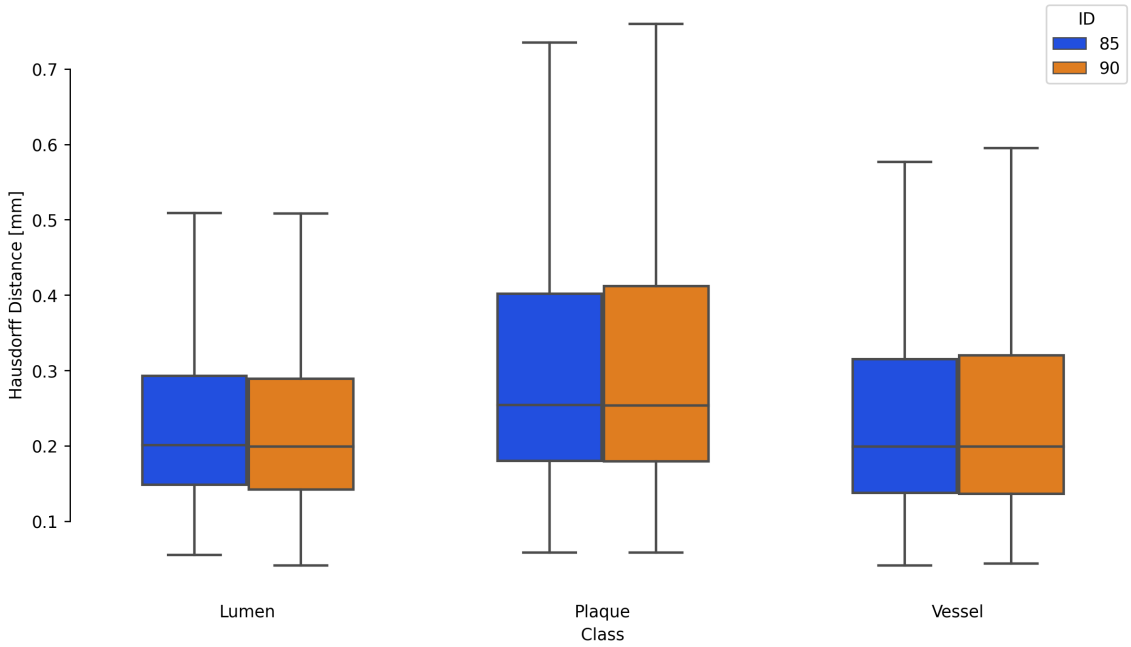
Source: Author

Figure 40 – IDs 85 and 90 Jaccard-Index



Source: Author

Figure 41 – IDs 85 and 90 Hausdorff Distance

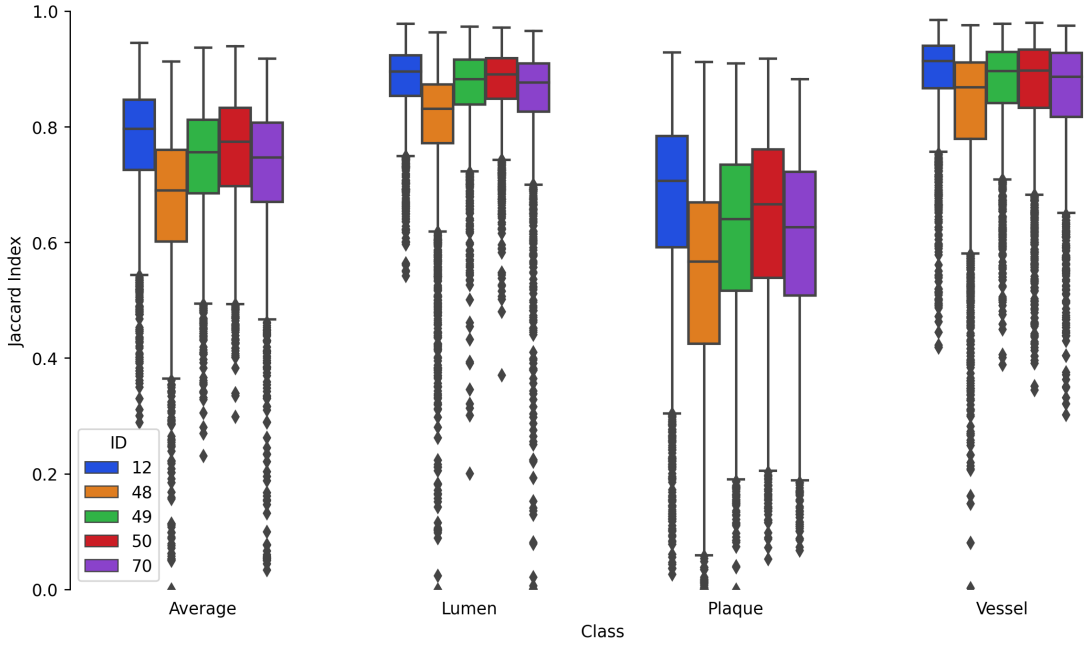


Source: Author

3.1.4 Node Structure

Figure 42 shows the IoU evaluation of several networks of varying node structures. Trainings were performed at 30 epochs with batch size of 8, using a learning rate of 0.001 and IoU loss on a U-Net++ with depth of 7 and base filters set to 32.

Figure 42 – Different Node Structure Comparisons

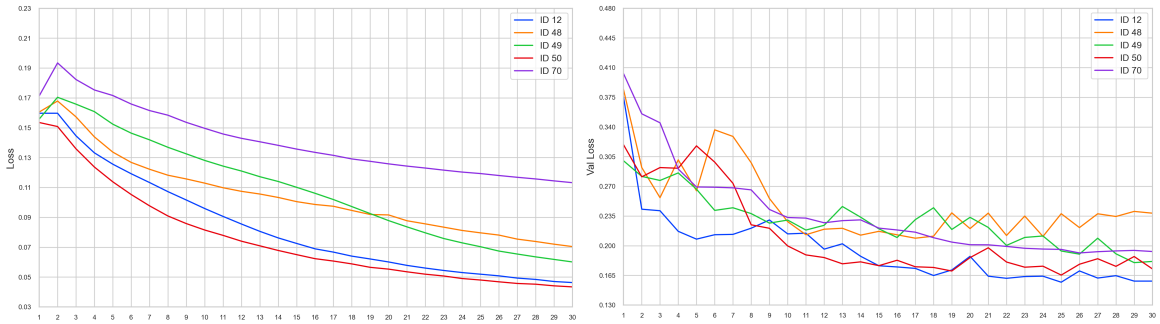


Source: Author

As can be seen, for the specified parameters the node type 4 (see Figure 22), a structure comprised of a convolution followed by batch normalization and finally ReLU activation (ID 12) successfully beat all other considered structures. Figure 43 shows the training curves for these trainings.

Figure 43 – Training Curves of Different Node Structures

(a) IoU Loss (b) Validation IoU Loss



Source: Author

Analysis of the training curves suggest training IDs 12 and 50 give the best results, with ID 50 differing from ID 12 only by the use of PReLU activations (HE et al., 2015) instead of

ReLUs (XU et al., 2015). While ID 50's training curve suggests faster convergence and shows a slightly improved loss value, ID 12's validation loss curve suggests better results, which are confirmed with Figure 42. It might be the case that PReLU activations increase the likelihood of overfitting, taking into consideration, however, that the trainings were not yet fully converged. Nevertheless, node type 4 (ID 12) presents a higher IoU scores across every class, specially in plaque segmentation. Table 7 further illustrates the results.

Table 7 – Jaccard-Index of Node Structures Comparison

	Mean				Median				IQR			
ID	Average	Lumen	Plaque	Vessel	Average	Lumen	Plaque	Vessel	Average	Lumen	Plaque	Vessel
12	0.7727	0.8800	0.6654	0.8915	0.7964	0.8957	0.7068	0.9143	0.1214	0.0699	0.1922	0.0742
48	0.6676	0.7980	0.5373	0.8201	0.6903	0.8317	0.5675	0.8689	0.1589	0.1017	0.2444	0.1323
49	0.7398	0.8665	0.6130	0.8718	0.7558	0.8825	0.6403	0.8968	0.1272	0.0773	0.2181	0.0883
50	0.7547	0.8738	0.6357	0.8666	0.7741	0.8904	0.6663	0.8975	0.1357	0.0703	0.2224	0.1003
70	0.7241	0.8465	0.6016	0.8584	0.7469	0.8769	0.6264	0.8865	0.1376	0.0841	0.2137	0.1112

Source: Author

3.1.5 Native vs. Hybrid Dataset

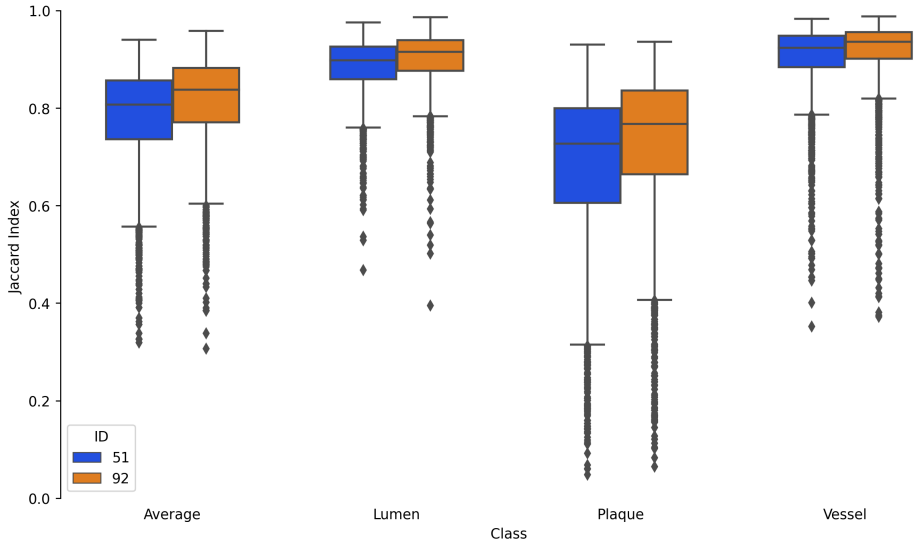
Performed nearly at the end of this work, it was desired to assess the effect of the data augmentation approach used on training and quality, although such study was not initially part of the scope, as mentioned in the introduction of Section 2.

Since the datasets have largely different sizes, the training partition of the hybrid dataset is roughly 21.5 times larger the training partition of the native dataset, the epoch unit means different amounts of training for each dataset. In other words, 1 epoch of training in the hybrid dataset represents roughly 21.5 epochs of training in the native dataset. Thus, in order to compare similar amounts of trainings on both datasets the number of epochs of training in the native dataset was multiplied by a factor of 21.5. This means that the standard $30 + 20n$ number of epochs become $21.5(30 + 20n) = 645 + 430n$ epochs.

Figure 44 showcases the comparison between two trainings at 30 epochs. Both refer to equal neural networks, consisting of U-Net++' skip connections, depth 8, 32 base filters, kernel size of 5 and a node structure consisting of a 2D Convolution followed by Batch Normalization and ReLU activation. Both where subjected to the equivalent trainings, with IoU loss, learning rate of 0.0003 and batch size of 8. A stark improvement was found in the removal of the data augmentation technique (that is, using the native dataset). While ID 51 reached a median Plaque IoU of 0.7273, ID 92 reached a median Plaque IoU of 0.7697.

The use of the native dataset clearly improved not only final segmentation quality, but convergence time (that is, the amount of training time needed) as well. Figure 45 shows that after around one third of the training (around 200 epochs on the native dataset), there is little to no

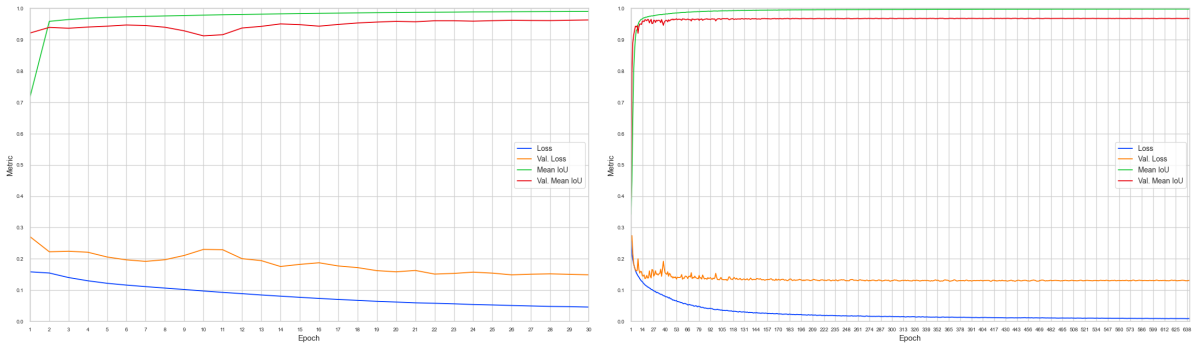
Figure 44 – ID 51 (Hybrid Dataset) vs. ID 92 (Native Dataset)



Source: Author

variation on the validation loss and metrics, while training on the hybrid dataset required further training.

Figure 45 – Training curves of ID's 51 (Hybrid Dataset) and ID 92 (Native Dataset)

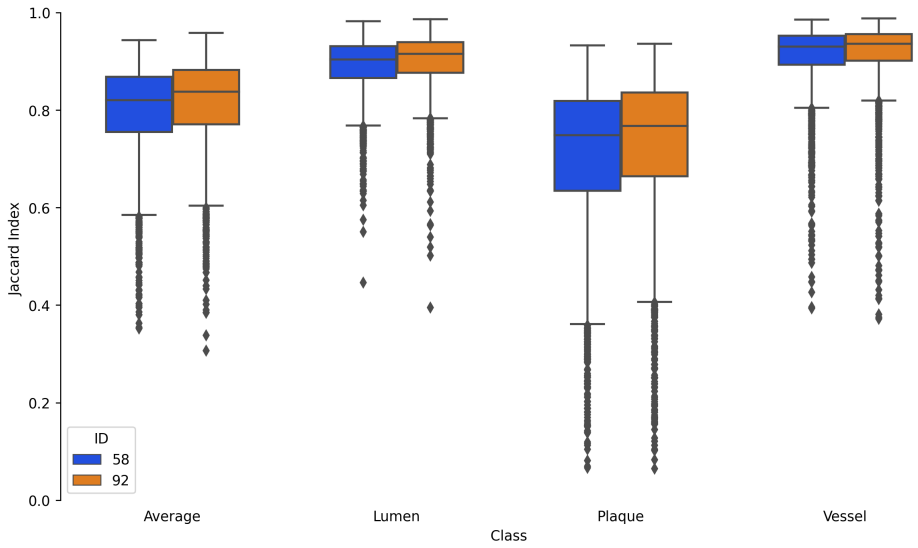


(a) Hybrid Dataset (b) Native Dataset

Source: Author

Even comparing against the further trained ID 58 (which is the neural network of ID 51 trained by 20 more epochs), there is still a strong contrast in quality (see Figure 46). While ID 58 reached a median Plaque IoU of 0.7486, ID 92 reached a median Plaque IoU of 0.7697.

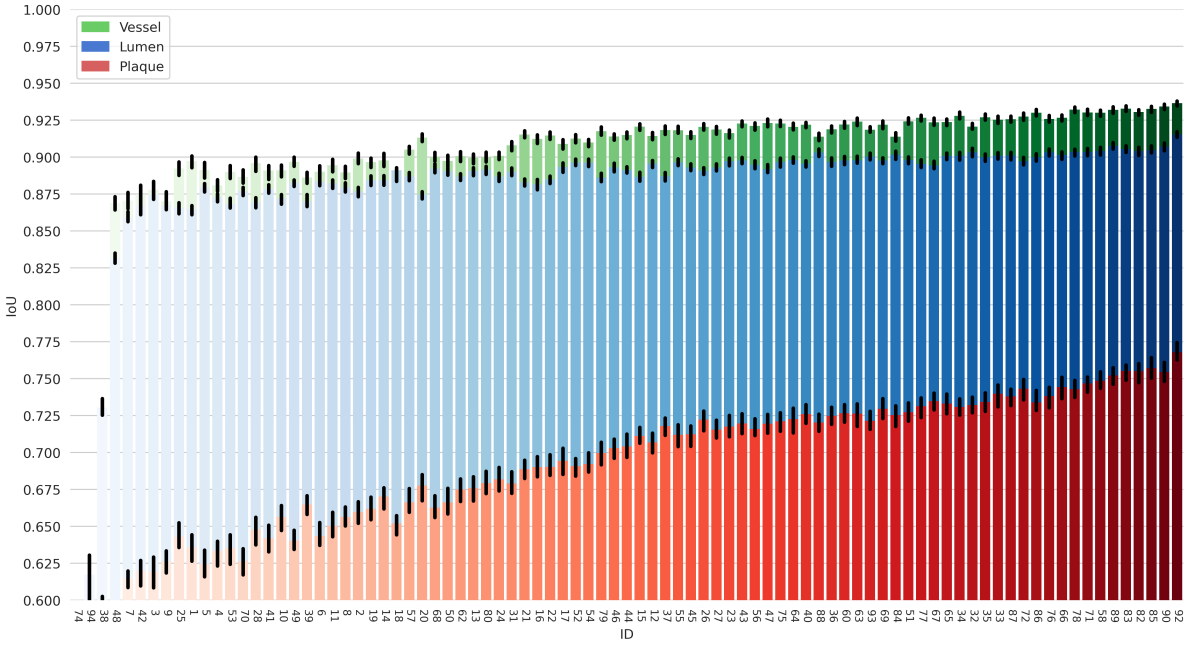
Figure 46 – ID 58 (Hybrid Dataset) vs. ID 92 (Native Dataset)



Source: Author

3.2 Best Model

Figure 47 – Training IDs ordered by median Average IoU



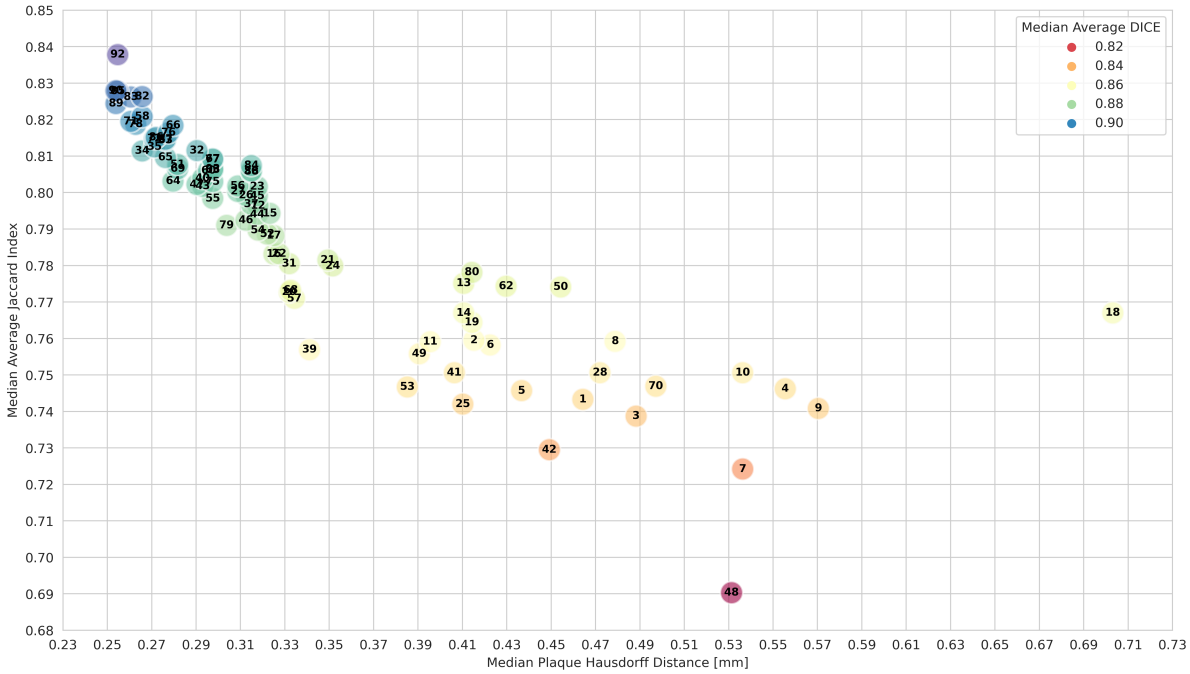
Source: Author

Figure 47 shows the trainings executed ordered by the median Average IoU observed in the validation dataset (error bars show a confidence interval of 95%). The best results were found in training ID 92, from which the specifications, detailed results and commentaries are presented in the following subsections.

Figure 48 shows the median Average IoU scores, as defined by Equation 3.1, further

contextualized by median Plaque Hausdorff Distance (Equation 3.2) and median Average DICE scores (calculation equivalent to Average IoU).

Figure 48 – Scatterplot of Average Jaccard-Index in relation to Plaque Hausdorff Distance and Average DICE



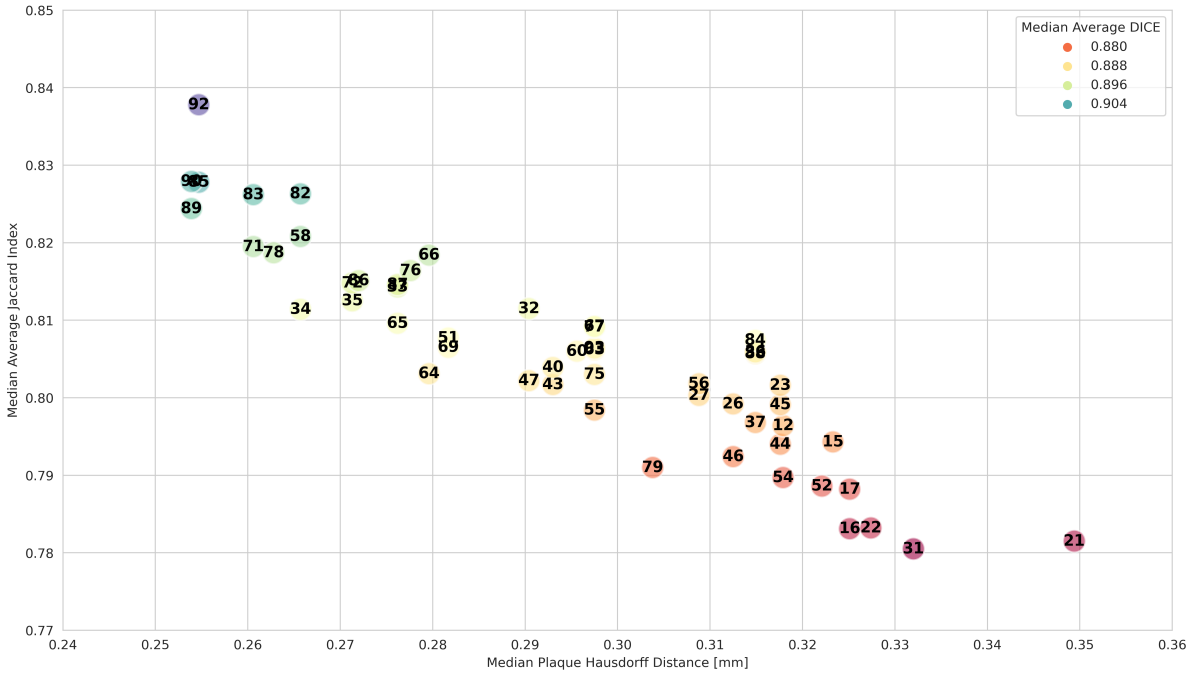
Source: Author

While lower quality IDs are more disperse, higher quality IDs form a thinner line, suggesting that the correlation between a high IoU score and a low Hausdorff Distance is higher on high quality networks. Both metrics are also correlated to the Average DICE. In the relatively lower IoU zone, between 0.72 and 0.78, there is clearly a much greater variation in the Plaque Hausdorff Distance, which raises the hypothesis that the IoU metric might be missing on some important aspect of segmentation quality evaluation in these ranges.

Figure 49 show a zoomed scatter plot of the aforementioned high quality networks. Note that the networks closer to ID 92 (IDs 82, 83, 85, 89 and 90) are all multichannel users, thus needing the data augmentation technique described in Section 2.2.2. They also needed extensive training time, with the least case scenario of ID 82 being trained for 50 epochs and the worst case scenario of ID 90 being trained for 110 epochs, while ID 92 was trained for only 30 epochs, and as mentioned in Section 3.1.5, actually needed less training to converge. They also have slightly more parameters (all had around 9.07×10^8 parameters as opposed to ID 92's 9.01×10^8).

It should be noted that a very small improvement (less than 0.01mm) to the median Plaque Hausdorff Distance can be observed in trainings 85, 89 and 90 in relation to ID 92. Nevertheless ID 92 is the clear winner, a conclusion drawn not only from the aforementioned issues but also due to the clear difference in the median Average IoU, which was chosen as the main evaluation metric, as described in the begin of this chapter.

Figure 49 – Zoomed Scatter Plot of Average Jaccard-Index in relation to Plaque Hausdorff Distance and Average DICE



Source: Author

3.2.1 Specifications

The network is essentially identical to the networks proposed by IDs 51 and 58, ID 58 being the previously best non-multichannel user. The difference is only in the training process, which was performed on the native dataset with an appropriately converted number of epochs, equivalent to roughly 30 epochs on the hybrid dataset (see Section 3.1.5). Table 8 presents the values of the hyperparameters that control the neural networks' structure, resulting in a network that has around 9.01×10^8 parameters.

Table 8 – Neural Network Hyperparameters

Hyperparameter	Value
Macro Structure	U-Net++
Node Structure	4
Depth	8
Base Filters	32
Kernel Size	5
Multi-channel	[1, 1]
Multi-Output	
Downsizing	1
Upsampling	Bilinear

Source: Author

Table 9 show the training related hyperparameters. That is, total epochs trained, initial

learning rate, optimizer used etc.

Table 9 – Training Hyperparameters

Hyperparameter	Value
Epochs	641
Learning Rate	0.0003
Batch Size	8
Optimizer	Adam
Loss	IoU
Base ID	None
Dataset	Native

Source: Author

Table 10 – TensorFlow Flags

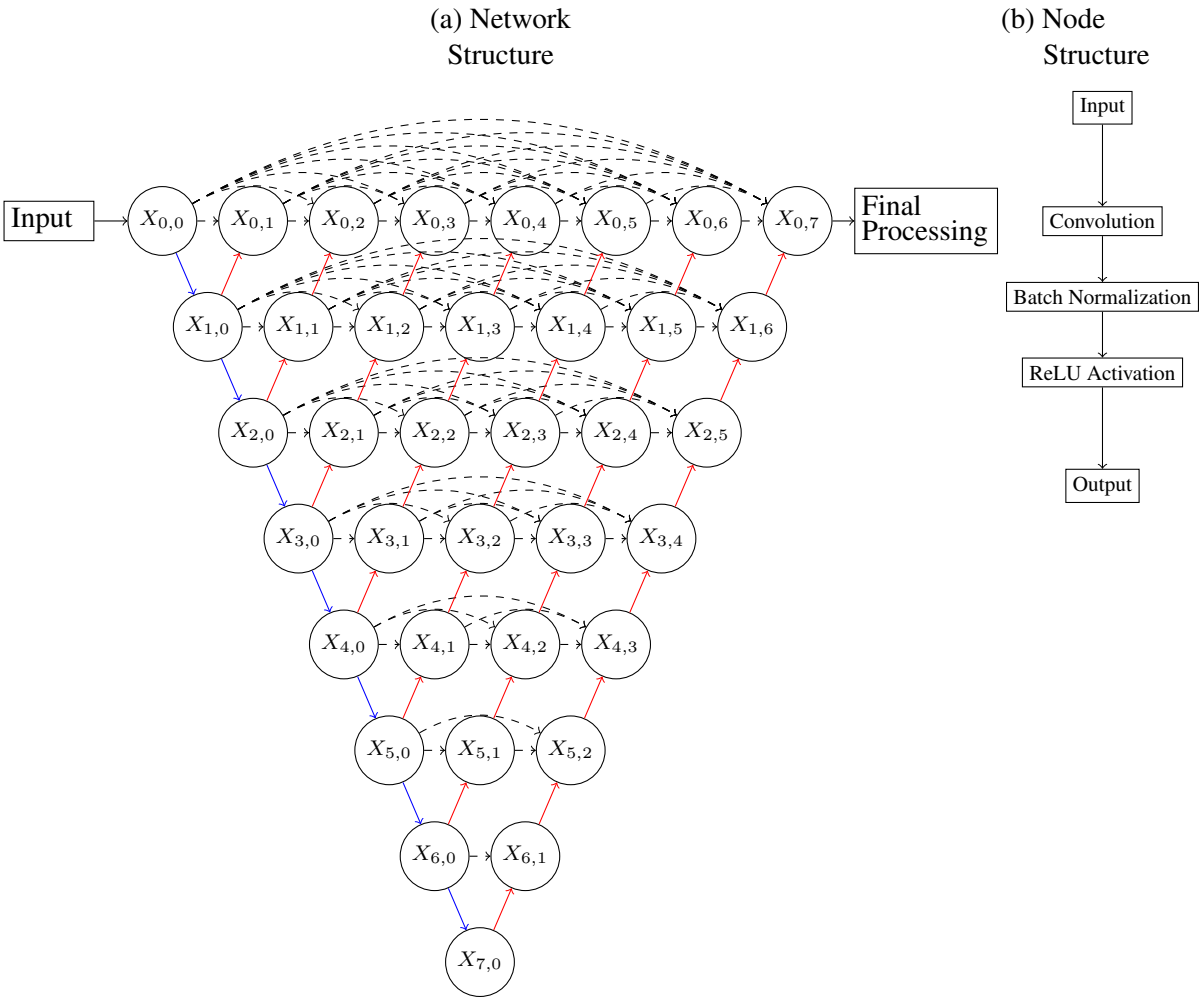
Flag	Value
Multi-GPU	True
Mixed Precision	True
Auto-Clustering	True

Source: Author

Table 10 shows the relevant TensorFlow flags used. *Auto-Clustering* refers to the XLA compilation routine used to improve training speed (TensorFlow Developers, 2022c), while *Mixed Precision* (TensorFlow Developers, 2022a) refers to the technique of using 16-bit float precision for some (usually most) computations. This results not only in less memory consumption, but also in a speed gain when training in some modern GPUs. The correct application of the technique usually keeps some essential operations in the typical 32-bit precision, such as the softmax operation (see Equation 2.5), due to numerical stability issues. Finally, *Multi-GPU* is self explanatory.

Figure 50 shows the resulting neural network for the specified hyperparameters. Blue arrows indicate maxpooling operations (Keras Team, 2022d), red arrows indicate upsampling operations (Keras Team, 2022f) and dashed arrows indicate skip connections (a node's output is simply passed as another's input). When a node's input is comprised of multiple outputs, such as node $X_{0,2}$ in Figure 50, they are concatenated before being processed. The node structure is equivalent to node type 4 (see Section 2.4.7.5), being comprised of a convolution operation (Keras Team, 2022b) followed by a batch normalization (Keras Team, 2022a) and finally a ReLU activation (Keras Team, 2022e). Pre-processing techniques include the conversion of pixel values from the typical [0, 255] integer range to the floating-point [0, 1) range. Final processing refers to the softmax operation and the following conversion into proper class labels.

Figure 50 – ID 92's Neural Network Structure



Source: Author

3.2.2 Segmentation Quality Analysis

This section describes the segmentation quality of the best network (ID 92) using the test partition as reference. Results for validation partition are described in Appendix A.

3.2.2.1 Jaccard Index

Figure 51 shows the violin plot showcases the resulting distribution for the Jaccard index of all 4 classes (Average, Lumen, Plaque and Vessel). As shown in Table 11, the model achieved a median Average IoU of 81.7%, the highest of the more than 70 trainings performed, with an IQR of 13.3%. The median Plaque IoU reached 73.9%, an overall good improvement over the bulk of other networks tested.

Figure 52 shows a scatter plot of the IoU metric taken for all 3 classes (Lumen, Plaque, Vessel), and the Average class separately. Note the localized valleys of Plaque segmentation performance.

Figure 51 – Validation partition's IoU Violin Plot

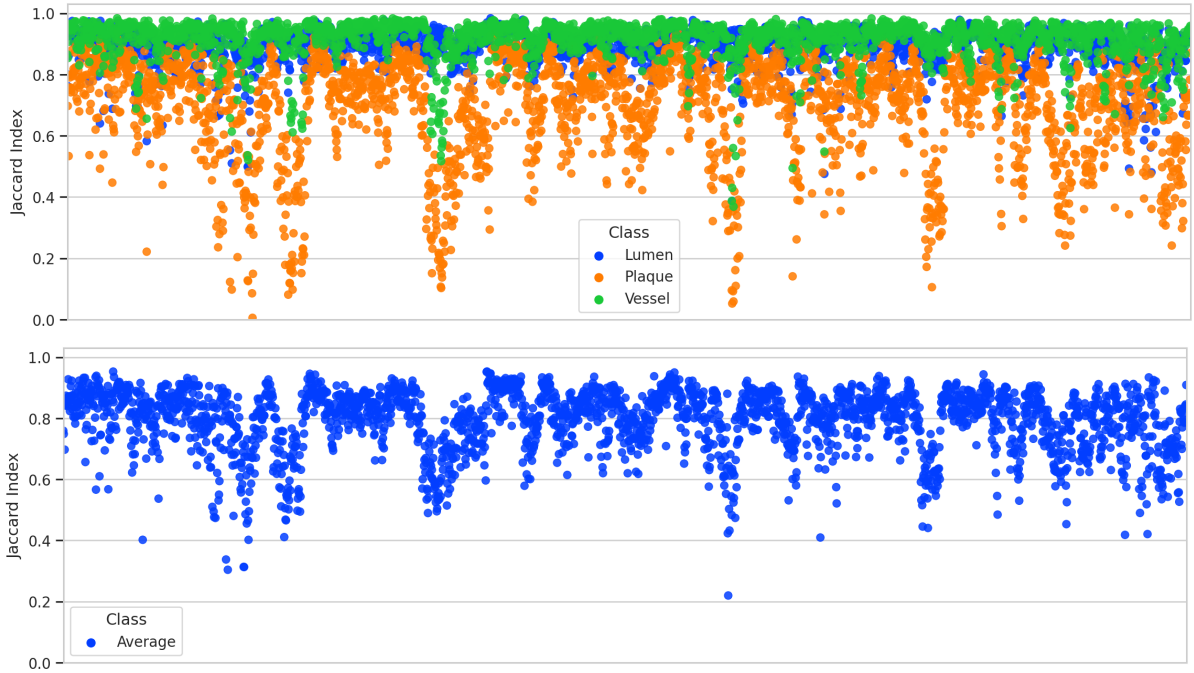
Source: Author

Table 11 – ID 92's IoU evaluation

	IoU			
	Average	Lumen	Plaque	Vessel
Mean	0.7928	0.8947	0.6908	0.9153
Std	0.1023	0.0599	0.1682	0.0636
IQR	0.1330	0.0616	0.2063	0.0514
Min	0.2197	0.3870	0.0056	0.3687
25%	0.7357	0.8728	0.6087	0.9015
50%	0.8169	0.9076	0.7392	0.9331
75%	0.8687	0.9345	0.8150	0.9528
Max	0.9532	0.9836	0.9415	0.9855

Source: Author

Figure 52 – Validation partition's IoU

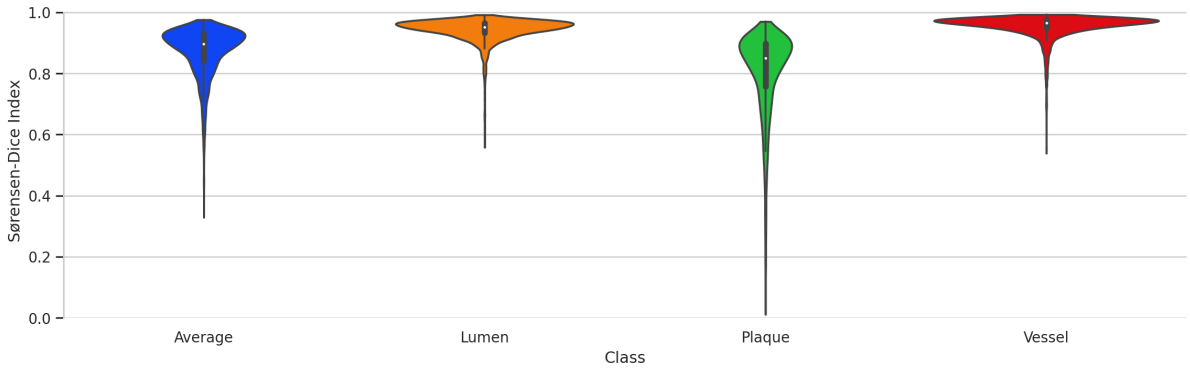


Source: Author

3.2.2.2 Sørensen-Dice Index

Figure 53 shows the violin plot showcases the resulting distribution for the Sørensen-Dice index of all 4 classes (Average, Lumen, Plaque and Vessel). It should be noted that the DICE metric, when compared to the IoU, tends to output a value closer to 1. Thus the "cleaner" display of Figures 53 and 54.

Figure 53 – Validation partition's DICE Violin Plot



Source: Author

As shown in Table 12, the model achieved a median Average DICE of 89.6%, the highest of the more than 70 trainings performed, with an IQR of 0.09%. The median Plaque DICE reached 85.0%, an overall improvement over the other networks tested.

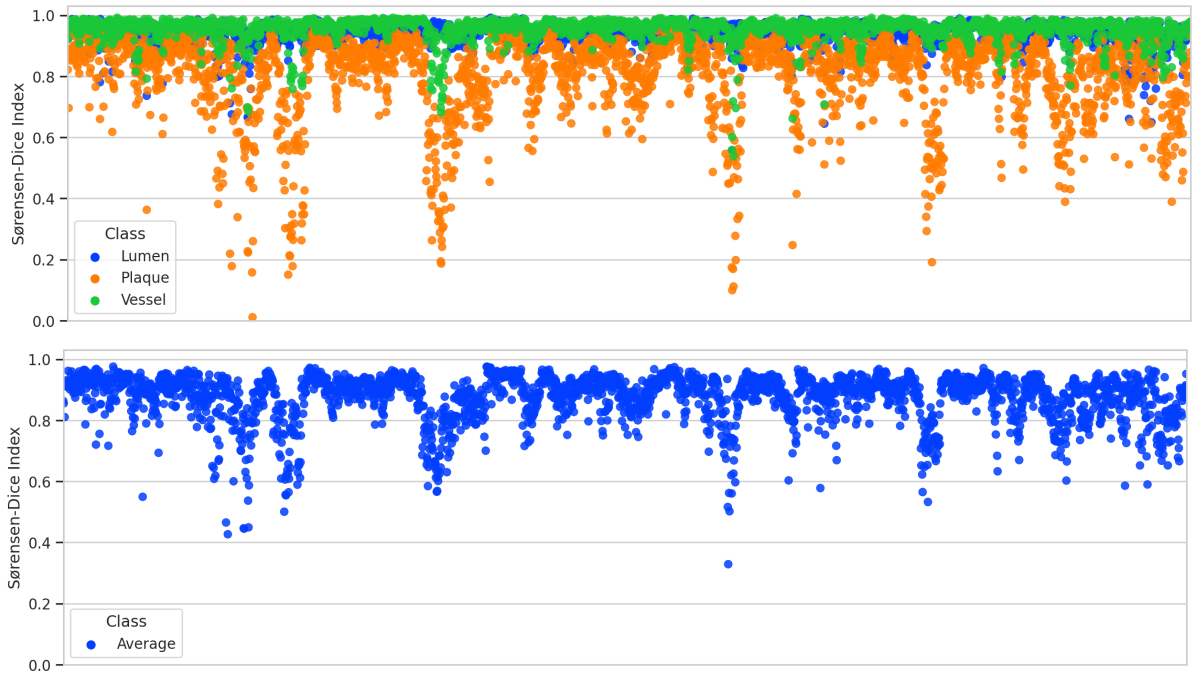
Table 12 – ID 92's DICE evaluation

	DICE			
	Average	Lumen	Plaque	Vessel
Mean	0.8733	0.9433	0.8033	0.9545
Std	0.0802	0.0365	0.1402	0.0391
IQR	0.0879	0.0340	0.1413	0.0277
Min	0.3288	0.5580	0.0111	0.5388
25%	0.8409	0.9321	0.7568	0.9482
50%	0.8963	0.9515	0.8500	0.9654
75%	0.9287	0.9661	0.8981	0.9758
Max	0.9759	0.9917	0.9699	0.9927

Source: Author

Figure 54 shows a scatter plot of the DICE metric taken for all 3 classes (Lumen, Plaque, Vessel), and the Average class separately. Note that due to the "cleanliness" previously mentioned, the localized valleys of Plaque segmentation performance are more pronounced.

Figure 54 – Validation partition's DICE

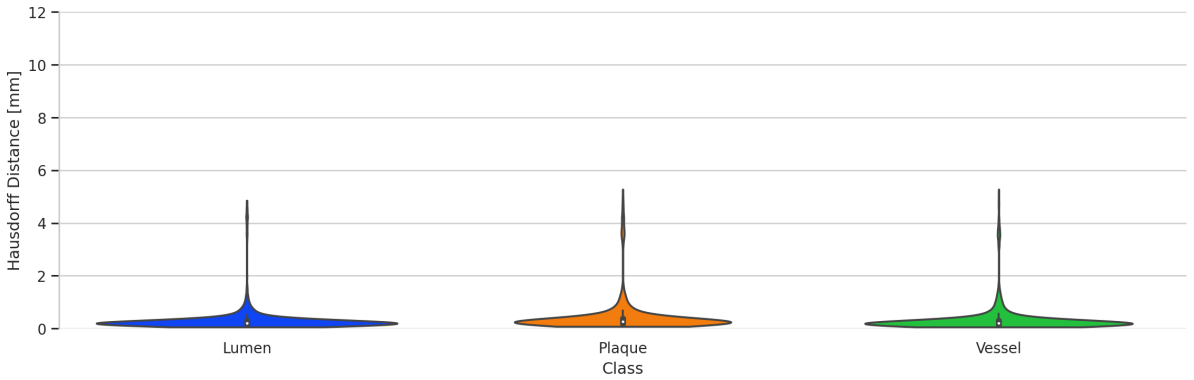


Source: Author

3.2.2.3 Hausdorf Distance

Figure 55 shows the violin plot showcases the resulting distribution for the Hausdorff distance of the 3 main classes (Lumen, Plaque and Vessel).

Figure 55 – Validation partition's Hausdorff Distance Violin Plot



Source: Author

As shown in Table 11, the model achieved a median Plaque Hausdorff Distance of 0.2620 mm, with an IQR of 0.2003 mm. Median Lumen and Vessel show values of 0.2104 mm and 0.2011 mm, respectively.

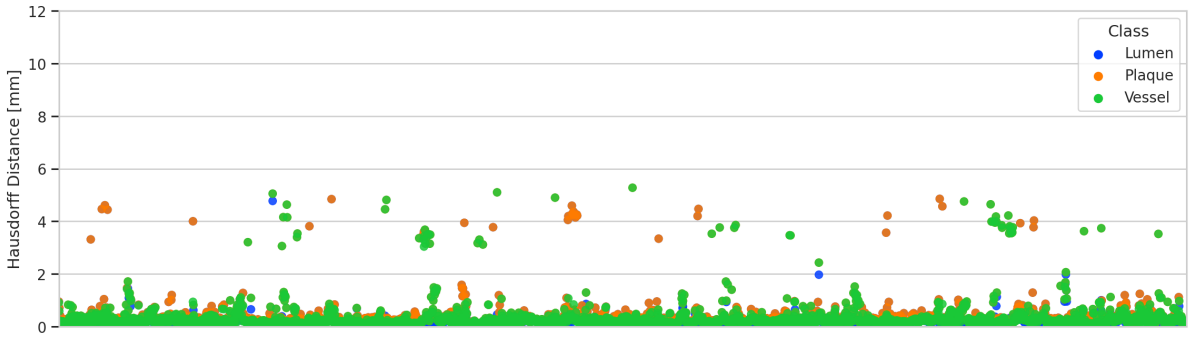
Figure 52 shows a scatter plot of the Hausdorff Distance metric taken for all 3 classes (Lumen, Plaque, Vessel). Note that the few peaks come, more often than not, from Vessel contour errors.

Table 13 – ID 92's Hausdorff Distance Evaluation

	Hausdorff Distance [mm]		
	Lumen	Plaque	Vessel
Mean	0.3129	0.4488	0.3437
Std	0.4833	0.6941	0.5663
IQR	0.1476	0.2003	0.1666
Min	0.0552	0.0781	0.0552
25%	0.1562	0.1924	0.1422
50%	0.2104	0.2620	0.2011
75%	0.3038	0.3927	0.3088
Max	4.8546	5.2771	5.2771

Source: Author

Figure 56 – Validation partition's Hausdorff Distance



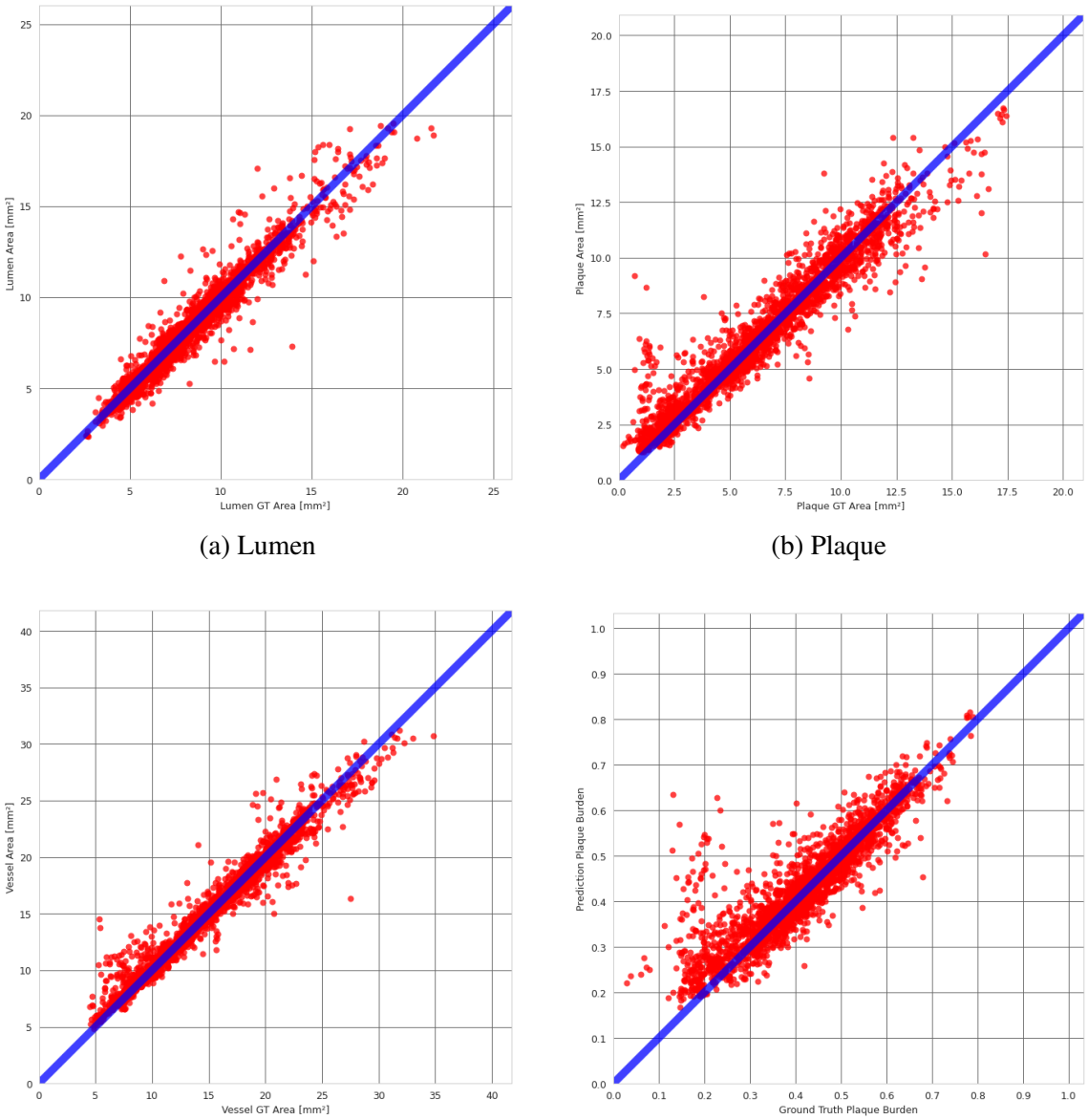
Source: Author

3.2.2.4 Area Estimation and Plaque Burden

Figure 57 showcase a comparison of the measured areas for each class alongside plaque burden. Comparing the three main classes, lumen is clearly the better behaved distribution, with values close to the blue line (Ground Truth Area = Prediction Area) and perhaps a slight bias, tending to underestimate the lumen's area. Plaque's distribution demonstrates the biggest dispersion, with outliers tending to overestimate plaque area when the ground truth's area is less than 10 mm^2 and underestimate it when above. Vessel's distribution show a behaved distribution, however sprinkled with a few zones of higher deviance, as can be seen specifically at ground truth's 5, 10, 20 and 25 mm^2 marks. Plaque Burden's distribution is relatively centered around the blue line, except for the lower end (ground truth's plaque burden between 0.1 and 0.3), where there is a clear bias towards overestimating it and a few outliers, also overestimating.

Further enriching analysis can be made through Bland-Altman plots (ALTMAN; BLAND, 1983), which compare two models (in this case, the provided ground truth and ID 92's neural network) without discriminating which is the correct one. Bland-Altman plots for each class and plaque burden are presented below. The dashed lines represent a 95% confidence interval.

Figure 57 – Comparison of Predictions and Ground Truth's Area and Plaque Burden Measures



(a) Lumen

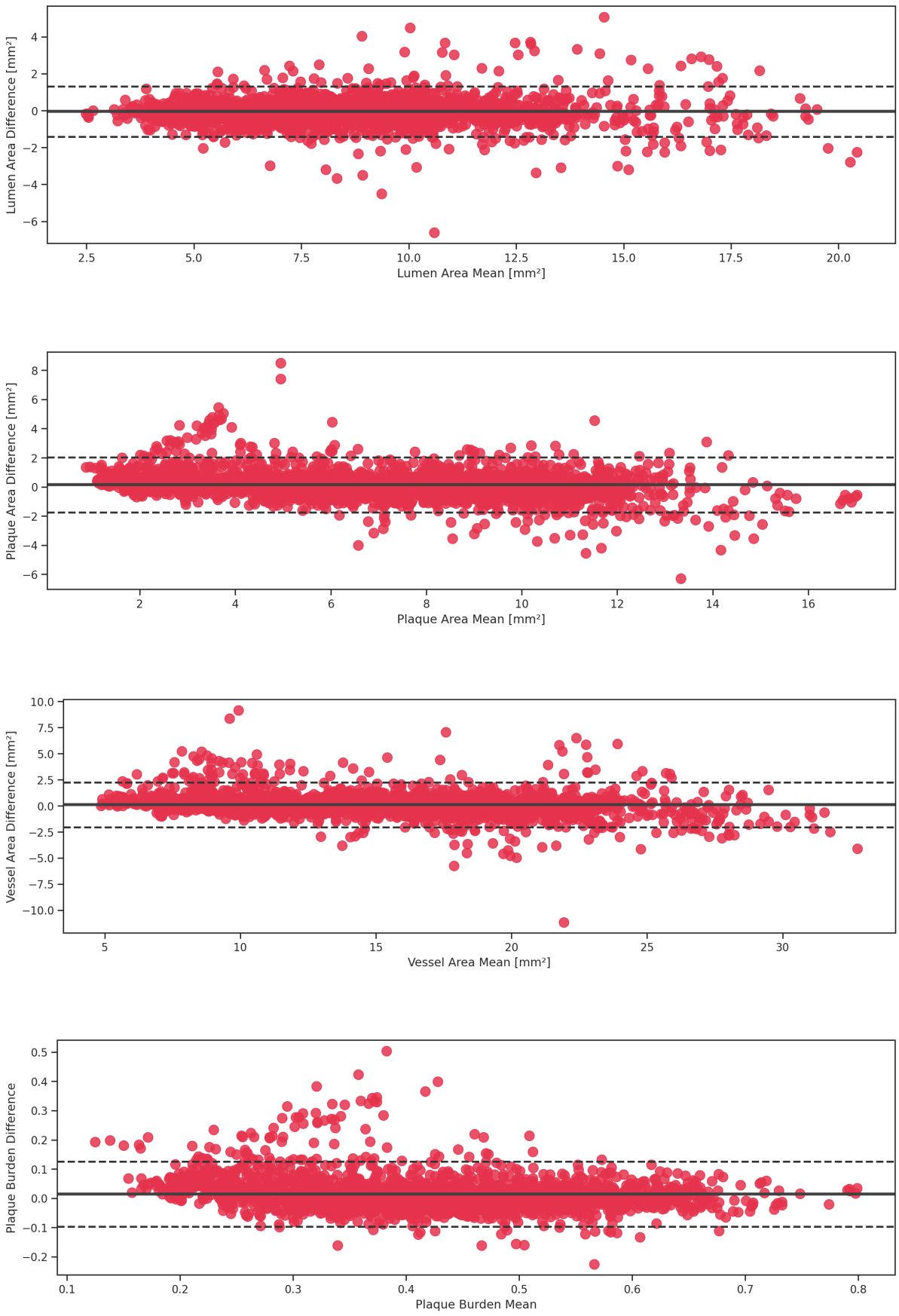
(b) Plaque

(c) Vessel

(d) Plaque Burden

Source: Author

Figure 58 – ID 92 and Ground Truth's Bland-Altman Analysis

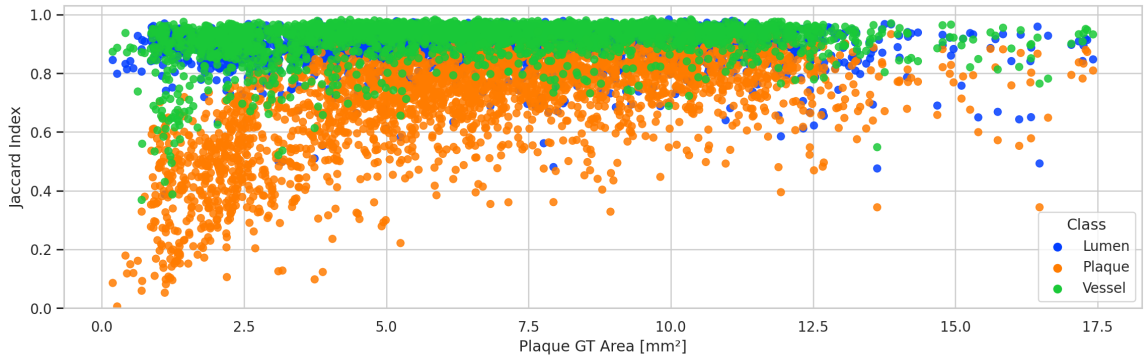


Source: Author

3.2.2.5 Performance by Metrics

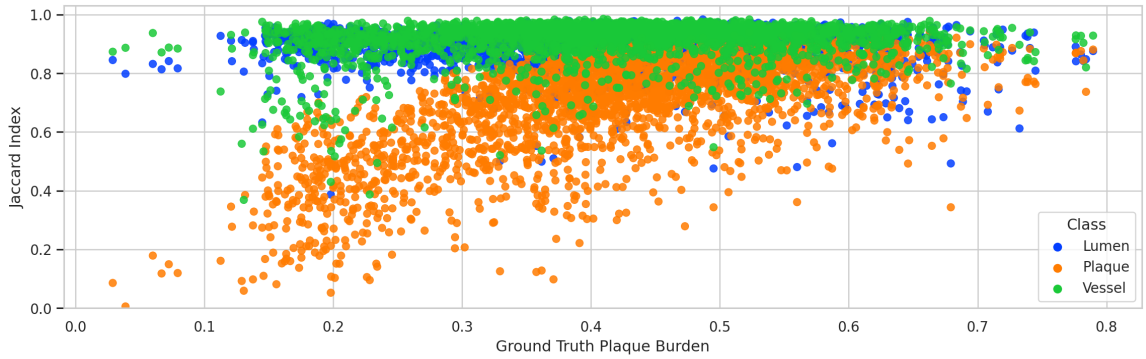
Figures 59 and 60 show the previously presented Jaccard-Index scores for the validation partition, ordered by the ground truth's plaque area and plaque burden, respectively. Note that the previously mentioned valleys of performance mostly accumulate in zones where the plaque area/burden is smaller. From the plaque burden perspective it is natural that segmentations with smaller plaque burdens are harder to reproduce, since the plaque region becomes an easy to miss thin ring-like structure, which also tends to happen with generally small areas.

Figure 59 – Jaccard-Index in relation to Ground Truth's Plaque Area



Source: Author

Figure 60 – Jaccard-Index in relation to Plaque Burden



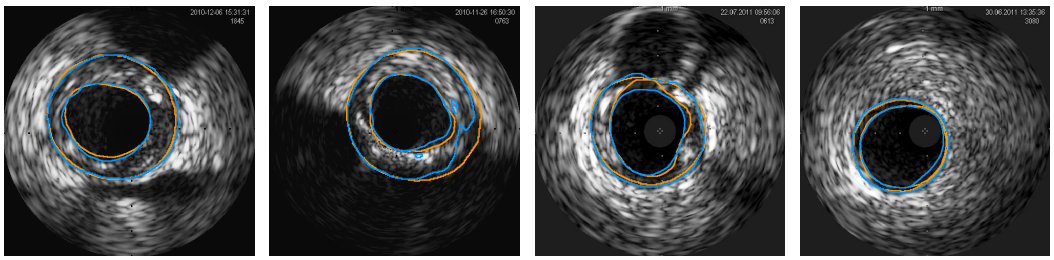
Source: Author

3.2.3 Segmentation Examples

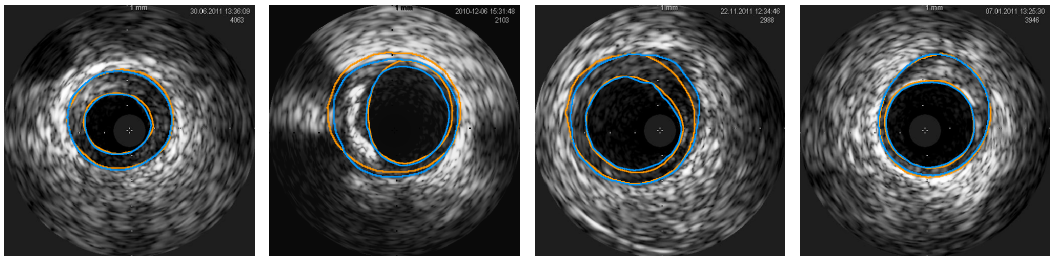
Some segmentation contour examples are presented below in Figure 61: Model predictions in blue, ground truth contours in orange and the segmentation's Average IoU score in the respective captions. As demonstrated by the samples, an Average IoU of around 0.84 usually foreshadows a good segmentation contour, with only minor deviations from the ground truth.

Figure 61 – ID 92's Segmentation Examples.

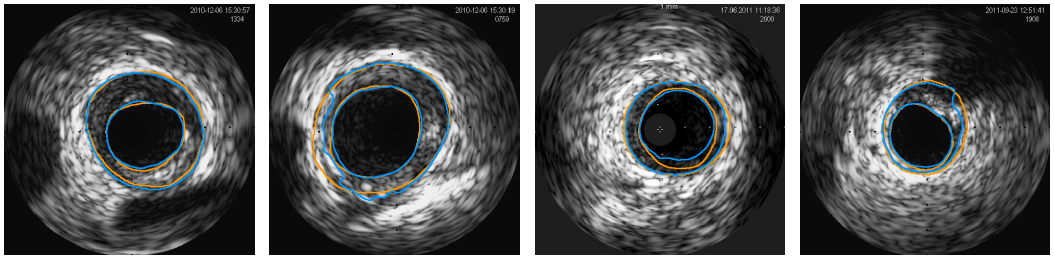
(a) 0.8157 (b) 0.8157 (c) 0.8158 (d) 0.8162



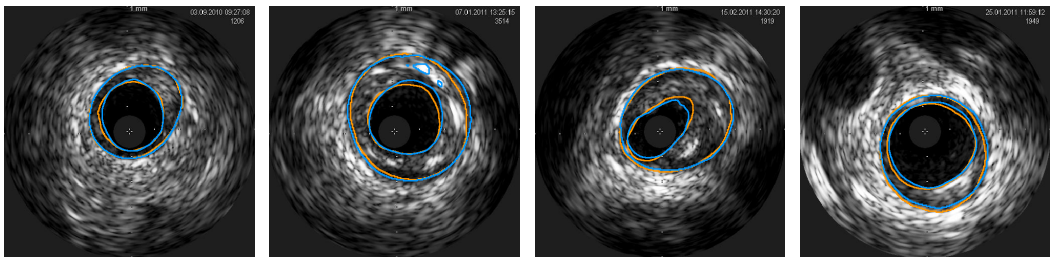
(e) 0.8163 (f) 0.8164 (g) 0.8165 (h) 0.8166



(i) 0.8167 (j) 0.8168 (k) 0.8169 (l) 0.8170



(m) 0.8172 (n) 0.8172 (o) 0.8176 (p) 0.8182



Source: Author

4 Conclusion and Future Research

During this work's execution, over 80 trainings with more than 40 different resource intensive networks were performed on HeMoLab's dataset of 160 pullbacks, totalling 287 thousand frames, of which 13 thousand are native frames and 274 thousand are artificially created frames. Each training took around one week to complete. Their summary results are presented in Appendix A.

The best results were found in training ID 92, a convolutional neural network with 9.01×10^8 parameters, from which specifications, detailed results and commentaries were described in Section 3.2. A median Average IoU of 0.8169 was achieved, with median lumen IoU of 0.9076, median plaque IoU of 0.7392 and median vessel IoU of 0.9331.

The proposed AI solution is not yet ready for full (non-supervised) automation, but is ready to be used as an AI assistant, with potential gains in reliability and acceleration of the typical exam workflow, reducing costs while assuring safety.

As explained in Section 3.2.2.5, from the perspective of the metric chosen to assess the models' quality, there is a greater difficulty in correctly segmenting the plaque region on healthy pullbacks, likely due to the easy to miss, ring-like geometrical form that the plaque will have. This presents a challenge that could be accounted for in a custom solution.

Regarding more future research possibilities, many hyperparameter options/combinations were not tested. Regarding custom losses, for instance, DICE loss and its linear combination with the crossentropy loss were never tested on the problem proposed, even though the original U-Net++ was trained with it (ZHOU et al., 2019). Arguably it could have little effect on the results, since the metric is similar in nature to the extensively used IoU, which was tested not only alone and but also linearly combined with crossentropy.

Nevertheless, when the linear combination of IoU and Crossentropy was in fact used as loss, the weights of such combination were kept constant (equal weights for both) through all tests. This custom loss could have been parameterized as $\alpha IoU + (1 - \alpha)Crossentropy$ in order to test for multiple weights, where $\alpha \in [0, 1]$. Multiple equally spaced samples of $\{\alpha \mid \alpha \in [0, 1]\}$ would be tested, such as $\alpha = 0, 0.1, 0.2, 0.3, \cdots, 1$. The concept could be generalized even further, including multiple losses, not just IoU and Crossentropy.

Still on the topic of hyperparameter options not tested, the way it is defined, a network's node structure can take countless possible forms, since it is simply required to have an entry point (input) and an exit point (output). For example, activation layers such as SELU (KLAMBAUER et al., 2017) and ELU (CLEVERT; UNTERTHINER; HOCHREITER, 2016) were not tested. Regarding base filters, due to the way the hyperparameter was proposed in Sec-

tion 2.4.4, the resulting neural network's filters will always form a geometric sequence of ratio 2 (eg. $32, 64, 128, \cdots, 2048$). Meanwhile, networks such as the SegNet (BADRINARAYANAN; KENDALL; CIPOLLA, 2016) use what would be equivalent as multiple node structures within the same network.

Finally, more specialized networks and preprocessing/postprocessing techniques might capitalize on the problems' characteristics (for instance, the output image is *always* a circuloid shape contained within another circuloid shape), improving results further. Attempts could also be made to automatically extract metrics such as plaque burden and lumen's area as regression targets. Refined methodologies such as transfer learning and multi-stage training where only specific parts of the network are trained each stage might also contribute to improved results. Studies regarding the bias-variance tradeoff could provide interesting results as well.

The author thanks LNCC and HeMoLab for the necessary computational resources to perform this work, without which it would not have been possible.

References

ALMBERG, S. S. et al. Training, validation, and clinical implementation of a deep-learning segmentation model for radiotherapy of loco-regional breast cancer. *Radiotherapy and Oncology*, v. 173, p. 62–68, ago. 2022. ISSN 01678140. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S0167814022022691>.

ALTMAN, D. G.; BLAND, J. M. Measurement in Medicine: The Analysis of Method Comparison Studies. *The Statistician*, v. 32, n. 3, p. 307, set. 1983. ISSN 00390526. Disponível em: <https://www.jstor.org/stable/2987937?origin=crossref>.

BADRINARAYANAN, V.; KENDALL, A.; CIPOLLA, R. *SegNet: A Deep Convolutional Encoder-Decoder Architecture for Image Segmentation*. arXiv, 2016. ArXiv:1511.00561 [cs]. Disponível em: <http://arxiv.org/abs/1511.00561>.

BEERS, F. van et al. Deep Neural Networks with Intersection over Union Loss for Binary Image Segmentation:. In: *Proceedings of the 8th International Conference on Pattern Recognition Applications and Methods*. Prague, Czech Republic: SCITEPRESS - Science and Technology Publications, 2019. p. 438–445. ISBN 978-989-758-351-3. Disponível em: <http://www.scitepress.org/DigitalLibrary/Link.aspx?doi=10.5220/0007347504380445>.

BUELENS, P. et al. Clinical evaluation of a deep learning model for segmentation of target volumes in breast cancer radiotherapy. *Radiotherapy and Oncology*, v. 171, p. 84–90, jun. 2022. ISSN 01678140. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S0167814022002213>.

BUSHBERG, J. T. (Ed.). *The essential physics of medical imaging*. 3rd ed. ed. Philadelphia: Wolters Kluwer Health/Lippincott Williams & Wilkins, 2012. ISBN 978-0-7817-8057-5.

CERAGIOLI, F. (Ed.). *System modeling and optimization: proceedings of the 22nd IFIP TC7 conference, July 18-22, 2005, Turin, Italy*. New York: Springer, 2006. (The International Federation for Information Processing, 199). Meeting Name: IFIP Conference on System Modeling and Optimization. ISBN 978-0-387-32774-7 978-0-387-33006-8.

CHEN, J. et al. *TransUNet: Transformers Make Strong Encoders for Medical Image Segmentation*. [S.l.], 2021. ArXiv:2102.04306 [cs] type: article. Disponível em: <http://arxiv.org/abs/2102.04306>.

CHEN, W. et al. S3D-UNet: Separable 3D U-Net for Brain Tumor Segmentation. In: CRIMI, A. et al. (Ed.). *Brainlesion: Glioma, Multiple Sclerosis, Stroke and Traumatic Brain Injuries*. Cham: Springer International Publishing, 2019. v. 11384, p. 358–368. ISBN 978-3-030-11725-2 978-3-030-11726-9. Series Title: Lecture Notes in Computer Science. Disponível em: <http://link.springer.com/10.1007/978-3-030-11726-9_32>.

CHOLLET, F.; others. *Keras*. 2015. Disponível em: <https://keras.io>.

CLEVERT, D.-A.; UNTERTHINER, T.; HOCHREITER, S. *Fast and Accurate Deep Network Learning by Exponential Linear Units (ELUs)*. arXiv, 2016. ArXiv:1511.07289 [cs]. Disponível em: <http://arxiv.org/abs/1511.07289>.

DICE, L. R. Measures of the Amount of Ecologic Association Between Species. *Ecology*, v. 26, n. 3, p. 297–302, jul. 1945. ISSN 00129658. Disponível em: <http://doi.wiley.com/10.2307/1932409>.

DUMOULIN, V.; VISIN, F. A guide to convolution arithmetic for deep learning. 2016. Publisher: arXiv Version Number: 2. Disponível em: <https://arxiv.org/abs/1603.07285>.

ESHTEHARDI, P. et al. Association of Coronary Wall Shear Stress With Atherosclerotic Plaque Burden, Composition, and Distribution in Patients With Coronary Artery Disease. *Journal of the American Heart Association*, v. 1, n. 4, p. e002543, ago. 2012. ISSN 2047-9980. Disponível em: <https://www.ahajournals.org/doi/10.1161/JAHA.112.002543>.

FELDMAN, A. et al. Utilizing a Deep Learning-Based Object Detection and Instance Segmentation Algorithm for the Delineation of Prostate and Prostate Cancer Segmentation. *International Journal of Radiation Oncology*Biology*Physics*, v. 105, n. 1, p. S197–S198, set. 2019. ISSN 03603016. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S0360301619310922>.

GOODFELLOW, I.; BENGIO, Y.; COURVILLE, A. *Deep learning*. MIT Press, 2016. OCLC: 1183962587. ISBN 978-0-262-33737-3. Disponível em: <https://www.deeplearningbook.org/>.

GORDIENKO, Y. et al. Deep Learning with Lung Segmentation and Bone Shadow Exclusion Techniques for Chest X-Ray Analysis of Lung Cancer. In: HU, Z. et al. (Ed.). *Advances in Computer Science for Engineering and Education*. Cham: Springer International Publishing, 2019. v. 754, p. 638–647. ISBN 978-3-319-91007-9 978-3-319-91008-6. Series Title: Advances in Intelligent Systems and Computing. Disponível em: <http://link.springer.com/10.1007/978-3-319-91008-6_63>.

GéRON, A. *Hands-on machine learning with Scikit-Learn, Keras, and TensorFlow: concepts, tools, and techniques to build intelligent systems*. Second edition. Beijing [China] ; Sebastopol, CA: O'Reilly Media, Inc, 2019. ISBN 978-1-4920-3264-9.

HE, K. et al. Delving Deep into Rectifiers: Surpassing Human-Level Performance on ImageNet Classification. *arXiv:1502.01852 [cs]*, fev. 2015. ArXiv: 1502.01852. Disponível em: <http://arxiv.org/abs/1502.01852>.

HONG, M.-K. *Coronary Imaging and Physiology*. [S.l.: s.n.], 2018. OCLC: 1026872003. ISBN 978-981-10-2787-1.

ISLAM, J.; ZHANG, Y. Towards Robust Lung Segmentation in Chest Radiographs with Deep Learning. 2018. Publisher: arXiv Version Number: 1. Disponível em: <https://arxiv.org/abs/1811.12638>.

JACCARD, P. THE DISTRIBUTION OF THE FLORA IN THE ALPINE ZONE.1. *New Phytologist*, v. 11, n. 2, p. 37–50, fev. 1912. ISSN 0028-646X, 1469-8137. Disponível em: <https://onlinelibrary.wiley.com/doi/10.1111/j.1469-8137.1912.tb05611.x>.

KASSANI, S. H. et al. Deep transfer learning based model for colorectal cancer histopathology segmentation: A comparative study of deep pre-trained models. *International Journal of Medical Informatics*, v. 159, p. 104669, mar. 2022. ISSN 13865056. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S1386505621002951>.

Keras Team. *Keras documentation: BatchNormalization layer*. 2022. Disponível em: <https://keras.io/api/layers/normalization_layers/batch_normalization/>.

Keras Team. *Keras documentation: Conv2D layer*. 2022. Disponível em: <https://keras.io/api/layers/convolution_layers/convolution2d/>.

Keras Team. *Keras documentation: Image segmentation metrics*. 2022. Disponível em: <https://keras.io/api/metrics/segmentation_metrics/>.

Keras Team. *Keras documentation: MaxPooling2D layer*. 2022. Disponível em: <https://keras.io/api/layers/pooling_layers/max_pooling2d/>.

Keras Team. *Keras documentation: ReLU layer*. 2022. Disponível em: <https://keras.io/api/layers/activation_layers/relu/>.

Keras Team. *Keras documentation: UpSampling2D layer*. 2022. Disponível em: <https://keras.io/api/layers/reshaping_layers/up_sampling2d/>.

KINGMA, D. P.; BA, J. Adam: A Method for Stochastic Optimization. *arXiv:1412.6980 [cs]*, jan. 2017. ArXiv: 1412.6980. Disponível em: <http://arxiv.org/abs/1412.6980>.

KLAMBAUER, G. et al. *Self-Normalizing Neural Networks*. arXiv, 2017. ArXiv:1706.02515 [cs, stat]. Disponível em: <http://arxiv.org/abs/1706.02515>.

LNCC. *SDumont - Sistema Brasileiro de Computação Petaflópica*. 2022. Disponível em: <https://sdumont.lncc.br/index.php>.

MCDANIEL, M. C. et al. Contemporary Clinical Applications of Coronary Intravascular Ultrasound. *JACC: Cardiovascular Interventions*, v. 4, n. 11, p. 1155–1167, nov. 2011. ISSN 19368798. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S1936879811006844>.

MURPHY, K. P. *Machine Learning: A Probabilistic Perspective*. Illustrated edition. Cambridge, MA: The MIT Press, 2012. ISBN 978-0-262-01802-9.

PASCAL 2. *PASCAL VOC2011 Example Segmentations*. 2012. Disponível em: <http://host.robots.ox.ac.uk/pascal/VOC/voc2012/segexamples/index.html>.

Radboud University Medical Center. *StructSeg2019 - Grand Challenge*. 2019. Disponível em: <https://structseg2019.grand-challenge.org/Evaluation/>.

ROCKAFELLAR, R. T.; WETS, R. J.-B. *Variational analysis*. Corr. 2nd print. Berlin: Springer, 2004. (Grundlehren der mathematischen Wissenschaften, 317). ISBN 978-3-540-62772-2.

RONNEBERGER, O.; FISCHER, P.; BROX, T. U-Net: Convolutional Networks for Biomedical Image Segmentation. 2015. Publisher: arXiv Version Number: 1. Disponível em: <https://arxiv.org/abs/1505.04597>.

SHELHAMER, E.; LONG, J.; DARRELL, T. Fully Convolutional Networks for Semantic Segmentation. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, v. 39, n. 4, p. 640–651, abr. 2017. ISSN 0162-8828, 2160-9292. Disponível em: <http://ieeexplore.ieee.org/document/7478072/>.

SKOURT, B. A.; HASSANI, A. E.; MAJDA, A. Lung CT Image Segmentation Using Deep Neural Networks. *Procedia Computer Science*, v. 127, p. 109–113, 2018. ISSN 18770509. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S1877050918301157>.

SøRENSEN, T. J. *A method of establishing groups of equal amplitude in plant sociology based on similarity of species and its application to analyses of the vegetation on Danish commons*. København: Munksgaard, 1948.

TAKU, N. et al. Auto-detection and segmentation of involved lymph nodes in HPV-associated oropharyngeal cancer using a convolutional deep learning neural network. *Clinical and Translational Radiation Oncology*, v. 36, p. 47–55, set. 2022. ISSN 24056308. Disponível em: <https://linkinghub.elsevier.com/retrieve/pii/S2405630822000520>.

TensorFlow Developers. *Mixed precision | TensorFlow Core*. 2022. Disponível em: <https://www.tensorflow.org/guide/mixed_precision>.

TensorFlow Developers. *TensorFlow*. Zenodo, 2022. Disponível em: <https://zenodo.org/record/4724125>.

TensorFlow Developers. *XLA: Optimizing Compiler for Machine Learning*. 2022. Disponível em: <https://www.tensorflow.org/xla>.

XU, B. et al. Empirical Evaluation of Rectified Activations in Convolutional Network. 2015. Publisher: arXiv Version Number: 2. Disponível em: <https://arxiv.org/abs/1505.00853>.

YAO, X. et al. A comprehensive survey on convolutional neural network in medical image analysis. *Multimedia Tools and Applications*, ago. 2020. ISSN 1380-7501, 1573-7721. Disponível em: <https://link.springer.com/10.1007/s11042-020-09634-7>.

ZHOU, Z. et al. UNet++: A Nested U-Net Architecture for Medical Image Segmentation. 2018. Publisher: arXiv Version Number: 1. Disponível em: <https://arxiv.org/abs/1807.10165>.

ZHOU, Z. et al. UNet++: A Nested U-Net Architecture for Medical Image Segmentation. In: STOYANOV, D. et al. (Ed.). *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*. Cham: Springer International Publishing, 2018. v. 11045, p. 3–11. ISBN 978-3-030-00888-8 978-3-030-00889-5. Series Title: Lecture Notes in Computer Science. Disponível em: <http://link.springer.com/10.1007/978-3-030-00889-5_1>.

ZHOU, Z. et al. UNet++: Redesigning Skip Connections to Exploit Multiscale Features in Image Segmentation. 2019. Publisher: arXiv Version Number: 2. Disponível em: <https://arxiv.org/abs/1912.05074>.

ZIABARI, A. et al. 2.5D Deep Learning for CT Image Reconstruction using a Multi-GPU implementation. 2018. Publisher: arXiv Version Number: 1. Disponível em: <https://arxiv.org/abs/1812.08367>.

ZIEMER, P. G. P. et al. Automated lumen segmentation using multi-frame convolutional neural networks in intravascular ultrasound datasets. *European Heart Journal - Digital Health*, v. 1, n. 1, p. 75–82, nov. 2020. ISSN 2634-3916. Disponível em: <https://academic.oup.com/ehjdh/article/1/1/75/5998645>.

ZIPES, D. P. et al. (Ed.). *Braunwald's heart disease: a textbook of cardiovascular medicine*. Eleventh edition, international edition. Philadelphia, PA: Elsevier, 2019. ISBN 978-0-323-46342-3 978-0-323-46299-0 978-0-323-55592-0.

Appendix

APPENDIX A – MASTER TABLE

A.1 Training Setup

A.1.1 Neural Network Configuration

ID	Macro Structure	Node Structure	Depth	Pool Factor	Base Filters	Kernel Size	Multi-channel	Multi-Output	Downsizing	Upsampling	T. Conv. Filters	Input Normalization	Parameters
1	U-Net++	4	6	2	32	3	[1, 1]		1	Bilinear	None		2.02×10^7
2	U-Net++	4	6	2	32	3	[1, 1]		1	Bilinear	None		2.02×10^7
3	U-Net++	4	6	2	32	3	[1, 1]		1	Bilinear	None		2.02×10^7
4	U-Net++	4	6	2	32	3	[1, 1]		1	Bilinear	None		2.02×10^7
5	U-Net++	4	6	2	32	3	[1, 1]		1	Bilinear	None		2.02×10^7
6	U-Net++	4	6	2	32	3	[1, 1]		1	Bilinear	None		2.02×10^7
7	U-Net++	4	6	2	32	3	[1, 1]		1	Bilinear	None		2.02×10^7
8	U-Net++	4	6	2	32	3	[1, 1]		1	Bilinear	None		2.02×10^7
9	U-Net	4	6	2	32	3	[1, 1]		1	Bilinear	None		1.57×10^7
10	U-Net	4	6	2	32	3	[1, 1]		1	Bilinear	None		1.57×10^7
11	U-Net++	4	6	2	64	3	[1, 1]		1	Bilinear	None		8.07×10^7
12	U-Net++	4	7	2	32	3	[1, 1]		1	Bilinear	None		8.10×10^7
13	U-Net++	4	6	2	32	3	[1, 1]	X	1	Bilinear	None		2.02×10^7
14	U-Net++	4	6	2	32	3	[1, 1]		1	Bilinear	None		2.02×10^7
15	U-Net++	4	6	2	32	5	[1, 1]		1	Bilinear	None		5.60×10^7
16	U-Net++	4	6	2	64	5	[1, 1]		2	Bilinear	None		2.24×10^8
17	U-Net++	4	6	2	32	7	[1, 1]		2	Bilinear	None		1.10×10^8
18	U-Net++	0	5	2	32	3	[1, 1]	X	1	Bilinear	None		4.98×10^6
19	U-Net++	4	6	2	32	3	[1, 1]		1	Bilinear	None		2.02×10^7
20	U-Net++	4	6	2	32	3	[1, 1]	X	2	Bilinear	None		2.02×10^7
21	U-Net++	4	6	2	32	3	[1, 1]		2	Bilinear	None		2.02×10^7
22	U-Net++	4	6	2	32	3	[1, 1]		2	Bilinear	None		2.02×10^7
23	U-Net++	4	8	2	32	3	[1, 1]		1	Bilinear	None		3.24×10^8
24	U-Net++	4	8	2	32	3	[1, 1]		2	Bilinear	None		3.24×10^8
25	U-Net++	5	6	2	32	3	[1, 1]		1	Bilinear	None		3.69×10^7
26	U-Net++	4	7	2	32	3	[1, 1]		1	Bilinear	None		8.10×10^7

Continued on next page

ID	Macro Structure	Node Structure	Depth	Pool Factor	Base Filters	Kernel Size	Multi-channel	Multi-Output	Downsizing	Upsampling	T. Conv. Filters	Input Normalization	Parameters
27	U-Net++	4	7	2	32	5	[1, 1]		2	Bilinear	None		2.25×10^8
28	U-Net++	4	6	2	32	3	[1, 1]		1	Bilinear	None		2.02×10^7
31	U-Net++	4	6	2	32	3	[1, 1]		4	Bilinear	None		2.02×10^7
32	U-Net++	4	8	2	32	3	[1, 1]		1	Bilinear	None		3.24×10^8
33	U-Net++	4	8	2	32	3	[1, 1]		1	Bilinear	None		3.24×10^8
34	U-Net++	4	7	2	32	5	[1, 1]		2	Bilinear	None		2.25×10^8
35	U-Net++	4	7	2	32	5	[1, 1]		2	Bilinear	None		2.25×10^8
36	U-Net++	4	8	2	32	5	[1, 1]		1	Bilinear	None		9.01×10^8
37	U-Net++	4	9	2	16	3	[1, 1]		1	Bilinear	None		3.24×10^8
38	U-Net++	4	9	2	8	3	[1, 1]		2	Bilinear	None		8.11×10^7
39	U-Net++	4	9	2	16	3	[1, 1]		2	Bilinear	None		3.24×10^8
40	U-Net++	4	8	2	32	3	[1, 1]		1	Bilinear	None		3.24×10^8
41	U-Net++	5	6	2	32	3	[1, 1]		1	Bilinear	None		3.69×10^7
42	U-Net++	5	6	2	32	3	[1, 1]		1	Bilinear	None		3.69×10^7
43	U-Net++	4	7	2	32	5	[1, 1]		2	Bilinear	None		2.25×10^8
44	U-Net++	4	6	2	32	3	[1, 1]		1	Bilinear	None		2.02×10^7
45	U-Net++	4	7	2	32	3	[1, 1]		1	Bilinear	None		8.10×10^7
46	U-Net++	4	8	2	32	3	[1, 1]		1	Bilinear	None		3.24×10^8
47	U-Net++	4	7	2	32	5	[1, 1]		2	Bilinear	None		2.25×10^8
48	U-Net++	1	7	2	32	3	[1, 1]		1	Bilinear	None		1.48×10^8
49	U-Net++	5	7	2	32	3	[1, 1]		1	Bilinear	None		1.48×10^8
50	U-Net++	7	7	2	32	3	[1, 1]		1	Bilinear	None		1.82×10^8
51	U-Net++	4	8	2	32	5	[1, 1]		1	Bilinear	None		9.01×10^8
52	U-Net++	5	7	2	32	3	[1, 1]		1	Bilinear	None		1.48×10^8
53	U-Net++	4	7	2	32	3	[3, 1]		1	Bilinear	None		8.10×10^7
54	U-Net++	4	7	2	32	3	[5, 1]		1	Bilinear	None		8.10×10^7
55	U-Net++	4	7	2	32	3	[7, 1]		1	Bilinear	None		8.10×10^7
56	U-Net++	4	7	2	32	3	[9, 1]		1	Bilinear	None		8.10×10^7
57	U-Net++	4	7	2	32	3	[11, 1]		1	Bilinear	None		8.10×10^7
58	U-Net++	4	8	2	32	5	[1, 1]		1	Bilinear	None		9.01×10^8
60	U-Net++	4	8	2	32	5	[11, 1]		1	Bilinear	None		9.01×10^8
62	S-Net++	10	3	2	128	7	[1, 1]		2	T. Conv.	None		2.81×10^7
63	U-Net++	4	7	2	64	[9, 5]	[11, 1]		2	Bilinear	None		9.07×10^8

Continued on next page

ID	Macro Structure	Node Structure	Depth	Pool Factor	Base Filters	Kernel Size	Multi-channel	Multi-Output	Downsizing	Upsampling	T. Conv. Filters	Input Normalization	Parameters
64	U-Net++	4	8	2	32	3	[1, 1]		1	Bilinear	None		3.24×10^8
65	U-Net++	4	8	2	32	3	[1, 1]		1	Bilinear	None		3.24×10^8
66	U-Net++	4	8	2	32	5	[1, 1]		1	Bilinear	None		9.01×10^8
67	U-Net++	4	8	2	32	5	[1, 1]	X	1	Bilinear	None		9.01×10^8
68	U-Net++	5	7	2	32	3	[1, 1]		1	Bilinear	None		1.48×10^8
69	U-Net++	4	8	2	32	5	[1, 1]		1	Bilinear	None		9.01×10^8
70	U-Net++	8	7	2	32	3	[1, 1]		1	Bilinear	None		1.48×10^8
71	U-Net++	4	8	2	32	5	[1, 1]		1	Bilinear	None		9.01×10^8
72	U-Net++	4	8	2	32	5	[1, 1]		1	Bilinear	None		9.01×10^8
74	U-Net	8	7	2	32	3	[7, 1]		1	Bilinear	None		1.26×10^8
75	U-Net++	4	7	2	64	[9, 5]	[5, 1]		2	Bilinear	None		9.07×10^8
76	U-Net++	4	7	2	64	[9, 5]	[7, 1]		2	Bilinear	None		9.07×10^8
77	U-Net++	4	7	2	64	[9, 5]	[9, 1]		2	Bilinear	None		9.07×10^8
78	U-Net++	4	7	2	64	[9, 5]	[11, 1]		2	Bilinear	None		9.07×10^8
79	U-Net++	4	6	2	128	[9, 3]	[11, 1]		2	Bilinear	None		9.19×10^8
80	S-Net++	10	4	2	128	7	[1, 1]		2	T. Conv.	None		1.26×10^8
82	U-Net++	4	7	2	64	[9, 5]	[7, 1]		2	Bilinear	None		9.07×10^8
83	U-Net++	4	7	2	64	[9, 5]	[11, 1]		2	Bilinear	None		9.07×10^8
84	U-Net++	4	8	2	32	5	[1, 1]	X	1	Bilinear	None	X	9.01×10^8
85	U-Net++	4	7	2	64	[9, 5]	[7, 1]		2	Bilinear	None		9.07×10^8
86	U-Net++	4	6	2	128	[9, 3]	[11, 1]		2	Bilinear	None		9.19×10^8
87	U-Net++	4	8	2	32	5	[11, 1]		1	Bilinear	None		9.01×10^8
88	U-Net++	4	8	2	32	5	[1, 1]	X	1	Bilinear	None		9.01×10^8
89	U-Net++	4	7	2	64	[9, 5]	[11, 1]		2	Bilinear	None		9.07×10^8
90	U-Net++	4	7	2	64	[9, 5]	[11, 1]		2	Bilinear	None		9.07×10^8
92	U-Net++	4	8	2	32	5	[1, 1]		1	Bilinear	None		9.01×10^8
93	U-Net++	5	5	4	64	[9, 5]	[1, 1]	X	2	Bilinear	None		1.07×10^8
94	U-Net++	4	5	4	64	[9, 5]	[1, 1]	X	2	Bilinear	None		5.95×10^7

A.1.2 Fit and TensorFlow Configuration

	Fit Config							Tensorflow		
	Epochs	Learning Rate	Batch Size	Optimizer	Loss	Base ID	Dataset	Multi-GPU	Mixed Precision	Auto-Clustering
ID										
1	30	0.001	8	Adam	Crossentropy		Hybrid			X
2	50	0.001	8	Adam	Crossentropy	1	Hybrid			X
3	30	0.001	8	Adam	IoU		Hybrid			X
4	30	0.001	32	Adam	Crossentropy		Hybrid	X		X
5	50	0.001	8	Adam	IoU	3	Hybrid			X
6	50	0.001	32	Adam	Crossentropy	4	Hybrid	X		X
7	30	0.001	4	Adam	IoU		Hybrid			X
8	30	0.001	32	Adam	IoU		Hybrid	X		X
9	30	0.001	32	Adam	Crossentropy		Hybrid	X		X
10	30	0.001	32	Adam	IoU		Hybrid	X		X
11	30	0.001	8	Adam	IoU		Hybrid	X		X
12	30	0.001	8	Adam	IoU		Hybrid	X		X
13	30	0.001	32	Adam	IoU		Hybrid	X		X
14	50	0.001	32	Adam	IoU	8	Hybrid	X		X
15	30	0.001	32	Adam	IoU		Hybrid	X		X
16	30	0.001	32	Adam	IoU		Hybrid	X	X	X
17	30	0.001	32	Adam	IoU		Hybrid	X	X	X
18	30	0.0003	32	Adam	IoU		Hybrid	X		X
19	30	0.001	32	Adam	IoU		Hybrid	X	X	X
20	30	0.001	32	Adam	IoU		Hybrid	X		X
21	30	0.001	32	Adam	Crossentropy		Hybrid	X		X
22	30	0.001	32	Adam	IoU		Hybrid	X		X
23	30	0.001	8	Adam	IoU		Hybrid	X		X
24	30	0.001	8	Adam	IoU		Hybrid	X	X	X
25	30	0.001	8	Adam	IoU		Hybrid	X		X
26	30	0.001	8	Adam	Crossentropy		Hybrid	X		X
27	30	0.001	8	Adam	IoU		Hybrid	X		X
28	30	0.0003	32	Adam	IoU		Hybrid	X		X
31	30	0.001	32	Adam	IoU		Hybrid	X		X

Continued on next page

	Fit Config							Tensorflow		
	Epochs	Learning Rate	Batch Size	Optimizer	Loss	Base ID	Dataset	Multi-GPU	Mixed Precision	Auto-Clustering
ID										
32	50	0.001	8	Adam	IoU	23	Hybrid	X		X
33	50	0.0003	8	Adam	IoU	23	Hybrid	X		X
34	50	0.001	8	Adam	IoU	27	Hybrid	X		X
35	50	0.0003	8	Adam	IoU	27	Hybrid	X		X
36	30	0.0001	8	Adam	IoU		Hybrid	X	X	X
37	30	0.001	8	Adam	IoU		Hybrid	X	X	X
38	30	0.001	32	Adam	IoU		Hybrid	X	X	X
39	30	0.001	8	Adam	IoU		Hybrid	X	X	X
40	30	0.001	8	Adam	Crossentropy		Hybrid	X		X
41	50	0.001	8	Adam	IoU	25	Hybrid	X		X
42	50	0.0003	8	Adam	IoU	25	Hybrid	X		X
43	30	0.001	8	Adam	Crossentropy		Hybrid	X		X
44	30	0.001	16	Adam	IoU		Hybrid	X		X
45	30	0.001	16	Adam	IoU		Hybrid	X		X
46	30	0.001	16	Adam	IoU		Hybrid	X		X
47	30	0.001	16	Adam	IoU		Hybrid	X		X
48	30	0.001	8	Adam	IoU		Hybrid	X		X
49	30	0.001	8	Adam	IoU		Hybrid	X		X
50	30	0.001	8	Adam	IoU		Hybrid	X		X
51	30	0.0003	8	Adam	IoU		Hybrid	X	X	X
52	50	0.001	8	Adam	IoU	49	Hybrid	X		X
53	30	0.001	8	Adam	IoU		Hybrid	X		X
54	30	0.0003	8	Adam	IoU		Hybrid	X		X
55	30	0.0003	8	Adam	IoU		Hybrid	X		X
56	30	0.0003	8	Adam	IoU		Hybrid	X		X
57	30	0.0003	8	Adam	IoU		Hybrid	X		X
58	50	0.0003	8	Adam	IoU	51	Hybrid	X	X	X
60	30	0.0003	8	Adam	IoU + Crossentropy		Hybrid	X	X	X
62	30	0.0003	32	Adam	IoU + Crossentropy		Hybrid	X		X
63	30	0.0003	16	Adam	IoU + Crossentropy		Hybrid	X	X	X
64	70	0.0001	8	Adam	IoU	33	Hybrid	X		X
65	70	0.0003	8	Adam	IoU	33	Hybrid	X		X

Continued on next page

ID	Fit Config							Tensorflow		
	Epochs	Learning Rate	Batch Size	Optimizer	Loss	Base ID	Dataset	Multi-GPU	Mixed Precision	Auto-Clustering
66	50	0.0001	8	Adam	IoU	36	Hybrid	X	X	X
67	30	0.0003	8	Adam	IoU		Hybrid	X	X	X
68	50	0.0003	8	Adam	IoU	49	Hybrid	X		X
69	30	0.0003	8	Adam	Crossentropy		Hybrid	X	X	X
70	30	0.001	8	Adam	IoU		Hybrid	X		X
71	70	0.0003	8	Adam	IoU	58	Hybrid	X	X	X
72	50	0.0003	8	Adam	Crossentropy	69	Hybrid	X	X	X
74	30	0.0003	16	Adam	IoU		Hybrid	X		X
75	30	0.0003	16	Adam	IoU + Crossentropy		Hybrid	X	X	X
76	30	0.0003	16	Adam	IoU + Crossentropy		Hybrid	X	X	X
77	30	0.0003	16	Adam	IoU + Crossentropy		Hybrid	X	X	X
78	50	0.0003	16	Adam	IoU + Crossentropy	63	Hybrid	X	X	X
79	30	0.0003	16	Adam	IoU + Crossentropy		Hybrid	X	X	X
80	30	0.0003	32	Adam	IoU + Crossentropy		Hybrid	X		X
82	50	0.0003	16	Adam	IoU + Crossentropy	76	Hybrid	X	X	X
83	70	0.0003	16	Adam	IoU + Crossentropy	78	Hybrid	X	X	X
84	70	0.0003	8	Adam	IoU + Crossentropy		Hybrid	X	X	X
85	70	0.0001	8	Adam	IoU + Crossentropy	82	Hybrid	X	X	X
86	50	0.0003	12	Adam	IoU + Crossentropy	79	Hybrid	X	X	X
87	50	0.0003	8	Adam	IoU + Crossentropy	60	Hybrid	X	X	X
88	70	0.0003	8	Adam	IoU + Crossentropy		Hybrid	X	X	X
89	90	0.0003	16	Adam	IoU + Crossentropy	83	Hybrid	X	X	X
90	110	0.0003	16	Adam	IoU + Crossentropy	89	Hybrid	X	X	X
92	641	0.0003	8	Adam	IoU		Native	X	X	X
93	107	0.0003	8	Adam	IoU + Crossentropy		Native	X		X
94	107	0.0003	8	Adam	IoU + Crossentropy		Native	X		X

A.2 Results

A.2.1 Jaccard Index - IoU

	Mean				Median				IQR			
	Average	Lumen	Plaque	Vessel	Average	Lumen	Plaque	Vessel	Average	Lumen	Plaque	Vessel
ID												
1	0.7223	0.8413	0.6034	0.8594	0.7433	0.8648	0.6359	0.8973	0.1393	0.0861	0.2159	0.1076
2	0.7291	0.8323	0.6259	0.8537	0.7595	0.8768	0.6597	0.8991	0.1470	0.0886	0.2153	0.1167
3	0.7065	0.8257	0.5873	0.8253	0.7387	0.8745	0.6193	0.8779	0.1684	0.1051	0.2482	0.1589
4	0.7190	0.8353	0.6028	0.8398	0.7462	0.8727	0.6334	0.8809	0.1585	0.1051	0.2181	0.1299
5	0.7086	0.8259	0.5914	0.8238	0.7457	0.8794	0.6245	0.8912	0.1769	0.0992	0.2602	0.1683
6	0.7386	0.8582	0.6191	0.8597	0.7582	0.8835	0.6436	0.8902	0.1479	0.0887	0.2207	0.1043
7	0.6918	0.8010	0.5826	0.8165	0.7242	0.8589	0.6146	0.8704	0.1865	0.1253	0.2405	0.1739
8	0.7404	0.8546	0.6262	0.8599	0.7592	0.8795	0.6564	0.8896	0.1395	0.0905	0.2147	0.1123
9	0.7091	0.8220	0.5962	0.8265	0.7408	0.8676	0.6262	0.8701	0.1636	0.1063	0.2246	0.1415
10	0.7145	0.8053	0.6237	0.8346	0.7506	0.8720	0.6563	0.8911	0.1729	0.1078	0.2172	0.1347
11	0.7349	0.8491	0.6208	0.8558	0.7591	0.8821	0.6503	0.8945	0.1397	0.0831	0.2211	0.1279
12	0.7727	0.8800	0.6654	0.8915	0.7964	0.8957	0.7068	0.9143	0.1214	0.0699	0.1922	0.0742
13	0.7478	0.8536	0.6420	0.8657	0.7751	0.8906	0.6759	0.8995	0.1515	0.0883	0.2237	0.1032
14	0.7488	0.8553	0.6423	0.8651	0.7670	0.8842	0.6701	0.8977	0.1408	0.0905	0.2066	0.1119
15	0.7718	0.8705	0.6731	0.8963	0.7943	0.8867	0.7113	0.9207	0.1291	0.0827	0.1979	0.0720
16	0.7535	0.8603	0.6467	0.8776	0.7831	0.8816	0.6902	0.9124	0.1475	0.0827	0.2226	0.0940
17	0.7578	0.8684	0.6473	0.8761	0.7882	0.8933	0.6943	0.9090	0.1358	0.0798	0.2177	0.0857
18	0.7527	0.8731	0.6322	0.8658	0.7670	0.8911	0.6520	0.8873	0.1259	0.0730	0.1960	0.1115
19	0.7391	0.8485	0.6297	0.8521	0.7644	0.8845	0.6618	0.8966	0.1622	0.0963	0.2339	0.1300
20	0.7518	0.8595	0.6440	0.8910	0.7727	0.8747	0.6777	0.9132	0.1278	0.0724	0.2064	0.0717
21	0.7627	0.8706	0.6547	0.8958	0.7815	0.8828	0.6886	0.9153	0.1262	0.0732	0.2022	0.0690
22	0.7637	0.8702	0.6573	0.8931	0.7832	0.8849	0.6903	0.9147	0.1162	0.0767	0.1888	0.0679
23	0.7803	0.8819	0.6787	0.8939	0.8016	0.8973	0.7175	0.9162	0.1265	0.0700	0.2021	0.0706
24	0.7609	0.8727	0.6490	0.8834	0.7799	0.8868	0.6818	0.9008	0.1196	0.0722	0.1970	0.0834
25	0.7116	0.8109	0.6123	0.8392	0.7420	0.8656	0.6431	0.8923	0.1721	0.1090	0.2283	0.1343
26	0.7762	0.8731	0.6792	0.9005	0.7992	0.8908	0.7223	0.9205	0.1293	0.0759	0.1983	0.0663
27	0.7777	0.8781	0.6773	0.9004	0.8003	0.8930	0.7154	0.9187	0.1217	0.0740	0.1918	0.0631
28	0.7143	0.8147	0.6139	0.8366	0.7506	0.8694	0.6477	0.8959	0.1696	0.1101	0.2381	0.1445
31	0.7591	0.8756	0.6426	0.8813	0.7805	0.8901	0.6791	0.9080	0.1316	0.0707	0.2101	0.0893
32	0.7912	0.8894	0.6930	0.8984	0.8115	0.9033	0.7321	0.9207	0.1185	0.0662	0.1896	0.0673
33	0.7928	0.8877	0.6979	0.9045	0.8143	0.9010	0.7398	0.9254	0.1219	0.0671	0.1921	0.0650
34	0.7903	0.8865	0.6941	0.9093	0.8114	0.9010	0.7309	0.9279	0.1137	0.0679	0.1807	0.0617
35	0.7908	0.8863	0.6954	0.9079	0.8125	0.9000	0.7342	0.9270	0.1178	0.0679	0.1896	0.0639
36	0.7812	0.8798	0.6825	0.8990	0.8059	0.8967	0.7248	0.9189	0.1209	0.0732	0.1854	0.0605
37	0.7753	0.8686	0.6820	0.8969	0.7968	0.8870	0.7178	0.9184	0.1223	0.0796	0.1879	0.0700
38	0.4962	0.7074	0.2850	0.5850	0.4998	0.7303	0.2829	0.5973	0.1316	0.1764	0.1881	0.1731
39	0.7371	0.8432	0.6310	0.8615	0.7569	0.8700	0.6648	0.8863	0.1302	0.1090	0.2001	0.1039
40	0.7841	0.8800	0.6882	0.9057	0.8039	0.8956	0.7260	0.9219	0.1192	0.0710	0.1853	0.0568
41	0.7190	0.8350	0.6031	0.8378	0.7506	0.8784	0.6421	0.8909	0.1703	0.0972	0.2571	0.1510
42	0.6906	0.7984	0.5828	0.8081	0.7295	0.8670	0.6198	0.8749	0.2159	0.1553	0.2726	0.1987
43	0.7824	0.8822	0.6825	0.9038	0.8017	0.8978	0.7196	0.9227	0.1252	0.0674	0.2011	0.0589
44	0.7756	0.8763	0.6750	0.8973	0.7940	0.8913	0.7040	0.9150	0.1191	0.0712	0.1881	0.0698
45	0.7767	0.8784	0.6750	0.8936	0.7991	0.8931	0.7123	0.9150	0.1213	0.0685	0.1865	0.0729
46	0.7701	0.8777	0.6625	0.8949	0.7924	0.8930	0.7029	0.9140	0.1225	0.0711	0.1922	0.0639
47	0.7795	0.8759	0.6832	0.8995	0.8022	0.8919	0.7195	0.9231	0.1155	0.0725	0.1830	0.0673
48	0.6676	0.7980	0.5373	0.8201	0.6903	0.8317	0.5675	0.8689	0.1589	0.1017	0.2444	0.1323

Continued on next page

	Mean				Median				IQR			
	Average	Lumen	Plaque	Vessel	Average	Lumen	Plaque	Vessel	Average	Lumen	Plaque	Vessel
ID												
49	0.7398	0.8665	0.6130	0.8718	0.7558	0.8825	0.6403	0.8968	0.1272	0.0773	0.2181	0.0883
50	0.7547	0.8738	0.6357	0.8666	0.7741	0.8904	0.6663	0.8975	0.1357	0.0703	0.2224	0.1003
51	0.7853	0.8843	0.6864	0.9033	0.8077	0.8980	0.7273	0.9242	0.1203	0.0665	0.1942	0.0649
52	0.7704	0.8819	0.6589	0.8902	0.7886	0.8964	0.6907	0.9126	0.1215	0.0687	0.1972	0.0775
53	0.7184	0.8413	0.5955	0.8542	0.7467	0.8684	0.6354	0.8899	0.1627	0.1058	0.2476	0.1136
54	0.7658	0.8800	0.6516	0.8847	0.7897	0.8963	0.6922	0.9100	0.1336	0.0739	0.2210	0.0791
55	0.7754	0.8791	0.6718	0.8946	0.7984	0.8965	0.7120	0.9182	0.1288	0.0747	0.2052	0.0735
56	0.7752	0.8759	0.6745	0.8952	0.8018	0.8949	0.7159	0.9211	0.1273	0.0775	0.1959	0.0728
57	0.7491	0.8730	0.6252	0.8806	0.7710	0.8871	0.6663	0.9051	0.1332	0.0749	0.2205	0.0812
58	0.7983	0.8907	0.7058	0.9091	0.8208	0.9041	0.7486	0.9301	0.1134	0.0654	0.1841	0.0594
60	0.7819	0.8798	0.6839	0.8963	0.8060	0.8977	0.7266	0.9221	0.1254	0.0720	0.1924	0.0746
62	0.7594	0.8729	0.6459	0.8754	0.7743	0.8869	0.6750	0.9007	0.1214	0.0710	0.1978	0.0946
63	0.7840	0.8831	0.6850	0.8984	0.8062	0.8979	0.7263	0.9240	0.1243	0.0684	0.1948	0.0676
64	0.7837	0.8842	0.6833	0.8996	0.8031	0.8980	0.7226	0.9204	0.1220	0.0703	0.1989	0.0677
65	0.7893	0.8868	0.6918	0.9024	0.8096	0.9008	0.7332	0.9238	0.1201	0.0685	0.1939	0.0660
66	0.7928	0.8850	0.7006	0.9059	0.8184	0.9015	0.7444	0.9267	0.1185	0.0676	0.1838	0.0625
67	0.7846	0.8815	0.6878	0.8977	0.8092	0.8944	0.7348	0.9236	0.1241	0.0704	0.1962	0.0690
68	0.7544	0.8765	0.6324	0.8759	0.7732	0.8923	0.6627	0.9001	0.1309	0.0709	0.2117	0.0883
69	0.7856	0.8828	0.6883	0.9014	0.8065	0.8978	0.7295	0.9219	0.1176	0.0699	0.1877	0.0632
70	0.7241	0.8465	0.6016	0.8584	0.7469	0.8769	0.6264	0.8865	0.1376	0.0841	0.2137	0.1112
71	0.7973	0.8899	0.7048	0.9088	0.8195	0.9034	0.7467	0.9302	0.1158	0.0650	0.1828	0.0612
72	0.7931	0.8841	0.7021	0.9077	0.8148	0.8973	0.7432	0.9274	0.1179	0.0698	0.1855	0.0637
74	0.0360	0.0010	0.0709	0.1506	0.0341	0.0000	0.0673	0.1445	0.0225	0.0000	0.0464	0.0733
75	0.7817	0.8820	0.6814	0.9002	0.8030	0.8966	0.7212	0.9228	0.1256	0.0688	0.1992	0.0710
76	0.7924	0.8877	0.6971	0.9069	0.8164	0.9032	0.7382	0.9259	0.1205	0.0670	0.1854	0.0617
77	0.7825	0.8782	0.6868	0.9037	0.8091	0.8954	0.7313	0.9267	0.1275	0.0717	0.2004	0.0656
78	0.7967	0.8892	0.7042	0.9104	0.8187	0.9035	0.7428	0.9321	0.1096	0.0651	0.1797	0.0583
79	0.7615	0.8665	0.6564	0.8901	0.7910	0.8863	0.6997	0.9176	0.1349	0.0780	0.2122	0.0751
80	0.7627	0.8759	0.6494	0.8737	0.7781	0.8915	0.6793	0.8997	0.1226	0.0687	0.2038	0.0995
82	0.8014	0.8908	0.7120	0.9100	0.8263	0.9041	0.7550	0.9305	0.1174	0.0656	0.1822	0.0595
83	0.8023	0.8908	0.7138	0.9114	0.8262	0.9055	0.7552	0.9328	0.1135	0.0662	0.1782	0.0587
84	0.7821	0.8861	0.6781	0.8878	0.8074	0.9016	0.7254	0.9139	0.1260	0.0722	0.1998	0.0811
85	0.8035	0.8921	0.7149	0.9126	0.8278	0.9048	0.7571	0.9326	0.1150	0.0654	0.1791	0.0568
86	0.7897	0.8839	0.6954	0.9063	0.8151	0.9001	0.7339	0.9300	0.1231	0.0709	0.1877	0.0669
87	0.7899	0.8849	0.6948	0.9035	0.8146	0.9011	0.7380	0.9256	0.1171	0.0726	0.1817	0.0666
88	0.7833	0.8892	0.6774	0.8877	0.8057	0.9031	0.7203	0.9139	0.1229	0.0683	0.1993	0.0777
89	0.8024	0.8929	0.7119	0.9104	0.8244	0.9076	0.7521	0.9320	0.1163	0.0640	0.1825	0.0588
90	0.8051	0.8936	0.7166	0.9115	0.8279	0.9071	0.7546	0.9343	0.1156	0.0652	0.1809	0.0585
92	0.8154	0.9006	0.7302	0.9155	0.8378	0.9153	0.7679	0.9365	0.1118	0.0626	0.1719	0.0546
93	0.7865	0.8865	0.6866	0.8985	0.8064	0.9003	0.7215	0.9186	0.1134	0.0652	0.1793	0.0660
94	0.4040	0.6041	0.2040	0.3417	0.3871	0.6173	0.1566	0.2950	0.2404	0.4083	0.1642	0.2066

A.2.2 Sørensen-Dice Index - DICE

	Mean				Median				IQR			
	Average	Lumen	Plaque	Vessel	Average	Lumen	Plaque	Vessel	Average	Lumen	Plaque	Vessel
ID												
1	0.8239	0.9097	0.7380	0.9200	0.8469	0.9275	0.7774	0.9458	0.1028	0.0499	0.1657	0.0611
2	0.8278	0.8996	0.7560	0.9149	0.8582	0.9343	0.7950	0.9468	0.1050	0.0510	0.1607	0.0662
3	0.8092	0.8957	0.7226	0.8957	0.8419	0.9331	0.7649	0.9350	0.1274	0.0609	0.1954	0.0930

Continued on next page

	Mean				Median				IQR			
ID	Average	Lumen	Plaque	Vessel	Average	Lumen	Plaque	Vessel	Average	Lumen	Plaque	Vessel
4	0.8212	0.9043	0.7381	0.9080	0.8483	0.9320	0.7756	0.9367	0.1142	0.0609	0.1674	0.0756
5	0.8089	0.8928	0.7251	0.8921	0.8474	0.9359	0.7689	0.9425	0.1329	0.0571	0.2030	0.0982
6	0.8360	0.9209	0.7510	0.9214	0.8570	0.9381	0.7832	0.9419	0.1052	0.0506	0.1662	0.0594
7	0.7996	0.8795	0.7197	0.8909	0.8323	0.9241	0.7613	0.9307	0.1426	0.0745	0.1903	0.1032
8	0.8374	0.9184	0.7564	0.9211	0.8584	0.9359	0.7926	0.9416	0.0995	0.0517	0.1602	0.0639
9	0.8127	0.8947	0.7307	0.8983	0.8446	0.9291	0.7701	0.9305	0.1243	0.0620	0.1738	0.0828
10	0.8157	0.8773	0.7542	0.9015	0.8522	0.9316	0.7925	0.9424	0.1281	0.0628	0.1626	0.0777
11	0.8332	0.9140	0.7524	0.9177	0.8580	0.9374	0.7881	0.9443	0.1025	0.0474	0.1665	0.0729
12	0.8601	0.9348	0.7853	0.9407	0.8836	0.9450	0.8282	0.9552	0.0821	0.0392	0.1353	0.0410
13	0.8421	0.9160	0.7682	0.9243	0.8688	0.9421	0.8066	0.9471	0.1085	0.0502	0.1638	0.0582
14	0.8440	0.9184	0.7697	0.9243	0.8631	0.9386	0.8025	0.9461	0.0999	0.0516	0.1509	0.0633
15	0.8605	0.9292	0.7919	0.9434	0.8823	0.9399	0.8313	0.9587	0.0880	0.0469	0.1388	0.0395
16	0.8459	0.9225	0.7693	0.9311	0.8747	0.9371	0.8167	0.9542	0.1046	0.0471	0.1612	0.0523
17	0.8472	0.9258	0.7686	0.9297	0.8778	0.9437	0.8196	0.9523	0.0942	0.0449	0.1564	0.0477
18	0.8470	0.9308	0.7632	0.9257	0.8627	0.9424	0.7894	0.9403	0.0882	0.0412	0.1451	0.0633
19	0.8355	0.9132	0.7579	0.9148	0.8618	0.9387	0.7965	0.9455	0.1146	0.0550	0.1741	0.0743
20	0.8464	0.9227	0.7701	0.9405	0.8680	0.9332	0.8079	0.9546	0.0895	0.0414	0.1505	0.0396
21	0.8539	0.9296	0.7781	0.9434	0.8737	0.9378	0.8156	0.9558	0.0883	0.0415	0.1451	0.0380
22	0.8551	0.9292	0.7810	0.9415	0.8751	0.9389	0.8168	0.9554	0.0806	0.0435	0.1353	0.0373
23	0.8660	0.9359	0.7961	0.9421	0.8863	0.9458	0.8355	0.9563	0.0850	0.0391	0.1409	0.0388
24	0.8531	0.9307	0.7755	0.9363	0.8721	0.9400	0.8108	0.9478	0.0834	0.0408	0.1429	0.0465
25	0.8144	0.8846	0.7443	0.9049	0.8456	0.9280	0.7828	0.9431	0.1280	0.0636	0.1737	0.0772
26	0.8634	0.9306	0.7962	0.9461	0.8858	0.9423	0.8388	0.9586	0.0878	0.0428	0.1376	0.0363
27	0.8649	0.9338	0.7960	0.9461	0.8864	0.9435	0.8341	0.9576	0.0806	0.0416	0.1339	0.0345
28	0.8159	0.8871	0.7448	0.9024	0.8523	0.9301	0.7862	0.9451	0.1218	0.0643	0.1805	0.0833
31	0.8505	0.9324	0.7685	0.9346	0.8724	0.9419	0.8089	0.9518	0.0923	0.0398	0.1530	0.0498
32	0.8738	0.9404	0.8073	0.9445	0.8936	0.9492	0.8453	0.9587	0.0787	0.0368	0.1301	0.0368
33	0.8750	0.9394	0.8106	0.9482	0.8955	0.9479	0.8504	0.9612	0.0801	0.0374	0.1310	0.0354
34	0.8738	0.9387	0.8089	0.9512	0.8931	0.9479	0.8446	0.9626	0.0757	0.0378	0.1237	0.0335
35	0.8741	0.9386	0.8097	0.9504	0.8939	0.9474	0.8467	0.9621	0.0790	0.0378	0.1298	0.0347
36	0.8668	0.9347	0.7989	0.9452	0.8893	0.9455	0.8404	0.9577	0.0811	0.0410	0.1283	0.0331
37	0.8636	0.9280	0.7991	0.9438	0.8835	0.9401	0.8357	0.9575	0.0813	0.0451	0.1303	0.0384
38	0.6245	0.8206	0.4284	0.7297	0.6292	0.8442	0.4411	0.7479	0.1370	0.1197	0.2307	0.1374
39	0.8356	0.9116	0.7595	0.9229	0.8562	0.9305	0.7987	0.9397	0.0941	0.0633	0.1475	0.0594
40	0.8695	0.9349	0.8041	0.9492	0.8889	0.9449	0.8412	0.9594	0.0787	0.0399	0.1281	0.0309
41	0.8179	0.9024	0.7333	0.9047	0.8510	0.9353	0.7820	0.9423	0.1285	0.0559	0.1972	0.0875
42	0.7945	0.8749	0.7140	0.8823	0.8362	0.9288	0.7653	0.9333	0.1645	0.0927	0.2155	0.1186
43	0.8680	0.9362	0.7999	0.9479	0.8870	0.9461	0.8369	0.9598	0.0842	0.0377	0.1398	0.0321
44	0.8640	0.9327	0.7954	0.9445	0.8821	0.9425	0.8263	0.9556	0.0805	0.0400	0.1321	0.0385
45	0.8641	0.9340	0.7942	0.9421	0.8851	0.9435	0.8320	0.9556	0.0816	0.0385	0.1302	0.0402
46	0.8592	0.9336	0.7849	0.9431	0.8808	0.9435	0.8255	0.9551	0.0835	0.0400	0.1366	0.0352
47	0.8662	0.9322	0.8003	0.9448	0.8878	0.9429	0.8369	0.9600	0.0772	0.0407	0.1266	0.0367
48	0.7795	0.8804	0.6785	0.8937	0.8078	0.9081	0.7241	0.9299	0.1269	0.0613	0.2054	0.0778
49	0.8364	0.9265	0.7463	0.9292	0.8555	0.9376	0.7807	0.9456	0.0948	0.0439	0.1658	0.0497
50	0.8473	0.9312	0.7634	0.9252	0.8679	0.9420	0.7997	0.9460	0.0963	0.0396	0.1641	0.0566
51	0.8696	0.9374	0.8017	0.9475	0.8913	0.9462	0.8421	0.9606	0.0809	0.0371	0.1343	0.0353
52	0.8595	0.9360	0.7830	0.9401	0.8780	0.9454	0.8171	0.9543	0.0832	0.0385	0.1410	0.0428
53	0.8180	0.9101	0.7260	0.9170	0.8493	0.9296	0.7771	0.9417	0.1185	0.0615	0.1909	0.0649
54	0.8540	0.9346	0.7734	0.9366	0.8786	0.9453	0.8181	0.9529	0.0918	0.0414	0.1593	0.0439
55	0.8622	0.9341	0.7903	0.9426	0.8843	0.9454	0.8318	0.9573	0.0863	0.0419	0.1443	0.0404
56	0.8621	0.9321	0.7921	0.9426	0.8866	0.9446	0.8344	0.9589	0.0856	0.0436	0.1371	0.0399
57	0.8412	0.9309	0.7516	0.9342	0.8662	0.9402	0.7998	0.9502	0.0955	0.0424	0.1637	0.0453
58	0.8787	0.9412	0.8162	0.9507	0.8992	0.9496	0.8562	0.9638	0.0744	0.0363	0.1238	0.0321
60	0.8670	0.9346	0.7993	0.9428	0.8903	0.9461	0.8417	0.9595	0.0822	0.0403	0.1330	0.0409

Continued on next page

	Mean				Median				IQR			
ID	Average	Lumen	Plaque	Vessel	Average	Lumen	Plaque	Vessel	Average	Lumen	Plaque	Vessel
62	0.8517	0.9308	0.7726	0.9310	0.8686	0.9401	0.8059	0.9477	0.0857	0.0401	0.1439	0.0531
63	0.8684	0.9366	0.8002	0.9442	0.8898	0.9462	0.8415	0.9605	0.0828	0.0382	0.1345	0.0369
64	0.8685	0.9374	0.7997	0.9454	0.8882	0.9463	0.8389	0.9585	0.0817	0.0393	0.1378	0.0371
65	0.8724	0.9389	0.8059	0.9470	0.8922	0.9478	0.8461	0.9604	0.0798	0.0382	0.1329	0.0360
66	0.8750	0.9378	0.8123	0.9489	0.8977	0.9482	0.8535	0.9620	0.0768	0.0377	0.1245	0.0340
67	0.8687	0.9358	0.8017	0.9435	0.8919	0.9443	0.8471	0.9603	0.0822	0.0394	0.1348	0.0377
68	0.8474	0.9326	0.7622	0.9316	0.8673	0.9431	0.7971	0.9474	0.0931	0.0399	0.1572	0.0496
69	0.8700	0.9365	0.8036	0.9466	0.8906	0.9461	0.8436	0.9594	0.0781	0.0391	0.1291	0.0345
70	0.8242	0.9105	0.7379	0.9204	0.8496	0.9344	0.7703	0.9399	0.1022	0.0482	0.1645	0.0635
71	0.8781	0.9407	0.8156	0.9505	0.8984	0.9493	0.8550	0.9638	0.0760	0.0361	0.1233	0.0331
72	0.8754	0.9373	0.8135	0.9501	0.8955	0.9459	0.8527	0.9623	0.0770	0.0390	0.1253	0.0346
74	0.0662	0.0020	0.1304	0.2582	0.0640	0.0000	0.1262	0.2526	0.0397	0.0000	0.0815	0.1116
75	0.8672	0.9361	0.7984	0.9457	0.8884	0.9455	0.8380	0.9599	0.0853	0.0385	0.1386	0.0388
76	0.8745	0.9392	0.8098	0.9497	0.8967	0.9491	0.8494	0.9615	0.0788	0.0372	0.1260	0.0336
77	0.8677	0.9337	0.8016	0.9476	0.8917	0.9448	0.8448	0.9620	0.0853	0.0402	0.1382	0.0357
78	0.8778	0.9403	0.8154	0.9515	0.8977	0.9493	0.8524	0.9649	0.0722	0.0362	0.1210	0.0315
79	0.8515	0.9263	0.7767	0.9394	0.8799	0.9397	0.8233	0.9570	0.0931	0.0442	0.1518	0.0413
80	0.8539	0.9326	0.7751	0.9298	0.8715	0.9426	0.8090	0.9472	0.0869	0.0387	0.1481	0.0560
82	0.8809	0.9411	0.8207	0.9511	0.9028	0.9496	0.8604	0.9640	0.0752	0.0364	0.1218	0.0321
83	0.8818	0.9411	0.8225	0.9519	0.9029	0.9504	0.8606	0.9652	0.0738	0.0367	0.1190	0.0317
84	0.8666	0.9384	0.7948	0.9382	0.8913	0.9483	0.8408	0.9550	0.0835	0.0403	0.1389	0.0448
85	0.8825	0.9419	0.8231	0.9527	0.9038	0.9500	0.8618	0.9651	0.0746	0.0362	0.1194	0.0306
86	0.8725	0.9370	0.8079	0.9490	0.8962	0.9474	0.8465	0.9637	0.0803	0.0396	0.1277	0.0363
87	0.8730	0.9376	0.8083	0.9475	0.8953	0.9480	0.8492	0.9614	0.0776	0.0405	0.1241	0.0363
88	0.8677	0.9403	0.7951	0.9380	0.8891	0.9491	0.8374	0.9550	0.0831	0.0380	0.1392	0.0429
89	0.8815	0.9423	0.8208	0.9513	0.9017	0.9516	0.8585	0.9648	0.0749	0.0354	0.1216	0.0318
90	0.8836	0.9428	0.8244	0.9519	0.9034	0.9513	0.8602	0.9661	0.0739	0.0360	0.1204	0.0316
92	0.8903	0.9466	0.8340	0.9542	0.9101	0.9558	0.8687	0.9672	0.0709	0.0344	0.1125	0.0293
93	0.8712	0.9387	0.8038	0.9448	0.8901	0.9475	0.8383	0.9576	0.0754	0.0364	0.1239	0.0362
94	0.5178	0.7252	0.3104	0.4795	0.5114	0.7634	0.2708	0.4556	0.2228	0.3178	0.2411	0.2449

A.2.3 Hausdorff Distance

	Mean			Median			IQR		
ID	Lumen	Plaque	Vessel	Lumen	Plaque	Vessel	Lumen	Plaque	Vessel
1	0.5415	1.0675	0.9396	0.2930	0.4643	0.3706	0.2428	0.7301	0.7081
2	0.5286	0.9774	0.8656	0.2657	0.4152	0.3411	0.2350	0.7419	0.7031
3	0.6197	1.2488	1.1712	0.2734	0.4883	0.4301	0.2607	1.0735	1.0693
4	0.6099	1.2504	1.1038	0.3051	0.5555	0.4883	0.3146	0.9532	0.8703
5	0.5541	0.9958	0.9286	0.2547	0.4367	0.3731	0.2536	0.9215	0.9239
6	0.4783	0.8096	0.7168	0.2657	0.4225	0.3585	0.2183	0.5511	0.5434
7	0.7984	1.5412	1.4270	0.2884	0.5363	0.4597	0.2678	1.7525	1.5562
8	0.5441	1.0702	0.9349	0.2720	0.4788	0.4007	0.2536	0.8170	0.8028
9	0.6057	1.1882	1.0878	0.3057	0.5704	0.5241	0.2922	1.0247	0.9671
10	0.6658	1.2634	1.1306	0.3032	0.5363	0.4543	0.3513	1.1445	1.0709
11	0.4710	0.9028	0.8173	0.2539	0.3955	0.3343	0.1930	0.6768	0.6900
12	0.2955	0.5344	0.4765	0.2278	0.3179	0.2620	0.1576	0.3047	0.2938
13	0.4676	0.9569	0.8532	0.2486	0.4106	0.3461	0.2271	0.7179	0.6567
14	0.5008	0.9789	0.8873	0.2471	0.4106	0.3417	0.2148	0.7321	0.7269

Continued on next page

ID	Mean Lumen	Mean Plaque	Mean Vessel	Median Lumen	Median Plaque	Median Vessel	IQR Lumen	IQR Plaque	IQR Vessel
15	0.3959	0.7262	0.6281	0.2539	0.3233	0.2447	0.1948	0.3297	0.2918
16	0.3290	0.5197	0.4661	0.2486	0.3251	0.2547	0.1793	0.3669	0.3662
17	0.3570	0.5400	0.4547	0.2348	0.3251	0.2624	0.1895	0.3233	0.2799
18	0.5934	1.5739	1.4731	0.2981	0.7031	0.6425	0.2756	1.0900	1.0780
19	0.4741	0.8549	0.7861	0.2539	0.4143	0.3564	0.2375	0.7460	0.7393
20	0.3170	0.4404	0.3756	0.2657	0.3320	0.2620	0.1659	0.2691	0.2488
21	0.3311	0.4673	0.3800	0.2734	0.3494	0.2547	0.1758	0.2802	0.2733
22	0.3113	0.4570	0.3934	0.2471	0.3274	0.2620	0.1723	0.2813	0.2666
23	0.3109	0.6972	0.6291	0.2278	0.3176	0.2620	0.1622	0.3210	0.2982
24	0.3044	0.4946	0.4378	0.2416	0.3516	0.2956	0.1731	0.3319	0.3188
25	0.6342	1.0331	0.9352	0.2720	0.4102	0.3326	0.2035	0.6461	0.6652
26	0.3463	0.5224	0.4116	0.2447	0.3125	0.2416	0.1769	0.2793	0.2373
27	0.3004	0.6822	0.6108	0.2352	0.3088	0.2416	0.1731	0.2614	0.2409
28	0.5787	0.9334	0.8471	0.3038	0.4720	0.3950	0.3201	0.7825	0.8131
31	0.3039	0.4602	0.4049	0.2360	0.3320	0.2720	0.1706	0.3400	0.3320
32	0.2821	0.4633	0.4093	0.2076	0.2904	0.2447	0.1403	0.2762	0.2564
33	0.2590	0.3796	0.3263	0.2113	0.2762	0.2210	0.1465	0.2366	0.2241
34	0.2657	0.3723	0.3116	0.2157	0.2657	0.2104	0.1470	0.2272	0.1924
35	0.2671	0.4080	0.3468	0.2184	0.2713	0.2104	0.1562	0.2206	0.1930
36	0.2839	0.7017	0.6417	0.2278	0.3149	0.2539	0.1611	0.2963	0.2620
37	0.2992	0.4309	0.3567	0.2352	0.3149	0.2352	0.1664	0.2671	0.2380
38	6.8533	7.7031	7.5773	7.2972	7.6708	7.5682	1.1054	1.0045	0.9968
39	0.3447	0.5013	0.4148	0.2384	0.3411	0.2884	0.1782	0.3097	0.2845
40	0.3375	0.4335	0.3364	0.2344	0.2930	0.2286	0.1669	0.2389	0.2000
41	0.4323	0.7546	0.6899	0.2447	0.4064	0.3494	0.1951	0.6471	0.6811
42	0.6330	1.0335	0.9595	0.2657	0.4492	0.3772	0.2660	0.8441	0.8530
43	0.2949	0.4334	0.3569	0.2348	0.2930	0.2344	0.1652	0.2354	0.1980
44	0.3063	0.5778	0.5105	0.2348	0.3176	0.2620	0.1640	0.2866	0.2566
45	0.3326	0.5130	0.4289	0.2344	0.3176	0.2569	0.1646	0.2868	0.2583
46	0.2945	0.4015	0.3393	0.2360	0.3125	0.2539	0.1664	0.2444	0.2169
47	0.3201	0.4848	0.3990	0.2344	0.2904	0.2210	0.1594	0.2456	0.2203
48	1.3130	1.9175	1.7180	0.3711	0.5313	0.4419	0.3744	1.2318	0.8486
49	0.3551	0.5702	0.4935	0.2539	0.3906	0.3320	0.1798	0.3582	0.3415
50	0.4430	1.3912	1.3025	0.2490	0.4543	0.4005	0.1771	0.7749	0.7746
51	0.2731	0.3783	0.3184	0.2184	0.2817	0.2286	0.1514	0.2338	0.2123
52	0.2904	0.4498	0.3951	0.2278	0.3221	0.2720	0.1509	0.2974	0.2912
53	0.3552	0.5353	0.4837	0.2657	0.3852	0.3149	0.2174	0.3711	0.3854
54	0.2799	0.5691	0.5261	0.2210	0.3179	0.2720	0.1575	0.2868	0.2829
55	0.2758	0.4100	0.3582	0.2184	0.2975	0.2376	0.1647	0.2681	0.2527
56	0.2971	0.5292	0.4667	0.2227	0.3088	0.2447	0.1676	0.3124	0.3035
57	0.3196	0.4463	0.3708	0.2416	0.3343	0.2762	0.1714	0.2719	0.2691
58	0.2558	0.3743	0.3169	0.2072	0.2657	0.2104	0.1449	0.2288	0.2003
60	0.3273	0.5035	0.4064	0.2157	0.2956	0.2348	0.1720	0.3115	0.2804
62	0.3622	0.8697	0.8029	0.2547	0.4297	0.3516	0.1868	0.5248	0.5445
63	0.3209	0.4970	0.3973	0.2227	0.2975	0.2348	0.1663	0.3041	0.2720
64	0.2633	0.3650	0.3144	0.2184	0.2796	0.2278	0.1562	0.2436	0.2194
65	0.2590	0.3602	0.3110	0.2148	0.2762	0.2278	0.1507	0.2398	0.2203
66	0.2790	0.4902	0.4179	0.2148	0.2796	0.2184	0.1578	0.2623	0.2153
67	0.3019	0.4997	0.4124	0.2184	0.2975	0.2360	0.1575	0.2837	0.2711
68	0.3021	0.4551	0.4032	0.2348	0.3326	0.2796	0.1605	0.3047	0.3030
69	0.2929	0.4323	0.3518	0.2184	0.2817	0.2278	0.1575	0.2529	0.2023
70	0.4070	0.9738	0.9097	0.2762	0.4972	0.4543	0.2109	0.5508	0.5703
71	0.2526	0.3617	0.3083	0.2039	0.2606	0.2076	0.1411	0.2236	0.2050
72	0.2787	0.3979	0.3241	0.2148	0.2713	0.2104	0.1550	0.2220	0.1898

Continued on next page

ID	Mean			Median			IQR		
	Lumen	Plaque	Vessel	Lumen	Plaque	Vessel	Lumen	Plaque	Vessel
74	2.0025	2.6170	2.6124	1.9531	1.9531	1.9531	0.0000	1.3562	1.3449
75	0.3194	0.4520	0.3525	0.2278	0.2975	0.2344	0.1663	0.2872	0.2483
76	0.3038	0.4008	0.3060	0.2148	0.2776	0.2227	0.1624	0.2379	0.2098
77	0.2878	0.3906	0.3176	0.2344	0.2975	0.2227	0.1742	0.2425	0.2150
78	0.2592	0.3646	0.3024	0.2104	0.2628	0.2039	0.1470	0.2290	0.1852
79	0.2966	0.4472	0.3930	0.2384	0.3038	0.2416	0.1664	0.2822	0.2679
80	0.3124	0.6632	0.6156	0.2486	0.4143	0.3516	0.1833	0.4782	0.5001
82	0.2864	0.4053	0.3196	0.2072	0.2657	0.2072	0.1513	0.2264	0.1923
83	0.2532	0.3844	0.3238	0.2011	0.2606	0.2011	0.1487	0.2263	0.1939
84	0.2796	0.4364	0.3780	0.2148	0.3149	0.2657	0.1529	0.3254	0.3140
85	0.2751	0.3692	0.2877	0.2011	0.2547	0.1992	0.1442	0.2221	0.1774
86	0.2746	0.3989	0.3384	0.2148	0.2720	0.2148	0.1575	0.2586	0.2297
87	0.2878	0.4328	0.3575	0.2104	0.2762	0.2184	0.1562	0.2586	0.2328
88	0.2662	0.4794	0.4296	0.2104	0.3149	0.2657	0.1569	0.3324	0.3125
89	0.2569	0.3692	0.3055	0.2011	0.2539	0.2011	0.1442	0.2325	0.1907
90	0.2575	0.3731	0.3057	0.1992	0.2539	0.1992	0.1470	0.2325	0.1836
92	0.2890	0.4516	0.3560	0.1953	0.2547	0.1953	0.1519	0.2561	0.2094
93	0.2734	0.3971	0.3394	0.2184	0.2975	0.2416	0.1578	0.2607	0.2247
94	3.6309	5.8682	5.4057	5.0516	6.3610	5.6352	6.0450	2.2558	2.9985

A.2.4 Area Ratio

ID	Mean			Median			IQR		
	Lumen	Plaque	Vessel	Lumen	Plaque	Vessel	Lumen	Plaque	Vessel
1	1.0565	0.9538	0.9994	1.0686	0.8901	0.9984	0.1462	0.2095	0.1111
2	1.0136	0.9440	0.9716	1.0514	0.8944	0.9878	0.1413	0.1853	0.1160
3	1.0233	0.8808	0.9466	1.0360	0.8805	0.9792	0.1519	0.2409	0.1639
4	0.9492	0.9615	0.9386	0.9860	0.9160	0.9561	0.1413	0.2199	0.1276
5	0.9800	0.8238	0.8975	1.0279	0.8583	0.9676	0.1535	0.2604	0.1868
6	0.9632	0.9551	0.9488	0.9762	0.9220	0.9558	0.1188	0.1983	0.1058
7	1.1540	1.0761	1.0943	1.0800	0.9777	1.0360	0.1883	0.2575	0.1742
8	1.0163	1.0064	1.0000	1.0263	0.9568	1.0016	0.1349	0.2127	0.1241
9	0.9807	1.0778	1.0032	1.0187	0.9978	1.0120	0.1561	0.2657	0.1518
10	0.9705	0.9894	0.9727	1.0349	0.9576	1.0077	0.1564	0.1969	0.1318
11	1.0407	0.8954	0.9661	1.0426	0.8909	0.9868	0.1414	0.1874	0.1306
12	0.9890	1.0311	0.9987	0.9923	1.0013	0.9939	0.1130	0.2117	0.0899
13	0.9686	1.0459	0.9867	0.9942	0.9901	0.9945	0.1228	0.2002	0.1143
14	0.9906	0.9793	0.9730	1.0026	0.9629	0.9891	0.1356	0.1976	0.1177
15	0.9689	1.0361	0.9872	0.9709	1.0163	0.9876	0.1008	0.1719	0.0732
16	0.9204	0.9563	0.9238	0.9338	0.9574	0.9496	0.1080	0.1853	0.0991
17	0.9413	0.9279	0.9320	0.9576	0.9302	0.9498	0.1104	0.1876	0.0963
18	0.9948	1.0447	0.9993	0.9963	0.9895	0.9931	0.0996	0.2240	0.1007
19	0.9547	0.9621	0.9476	0.9886	0.9621	0.9832	0.1483	0.2249	0.1411
20	0.9942	1.0020	0.9827	0.9897	0.9704	0.9820	0.1396	0.1839	0.0818
21	1.0522	1.0482	1.0368	1.0434	0.9867	1.0196	0.1290	0.1860	0.0900
22	1.0155	0.9832	0.9900	1.0145	0.9539	0.9871	0.1240	0.1818	0.0833
23	0.9740	1.1216	1.0293	0.9783	1.0574	1.0105	0.1089	0.2193	0.0982
24	1.0140	1.0904	1.0385	1.0127	1.0474	1.0206	0.1220	0.2565	0.1265
25	1.1845	0.9886	1.0676	1.0385	0.9647	1.0100	0.1821	0.2105	0.1201

Continued on next page

	Mean			Median			IQR		
	Lumen	Plaque	Vessel	Lumen	Plaque	Vessel	Lumen	Plaque	Vessel
ID									
26	0.9667	1.0764	1.0099	0.9700	1.0292	0.9945	0.1089	0.1812	0.0789
27	1.0321	1.0684	1.0325	1.0200	1.0239	1.0193	0.1107	0.2037	0.0830
28	0.9392	0.9190	0.9219	0.9908	0.9360	0.9747	0.1714	0.2335	0.1543
31	1.0004	0.9561	0.9686	1.0009	0.9249	0.9703	0.1151	0.1948	0.0980
32	1.0088	1.1396	1.0551	1.0057	1.0675	1.0296	0.1018	0.1999	0.0796
33	0.9767	1.0852	1.0144	0.9772	1.0317	1.0011	0.0996	0.1756	0.0762
34	1.0260	0.9781	0.9909	1.0186	0.9509	0.9882	0.1033	0.1599	0.0676
35	1.0148	1.0063	0.9980	1.0077	0.9762	0.9920	0.1028	0.1660	0.0667
36	0.9545	1.0768	1.0037	0.9583	1.0301	0.9870	0.0989	0.2137	0.0817
37	0.9394	1.1017	1.0038	0.9445	1.0531	0.9910	0.1055	0.2011	0.0756
38	1.2738	2.2678	1.6127	1.2112	1.8285	1.5057	0.3417	1.4953	0.6713
39	0.9672	0.8928	0.9275	0.9762	0.8899	0.9347	0.1277	0.1995	0.0960
40	1.0064	1.0688	1.0237	1.0025	1.0273	1.0123	0.1080	0.1991	0.0804
41	1.0649	0.8782	0.9600	1.0006	0.8978	0.9629	0.1483	0.2313	0.1361
42	1.0714	0.9326	0.9792	0.9646	0.9434	0.9662	0.1663	0.2601	0.1612
43	1.0158	1.0280	1.0132	1.0114	0.9839	0.9967	0.1027	0.1856	0.0806
44	0.9903	1.0987	1.0299	0.9946	1.0525	1.0210	0.1083	0.1863	0.0844
45	0.9681	1.1098	1.0180	0.9701	1.0505	1.0050	0.1043	0.2281	0.0945
46	0.9947	0.9847	0.9766	0.9911	0.9465	0.9676	0.1087	0.1892	0.0813
47	1.0245	1.0224	1.0146	1.0254	0.9772	1.0032	0.1059	0.1734	0.0695
48	1.2272	1.1970	1.1939	1.1596	1.0617	1.1117	0.1645	0.3473	0.1845
49	1.0181	0.9780	0.9919	1.0121	0.9294	0.9779	0.1355	0.2687	0.1125
50	1.0415	1.2037	1.1032	1.0348	1.0770	1.0494	0.1107	0.3039	0.1284
51	1.0113	1.0490	1.0195	1.0102	0.9959	1.0030	0.1078	0.1933	0.0788
52	1.0202	0.9936	0.9971	1.0136	0.9512	0.9866	0.1107	0.1955	0.0868
53	0.8977	0.9177	0.8955	0.9154	0.9253	0.9200	0.1412	0.2127	0.1262
54	0.9613	0.9481	0.9504	0.9699	0.9421	0.9604	0.1064	0.2061	0.1039
55	0.9663	1.0094	0.9798	0.9757	1.0017	0.9907	0.1155	0.1811	0.0936
56	0.9747	1.0312	0.9949	0.9878	1.0117	1.0033	0.1258	0.1913	0.0948
57	1.0165	0.8750	0.9493	1.0217	0.8909	0.9637	0.1171	0.2027	0.1090
58	0.9993	1.0626	1.0169	0.9999	1.0041	0.9996	0.0989	0.1681	0.0634
60	0.9956	1.1270	1.0469	1.0074	1.0373	1.0177	0.1202	0.2223	0.0826
62	1.0031	1.1622	1.0556	0.9990	1.0741	1.0286	0.1240	0.2695	0.1009
63	1.0090	1.1188	1.0499	1.0106	1.0295	1.0183	0.1048	0.1939	0.0807
64	0.9703	1.0229	0.9843	0.9721	0.9822	0.9765	0.1049	0.1684	0.0778
65	0.9682	1.0531	0.9959	0.9701	1.0032	0.9840	0.1019	0.1699	0.0780
66	0.9654	1.1115	1.0210	0.9677	1.0431	0.9975	0.0938	0.1965	0.0718
67	0.9682	1.1761	1.0490	0.9681	1.0835	1.0137	0.1038	0.2147	0.0851
68	0.9915	0.9307	0.9493	0.9859	0.9042	0.9509	0.1126	0.2136	0.0985
69	0.9828	1.0883	1.0260	0.9880	1.0304	1.0050	0.1069	0.2053	0.0911
70	0.9954	0.9962	0.9760	1.0088	0.9278	0.9785	0.1597	0.2133	0.1277
71	0.9956	1.0539	1.0099	0.9960	0.9965	0.9942	0.0983	0.1633	0.0631
72	0.9854	1.0885	1.0225	0.9874	1.0362	1.0042	0.1027	0.1788	0.0701
74	0.0015	18.7878	7.4093	0.0000	14.7479	6.8181	0.0000	10.7827	3.3127
75	1.0076	1.0418	1.0102	0.9997	0.9887	0.9978	0.1014	0.1657	0.0746
76	0.9888	1.0176	0.9931	0.9899	0.9803	0.9843	0.0962	0.1550	0.0706
77	0.9876	1.0588	1.0062	0.9965	0.9974	0.9954	0.1107	0.1751	0.0732
78	1.0065	1.0458	1.0140	1.0089	0.9834	0.9937	0.1018	0.1584	0.0600
79	0.9495	0.9804	0.9548	0.9588	0.9531	0.9591	0.1212	0.1601	0.0935
80	1.0012	1.2160	1.0794	0.9954	1.0973	1.0432	0.1279	0.3168	0.1154
82	0.9892	1.1035	1.0283	0.9898	1.0337	1.0068	0.0943	0.1685	0.0617
83	0.9688	1.0784	1.0070	0.9726	1.0130	0.9895	0.0950	0.1571	0.0601
84	0.9871	1.1820	1.0479	0.9871	1.0762	1.0223	0.0967	0.2358	0.0896
85	0.9854	1.0647	1.0084	0.9842	1.0017	0.9904	0.0928	0.1565	0.0576

Continued on next page

ID	Mean			Median			IQR		
	Lumen	Plaque	Vessel	Lumen	Plaque	Vessel	Lumen	Plaque	Vessel
86	0.9820	1.0016	0.9830	0.9873	0.9695	0.9819	0.1053	0.1458	0.0707
87	0.9801	1.0721	1.0116	0.9872	1.0066	0.9933	0.1047	0.1809	0.0696
88	1.0051	1.1580	1.0518	1.0010	1.0364	1.0123	0.0986	0.2435	0.0972
89	0.9839	1.0540	1.0042	0.9845	0.9918	0.9851	0.0945	0.1507	0.0571
90	0.9823	1.0714	1.0110	0.9847	1.0058	0.9900	0.0886	0.1529	0.0559
92	0.9870	1.0831	1.0197	0.9849	1.0198	0.9983	0.0699	0.1428	0.0529
93	0.9887	1.0692	1.0065	0.9853	1.0160	0.9963	0.0868	0.1894	0.0737
94	1.6014	7.5991	3.7600	1.2300	4.9819	3.0769	0.8889	6.0595	2.6114

A.2.5 Plaque Burden

ID	Mean			Median			IQR		
	Prediction	Ground Truth	Ratio	Prediction	Ground Truth	Ratio	Prediction	Ground Truth	Ratio
1	0.4243	0.4542	0.9428	0.4213	0.4540	0.9188	0.1862	0.1782	0.1569
2	0.4363	0.4542	0.9705	0.4326	0.4540	0.9357	0.1904	0.1782	0.1527
3	0.4150	0.4542	0.9249	0.4083	0.4540	0.9148	0.1779	0.1782	0.1462
4	0.4521	0.4542	1.0182	0.4486	0.4540	0.9765	0.1694	0.1782	0.1556
5	0.4130	0.4542	0.9218	0.4070	0.4540	0.9070	0.1710	0.1782	0.1501
6	0.4454	0.4542	0.9954	0.4442	0.4540	0.9783	0.1754	0.1782	0.1410
7	0.4376	0.4542	0.9830	0.4369	0.4540	0.9571	0.1660	0.1782	0.1649
8	0.4456	0.4542	0.9954	0.4469	0.4540	0.9721	0.1644	0.1782	0.1395
9	0.4677	0.4542	1.0582	0.4642	0.4540	1.0078	0.1682	0.1782	0.1669
10	0.4631	0.4542	1.0283	0.4572	0.4540	0.9803	0.2034	0.1782	0.1606
11	0.4168	0.4542	0.9244	0.4116	0.4540	0.9180	0.1724	0.1782	0.1320
12	0.4590	0.4542	1.0249	0.4613	0.4540	1.0069	0.1863	0.1782	0.1444
13	0.4676	0.4542	1.0534	0.4703	0.4540	1.0082	0.1706	0.1782	0.1491
14	0.4483	0.4542	1.0030	0.4501	0.4540	0.9830	0.1652	0.1782	0.1345
15	0.4651	0.4542	1.0424	0.4703	0.4540	1.0219	0.1734	0.1782	0.1276
16	0.4568	0.4542	1.0264	0.4582	0.4540	1.0074	0.1702	0.1782	0.1271
17	0.4459	0.4542	0.9865	0.4496	0.4540	0.9844	0.1977	0.1782	0.1386
18	0.4581	0.4542	1.0324	0.4605	0.4540	0.9976	0.1509	0.1782	0.1515
19	0.4504	0.4542	1.0059	0.4531	0.4540	0.9924	0.1725	0.1782	0.1398
20	0.4514	0.4542	1.0155	0.4439	0.4540	0.9909	0.1745	0.1782	0.1383
21	0.4480	0.4542	1.0028	0.4429	0.4540	0.9745	0.1700	0.1782	0.1352
22	0.4421	0.4542	0.9857	0.4398	0.4540	0.9735	0.1741	0.1782	0.1392
23	0.4814	0.4542	1.0798	0.4843	0.4540	1.0463	0.1824	0.1782	0.1478
24	0.4657	0.4542	1.0402	0.4721	0.4540	1.0200	0.1843	0.1782	0.1545
25	0.4190	0.4542	0.9423	0.4124	0.4540	0.9478	0.1814	0.1782	0.1644
26	0.4755	0.4542	1.0597	0.4755	0.4540	1.0329	0.1802	0.1782	0.1350
27	0.4571	0.4542	1.0265	0.4615	0.4540	1.0049	0.1665	0.1782	0.1396
28	0.4484	0.4542	0.9985	0.4468	0.4540	0.9815	0.1758	0.1782	0.1549
31	0.4360	0.4542	0.9737	0.4347	0.4540	0.9547	0.1824	0.1782	0.1358
32	0.4768	0.4542	1.0703	0.4796	0.4540	1.0372	0.1738	0.1782	0.1335
33	0.4738	0.4542	1.0618	0.4746	0.4540	1.0316	0.1706	0.1782	0.1240
34	0.4390	0.4542	0.9822	0.4409	0.4540	0.9655	0.1681	0.1782	0.1206
35	0.4483	0.4542	1.0039	0.4499	0.4540	0.9850	0.1707	0.1782	0.1243
36	0.4774	0.4542	1.0647	0.4785	0.4540	1.0407	0.1875	0.1782	0.1455
37	0.4861	0.4542	1.0888	0.4872	0.4540	1.0569	0.1868	0.1782	0.1527
38	0.5615	0.4542	1.3290	0.5722	0.4540	1.2244	0.1406	0.1782	0.4946

Continued on next page

ID	Mean Prediction	Ground Truth	Ratio	Median Prediction	Ground Truth	Ratio	IQR Prediction	Ground Truth	Ratio
39	0.4320	0.4542	0.9568	0.4363	0.4540	0.9490	0.1921	0.1782	0.1627
40	0.4638	0.4542	1.0370	0.4692	0.4540	1.0153	0.1808	0.1782	0.1396
41	0.4089	0.4542	0.9175	0.4031	0.4540	0.9329	0.1794	0.1782	0.1647
42	0.4264	0.4542	0.9646	0.4237	0.4540	0.9728	0.1765	0.1782	0.1791
43	0.4520	0.4542	1.0070	0.4512	0.4540	0.9883	0.1830	0.1782	0.1312
44	0.4743	0.4542	1.0609	0.4724	0.4540	1.0303	0.1793	0.1782	0.1338
45	0.4793	0.4542	1.0792	0.4806	0.4540	1.0409	0.1722	0.1782	0.1602
46	0.4472	0.4542	1.0023	0.4507	0.4540	0.9770	0.1734	0.1782	0.1355
47	0.4490	0.4542	1.0009	0.4484	0.4540	0.9788	0.1784	0.1782	0.1306
48	0.4430	0.4542	0.9896	0.4432	0.4540	0.9597	0.1899	0.1782	0.1729
49	0.4380	0.4542	0.9729	0.4375	0.4540	0.9542	0.1996	0.1782	0.1845
50	0.4782	0.4542	1.0721	0.4748	0.4540	1.0326	0.1839	0.1782	0.1659
51	0.4574	0.4542	1.0207	0.4570	0.4540	0.9948	0.1842	0.1782	0.1346
52	0.4425	0.4542	0.9881	0.4407	0.4540	0.9677	0.1795	0.1782	0.1388
53	0.4551	0.4546	1.0190	0.4599	0.4547	1.0022	0.1691	0.1790	0.1523
54	0.4468	0.4546	0.9892	0.4475	0.4547	0.9840	0.1832	0.1790	0.1432
55	0.4616	0.4546	1.0258	0.4634	0.4547	1.0130	0.1857	0.1790	0.1237
56	0.4643	0.4546	1.0310	0.4662	0.4547	1.0153	0.1872	0.1790	0.1364
57	0.4165	0.4546	0.9130	0.4238	0.4547	0.9236	0.1833	0.1790	0.1356
58	0.4632	0.4542	1.0369	0.4630	0.4540	1.0069	0.1715	0.1782	0.1232
60	0.4762	0.4546	1.0643	0.4705	0.4547	1.0237	0.1929	0.1790	0.1544
62	0.4800	0.4542	1.0860	0.4797	0.4540	1.0393	0.1655	0.1782	0.1761
63	0.4717	0.4546	1.0533	0.4722	0.4547	1.0160	0.1829	0.1790	0.1354
64	0.4619	0.4542	1.0315	0.4631	0.4540	1.0088	0.1726	0.1782	0.1217
65	0.4687	0.4542	1.0490	0.4679	0.4540	1.0210	0.1704	0.1782	0.1225
66	0.4816	0.4542	1.0799	0.4796	0.4540	1.0452	0.1793	0.1782	0.1377
67	0.4919	0.4542	1.1082	0.4908	0.4540	1.0607	0.1720	0.1782	0.1519
68	0.4334	0.4542	0.9724	0.4338	0.4540	0.9526	0.1667	0.1782	0.1421
69	0.4733	0.4542	1.0532	0.4718	0.4540	1.0284	0.1907	0.1782	0.1424
70	0.4504	0.4542	1.0181	0.4426	0.4540	0.9646	0.1686	0.1782	0.1712
71	0.4620	0.4542	1.0356	0.4593	0.4540	1.0034	0.1741	0.1782	0.1242
72	0.4730	0.4542	1.0590	0.4688	0.4540	1.0328	0.1802	0.1782	0.1325
74	0.9999	0.4546	2.4324	1.0000	0.4547	2.1994	0.0000	0.1790	0.8937
75	0.4565	0.4546	1.0232	0.4528	0.4547	0.9948	0.1716	0.1790	0.1141
76	0.4574	0.4546	1.0199	0.4530	0.4547	0.9960	0.1876	0.1790	0.1159
77	0.4650	0.4546	1.0449	0.4639	0.4547	1.0058	0.1746	0.1790	0.1361
78	0.4584	0.4546	1.0237	0.4589	0.4547	0.9910	0.1774	0.1790	0.1244
79	0.4581	0.4546	1.0223	0.4582	0.4547	0.9978	0.1724	0.1790	0.1234
80	0.4906	0.4542	1.1091	0.4936	0.4540	1.0582	0.1792	0.1782	0.1911
82	0.4745	0.4546	1.0650	0.4738	0.4547	1.0266	0.1784	0.1790	0.1249
83	0.4740	0.4546	1.0627	0.4723	0.4547	1.0252	0.1754	0.1790	0.1188
84	0.4867	0.4542	1.1109	0.4827	0.4540	1.0518	0.1540	0.1782	0.1537
85	0.4673	0.4546	1.0482	0.4627	0.4547	1.0127	0.1767	0.1790	0.1168
86	0.4552	0.4546	1.0133	0.4516	0.4547	0.9933	0.1742	0.1790	0.1125
87	0.4700	0.4546	1.0519	0.4643	0.4547	1.0157	0.1819	0.1790	0.1308
88	0.4766	0.4542	1.0812	0.4740	0.4540	1.0236	0.1609	0.1782	0.1604
89	0.4650	0.4546	1.0408	0.4633	0.4547	1.0069	0.1756	0.1790	0.1175
90	0.4691	0.4546	1.0511	0.4665	0.4547	1.0162	0.1754	0.1790	0.1162
92	0.4702	0.4542	1.0520	0.4689	0.4540	1.0201	0.1697	0.1782	0.1105
93	0.4658	0.4542	1.0518	0.4665	0.4540	1.0174	0.1557	0.1782	0.1343
94	0.7418	0.4542	1.8063	0.7668	0.4540	1.6087	0.1121	0.1782	0.7178