Benchmarking of Vascular Segmentation Methods

oscar.morand
(supervisor: Elodie Puybareau)

June 2024

Vascular segmentation holds significant importance in medical image analysis, as the thorough examination of blood vessels is essential for diagnosis, monitoring and treatment planning, and segmentation techniques can support clinicians in these tasks. During this project, we will perform an honest review of existing methods, by benchmarking machine learning and deep learning algorithms, identified either as highly promising or currently regarded as the gold standard. The criteria used to evaluate the selected models are metrics that assess both the accuracy and efficiency of our methods. However, most accuracy metrics are unsatisfactory and have limitations for fine objects such as blood vessels. The development of a new metric, designed to evaluate the segmentation performance of blood vessels of different sizes, will be presented in a second part.

La segmentation vasculaire revêt une importance significative dans l'analyse des images médicales, car l'examen approfondi des vaisseaux sanguins est essentiel pour le diagnostic, la surveillance et la planification des traitements. Les techniques de segmentation peuvent soutenir les cliniciens dans ces tâches. Au cours de ce projet, nous effectuerons une revue honnête des méthodes existantes en comparant les algorithmes d'apprentissage automatique et d'apprentissage profond, identifiés soit comme très prometteurs, soit actuellement considérés comme la référence. Les critères utilisés pour évaluer les modèles sélectionnés sont des métriques qui évaluent à la fois la précision et l'efficacité de nos méthodes. Cependant, la plupart des métriques de précision sont insatisfaisantes et présentent des limitations pour les objets fins tels que les vaisseaux sanguins. Le développement d'une nouvelle métrique, conçue pour évaluer la performance de segmentation des vaisseaux sanguins de différentes tailles, sera présenté dans une deuxième partie.

Keywords
Vascular segmentation, Benchmarking, Deep Learning, Loss function, Metric

Laboratoire de Recherche de l'EPITA
14-16, rue Voltaire
94270 Le Kremlin-Bicêtre CEDEX
France

Copying this document

Contents

Chapter 1

Introduction

Before we delve into our discussion, it is crucial to clarify a few terms. Benchmarking is a performance evaluation of a computer system aimed at comparing its capabilities with others.

Segmentation, on the other hand, is an image processing technique used to automatically partition an image into regions where pixels share similar characteristics belonging to the same object class. This technique is fundamental in various applications, particularly in medical imaging and computer vision tasks, enabling precise analysis and interpretation of complex visual data.

In our case, we are talking about vascular segmentation, which means the segmentation of blood vessels. This is very important because examining blood vessels is essential for diagnosis, monitoring, and treatment planning. Segmentation techniques can assist clinicians in these tasks, but it can be beneficial to compare these techniques to determine which ones are suitable for which situations, on which parts of the body, and with which medical imaging technology.

This report focuses solely on binary semantic segmentation, not instance or multi-class segmentation 1.1.

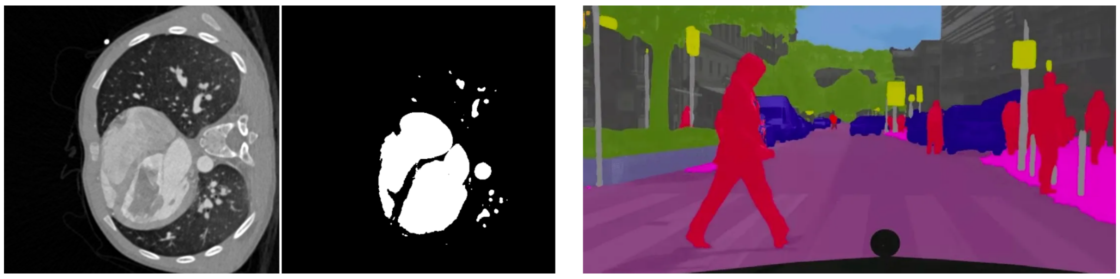


Figure 1.1: Example of binary segmentation (left) Barrowclough et al. (2021), and multi-class segmentation (right) Hamza (2023) illustrating accurate classification of each pixel into its respective class.

In semantic segmentation tasks, each pixel in an image is classified into a category corresponding to a specific object class. Assuming there is only one type of object to predict, this becomes a binary semantic task. The following mapping can be assigned:

- 0 for the background class.
- 1 for the object of interest.

Most of the images that will be used, particularly for initial training and metric testing, will be 2D data, as they are easier to handle and require less computational resources. Specifically, we will use fundus images in our subsequent studies. However, since the goal is to include images from CT scans or MRIs, it may be beneficial to extend the benchmarking to all types of data, including 3D images, as they represent a significant portion of medical images today. This will allow us to cover a wide range of modalities and vascular networks from various organs such as lungs, kidneys, brain, retina, and heart.

To ensure a comprehensive and unbiased benchmarking process, we will utilize data from several public databases encompassing diverse modalities and anatomical regions. One dataset that will be heavily employed for training our model and subsequently testing new metrics is the DRIVE dataset Staal et al. (2004). DRIVE consists of retinal blood vessel images, featuring 20 annotated images in the training set. These 2D color images have a resolution of 540x540 pixels, providing detailed visual information for segmentation tasks (see Figure 1.2). The dataset is well-regarded for its high-quality annotations and manageable complexity, ensuring consistency and reliability across the images used in our study. This consistency makes it straightforward for our model to learn effectively from the dataset.

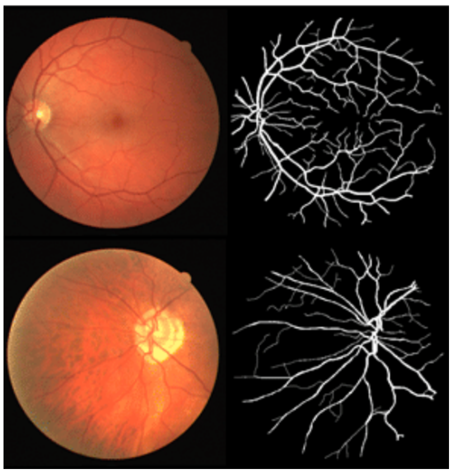


Figure 1.2: Example images from the DRIVE dataset, introduced by Staal et al. (2004): original images (left) and corresponding manually annotated binary segmentations (right), used as ground truth.

Chapter 2

The benchmarking, first part

This section begins in February at the start of the semester and continues until mid-April, when we paused our work on this topic to explore the development of a new metric, which will be discussed in the second part.

2.1 State of the art

To discuss the state of the art, we begin with the current reference in segmentation models, the well-known U-Net Ronneberger et al. (2015), characterized by its U-shaped architecture 2.1. This model comprises an encoding phase that reduces the image's spatial resolution while capturing features, followed by a decoding phase that restores resolution and produces segmentation maps.

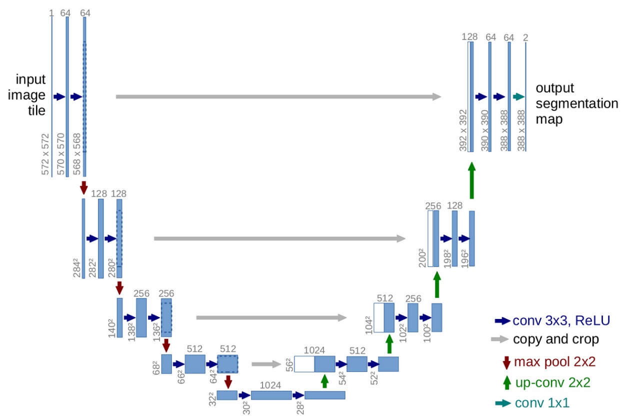


Figure 2.1: The U-Net model architecture: the contracting path (or encoder) on the left, then the Bottleneck, and finally the expansive path (or decoder).

For volumetric images, a notable variant is U-Net 3D. Additionally, there are several derivatives of U-Net such as V-Net Milletari et al. (2016), U-Net++ Zhou et al. (2018), and more recently nnU-Net Isensee et al. (2021), recognized for its flexibility and impressive performance in vascular segmentation. All these models are fully convolutional networks (FCNs).

In parallel, Generative Adversarial Networks (GANs) Luc et al. (2016), typically used for image synthesis, have been adapted for image segmentation. In this approach, a generator produces segmented images, while a discriminator distinguishes between predicted segmentations and ground truth. Vision transformer-based models Dosovitskiy et al. (2020), less common than FCNs but still present, could also be considered for benchmarking once the benchmarking pipeline is ready.

Regarding benchmarking, there are few recent comprehensive studies on vascular segmentation methods. One of the most recent, "Blood vessel segmentation algorithms—review of methods, datasets and evaluation metrics" Moccia et al. (2018), laid the groundwork, particularly in terms of evaluation metrics. However, due to the rapid evolution in this field, a few years can make a significant difference in the models used, underscoring the need for a more recent study.

While not strictly benchmarks, challenges like the PARSE challenge Luo et al. (2023) are valuable as they provide methodologies for evaluating and ranking models. This challenge has been particularly useful for its evaluation methodology, metrics used, and insights into models employed by top-performing teams.

2.2 Setting up the benchmarking, first model training

To initiate this exploration and gain initial experience with deep learning, the first task involved training a network on a "simple" dataset using U-Net applied to the DRIVE dataset.

This foundational step allowed for the grasp of fundamental concepts and workflows in deep learning. Once satisfactory training results were achieved, the focus shifted to adapting the benchmarking pipeline.

This adaptation aimed to ensure a fair and comprehensive evaluation of different models across various datasets. Prioritizing model convergence before initiating benchmarking aimed to establish a solid groundwork for precise and informative evaluations.

At this stage, the objective is obviously not to outperform the current state-of-the-art, but rather to meticulously analyze how hyperparameters and loss functions impact the model's performance.

2.2.1 Work Accomplished

Metrics

The first step to compute a huge part of metrics used in segmentation tasks is to compute the values of the confusion matrix such as TP (true positives), TN (true negatives), FP (false positives), and FN (false negatives).

These values are essential for calculating overlaps between two binarized images: the ground truth image and the predicted image 2.2. TP refers to pixels where both the ground truth and the prediction agree that the pixel belongs to the object being segmented, in this case, a blood vessel. TN represents correctly identified background areas, while FP and FN denote disagreements between prediction and ground truth.

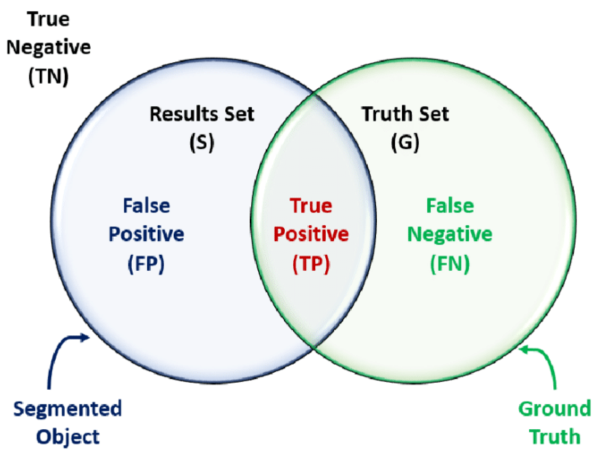


Figure 2.2: Diagram from Mostapha (2014) depicting the area-based computation of segmentation confusion matrix, including true positives (TP), false positives (FP), true negatives (TN), and false negatives (FN).

These metrics are instrumental in assessing the segmentation performance of our models. Here is the list of metrics used:

- **Accuracy:** This metric assigns a score of 1 to pixels that are correctly predicted and 0 otherwise. It is straightforward but not suitable for imbalanced datasets like vascular segmentation, where the background predominates. The formula is:

$$\text{Accuracy} = \frac{TP + TN}{TP + TN + FP + FN}$$

- **Precision, Recall and Specificity:**.

$$\text{Precision / Positive predictive value (PPV)} = \frac{\text{TP}}{\text{TP} + \text{FP}}$$

$$\text{Recall / Sensitivity} = \frac{TP}{TP + FN}$$

$$\text{Specificity} = \frac{TN}{TN + FP}$$

- **Dice Score / F1 Score:** The F1 score balances precision and recall, providing a harmonious measure of model performance. It is calculated as:

$$F1 = \frac{2 \times \text{TP}}{2 \times \text{TP} + \text{FP} + \text{FN}} = \frac{2 \times \text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}$$

- **False Positive Rate (FPR), Negative Predictive Value (NPV), Matthews Correlation Coefficient (MCC):** These metrics provide additional insights into the model's performance, each with their respective formulas:

$$\text{False Positive Rate (FPR) / Fallout} = 1 - \text{Specificity} = \frac{FP}{TN + FP}$$

$$\text{Negative Predictive Value (NPV)} = \frac{TN}{TN + FN}$$

$$\text{MCC} = \frac{TP \times TN - FP \times FN}{\sqrt{(TP + FP)(TP + FN)(TN + FP)(TN + FN)}}$$

These metrics are crucial for evaluating and refining our segmentation models, ensuring robust and reliable performance assessments.

Their values range between 0 and 1, with 0 indicating the worst possible score and 1 the optimal value to achieve. There are two exceptions: the Matthews Correlation Coefficient (MCC), which ranges from -1 to 1, and the false positive rate, which is better when closer to 0.

Cost Function

A classic cost function for segmentation tasks is the Binary Cross entropy (BCE):

$$\text{BCE} = -\frac{1}{N}\sum_{i=0}^{N} y_i \cdot \log(\hat{y}_i) + (1 - y_i) \cdot \log(1 - \hat{y}_i)$$

Here we had to modify the BCE to address class imbalance between background and blood vessels, because in our case the background pixels are much more prevalent.
The goal is to get a more precise vessel segmentation, achieved by assigning weights inversely proportional to the frequency of each class. For instance, if an image contains 1000 background pixels and 100 pixels of the object of interest, the weights assigned would be 1 for background and 10 for the vessel class (calculated as 100 / 10).

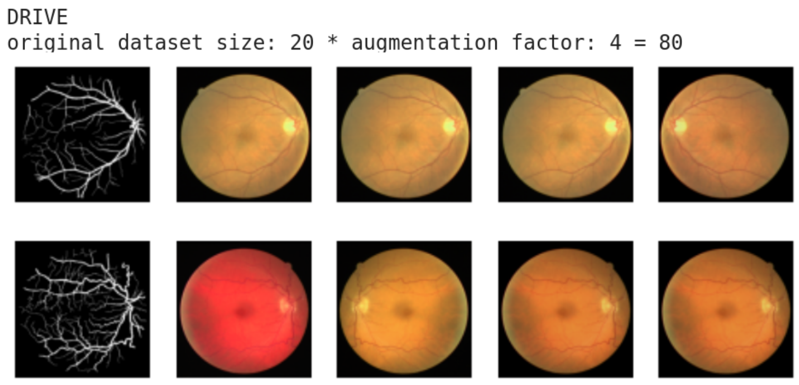
Data Preprocessing

Minimal preprocessing is applied to avoid biasing the data, including:

- Conversion to tensors,
- Normalization using z-score normalization, which centers the mean at 0 and standard deviation at 1, accelerating convergence and enhancing model stability,
- Resizing to 512x512 pixels.

Data augmentation techniques are crucial due to the limited dataset of only 20 annotated images. By applying an augmentation factor, such as 10, the dataset size is artificially increased to 200 annotated images. This augmentation is implemented by overriding the __getitem__ method of the Dataset class in Pytorch.

Augmentation operations include horizontal flipping and "color jitter," which adjusts brightness, contrast, saturation, and hue (2.3). These techniques contribute to models that generalize better and mitigate overfitting risks associated with training on a small dataset.



```
DRIVE
original dataset size: 20 * augmentation factor: 4 = 80
```

Figure 2.3: Example of data augmentation with an augmentation factor of 4, each line refers to one same original image, with the corresponding original ground truth (left)

Tools for the Benchmarking Program

To facilitate the benchmarking process, a command-line program was developed to perform benchmarking on multiple models and datasets. Although it has not yet been utilized, this program provides a solid foundation that will likely prove useful as the project progresses. The program includes the capability to train one or more models on specific datasets and then test these models on different datasets, enabling a robust comparison of their performance.

Compatibility between different models and data types has been carefully considered, addressing potential issues such as applying a simple U-Net to 3D data. The program also handles various errors and interactions with the user. Classes were created to be as general as possible, ensuring they work seamlessly with the command-line program. This design allows for adding new models or datasets with minimal effort.

Significant effort was put into incorporating extensive data visualization capabilities. This includes plots, graphs, and statistics generated at the end of each epoch, training session, and testing phase. Consequently, the benchmarking program offers more than just a table of metrics; it provides users with comprehensive tools to analyze results, make informed choices, and effectively compare models on different data types or modalities.

Progress on this program paused when transitioning to Kaggle and focusing on creating and training the first model. Nonetheless, the groundwork laid will be immensely beneficial for future phases of the project.

2.2.2 Project Realization Story

Initially, there was a need to familiarize myself with PyTorch and tensors. To gain a better understanding, I decided to manually handle most aspects of the project, including the development of my model. Even though this approach took a considerable amount of time, I am confident that it was a good decision because I gained a lot of valuable knowledge. It certainly wasn't lost time.

A significant challenge quickly emerged, as I was working locally on my computer, which lacked the computational power necessary for training this type of model. Although the possibility of accessing a lab computer was discussed, it never materialized.

While waiting for access to a more powerful machine, I switched to using Kaggle, which completely disrupted the structure of my program. Having never completed an entire project in a notebook before, I simply copied and pasted my code, including all its classes, without recognizing the potential issues. This approach limited my ability to leverage the flexibility that notebooks offer. I failed to distinguish between the exploratory data analysis and rapid training phases that Kaggle facilitates, and creating a deliverable that was both usable and maintainable in the context of benchmarking. This has led to other complications, which I will discuss below.

However, once I had the necessary computational power, I was able to start my initial training sessions. An unexpected problem arose in our effort to genuinely increase the size of our dataset through data augmentation and not just create a new variation of our dataset randomly. I struggled significantly to ensure that the same random transformations were applied to both the images and their ground truth annotations. This involved dealing with numerous different seeds in NumPy, Python, and PyTorch. Even though this synchronization problem may seem straightforward, it was a real challenge.

I also faced a significant setback due to a data issue that I had overlooked. Being entirely new to this, I didn't necessarily have the right instincts and wasn't asking the right questions initially. With Elodie's advice, I was able to identify the problem. This caused a slight delay in the benchmarking process, but it allowed me the opportunity to test potential solutions in an attempt to improve my model's convergence. We deliberated extensively, particularly on the cost function: whether changing it would improve outcomes, or if modifying it (such as adjusting weights) would be more effective. Ultimately, once the project was unblocked, all other parameters had been so thoroughly reassessed that everything worked perfectly.

2.3 Training Results

2.3.1 Training Configuration

After resolving the project blockage, we conducted the initial substantial training of our U-Net model. Presented here is the latest result: using a data augmentation factor of 4 (resulting in 80 images in the dataset), and employing a 5-fold cross-validation (with 64 images in the training dataset and 16 in the validation dataset), weighted binary cross-entropy loss with a learning rate of 1e-4, 100 epochs, and a batch size of 2. The obtained results will be discussed in the subsequent section.

2.3.2 Graphical Interpretation of Results

First, the loss evolution curve (2.4) shows a convergence rate that is reasonably good, neither too fast nor too slow, suggesting that the learning rate is well-suited for this configuration. There is also only a small gap between the training loss and validation loss, indicating that the model generalizes well on the data with minimal overfitting.
This training was arbitrarily stopped at 100 epochs; however, there is no impediment to running more epochs. As long as the test loss curve continues to decrease, the model is still learning without overfitting.

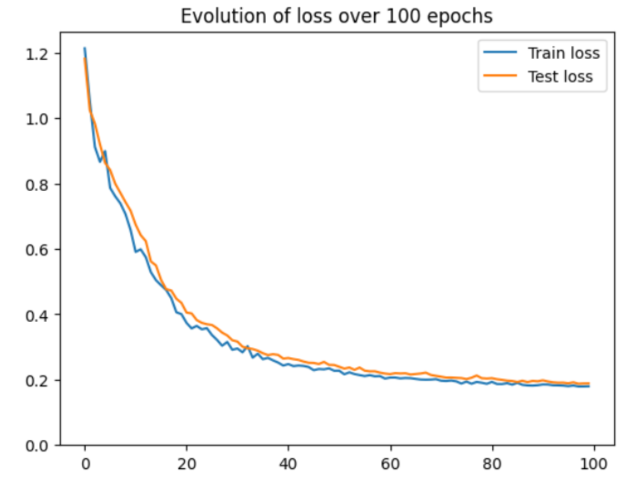


Figure 2.4: Evolution of the loss function (BCE) for one fold, over 100 epochs.

The evolution of various metrics (2.5) reinforces this notion; indeed, none of them exhibit worse results than the others, suggesting a successful training process that does not sacrifice one aspect of segmentation for another. This was a problem we encountered at length, where specificity overshadowed sensitivity, resulting in a nearly empty prediction mask towards the end.

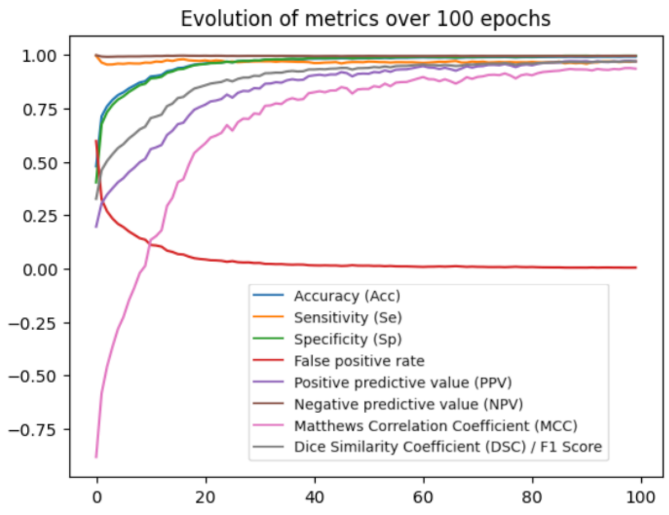


Figure 2.5: Evolution of the different metrics for one fold, over 100 epochs

We can ensure the robustness of our model by examining the results across different folds (2.6): we observe a relatively low variance among the folds, indicating that the model generalizes well to the test data, regardless of the training dataset used.

Visually, we can confirm this because the predictions closely approximate the ground truth in all cases (2.7), and the error map does not reveal any particular issues: there are no holes,

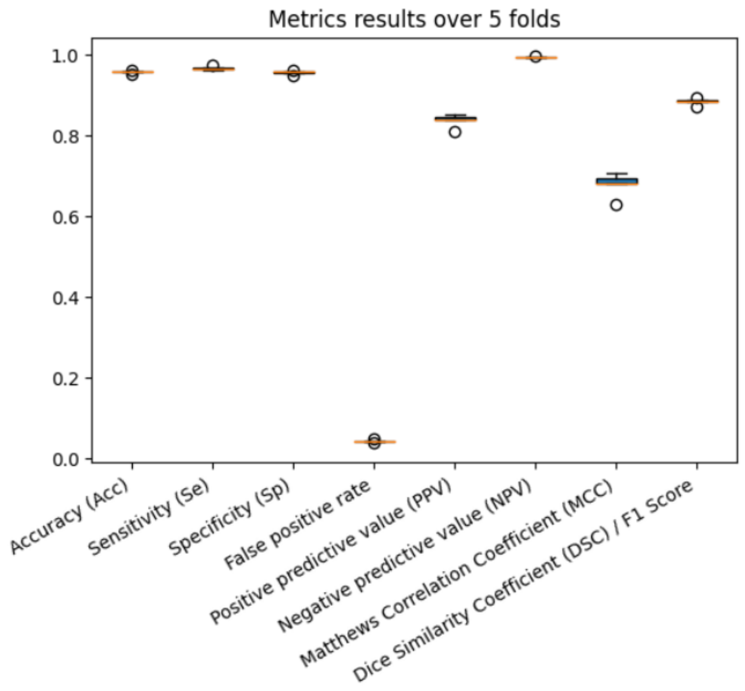


Figure 2.6: Statistical distribution of metric results across 5 folds, with the central bar centered around the median and the points indicating the minimum and maximum values.

discontinuities, omitted areas, under-segmented or over-segmented regions. The only potential issue lies with the borders (2.8), which are not always perfectly predicted by the model.

2.4 Reflection and Reevaluation of the Topic

All inquiries regarding these cost functions were initially discussed at the outset of the project setup (notably in the first abstract and previous lightning talks). Consequently, this issue was revisited, and it was decided to explore this avenue for a period. We recognized that the data we were working with were quite unique, due to the complex and diverse structures of blood vessels, and current metrics/cost functions did not necessarily accurately reflect the real performance of a model. Through literature review, we confirmed our suspicions upon realizing the extensive use of U-Net or its derivatives, prompting us to consider the value in studying the impact of different cost functions on the same model, especially in the context of vascular segmentation.

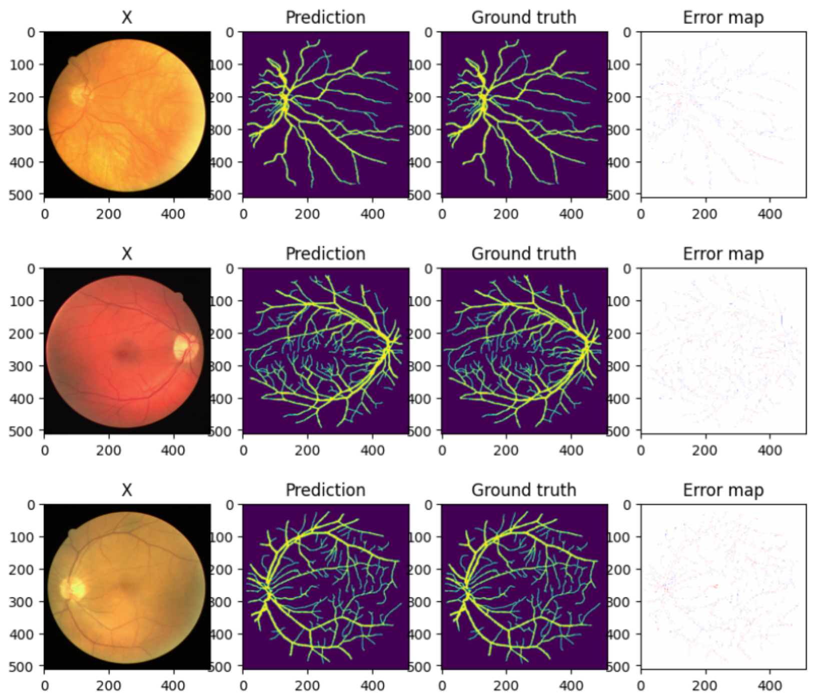


Figure 2.7: Inference results after training the model for 100 epochs, original image (left), followed by the model prediction, ground truth mask, and error map (right).

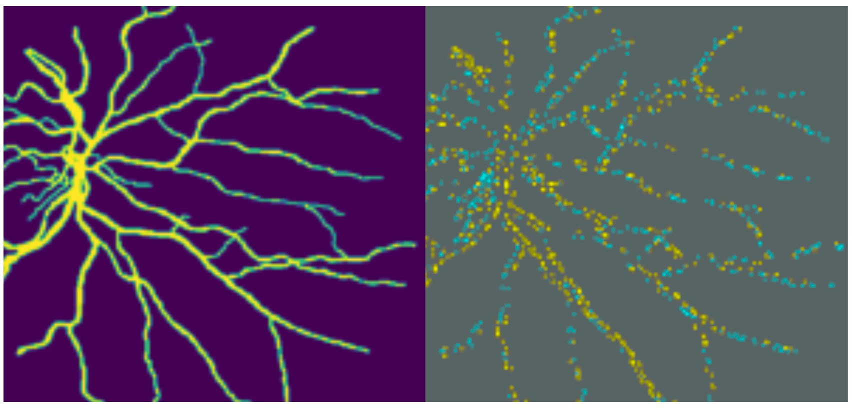


Figure 2.8: Zoom on the error map of a prediction, where yellow indicates false positives and blue indicates false negatives (with contrast enhancement to highlight errors).

Chapter 3

Development of a New Metric for Vascular Segmentation

In this chapter, we engage in exploratory research without a clear initial direction. This lack of clarity is reflected in the numerous tests conducted, which did not yield conclusive results as new ideas continually replaced previous ones.

3.1 Challenges in Segmentation of Thin Blood Vessels

Firstly, there is the issue of the partial volume effect (3.1), an acquisition artifact related to imaging methods that appears when the acquisition resolution is low relative to the structures being studied.

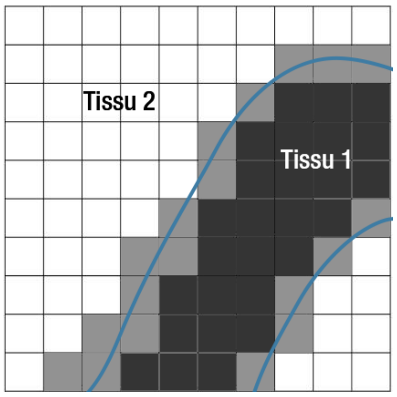


Figure 3.1: Example of the partial volume effect. The gray pixels correspond to partial volume artifacts and have an intermediate value between tissues 1 and 2.

In such cases, there are areas of uncertainty at the boundary between two tissues, where the pixels/voxels are a mix of the values of the surrounding tissues. This uncertainty is relevant for creating ground truth data, implying that at the edges of blood vessels, the ground truth will reflect the annotator's choice, and there is no single, definitive ground truth in these areas.

In the example below, taken from the STARE dataset: a dataset of fundus images, with two ground truths made by two different people. If we take an image with its associated ground truths, by zooming in on a small blood vessel in the two ground truths, we notice that the contours are not exactly the same, resulting in a Dice score of 0.4 between the two ground truths! While it is clear that the two vessels are the same, we can thus understand that the ground truths differ by the interpretation of the two people making them.

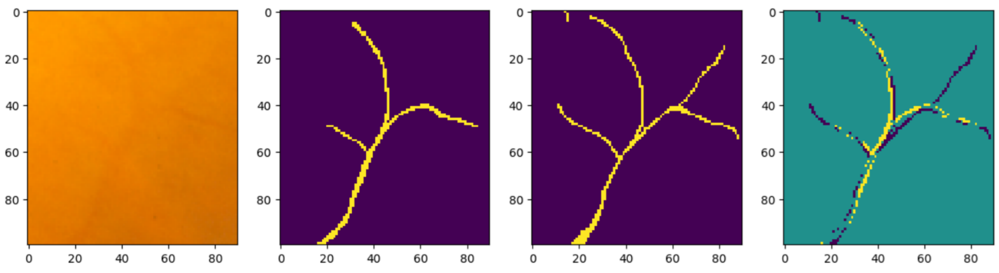


Figure 3.2: Image of the blood vessel (left), ground truth made by person number 1, ground truth number 2, map of the differences between the two ground truths (right).

This effect significantly impacts thin blood vessels (or small structures in general) because a vessel only two pixels wide will have a large portion of its surface in an area of uncertainty, whereas a large blood vessel's edges represent only a limited part of its area.

Moreover, there is an issue with using the Dice coefficient in vascular segmentation (or surface-based metrics in general). In scenarios involving vessels of varying sizes, coupled with the partial volume effect, these metrics are area-based, meaning that each element's contribution to the metric calculation is proportional to its area. For vessels of the same length, a large vessel will have a significantly greater total area and will contribute much more to the metric. Consequently, segmentation errors in small blood vessels may be under-penalized, which we want to avoid to ensure high-quality segmentation.

3.2 First Approach: Smoothing the segmentation masks

Our initial approach stems from the observation that ground truth data may not always be accurate at the edges. Therefore, why penalize the model for minor errors at these edges? As long as the overall structure of the prediction resembles the ground truth, who can definitively say which of the two is correct when there is only a one-pixel discrepancy at the edges? The idea of using a non-binary ground truth, incorporating transition zones between classes 0 and 1, would reduce the penalty on the model for discrepancies at the edges. The next step is to determine the type of transition and how to adapt the metrics to this new form of ground truth.

One initial idea was to simply blur the ground truth using a Gaussian blur (3.3). However, this approach was set aside because blurring the mask also altered the values at the centers of the blood vessels, which should remain at 1. It was preferable to use a distance function with an exponential decay to achieve the desired shape (3.4): smoothed edges and a plateau at a value of 1 corresponding to the center of the blood vessel.

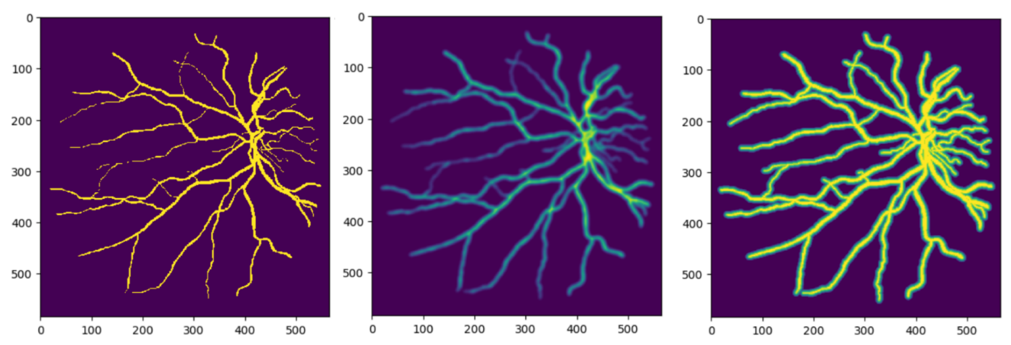


Figure 3.3: Binary ground truth mask (left), ground truth blurred with a Gaussian function (center), ground truth smoothed with a distance function and exponential decay (right).

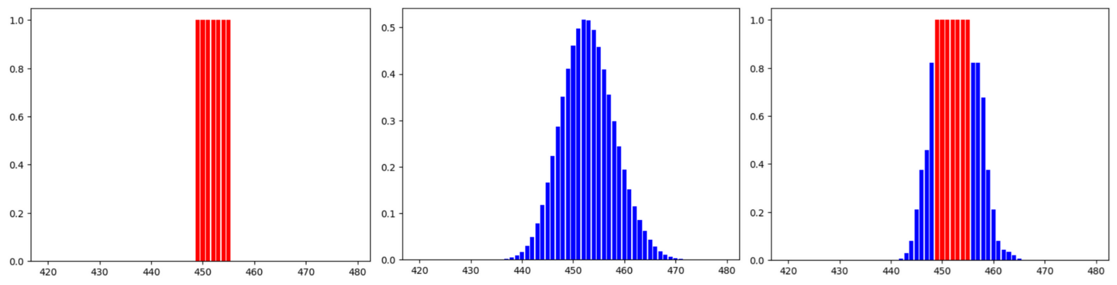


Figure 3.4: Cross-sectional view of the masks of the same blood vessel segment. Binary ground truth mask (left), ground truth blurred with a Gaussian function (center), ground truth smoothed with a distance function and exponential decay (right). Red bars indicate a value exactly of 1.

These new values for the ground truth require a new calculation for TP, TN, FP, and FN. Previously, these could be considered area calculations (3.5), but now we have intermediate values between 0 and 1, which resemble volume calculations or integral calculations under the curve (3.5).

We can now simply use the new confusion matrix values in the calculation of the same metrics as in section 2.2.1. This results in new metric values without changing their definitions, allowing for comparisons across different examples. We can test with modified versions of the ground truth by applying transformations to simulate segmentation errors, such as over-segmentation with a dilated ground truth.

3.2.1 Analysis of Results and Issues

With a slight shift (1 pixel) of the prediction relative to the ground truth, we observe fewer false positives and false negatives in the smoothed version (3.7) compared to the original binary version (3.6). The metrics (3.8) seem to penalize slight shifts less, which is desired in our study. Additionally, the difference between the metrics is significant, particularly for the MCC, which shows a difference of 0.23 between the two versions. This demonstrates a method to be significantly less sensitive to minor edge variations.

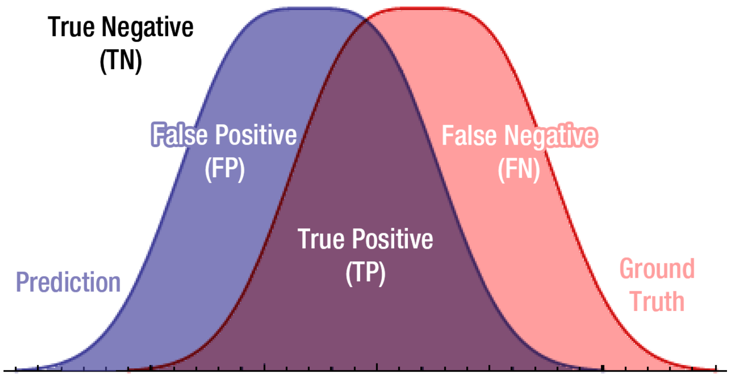


Figure 3.5: Diagram of the new calculation of confusion matrix values, with smoothed ground truth and prediction.

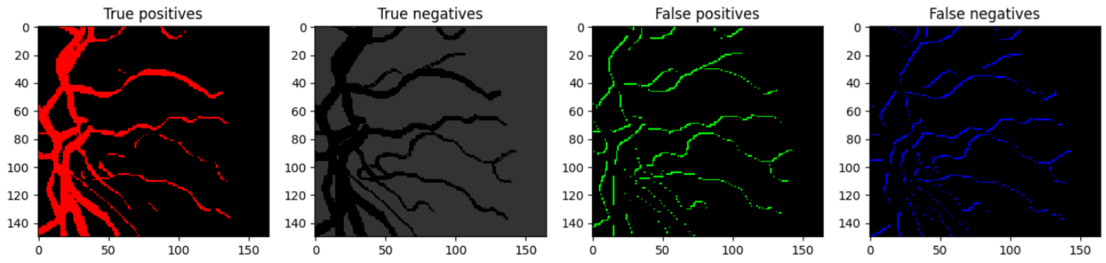


Figure 3.6: Logical map of the shifted prediction with the binary original ground truth segmentation. True positives in red (left), true negatives in gray, false positives in green, and false negatives in blue (right).

If we instead perform a morphological opening on the prediction (erosion followed by dilation), some of the thinner blood vessels disappear from the mask. Comparing our two methods in this scenario, we observe a smaller difference in their results (3.9, 3.10). Both methods penalize holes in the prediction similarly, which is expected, and with even lower metrics values (3.11) for the smooth version.

To conclude this section, we propose not only a new metric but a new system for calculating the values of the confusion matrix. However, analyzing the metrics reveals that the values are still too high; the disappearance of small blood vessels did not significantly impact the metrics, which should be more strongly penalized for such segmentation issues. Nevertheless, this idea is promising and can potentially be used in the calculation of other metrics to compensate for this problem, which will be explored in a subsequent section.

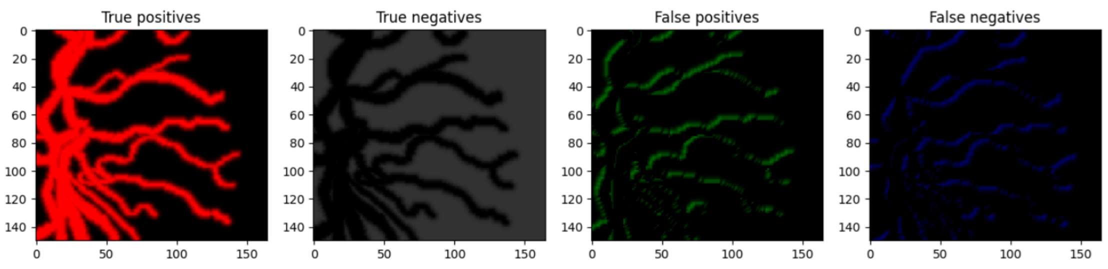
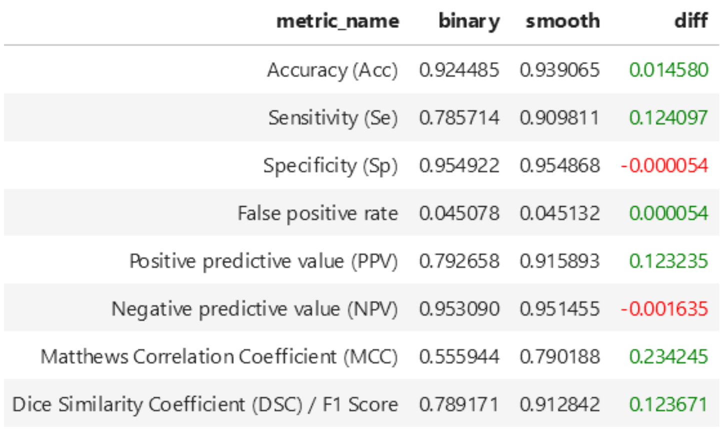


Figure 3.7: Logical map of the shifted prediction with the smoothed ground truth segmentation.



metric_name	binary	smooth	diff
Accuracy (Acc)	0.924485	0.939065	0.014580
Sensitivity (Se)	0.785714	0.909811	0.124097
Specificity (Sp)	0.954922	0.954868	-0.000054
False positive rate	0.045078	0.045132	0.000054
Positive predictive value (PPV)	0.792658	0.915893	0.123235
Negative predictive value (NPV)	0.953090	0.951455	-0.001635
Matthews Correlation Coefficient (MCC)	0.555944	0.790188	0.234245
Dice Similarity Coefficient (DSC) / F1 Score	0.789171	0.912842	0.123671

Figure 3.8: Comparative table of the metrics, applying each metric to both methods (binary masks and smoothed masks), and displaying the difference between them.

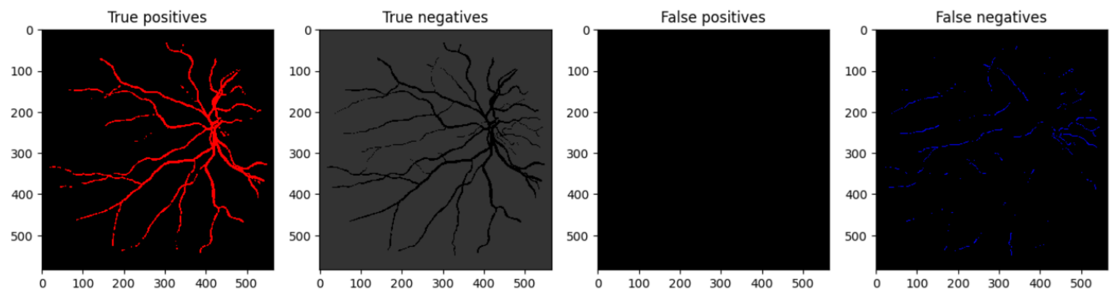


Figure 3.9: Logical map of the opened prediction with the binary original ground truth segmentation. True positives in red (left), true negatives in gray, false positives in green, and false negatives in blue (right).

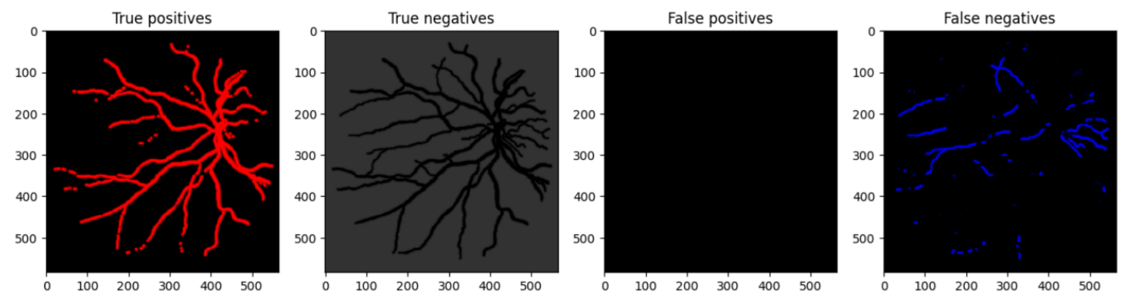
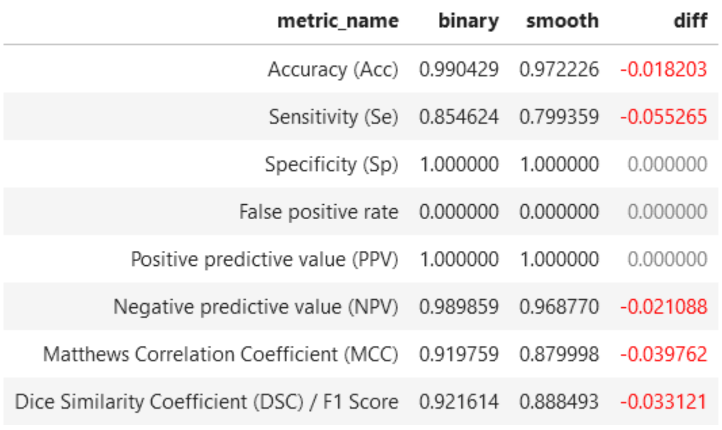


Figure 3.10: Logical map of the opened prediction with the smoothed ground truth segmentation.



metric_name	binary	smooth	diff
Accuracy (Acc)	0.990429	0.972226	-0.018203
Sensitivity (Se)	0.854624	0.799359	-0.055265
Specificity (Sp)	1.000000	1.000000	0.000000
False positive rate	0.000000	0.000000	0.000000
Positive predictive value (PPV)	1.000000	1.000000	0.000000
Negative predictive value (NPV)	0.989859	0.968770	-0.021088
Matthews Correlation Coefficient (MCC)	0.919759	0.879998	-0.039762
Dice Similarity Coefficient (DSC) / F1 Score	0.921614	0.888493	-0.033121

Figure 3.11: Comparative table of the metrics, applying each metric to both methods (binary masks and smoothed masks), and displaying the difference between them.

3.3 Second Approach: Skeleton-Based Metric

After some reflection and bibliographic research, a new approach came to us, which notably uses an existing metric that will be presented below.

3.3.1 clDice

The clDice Shit et al. (2021) is a cost function specialized in the segmentation of tubular objects such as blood vessels. It takes into account not each pixel/voxel of the image but the skeleton itself of the blood vessels, in order to preserve the connectivity and topology of the prediction.

The formula for clDice resembles the dice formula at first glance:

$$\text{clDice}(V_P, V_L) = 2 \times \frac{\text{Tprec}(S_P, V_L) \times \text{Tsens}(S_L, V_P)}{\text{Tprec}(S_P, V_L) + \text{Tsens}(S_L, V_P)}$$

With:

$$\text{Tprec}(S_P, V_L) = \frac{|S_P \cap V_L|}{|S_P|} \quad ; \quad \text{Tsens}(S_L, V_P) = \frac{|S_L \cap V_P|}{|S_L|}$$

where V_L is the ground truth mask, V_P is the segmentation prediction, S_L is the skeleton of the ground truth, and S_P is the skeleton of the prediction.

An implementation of a differentiable skeletonization algorithm is also proposed by Shit et al. (2021), using a series of pooling operations and activation functions (3.12). Despite imperfect results when re-implementing this algorithm, it will be usable when we want to adapt our metric as a cost function.

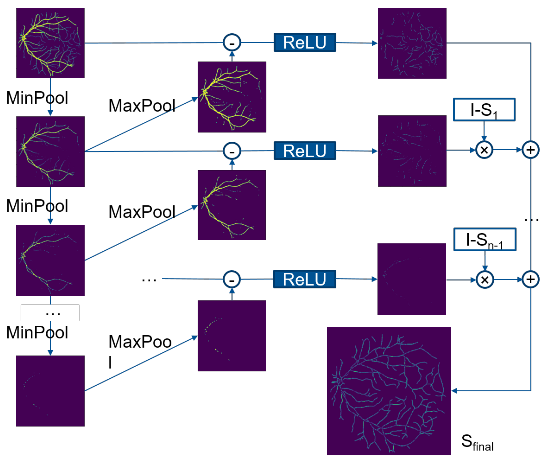


Figure 3.12: Differentiable skeletonization algorithm of the segmentation mask

3.3.2 Merging Metrics

After testing, we observe that for thin vessels, the clDice is not sufficient because as soon as the blood vessel deviates slightly from the ground truth, it is so thin that its skeleton directly exits the mask and the metric penalizes it.

Thus, it was imagined to merge this metric with the mask smoothing technique presented in section 3.2, thus creating a metric we called "smooth_clDice" (3.13). We can repeat the tests with different operations on the prediction to observe how our metrics behave (3.14, 3.15).

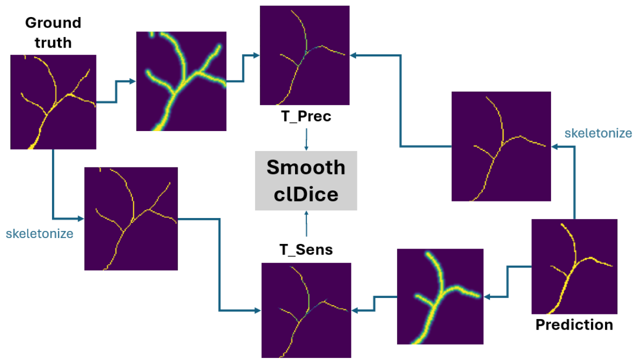


Figure 3.13: Algorithm of the "smooth_clDice", with the addition of the masks smoothing before the computation of T_{sens} and T_{prec}

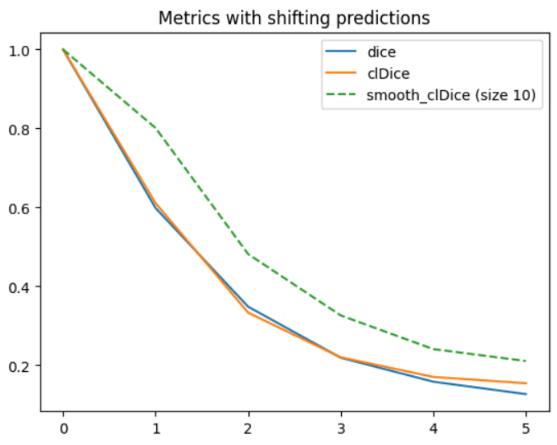


Figure 3.14: Evolution of metrics (dice, clDice, smooth_clDice) with respect to the strength of the shifting applied to the prediction (in pixels).

We can see that due to the problem of thin vessels mentioned earlier, the clDice does not allow for less penalization of small edge variations and gets almost the same values as the classic dice (3.14). However, if we apply our mask smoothing to the ground truth and the prediction, we observe a certain tolerance, especially for the smallest modifications, which is what we are aiming for here.

For the morphological opening operations, which make the thin blood vessels disappear more and more as the kernel radius increases, we can see that the clDice and smooth_clDice penalize the prediction more than the classic dice (3.15), which is also expected since we aim to

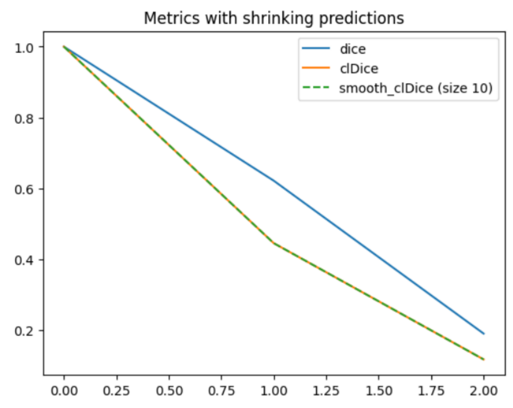


Figure 3.15: Evolution of metrics (dice, clDice, smooth_clDice) with respect to the kernel radius of the morphological opening applied to the prediction (in pixels).

ensure the detection of all vessels, even the smallest ones.

We used a smoothing size of 10 here to amplify the phenomenon, but the power of the decay function we use for the distance map allows us to modulate this effect.

Chapter 4

Future Work

4.1 Finalizing the Metric (smooth_clDice)

4.1.1 Testing the Metric in Real Cases

Add the smooth_clDice metric to the benchmarking metrics to test it in real cases and observe how it performs on real images.

4.1.2 Adapting the Metric as a Cost Function

To adapt our metric to a cost function, we need to ensure that each step of the metric calculation is differentiable. I never had the chance to write my very own cost function, so it will be an opportunity to discover and experiment with PyTorch.

Small training sessions should be run to see if it allows the model to converge and, if so, towards a coherent result. In the best case, the result would not only be coherent but also minimize the impact of the partial volume effect and especially minimize errors in the topology of blood vessels by reducing connectivity problems.

4.2 Continuation of Benchmarking

4.2.1 Efficiency Metrics

Since I don't have a stable environment to perform my benchmarks, nor a dedicated workstation/machine for this task, adding metrics that measure the efficiency of our models is not coherent. The results can depend on the environment in which the benchmark is executed or the context and other programs running simultaneously, introducing too much variability, thus not interesting for now. However, these metrics remain important criteria, especially for companies, researchers, or anyone who wants to use the models, as not everyone has unlimited memory and computing power resources. Others, on the contrary, may consider the most demanding solutions to meet their constraints (e.g., real-time applications).

There are some efficiency metrics we could introduce to the benchmark:

- Training time

- Running time or inference time, the time a model takes to make a prediction from an input image
- Maximum GPU memory used during model execution, important because graphics cards, especially older ones, are quite limited in terms of memory, so knowing the maximum necessary resources at a given time is essential
- Storage used by the model weights/biases

4.2.2 Integration and Benchmarking Continuation

We would firstly test to benchmark on more complex datasets since we have seen that with DRIVE, a simple UNet with minimal modifications was enough to achieve very acceptable results. Therefore, we need to expand our dataset base to try and challenge the basic UNet and then test our new cost function to see if it has any benefit.

It is important to test the inter-dataset robustness using the benchmarking tool that was previously presented. For this, we would need to finish the command-line tool to make it as user-friendly as possible.

Perhaps providing the user with the ability to add new datasets or even metrics themselves could be an interesting feature. It would mean to write some modular code to offer a usable and maintainable benchmarking tool.

Also, we could consider expanding the study and metrics to other types of data. For example, with 3D data, the calculation of metrics and cost functions would need to be modified.

Chapter 5

Conclusion

In this study, we have explored various aspects of vascular segmentation, highlighting its critical role in medical image analysis for diagnosis, monitoring, and treatment planning. Our work began with a state of the art of existing segmentation methods, especially deep learning algorithms. To ensure the reliability of our approach, we tested the most prevalent model in current literature: the U-Net, on a well-known eye fundus dataset, to verify that it converges and provides coherent results. We then established a benchmarking pipeline to be ready for integrating more models and datasets.

Our exploration revealed several challenges, particularly in segmenting thin blood vessels. We tried to address these challenges by developing a new metric, smooth_clDice, which aims to improve segmentation accuracy by focusing on the skeleton of the vessels rather than individual pixels and by adding a confidence zone to the ground truth and segmentation masks to avoid overly penalizing small deviations in thin vessels. This approach should help maintain the connectivity and topology of the vessels, which are crucial for accurate medical assessments. While the initial results are promising, further testing and adaptation are required to refine the metric and ensure its effectiveness as a cost function in model training.

Future work will focus on finalizing the smooth_clDice metric, integrating the new metric into our benchmarking framework and testing it with real images to observe its performance, continuing the benchmarking with more complex datasets, and improving the robustness of our models across different datasets. Additionally, we will explore efficiency metrics to provide a comprehensive evaluation of model performance, considering factors such as training time, inference time, GPU memory usage, and model storage requirements.

Overall, our study provides a solid foundation for improving vascular segmentation techniques, with the potential to enhance the accuracy and reliability of medical image analysis. The development of new metrics and continued benchmarking efforts will be crucial in advancing this field and supporting clinicians in their diagnostic and treatment planning processes.

Chapter 6

Bibliography

Barrowclough, O. J., Muntingh, G., Nainamalai, V., and Stangeby, I. (2021). Binary segmentation of medical images using implicit spline representations and deep learning. *Computer Aided Geometric Design*, 85:101972. (page 4)

Dosovitskiy, A., Beyer, L., Kolesnikov, A., Weissenborn, D., Zhai, X., Unterthiner, T., Dehghani, M., Minderer, M., Heigold, G., Gelly, S., et al. (2020). An image is worth 16x16 words: Transformers for image recognition at scale. *arXiv preprint arXiv:2010.11929*. (page 7)

Hamza, M. (2023). Multi-class semantic segmentation with u-net (pytorch). *Medium*. Accessed: 2024-06-15. (page 4)

Isensee, F., Jaeger, P. F., Kohl, S. A., Petersen, J., and Maier-Hein, K. H. (2021). nnu-net: a self-configuring method for deep learning-based biomedical image segmentation. *Nature methods*, 18(2):203–211. (page 7)

Luc, P., Couprie, C., Chintala, S., and Verbeek, J. (2016). Semantic segmentation using adversarial networks. *arXiv preprint arXiv:1611.08408*. (page 7)

Luo, G., Wang, K., Liu, J., Li, S., Liang, X., Li, X., Gan, S., Wang, W., Dong, S., Wang, W., et al. (2023). Efficient automatic segmentation for multi-level pulmonary arteries: The parse challenge. *arXiv preprint arXiv:2304.03708*. (page 7)

Milletari, F., Navab, N., and Ahmadi, S.-A. (2016). V-net: Fully convolutional neural networks for volumetric medical image segmentation. In *2016 fourth international conference on 3D vision (3DV)*, pages 565–571. Ieee. (page 6)

Moccia, S., De Momi, E., El Hadji, S., and Mattos, L. S. (2018). Blood vessel segmentation algorithms—review of methods, datasets and evaluation metrics. *Computer methods and programs in biomedicine*, 158:71–91. (page 7)

Mostapha, M. (2014). A novel diffusion tensor imaging-based computer-aided diagnostic system for early diagnosis of autism. (page 8)

Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. In *Medical Image Computing and Computer-Assisted Intervention–MICCAI 2015: 18th International Conference, Munich, Germany, October 5-9, 2015, Proceedings, Part III 18*, pages 234–241. Springer. (page 6)

Shit, S., Paetzold, J. C., Sekuboyina, A., Ezhov, I., Unger, A., Zhylka, A., Pluim, J. P., Bauer, U., and Menze, B. H. (2021). cldice-a novel topology-preserving loss function for tubular structure segmentation. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pages 16560–16569. (page 21)

Staal, J., Abramoff, M., Niemeijer, M., Viergever, M., and van Ginneken, B. (2004). Ridge-based vessel segmentation in color images of the retina. *IEEE Transactions on Medical Imaging*, 23(4):501–509. (page 5)

Zhou, Z., Rahman Siddiquee, M. M., Tajbakhsh, N., and Liang, J. (2018). Unet++: A nested u-net architecture for medical image segmentation. In *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support: 4th International Workshop, DLMIA 2018, and 8th International Workshop, ML-CDS 2018, Held in Conjunction with MICCAI 2018, Granada, Spain, September 20, 2018, Proceedings 4*, pages 3–11. Springer. (page 6)