VIETNAM NATIONAL UNIVERSITY, HO CHI MINH CITY
UNIVERSITY OF TECHNOLOGY
FACULTY OF COMPUTER SCIENCE AND ENGINEERING

Programming Intergration Project (CO3101)

Deep learning in Medical Researches: Lymphatic Vessel Segmentation

Advisor: Ph.D Nguyễn An Khương
Students: Vũ Hoàng Tùng - 2252886

HO CHI MINH CITY, May 2025

Contents

1 Abstract

The lymphatic system - often referred to as the "forgotten circulation"−plays a crucial role in maintaining fluid balance, immune response, and cancer metastasis monitoring. Identifying their geometry shape and analyzing their diameter provides valuable insights into potential pathologies and disease states. However, there has been difficulty in visualizing as a result of limited image technologies.

Therefore in this work, we provide a pipeline that segments and measures lymphatic vessels from very few samples as video, using U−net architecture as well as Data Augmentation techniques to produce State-of-Art performance in segmentation and diameter tracking. This work lays the groundwork for the future development of our end-to-end lymphatic research system, aiming to support laboratory studies and promote the application of computer vision in bioinformatics.

2 Introduction

2.1 Motivation

Lymphatic vessels play vital roles in immune function and fluid balance. Structural and functional integrity of lymphatic vessels is essential for maintaining tissue equilibrium and immune responsiveness. Among the key parameters influencing lymphatic function is vessel diameter, which can be indicative of physiological or pathological states. Tracking and analyzing their diameter provides valuable insights into how fluid travel through vessels and abnomalities in flow and vessels, from which detecting potential pathologies and disease states.

One popular condition associated with altered lymphatic vessel diameter is *lymphedema*. This disorder results from impaired lymphatic drainage, leading to chronic tissue swelling, inflammation, and fibrosis. Lymphedema can be primary (due to developmental defects) or secondary (acquired). The latter is especially common among cancer patients, particularly breast cancer survivors. Studies report that 20−40% of these individuals may develop lymphedema following treatment, especially when lymph nodes are removed or irradiated[1].

Lymphangiectasia is another condition marked by dilated lymphatic vessels. It commonly affects the intestines or lungs and can lead to protein-losing enteropathy or respiratory complications. This condition may be congenital or arise due to lymphatic obstruction, and is typically diagnosed through imaging or endoscopy[2].

Lymphatic malformations are congenital anomalies involving clusters of abnormally dilated lymphatic vessels. These malformations are most frequently located in the head and neck and often present in infancy. Depending on their size and location, they can cause functional impairment, disfigurement, or recurrent infections[3].

In the context of cancer, *tumor-induced lymphatic obstruction* is a significant cause of lymphatic dysfunction. Tumor cells can invade or compress lymphatic vessels and lymph nodes, disrupting normal flow. This phenomenon is particularly observed in malignancies such as breast, prostate, and skin cancers. In addition to mechanical blockage, tumors can stimulate abnormal lymphangiogenesis, contributing to metastasis and disease progression[4].

Finally, several genetic syndromes are associated with lymphatic abnormalities. Conditions such as Noonan syndrome, Turner syndrome, and Klippel-Trénaunay syndrome often feature dilated or malformed lymphatic vessels. Increased availability of genetic testing and high-resolution imaging has led to improved identification and classification of these anomalies[5].

2.2 Recent achievements

In summary, changes in lymphatic vessel diameter serve as key indicators of both localized and systemic diseases. A deeper understanding of these alterations, supported by imaging and molecular studies, holds promise for improving diagnosis, monitoring, and targeted treatment strategies across a range of conditions. However, it has historically been challenging to visualize and study because of limitations in imaging technologies. Recent advances in computer vision and deep learning techniques have revolutionized our ability to segment biomedical image, particularly for complex anatomical structures such as lymphatic vessels.

Among these, the U-Net architecture [6] has emerged as a foundational model for semantic segmentation tasks. Designed specifically for biomedical applications, U-Net addresses the two primary limitations of early CNN-based models: the requirement for large annotated datasets and the loss of spatial resolution due to successive pooling operations. It achieves this through a symmetric encoder-decoder structure, where a contracting path captures contextual features and an expansive path restores spatial precision. Additionally, U-Net incorporates data augmentation techniques, such as elastic deformations, to improve generalization in scenarios with limited training samples. Its performance on the ISBI challenge for neuron segmentation in electron microscopy underscores its capability, achieving state-of-the-art results while maintaining high computational efficiency.

Despite these advancements, certain lymphatic structures, such as mediastinal lymph nodes (LNs) and meningeal lymphatic vessels (MLVs), remain particularly challenging to segment due to their small size, ambiguous boundaries, and variability across patients. Recent research has sought to address these issues by building on the U-Net framework. For example, a 2021 study introduced an ensemble of 3D CNNs guided by anatomical priors for mediastinal LN segmentation. This method integrates slab-wise training—capturing high-resolution local features—with full-volume training, which retains global anatomical context. A pixel-wise maximum operator fuses the outputs from both strategies, enhancing segmentation accuracy by leveraging the strengths of each approach.

More recently, the MLV2-Net framework (2024) has extended the segmentation paradigm to include rater variability, a significant factor in datasets with high inter-observer disagreement. This model augments the nnU-Net architecture by encoding expert annotations as separate input channels, enabling the network to learn rater-specific segmentation patterns. These outputs are subsequently merged using a weighted majority voting scheme to yield a consensus segmentation and an uncertainty map, providing insight into regions of annotation disagreement.

2.3 Method

In contrast to prior work focused primarily on static images, this study focus on the segmentation from video. We propose a efficient pipeline that includes frame extraction from videos and the training of a constructed U-Net model, to achieve state-of-the-art performance in segmentation and diameter measurement. This end-to-end system is intended to support and advance an ongoing research in bio−infomatic: **Evidence of functional ryanodine receptors in rat mesenteric collecting lymphatic vessels** [7].

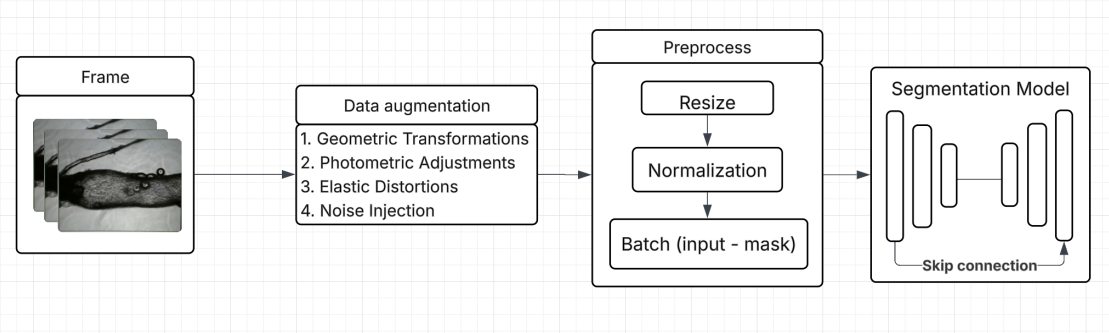


Figure 1: Model training pipeline

A visual summary of the full segmentation pipeline is presented in Figure 1 (shown above), with input preprocessing, data augmentation for rich training context and segmentational completion. The segmentation model developed in this study adopts the U-Net architecture, a convolutional neural network design widely used for image segmentation tasks. U-Net's structure consists of two primary components: an encoder (contracting path) and a decoder (expanding path), connected by skip connections. The encoder is responsible for progressively reducing the spatial dimensions of the input while capturing increasingly abstract and hierarchical feature representations. The decoder then restores the original spatial resolution by upsampling these features and refining them using high-resolution information passed through skip connections. This architecture enables the model to learn both high-level semantic features and fine-grained spatial details, effectively addressing the trade-off between localization accuracy and contextual understanding.

3 Data

The dataset consists of three videos recorded in a bioinformatics laboratory using professional protocols. Each video shows a segment of the lymphatic vessel from different areas of the body. To assess the model's generalization ability, we train it on two of these videos and validate it on the third video, the video that depicts a lymphatic section the model hasn't encountered before.

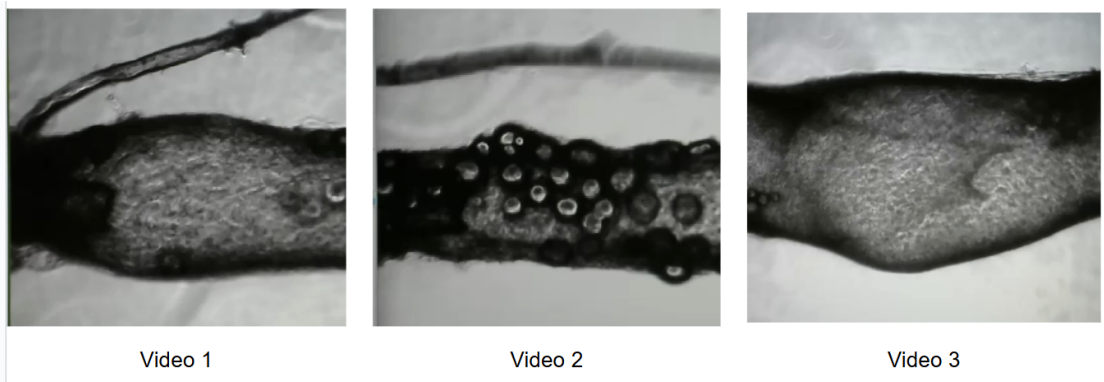


Figure 2: Lymphatic Vessels depicted in video dataset

The above figure is the extracted images from the three mentioned videos, with video 1 used

as validation data due to its balance characteristics and diameter fluctuation between the other two segments. We will train using the two remaining videos.

4 Data Preproccessing Pipeline

Effective preprocessing of image data is a critical component in the development of high-performing computer vision models. The primary objective of this stage is to ensure that the input data adheres to a consistent format and quality, enabling the learning algorithm to generalize well across diverse samples. In usual cases, the preprocessing pipeline is designed to standardize inputs, introduce controlled variability for regularization, and ensure compatibility with the assumptions of the model architecture. In this work, we add augmentation step to the preproccessing pipeline to increase the amount of data, as well as forcing the model to handle the background better, which is beneficial to the boundary segmentation needed for diameter measurement, as we will explain and prove in the next section.

4.1 Preproccess input

The original input images are captured in a resolution of $640 \times 480 \times 3$ (width $\times$ height $\times$ color channels). In order to prepare them for processing by convolutional neural networks, we apply a structured normalization pipeline.

First, images are resized such that their longest edge is limited to 256 pixels. This operation helps to bring the image dimensions within a consistent scale, while preserving aspect ratio. Subsequently, a random crop of 256×256 is taken from the resized image. This step serves two purposes: ensuring uniform input dimensions and acting as a minimal form of data augmentation by providing varied spatial perspectives during training. Below is a conversion from a picture to 256 * 256 size picture with preserved ratio, mask applied, appropriate for model training and ready to be normalized.

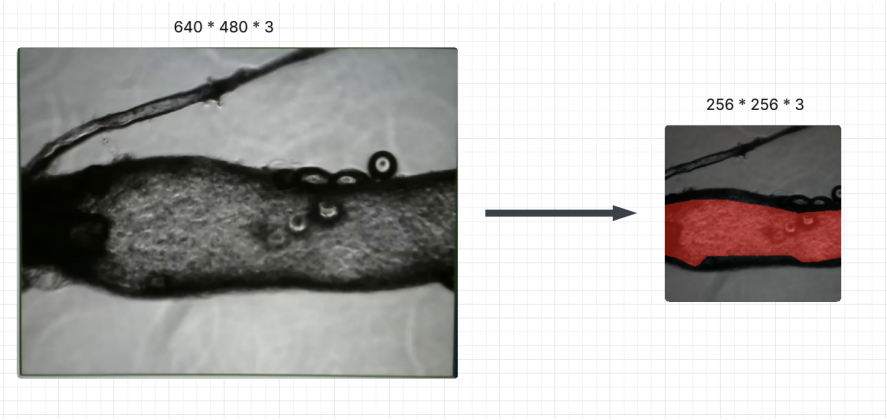


Figure 3: Resize with mask

Pixel values are then normalized channel-wise according to the ImageNet dataset statistics, with mean values of $(0.485, 0.456, 0.406)$ and standard deviations of $(0.229, 0.224, 0.225)$. This normalization aligns the data distribution with the expectations of pretrained models and supports stable training dynamics. Finally, images are transformed into PyTorch-compatible tensor

format, including a reordering of the channel dimension and conversion to floating-point representation. This completes the standardization pipeline for inputs without additional augmentations.

4.2 Data Augmentation

To enhance the training phase, we introduce a comprehensive augmentation strategy to increase dataset variability and promote robustness in learned features, especially for tasks involving fine-grained boundary segmentation. These transformations help models less sensitive to background noises and focus more on content. From images of constant size of 256 * 256, a structured sequence of augmentation operations is applied, including:

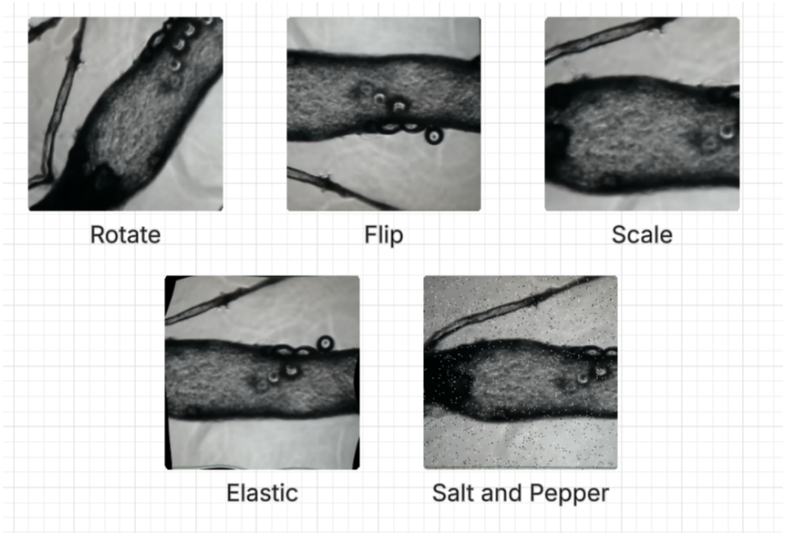


Figure 4: Some of the data Augmentation

Geometric Transformations: Random affine transformations such as rotation (up to 180 degrees) and horizontal flipping simulate changes in orientation. Additionally, alternative spatial transformations are applied via random scaling, padding, and center cropping to simulate zoom and framing effects.

Photometric Adjustments: Random changes in brightness and contrast are applied to model varying lighting conditions. These photometric shifts help the model to become invariant to common visual variations.

Elastic and Perspective Distortions: Non-linear deformations such as elastic transformations, grid distortions, and perspective warping emulate spatial irregularities and lens-related artifacts that may be present in real imaging conditions.

Noise Injection: To simulate sensor noise or corrupted input, Gaussian noise and salt-and-pepper noise are randomly applied. These disturbances force the model to learn more robust features by discouraging reliance on high-frequency noise patterns.

Each augmentation is probabilistically applied in a nested structure using controlled randomness, ensuring that each training sample is a unique variation, while maintaining semantic integrity. After all augmentations, the image is normalized using the same ImageNet-based standard and converted to tensor format for model compatibility. This extensive preprocessing pipeline supports generalization by introducing a rich distribution of training conditions and standardizing inputs for efficient model training and evaluation.

5 Models training with Unet

5.1 Model architecture

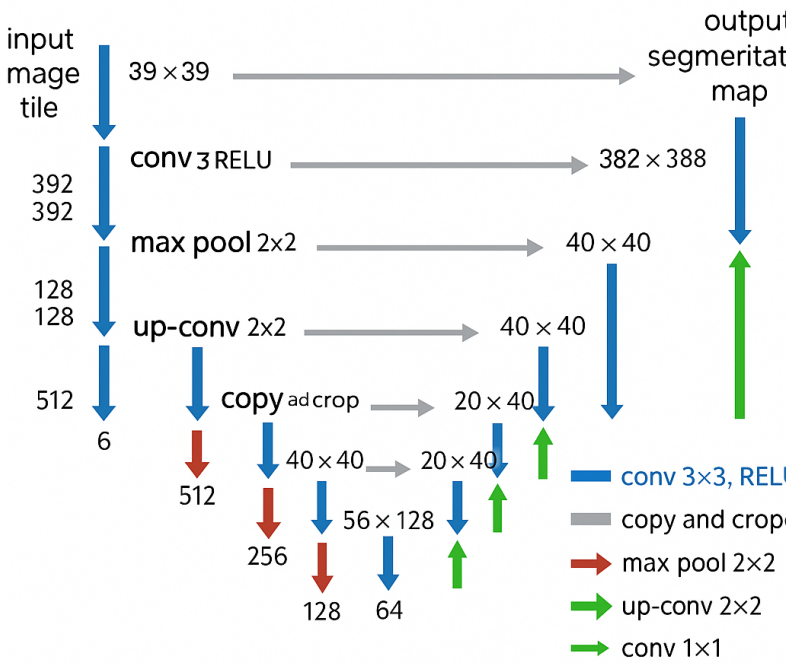


Figure 5: Unet model architecture

In this work, the segmentation system adopts the U-Net model architecture, which is characterized by its symmetric encoder-decoder structure. The encoder path (also referred to as the contracting path) consists of a sequence of convolutional layers followed by max-pooling operations, which progressively reduce the spatial dimensions of the input image while simultaneously capturing increasingly abstract and high-level feature representations. This hierarchical feature extraction enables the model to learn both local and global contextual information, which is essential for achieving accurate segmentation. The encoder functions as the backbone of the model, and in our study, we use ResNet34 as the encoder backbone.

ResNet34 is a deep convolutional neural network known for its residual learning framework[8], which facilitates the training of deeper networks by introducing identity shortcut connections. These residual connections help to mitigate the vanishing gradient problem and improve feature propagation across layers, thereby enhancing the representational capacity of the encoder. By integrating ResNet34 into the U-Net architecture, the model benefits from both the powerful feature extraction capabilities of ResNet and the spatial localization advantages of U-Net, resulting in improved segmentation performance.

Given an input batch of 4 images with size 256×256 and 3 channels (RGB), the encoder proceeds as follows:

Table 1: Detailed Encoder Architecture of U-Net with ResNet Backbone

Layer	Operation	Kernel/Stride/Pad	Output Shape	Details
Conv1	Conv2d + BN + ReLU	7×7 / 2 / 3	(4, 64, 128, 128)	Large kernel, downsampling
MaxPool	MaxPool2d	3×3 / 2 / 1	(4, 64, 64, 64)	Further downsampling
Layer1	2 × BasicBlock	3×3 / 1 / 1	(4, 64, 64, 64)	No downsampling
Layer2	2 × BasicBlock	3×3 / 2,1 / 1	(4, 128, 32, 32)	Downsampling via first block
Layer3	2 × BasicBlock	3×3 / 2,1 / 1	(4, 256, 16, 16)	Downsampling via first block
Layer4	2 × BasicBlock	3×3 / 2,1 / 1	(4, 512, 8, 8)	Final encoder layer

The encoder architecture follows a hierarchical design that progressively reduces spatial dimensions while increasing feature depth. It begins with a convolutional layer using a large kernel (7×7) with a stride of 2, followed by batch normalization and ReLU activation. This initial stage captures broad contextual information and reduces the input resolution, thereby lowering the computational burden in subsequent layers.

A max pooling operation with a 3×3 kernel and stride of 2 is then applied to further reduce the spatial resolution while preserving the number of channels. This step enlarges the receptive field and aids in discarding redundant spatial information.

The core of the encoder consists of four sequential stages composed of `BasicBlock` residual modules, which follow the design of ResNet. Each stage contains two residual blocks, with the first block in each stage performing spatial downsampling via a stride-2 convolution. Across the stages, the number of feature channels increases (from 64 to 512), enabling the network to encode increasingly abstract representations of the input.

Residual connections within each block facilitate gradient flow during backpropagation, addressing the degradation problem associated with deeper networks. This is particularly beneficial in encoder-decoder frameworks such as U-Net, where feature propagation across many layers is critical. Throughout the encoder:

- The spatial resolution is reduced by a factor of 2 at each downsampling stage.
- The number of feature channels is progressively increased to capture more complex semantic information.

The final encoder output is a compact feature representation with high semantic richness and low spatial resolution. This feature map serves as the input to the decoder, where spatial resolution is gradually restored. The encoder's hierarchical structure, combined with residual learning, ensures efficient feature extraction and forms a robust foundation for downstream segmentation tasks.

Table 2: Decoder Architecture of U-Net with ResNet Backbone

Layer	Output Shape	Description
Decoder4	(4, 256, 16, 16)	Upsample; add Encoder Layer3
Decoder3	(4, 128, 32, 32)	Upsample; add Encoder Layer2
Decoder2	(4, 64, 64, 64)	Upsample; add Encoder Layer1
Decoder1	(4, 64, 128, 128)	Upsample; add MaxPool output
FinalConv	(4, 1, 256, 256)	1×1 convolution for logits

The decoder is constructed as a series of `UnetDecoderBlock` modules, each designed to upsample spatial resolution and integrate high-resolution features from the encoder via skip connections.

This process enables the network to recover fine-grained spatial information lost during encoding. Each `Unet−DecoderBlock` consists of two convolutional sub-blocks (`Conv2d` → `BatchNorm2d` → `ReLU`) interleaved with optional attention modules, which in this configuration are identity functions. The structure maintains high fidelity in feature refinement without the additional computational overhead from attention.

Mathematically, the decoder at stage i reconstructs features $\mathbf{F}_i$ from the previous decoder stage $\mathbf{F}_{i+1}$ and the corresponding encoder features $\mathbf{E}_i$ as follows:

$$\mathbf{F}_i = \mathcal{C}_i \left(\text{Upsample}(\mathbf{F}_{i+1}) \oplus \mathbf{E}_i \right),$$

where $\mathcal{C}_i$ denotes the two-layer convolutional refinement block at stage i, and $\oplus$ is the channel-wise concatenation. The upsampling operation doubles the spatial resolution of the decoder features, while the concatenated encoder features inject high-resolution context from earlier layers. The channel dimensions across the decoder are gradually reduced, matching the encoder's multi-scale feature resolutions in the following sequence:

$$768 \to 256 \to 128 \to 64 \to 32 \to 16$$

At the end of the decoder, a `SegmentationHead` is applied to generate pixel-wise predictions. This head consists of a 3×3 convolution that maps the decoder output $\mathbf{F}_0 \in \mathbb{R}^{B \times H \times W \times 16}$ to a single-channel logit map:

$$\mathbf{O} = \mathcal{C}_{3\times3}(\mathbf{F}_0), \quad \mathbf{O} \in \mathbb{R}^{B \times H \times W \times 1}.$$

Since the task is binary segmentation, a sigmoid activation function is applied to obtain per-pixel class probabilities:

$$\hat{\mathbf{Y}} = \sigma(\mathbf{O}) = \frac{1}{1 + e^{-\mathbf{O}}},$$

where $\hat{\mathbf{Y}} \in [0, 1]^{B \times H \times W \times 1}$.

A final thresholding step is used to produce binary predictions:

$$\hat{\mathbf{Y}}_{\text{bin}}(x, y) = \begin{cases} 1, & \text{if } \hat{\mathbf{Y}}(x, y) > 0.5 \\ 0, & \text{otherwise} \end{cases}.$$

This binary output $\hat{\mathbf{Y}}_{\text{bin}}$ represents the model's segmentation mask, where each pixel is classified as either belonging to the foreground (class 1) or background (class 0). The decoder's design, combining learned upsampling with skip connections, ensures accurate boundary reconstruction and spatial detail preservation critical for segmentation performance.

5.2 Training

The training process of the segmentation model follows a standardized pipeline optimized for stability, convergence, and generalization. This subsection outlines the input preprocessing strategy, batching procedure, and training duration in terms of epochs.

To align with the requirements of the ResNet34 backbone pretrained on ImageNet, input images are normalized using the dataset's mean and standard deviation statistics. Specifically, each RGB channel is normalized independently by subtracting the channel-wise mean $\mu = [0.485, 0.456, 0.406]$ and dividing by the standard deviation $\sigma = [0.229, 0.224, 0.225]$. This transformation ensures that the input distribution matches the one observed during ResNet34's pretraining phase, thereby enhancing transfer learning effectiveness and accelerating convergence:

$$x_{\text{norm}} = \frac{x - \mu}{\sigma}$$

Training is conducted using mini-batches to exploit parallelism and stabilize the optimization process. A fixed batch size of $N = 4$ is employed, meaning that each iteration processes four images and their corresponding binary masks simultaneously. Batching improves computational efficiency and reduces the variance of gradient estimates, resulting in smoother and more consistent updates to the model parameters.

The training process spans 10 epochs, where one epoch corresponds to a complete pass through the entire training dataset. To optimize the model parameters, the binary cross-entropy loss with logits (`BCEWithLogitsLoss`) is employed as the training objective. This loss function make senses for binary segmentation tasks, as it combines a sigmoid activation with binary cross-entropy in a numerically stable formulation:

$$\mathcal{L}_{\text{BCE}} = -\left[y \cdot \log(\sigma(x)) + (1 - y) \cdot \log(1 - \sigma(x))\right]$$

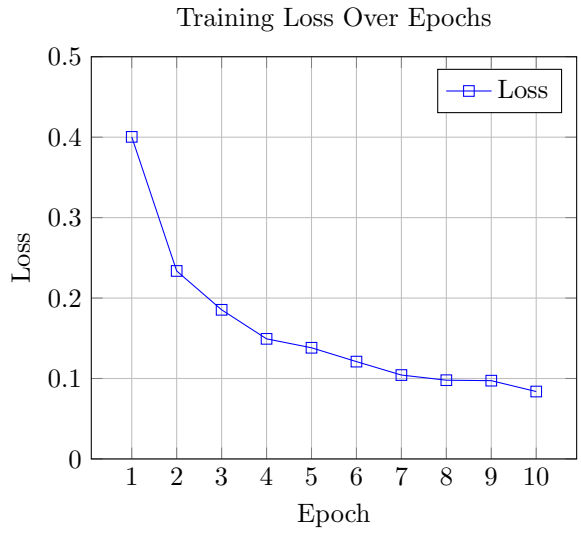
where $\sigma(x)$ denotes the sigmoid function applied to the raw output logits. This formulation allows direct training on the unnormalized outputs of the model.

To perform gradient-based optimization, the Adam optimizer is utilized with a learning rate of 1×10^{-4}. Adam combines the benefits of momentum and adaptive learning rates, enabling efficient and stable convergence, especially in the presence of sparse gradients and noisy data. Throughout training, loss and validation metrics are monitored to detect overfitting and inform learning rate adjustments or early stopping strategies. These combined practices ensure a stable optimization trajectory tailored for binary semantic segmentation.

```
model = smp.Unet(
    encoder_name="resnet34",        # Use ResNet-34 as the encoder
    encoder_weights="imagenet",     # Pretrained on ImageNet
    in_channels=3,                  # Input has 3 RGB channels
    classes=1                       # Output single channel for binary segmentation
)
criterion = nn.BCEWithLogitsLoss()
optimizer = optim.Adam(model.parameters(), lr=1e-4)
```

Listing 1: Model and optimizer setup

The above code snippet describes the model configuration, loss function, and optimizer. The model architecture is built upon the *segmentation_models_pytorch*[9] library. For this work, we use GPU to perform calculation and build models. Result after 10 epochs reveals a consistent decrease in training loss across the epochs, indicative of effective convergence during the training phase. Notably, there is a steep reduction in loss during the initial epochs, particularly between Epoch 1 and Epoch 3, dropping from 0.4004 to 0.1854. This rapid initial decrease suggests a swift adaptation of the model parameters to the underlying data distribution.



5.3 Models Evaluation

To assess the performance of segmentation models, four common metrics were used: Intersection over Union (IoU), Dice coefficient, Pixel Accuracy, and Boundary F1 Score. These metrics are defined as follows:

Intersection over Union (IoU): Also known as the Jaccard Index, IoU quantifies the overlap between the predicted segmentation mask P and the ground truth mask G:

$$\text{IoU} = \frac{|P \cap G|}{|P \cup G|} = \frac{\sum_i (P_i \wedge G_i)}{\sum_i (P_i \vee G_i)} \tag{1}$$

where P_i and G_i denote the binary indicators of pixel i being predicted as object in P and G, respectively. $|P \cap G|$ represents true positives, and $|P \cup G|$ represents both true positives and false predictions.

Dice Coefficient (F1-score): Dice measures the similarity between predicted and ground truth masks:

$$\text{Dice} = \frac{2|P \cap G|}{|P| + |G|} = \frac{2\sum_i (P_i \wedge G_i)}{\sum_i P_i + \sum_i G_i} \tag{2}$$

Here, $|P|$ and $|G|$ are the total number of predicted positive pixels and ground truth positive pixels, respectively. Dice is especially effective in scenarios with class imbalance.

Pixel Accuracy: Pixel accuracy evaluates the proportion of correctly predicted pixels (both foreground and background):

$$\text{Pixel Acc} = \frac{\sum_i (P_i = G_i)}{N} \tag{3}$$

where N is the total number of pixels, and $P_i = G_i$ indicates a correct prediction for pixel i.

Boundary F1 Score (BF Score): BF Score's purpose is to assess how well the predicted boundaries align with ground truth, emphasizing contour quality. Evaluation function works exactly like Dice Score, but we use border pixel only:

$$\text{BF Score} = \frac{2 \cdot \text{Precision}_b \cdot \text{Recall}_b}{\text{Precision}_b + \text{Recall}_b} \tag{4}$$

where:

$$\text{Precision}_b = \frac{\text{True Boundary Pixels}}{\text{Predicted Boundary Pixels}}, \quad \text{Recall}_b = \frac{\text{True Boundary Pixels}}{\text{Ground Truth Boundary Pixels}}$$

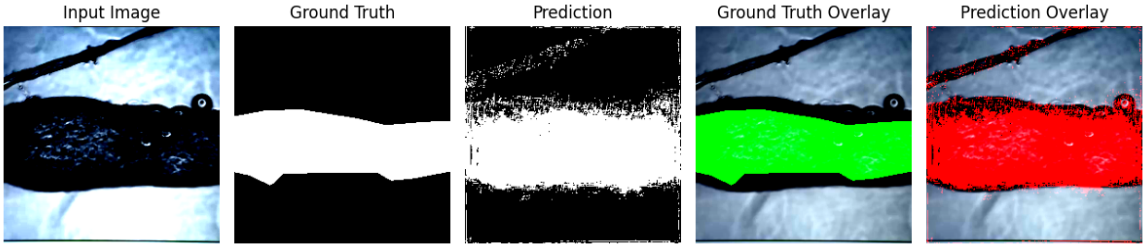
Boundaries are typically extracted from P and G using contour extraction or edge detection methods, and a tolerance δ is used to match boundary pixels.

After 5 independent run test with seperated testing data, The U-Net model was evaluated under two conditions: with and without data augmentation. The results are shown below:

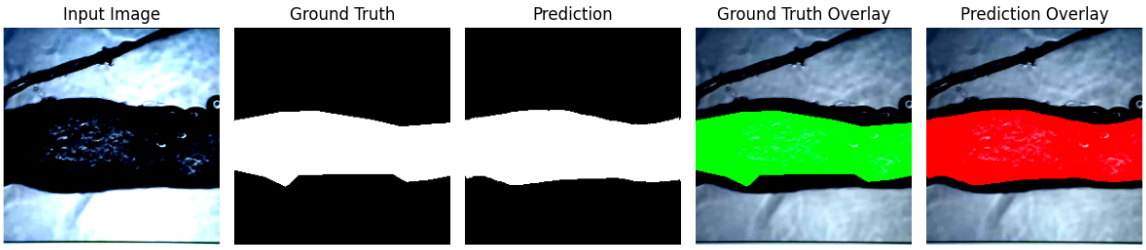
Table 3: Evaluation Metrics for U-Net

Model Name	IoU	Dice	Pixel Acc	Boundary F1
U-Net (no aug)	0.6957	0.8195	0.8856	0.0024
U-Net (with aug)	**0.8654**	**0.9267**	**0.9605**	**0.5476**

As seen above, Unet shows a strong capacity to segment bio-image with the Dice Score of 0.8, but falls short on boundary segmentation due to background noise. Data augmentation significantly improves the segmentation performance of U-Net across all metrics, esspecially the Boundary F1 Score, which increased from 0.0024 to 0.5476, indicating better boundary delineation. Since boundary segmentation is important for diameter measurement, our methodology has been proven correct to improve performance and construct a robust model for noise and abnormal cases.



(a) Segmentation results without data augmentation



(b) Segmentation results with data augmentation

Figure 6: Visualization of the Unet segmentation result. The first image (a) shows chaotic and out-of-bound binary masks while in the second image (b), the segmented regions align more closely with the ground truth, indicating better boundary detection and object integrity.

To further evaluate the effectiveness of U-Net, we compare its augmented variant with other popular segmentation architectures using the same dataset and evaluation settings.

Table 4: Performance Comparison with Other Models (With Augmented Data)

Model Name	IoU	Dice	Pixel Acc	Boundary F1
FPN	0.8450	0.9146	0.9565	0.8194
MANet	**0.8827**	**0.9366**	**0.9664**	0.5708
PSPNet	0.8603	0.9235	0.9602	**0.8472**
U-Net	0.8654	0.9267	0.9605	0.5476

While U−net achieves high IoU and Dice scores, PSPNet with augmentation completely outperforms U-Net in boundary accuracy, and MANet achieves the highest pixel accuracy, Dice score and IoU. Although U-Net offers a strong trade-off between simplicity and effectiveness, it makes sense when a model designed in 2015 fall behinds latest state-of-the-art models when it comes to accuracy.

To further improve segmentation performance, we examine U-Net++ (also known as Nested U-Net), an enhanced version of U-Net designed to address some of the limitations of the original architecture. U-Net++ introduces the structure of the skip connections. While U-Net directly connects encoder and decoder feature maps at corresponding levels, U-Net++ introduces nested dense skip connections that include intermediate convolutional layers. This design allows encoder features to be gradually refined before merging into the decoder, effectively narrowing the semantic gap between encoder and decoder representations. As a result, U-Net++ tends to produce more precise segmentation outputs, especially for fine structures and object boundaries.

The implemented U-Net++ architecture integrates a ResNet-34 backbone as the encoder, consisting of five hierarchical stages: an initial stem block followed by four sequential residual layers with output channels 64, 128, 256, and 512, progressively downsampling the input while capturing multi-scale features. The decoder follows the nested skip connection design of U-Net++, where feature maps from various encoder depths are densely connected to decoder sub-stages through intermediate convolutional blocks, enhancing feature refinement and spatial detail recovery. Decoder blocks perform bilinear upsampling and concatenation with aggregated features from earlier stages, followed by convolutional operations. The final segmentation head employs a 3×3 convolution to project the output to class logits. For each node $X_{i,j}$ in the decoder path of U-Net++, the computation is defined as:

$$X_{i,j} = \begin{cases} \text{Conv}\left(\text{Concat}(X_{i,0}, X_{i,1}, \ldots, X_{i,j-1}, \text{Up}(X_{i+1,j-1}))\right), & \text{if } j > 0 \\ \text{Conv}(X_i^{\text{enc}}), & \text{if } j = 0 \end{cases}$$

where:

- $i \in \{0, 1, \ldots, D\}$ is the depth index (with D being the maximum depth),
- $j \in \{0, 1, \ldots, D - i\}$ is the stage index within the nested decoder,
- X_i^{enc} is the output of the encoder at level i,
- $\text{Up}(\cdot)$ denotes the upsampling operation,
- $\text{Concat}(\cdot)$ denotes channel-wise concatenation,
- $\text{Conv}(\cdot)$ denotes convolution + activation + normalization.

The final segmentation output is computed from $X_{0,N}$, where N is the maximum depth of nested decoders (typically $N = D$).

Table 7 compares U-Net++ with several other segmentation models, including FPN, MANet, PSPNet, and the baseline U-Net, trained on the same augmented data set. U-Net++ shows a slight improvement over the other models in IoU, Dice, and pixel accuracy, outperforming them by approximately 0.01 in these metrics. However, it falls slightly short in boundary delineation, where PSPNet surpasses U-Net++ by about 0.05 in boundary F1-score.

Table 5: Performance Comparison with Other Models (Augmented Data)

Model Name	IoU	Dice	Pixel Acc	Boundary F1
FPN	0.8450	0.9146	0.9565	0.8194
MANet	0.8827	0.9366	**0.9664**	0.5708
PSPNet	0.8603	0.9235	0.9602	**0.8472**
U-Net	0.8654	0.9267	0.9605	0.5476
U-Net++	**0.8748**	**0.9312**	0.9628	0.7967

In summary, U-Net++ provides a balanced enhancement over the original U-Net, and arguably other state-of-the-art models, by leveraging improved feature aggregation and boundary refinement, yielding competitive results across all evaluated metrics.

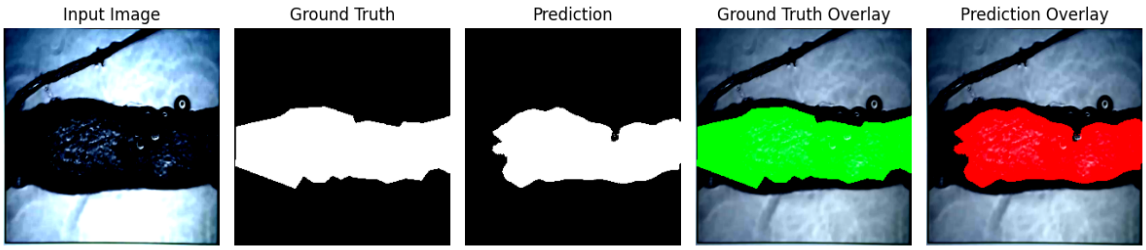
6 Conclusion

6.1 Performance Analysis

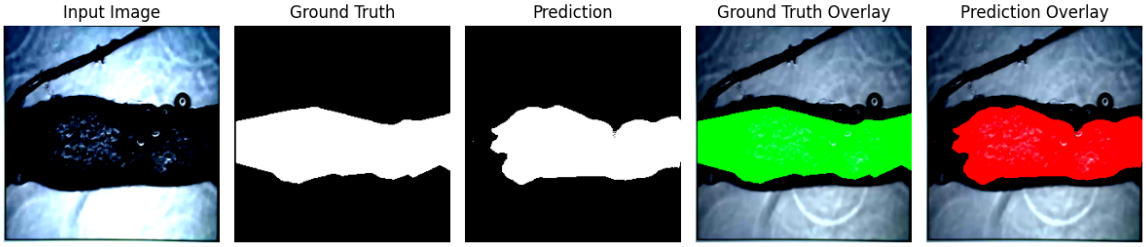
The comparative analysis across five segmentation models demonstrates that U-Net-based architectures, particularly U-Net++, achieve state-of-the-art performance in the current experimental setting. U-Net++ records the highest Intersection over Union (IoU) score of **0.8748** and a Dice coefficient of **0.9312**, underscoring its superior capability in accurately segmenting relevant regions. These improvements are attributed to its nested dense skip connections and deep supervision strategy, which facilitate richer feature reuse and more robust gradient flow during training. While PSPNet slightly outperforms others in Boundary F1 Score (**0.8472**), U-Net++ provides a more consistent balance across all metrics, validating its generalization capacity in diverse scenarios.

6.2 Limitation

Despite its strong performance in region-based segmentation, U-Net++ exhibits limitations in boundary precision. Its Boundary F1 score (0.7967) is notably lower than that of PSPNet (0.8472), indicating a relative weakness in capturing fine-grained boundary details. This pattern is also evident in the original U-Net, which have a noticably lower Boundary F1 score (0.5476). Such results suggest that while U-Net variants excel in global structure delineation, they tend to oversmooth contours, thereby underperforming in tasks where precise boundary definition is essential.



(a) Case 1



(b) Case 2

Figure 7: Underperforming cases of Unet++

The images above illustrate two cases where the segmentation results are notably lower, with overall performance scores falling below 0.8 (approximately 0.76 and 0.70, respectively). Although the IoU and Dice scores remain relatively high (above 0.82 and 0.90), indicating that the model captures the general regions well, the low Boundary F1 scores (0.57 and 0.42) reveal significant challenges in accurately capturing object boundaries. This suggests that while the model performs adequately in terms of overall pixel-wise accuracy, it struggles to delineate finer details and precise contours in these more complex or irregular cases. These examples underscore the need for further refinement in boundary detection capabilities, particularly for challenging segmentation scenarios.

Another major problem is the scarcity in training data. To prove this, we train 2 new models. For the first model, we train using only the first video, and validate on the remaining videos. For the second model, instead of using just 2 videos, we train a model with all videos, with a quarter of frames in each video as validation dataset. So although the model hasn't seen those specific images, it has seen other images of the segment before. Below is the evaluation result of both model.

Table 6: Performance of Unet++ trained on 1 video only

Model Name	IoU	Dice	Pixel Acc	Boundary F1
U-Net++	0.7498	0.8443	0.9434	0.298

Table 7: Performance of Unet++ trained on all 3 videos' data

Model Name	IoU	Dice	Pixel Acc	Boundary F1
U-Net++	0.9613	0.9802	0.9924	0.9971

While using only a small part of the data reduces its performance significantly on all metrics, except for Pixel Accuracy with minor degradation, using the full training dataset leads to strongly improved performance across all evaluation metrics. Moreover, Boundary F1 suffers the most from lacks of data. This proves the relationship between generalization, boundary accuracy and data richness. While this statement consolidate the correctness of our research method, it also shows limitations in our pipeline. In conclusion, the model's ability to capture universal patterns in human lymphatic vessels, an essential capability for accurately segmenting previously unseen vessel regions, need further improvements with more data of different regions.

6.3 Future Improvement

Future work should focus on enhancing the boundary segmentation capability and robustness to noise. One promising direction is the incorporation of boundary-aware loss functions that explicitly account for edge information. Loss functions such as Boundary Loss or composite loss formulations that combine Dice with Hausdorff Distance or contour-based penalties can provide stronger supervision for boundary refinement. These approaches could drive the model to maintain high region accuracy while also improving contour fidelity.

The strong performance of PSPNet in boundary segmentation and noise resilience can be attributed to its *pyramid pooling module*[10], which enables the aggregation of multi-scale contextual information. By extracting features at different spatial resolutions and integrating them via *upsampling* and *concatenation*, PSPNet maintains a rich representation of both global semantics and local boundary cues. This approach helps the model preserve fine structural details and remain robust against noise, explaining its superior Boundary F1 scores compared to models that rely on single-scale features.

A focused study of the architecture of PSPNet, in particular its multiscale fusion and feature integration, can inform future developments in segmentation models. Such research may contribute to boundary-sensitive designs, context-sensitive attention mechanisms, or hybrid frameworks that combine pyramid clustering with transformers, ultimately improving performance in precise segmentation tasks such as medical imaging, autonomous driving, and remote sensing.

7 Source code

https://github.com/TUng1872004/Lymphatic-vessel

References

Notes

[1]DiSipio, T., Rye, S., Newman, B., & Hayes, S. (2013). Incidence of unilateral arm lymphoedema after breast cancer: a systematic review and meta-analysis. *The Lancet Oncology*, 14(6), 500–515. https://doi.org/10.1016/S1470-2045(13)70076-7

[2]Mountford, C., Davies, P., & Mallick, A. (2017). Primary intestinal lymphangiectasia: A rare cause of protein-losing enteropathy. *Frontiers in Pediatrics*, 5, 219. https://doi.org/10.3389/fped.2017.00219

[3]Smith, M. C., Zimmerman, M. B., & Bauman, N. M. (2019). Contemporary management of lymphatic malformations. *JAMA Otolaryngology–Head & Neck Surgery*, 145(8), 749–754. https://doi.org/10.1001/jamaoto.2019.0711

[4]Padera, T. P., Meijer, E. F. J., & Munn, L. L. (2016). The lymphatic system in disease processes and cancer progression. *Science*, 352(6282), aaf6545. https://doi.org/10.1126/science.aaf6545

[5]Brouillard, P., & Vikkula, M. (2014). Genetic causes of vascular malformations. *Cold Spring Harbor Perspectives in Medicine*, 4(11), a015818. https://doi.org/10.1101/cshperspect.a015818

[6]Olaf Ronneberger, Philipp Fischer, Thomas Brox (2015). U-Net: Convolutional Networks for Biomedical Image Segmentation. https://arxiv.org/pdf/1505.04597

[7]Michiko Jo, Andrea N Trujillo, Ying Yang, Jerome W Breslin (2019). Evidence of functional ryanodine receptors in rat mesenteric collecting lymphatic vessels https://journals.physiology.org/doi/full/10.1152/ajpheart.00564.2018

[8]Kaiming He, Xiangyu Zhang, Shaoqing Ren, Jian Sun (2015). Deep Residual Learning for Image Recognition https://arxiv.org/abs/1512.03385

[9]Reference for Python library: https://smp.readthedocs.io/en/latest/models.html#unet

[10]Zhao, Hengshuang, et al. "Pyramid Scene Parsing Network." *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*. 2017.