UC Merced

UC Merced Electronic Theses and Dissertations

Title

Vascular Image Segmentation Architecture for 2D and 3D Biomedical Images

Permalink

https://escholarship.org/uc/item/02z4x9r8

ISBN

9798173794529

Author

Kaur, Asees

Publication Date

2026-07-30

Peer reviewed|Thesis/dissertation

eScholarship.org Powered by the California Digital Library
University of California

UNIVERSITY OF CALIFORNIA, MERCED

Vascular Image Segmentation Architecture for 2D and 3D Biomedical Images

A dissertation submitted in partial satisfaction of the
requirements for the degree Doctor of Philosophy

in

Applied Mathematics

by

Asees Kaur

Committee in charge:

Dr. Erica Rutter, Chair

Dr. Suzanne Sindi, Co-Chair

Dr. Arnold Kim

Dr. Tomas Rube

2026

All chapters © 2026 Asees Kaur

The dissertation of Asees Kaur is approved, and it is acceptable in quality and form for publication on microfilm and electronically:

(Dr. Arnold Kim)

(Dr. Tomas Rube)

(Dr. Suzanne Sindi, Co-Chair)

(Dr. Erica Rutter, Chair)

University of California, Merced

2026

TABLE OF CONTENTS

Signature Page . . . iii

List of Figures . . . vi

List of Tables . . . x

Acknowledgements . . . xii

Curriculum Vitae . . . xiv

Abstract . . . xix

Chapter 1 Introduction . . . 1
1.1 Introduction and Background for Biomedical Image Segmentation . . . 1
1.1.1 What are Convolutional Neural Networks? . . . 3
1.1.2 CNNs for Biomedical image segmentation . . . 4
1.2 Dissertation Structure . . . 6

Chapter 2 VISA: Vascular Image Segmentation Architecture for Two-Dimensional Digital Subtraction Angiography Images . . . 7
2.1 Introduction and Motivation . . . 9
2.2 Methods . . . 11
2.2.1 Data . . . 11
2.2.2 Training . . . 12
2.2.3 Testing . . . 19
2.3 Results and Analysis . . . 24
2.4 Discussion . . . 29
2.5 Acknowledgments . . . 30
2.6 Supplementary Material . . . 31

Chapter 3 UI-VISA: U-Net Initialized Vascular Image Segmentation Architecture 38
3.1 Introduction and Motivation . . . 38
3.2 Methods . . . 39
3.2.1 Data . . . 40
3.2.2 Data preparation . . . 41
3.2.3 Training . . . 42

3.2.4 Testing . . . 45
3.2.5 Post-processing . . . 48
3.3 Results and Analysis . . . 48
3.3.1 Statistical Inference . . . 51
3.4 Discussion . . . 52
3.5 Supplementary Material . . . 55

Chapter 4 3D-VISA: Three dimensional Vacular Image Segmentation Architecture . . . 59
4.1 Introduction and Motivation . . . 59
4.2 Data . . . 61
4.3 Training . . . 62
4.3.1 Data Preparation and Snap Extraction . . . 62
4.3.2 Addressing Class Imbalance . . . 64
4.3.3 Network Architecture and Training . . . 64
4.4 Testing . . . 65
4.4.1 3D Region Growing Algorithm (3D RGA) . . . 68
4.4.2 Post Processing . . . 71
4.5 Results and Analysis . . . 72
4.6 Discussion . . . 73
4.7 Acknowledgments . . . 76

Chapter 5 Conclusions and Future Work . . . 77

Bibliography . . . 78

LIST OF FIGURES

Figure 1.1: Example of semantic and instance image segmentation. 2
Figure 1.2: Unet architecture 5

Figure 2.1: Proposed Vascular Image Segmentation Architecture (VISA) Framework: (a) Training Pipeline Using 80×80 Image Snaps and 3×3 Ground Truth(GT) Mask Snaps for Neural Network Training, and (b) Testing Pipeline with Iterative Seed Expansion and Network Prediction for Vessel Segmentation 8
Figure 2.2: Left (a) is an example of a typical DSA Image, acquired during angiography, and on the right (b) is its corresponding manually traced mask we use as ground truth. 12
Figure 2.3: Example illustrating the creation of snaps for training images. The left panel displays a sample training image, while the middle panel offers a close-up view of a referenced section from the left image. The right panel demonstrates the generation of 80 x 80 snaps from the zoomed-in section shown in the middle panel. 14
Figure 2.4: Example illustrating the creation of snaps for training masks. The left panel displays a sample training mask, while the middle panel offers a close-up view of a referenced section from mask in the left panel. The right panel demonstrates the generation of 3×3 snaps from the zoomed-in section shown in the middle panel. 15
Figure 2.5: Network Architecture: Convolutional Neural Network (CNN) architecture consisting of layers with double convolution and ReLU (purple), max pooling (yellow), and a final single convolution (red). The network takes an $80 \times 80 \times 1$ grayscale snap as input. The dimensions labeled on each layer reflect the output of that layer, starting from $80 \times 80 \times 64$ after the first double convolution and ending at $3 \times 3 \times 1$. 17
Figure 2.6: The modified U-Net architecture used 18

Figure 2.7: Example illustrating the region growing method. The process begins with a single initial foreground pixel (yellow) as shown in the first panel. The second panel identifies the next candidate pixels (red) that will be evaluated by the trained network. The third panel shows the predicted classification of pixels into foreground (yellow) and background (purple). Finally, the fourth panel highlights the next set of candidate pixels (red) for the subsequent iteration of the network evaluation. Yellow indicates foreground pixels, purple indicates background pixels, and red indicates the next candidate pixels for classification. 21
Figure 2.8: Reconstruction process of the predicted test image. The series of subplots illustrates the progressive refinement and development of the image as the algorithm iteratively updates the predictions. 23
Figure 2.9: Qualitative comparison between U-Net and VISA predictions across three representative test cases. Each row shows an original DSA image (left), U-Net prediction (middle), and VISA prediction (right), with their respective Dice and clDice scores displayed below each prediction. The color-coded scheme indicates: white (true positives), black (true negatives), red (false positives), and yellow (false negatives). Row 1 (Image 21, Fold 4): U-Net performs better. Row 2 (Image 08, Fold 2): VISA performs better. Row 3 (Image 20, Fold 1): Both methods perform similarly. 25

Figure 3.1: Proposed UI-VISA framework for vessel segmentation. The left panel shows a test image segmented by U-Net, with predicted foreground pixels (red) used as initial seeds. The middle panel illustrates the iterative seed expansion pipeline: 80×80 image snaps centered on current seeds are fed into a trained CNN to predict 3×3 mask snaps, with newly predicted foreground pixels added as seeds and the process repeated until convergence. The right panel shows the final reconstructed vessel segmentation. 40

Figure 3.2: Architecture of the modified U-Net. The encoder (left) consists of four double convolution blocks with ReLU activations, each followed by max pooling, progressively reducing the spatial dimensions from 128×128 to 8×8 while increasing the feature channels from 64 to 512. A bottleneck block at the deepest level expands the channels to 1024. The decoder (right) mirrors the encoder, restoring spatial resolution through upsampling and concatenating encoder feature maps via skip connections (horizontal arrows). A final 1×1 convolution produces the output segmentation mask of size $128 \times 128 \times 1$. (Note that this figure is identical to Figure 2.6 from Chapter 2.) 43

Figure 3.3: VISA CNN Network Architecture: The architecture consists of layers with double convolution and ReLU (purple), max pooling (yellow), and a final single convolution (red). The network takes an $80 \times 80 \times 1$ grayscale snap as input. The dimensions labeled on each layer reflect the output of that layer, starting from $80 \times 80 \times 64$ after the first double convolution and ending at $3 \times 3 \times 1$. (Note that this figure is identical to Figure 2.5 from Chapter 2 44

Figure 3.4: Left: raw DSA test image. Middle: U-Net foreground predictions (white) thresholded at 0.5. Right: U-Net foreground predictions shown in red, used as initial seeds for the Region Growing Algorithm. 46

Figure 3.5: Qualitative comparison of segmentation outputs from VISA, U-Net, and UI-VISA approaches across three test images. Dice and clDice scores are reported under each prediction. White: true positives, Black: true negatives, Red: false positives, Yellow: false negatives. . 50

Figure 4.1: Overview of training phase of 3D-VISA. *(Top)* Shows the creation of volumetric snaps that are extracted from the full image volume ($40 \times 40 \times 40$) and its corresponding mask volume($3 \times 3 \times 3$). *(Bottom)* Each $40 \times 40 \times 40$ volumetric image snap is passed through a 3D CNN, producing a predicted $3 \times 3 \times 3$ volumetric mask snap as output. . . 63

Figure 4.2: Training and validation loss curve for the 3D CNN model over 30 epochs; blue curve shows the training loss and orange corve shows the validation loss 66

Figure 4.3: 3D Network Architecture. Convolutional Neural Network (CNN) architecture consisting of layers with double 3D convolution and 3D ReLU (blue), 3D max pooling (green), and a final single 3D convolution (red). The network takes a $40 \times 40 \times 40$ grayscale patch as input. The dimensions labeled on each layer reflect the output of that layer, starting from $40 \times 40 \times 40 \times 64$ after the first double convolution and ending at $3 \times 3 \times 3 \times 1$. 67

Figure 4.4: Example illustrating the 3D region growing method. The process begins with a single initial foreground voxel (yellow) as shown in the first panel. The second panel identifies the next candidate voxels (red) that will be evaluated by the trained network. The third panel shows the predicted classification of voxels into foreground (yellow) and background (purple). Finally, the fourth panel highlights the next set of candidate voxels (red) for the subsequent iteration of the network evaluation. Yellow indicates foreground voxels, purple indicates background voxels, and red indicates the next candidate voxels for classification. 69

Figure 4.5: Qualitative results of the 3D region growing algorithm. All volumes are cropped to the z-range of the largest missed vessel and rendered in the YZ plane. From left to right: raw image volume, ground truth mask, predicted segmentation with false negatives (yellow) and false positives (red), and a zoomed-in view of the largest missed vessel. . . 74

Figure 4.6: Confusion matrix: normalized across the rows; evaluating the proposed 3D vessel segmentation framework against ground truth labels. Quadrants display both absolute voxel counts and normalized rates: True Positive Rate (TPR), False Negative Rate (FNR), False Positive Rate (FPR), and True Negative Rate (TNR). 75

LIST OF TABLES

Table 2.1: Distribution of Training Mask Snap Sums: Table showing the counts of training image/mask patches for each sum, from 0 to 9, highlighting a predominance of background patches (sum 0) and the fewest patches at sum 8. 15
Table 2.2: Summary of Cross-Validation Results for U-Net and VISA Models without any augmentations 27
Table 2.3: Summary of Cross-Validation Results for U-Net (aug) and VISA (aug + resampling) Models 27
Table 2.4: Summary of Cross-Validation Results with Relaxed 3×3 Window Evaluation for U-Net and VISA Models without augmentations 28
Table 2.5: Summary of Cross-Validation Results with Relaxed 3×3 Window Evaluation for U-Net (aug) and VISA (aug + resampling) Models . . . 28
Table 2.6: VISA without augmentations Results - Fold 0 31
Table 2.7: VISA with augmentations and resampling Results - Fold 0 31
Table 2.8: U-Net without augmentations Results - Fold 0 32
Table 2.9: U-Net with augmentations Results - Fold 0 32
Table 2.10: VISA without augmentations Results - Fold 1 32
Table 2.11: VISA with augmentations and resampling Results - Fold 1 33
Table 2.12: U-Net without augmentations Results - Fold 1 33
Table 2.13: U-Net with augmentations Results - Fold 1 33
Table 2.14: VISA without augmentations Results - Fold 2 34
Table 2.15: RGC with augmentations and resampling Results - Fold 2 34
Table 2.16: U-Net without augmentations Results - Fold 2 34
Table 2.17: U-Net with augmentations Results - Fold 2 35
Table 2.18: VISA without augmentations Results - Fold 3 35
Table 2.19: VISA with augmentations and resampling Results - Fold 3 35
Table 2.20: U-Net without augmentations Results - Fold 3 36
Table 2.21: U-Net with augmentations Results - Fold 3 36
Table 2.22: VISA without augmentations Results - Fold 4 36
Table 2.23: VISA with augmentations and resampling Results - Fold 4 37
Table 2.24: U-Net without augmentations Results - Fold 4 37
Table 2.25: U-Net with augmentations Results - Fold 4 37

Table 3.1: Summary of Cross-Validation Results for VISA, U-Net, and UI-VISA Models 49
Table 3.2: Testing Time per Fold for VISA, U-Net, and UI-VISA Models 51

Table 3.3: Per-image Dice and clDice scores for U-Net and UI-VISA on the 26 test images. Each image appears in exactly one test fold. The difference d is computed as UI-VISA minus U-Net, so positive values favour UI-VISA. 53
Table 3.4: UI-VISA Results - Fold 0 55
Table 3.5: U-Net with Augmentations Results - Fold 0 55
Table 3.6: UI-VISA Results - Fold 1 56
Table 3.7: U-Net with Augmentations Results - Fold 1 56
Table 3.8: UI-VISA Results - Fold 2 56
Table 3.9: U-Net with Augmentations Results - Fold 2 57
Table 3.10: UI-VISA Results - Fold 3 57
Table 3.11: U-Net with Augmentations Results - Fold 3 57
Table 3.12: UI-VISA Results - Fold 4 58
Table 3.13: U-Net with Augmentations Results - Fold 4 58

Table 4.1: Number of training snaps per sum category before balancing 65
Table 4.2: Quantitative evaluation results of the proposed 3D region growing segmentation approach on test volume. 73

ACKNOWLEDGEMENTS

My PhD journey would not have existed without my advisors, Dr. Erica Rutter and Dr. Suzanne Sindi.

I still remember the day I walked into Dr. Rutter's office after reading her paper on region growing algorithms for biomedical images. I was someone who had never been drawn to biomedical data, with no idea that this would become my entire research. Dr. Rutter has supported me not only academically but through the personal ups and downs along the way, and her mentorship has shaped not just this work but the kind of researcher I've become. I am equally indebted to Dr. Sindi, who graciously stepped in as my committee chair during my qualifying exam and has been my advisor ever since. I could not be more thankful for her support, including in the moment I broke down right before that exam, and for the insight she has brought to my work. Her guidance has been instrumental in shaping the direction of my research projects.

I would also like to thank my committee members, Dr. Tomas Rube and Dr. Arnold Kim, for always providing me with thoughtful insights and feedback on my research. I have learned so much from Dr. Rube, both as a researcher and as his teaching assistant, and I have long admired his work ethic. I will always be grateful to Dr. Kim for stepping in as a committee member and staying ever since. I am thankful for the many directions he guided me in, the care he gave my research, and the way he encouraged not just the work in front of me but my belief that I could do this at all.

I am also grateful to the mentors who shaped my growth as a researcher beyond UC Merced. At Lawrence Livermore National Laboratory, Brian Gallagher, Cindy Gonzales, and Mary Silva guided me through the Data Science Challenge and showed me what applied research looks like outside the university. Even now, long after the internship ended, they continue to support me, from building my resume to opening doors and networking opportunities. At the Joint Genome Institute, Igor Grigoriev and Asaf Salamov welcomed me into the genomics work with the Tiberius model and trusted me with machine learning at a scale I had never worked at before. I carry what I learned from each of them into everything I do.

On a personal note, I would first and foremost like to thank my grandparents, Surinder Singh and Harjit Kaur, for always having faith in me. I would also like to thank my uncle, Ramanpreet Singh, my aunt, Jasleen Kaur, and my cousin, Daya, for welcoming me into their home when I moved to the US. They stood by me through all my ups and downs,

always believed in me, and supported me throughout this journey. I will be eternally grateful to them for giving me this opportunity, because without them I would not be here. I cannot thank my parents, Gurpreet Singh and Harvinder Kaur, enough for always having faith in me and supporting my dreams since childhood. They never let me believe there was anything in this world I could not do. Even when we are in different countries, I feel them standing right beside me at every step, supporting and guiding me. I am so grateful for my siblings, Sachneet and Sachveer, for the endless calls, for encouraging me every step of the way, and for wiping away my tears from afar even when I had given up. And lastly, I would like to thank my best friend, Ambika. I could not have done this without her. She has been my stress reliever, the person who would stay on the phone with me for hours so I could focus and quiet my anxiety, and the one who could always make me laugh.

ASEES KAUR

510-815-3440 | 1aseeskaur@gmail.com | linkedin.com/in/1aseeskaur | github.com/aseeskaur

Objective

PhD in Applied Mathematics seeking data science or research scientist position. Expert in deep learning, convolutional neural networks, image segmentation, and statistical analysis with strong programming skills in Python, PyTorch, and HPC systems.

Technical Skills

Programming Languages: Python, R, SQL, MATLAB, LaTeX
Machine Learning & AI: PyTorch, TensorFlow, Scikit-learn, CNNs, Deep Learning, Image Segmentation
Data Science & Analytics: Statistical Modeling, Predictive Analytics, Data Visualization, Data Cleaning, Feature Engineering
Tools & Technologies: Git, Linux/Unix, Jupyter Notebooks, HPC, Matplotlib, Seaborn, NumPy, Pandas

Education

University of California, Merced — Merced, CA
PhD Candidate in Applied Mathematics; GPA: 3.77 — Expected Aug 2026

California State University, East Bay — Hayward, CA
Master of Science in Statistics, Data Science Concentration; GPA: 4.00 — May 2020

California State University, East Bay — Hayward, CA
Bachelor of Science in Applied Mathematics; GPA: 3.73 — June 2018

Internship Experience

Graduate Student Intern — May 2025 – July 2025
Joint Genome Institute, Lawrence Berkeley National Lab — Berkeley, CA

- Built gene prediction pipeline using Tiberius model and Python for genomic sequence analysis

- Collaborated in team environment utilizing HPC clusters and Linux systems for computational biology workflows
- Applied machine learning and statistical modeling to improve gene annotation accuracy

Team Lead in Data Science Challenge — July 2023
Lawrence Livermore National Lab — Livermore, CA

- Led cross-functional team applying machine learning and computer programming to develop ML pipeline for 12-lead ECG analysis
- Mentored 4 undergraduate researchers demonstrating leadership and teaching experience in technical communication
- Performed data cleaning, feature engineering, and statistical analysis of medical datasets to support clinical decision-making

Data Analyst Intern — May 2022 – August 2022
Lawrence Berkeley National Lab — Berkeley, CA

- Optimized I/O performance on supercomputers
- Analyzed expansive datasets to identify performance bottlenecks using Python and HPC tools
- Analyzed I/O library performance across computing architectures with strong technical communication skills

Data Analyst — December 2019 – August 2020
United Sikhs — Remote

- Conducted cross-platform data integration
- Streamlined data transfer via PayPal and Salsa Lab
- Proficiently manipulated, analyzed, and presented data using R, Python, and Excel

Data Analyst — December 2018 – May 2020
California State University East Bay — Hayward, CA

- Analyzed student data using Excel, R, and SQL to inform tutoring services and success strategies

- Retrieved and manipulated data from PeopleSoft and EAB databases for predictive improvements in academic support programs

Research Projects

Region Growing with Convolutional Neural Networks (CNNs) — Aug 2022 – Present
University of California, Merced — Merced, CA

- Developed 2D region growing CNN, U-Net integrated region growing, and 3D volumetric CNN pipelines for medical image segmentation using PyTorch and HPC systems
- Trained and validated models on Digital Subtraction Angiography datasets using Git version control for collaborative development
- Combined deep learning with traditional computer vision techniques and applied statistical analysis with data visualization tools (Matplotlib, Seaborn) to validate results

Black Scholes Model — Jan 2021 – May 2021
University of California, Merced — Merced, CA

- Reduced Black Scholes equation to heat equation to predict the dynamic movements of financial market
- Used Crank Nicolson method to solve the Numerical Problem

Detecting Epileptic Seizures using Fractal Geometry and Data Science — Aug 2019 – May 2020
California State University, East Bay — Hayward, CA

- Used fractal geometry and data science methods to analyze seizure detection
- Increased efficiency of data analysis techniques to detect epileptic seizures

Predicting Data Science Salaries — Jan 2019 – May 2019
California State University, East Bay — Hayward, CA

- Predicted salaries for data science jobs using various job skills as predictor variables

- Compared modeling approaches such as linear regression with stepwise selection, LASSO, and random forests
- Used cross-validation to evaluate model performance

Teaching Experience

Teaching Assistant — Aug 2020 – Present
University of California, Merced — Merced, CA

- Taught Calculus, Probability, and Statistics (R and RStudio)
- Responsibilities encompass virtual and in-person discussion design, homework assistance, and assignment/exam grading

Teaching Associate — Aug 2018 – May 2020
California State University, East Bay — Hayward, CA

- Instructed College Algebra (IOR), Probability, and Statistics (TA) courses
- Duties included lesson planning, homework assistance, and grading, with collaboration among instructors, TAs, and undergraduate peer support

Lead Tutor, Student Center for Academic Achievement — Sep 2017 – Jan 2020
California State University, East Bay — Hayward, CA

- Supported upper-division mathematics and statistics students in understanding concepts and problem-solving
- Provided training and mentorship to elevate fellow tutors' performance

Invited Talks

Joint Mathematics Meetings (JMM) — January 2025
Seattle, WA

- "Improving Digital Subtraction Angiography Image Segmentation with CNNs"

Poster Presentations

SIAM Northern California Conference (NCC) — October 2025
Lawrence Berkeley National Lab

- "Training Tiberius for JGI Relevant Genomes"

SIAM Northern California Conference (NCC) — October 2024
Merced, CA

- "Improving Digital Subtraction Angiography Image Segmentation with CNNs"

AWM Workshop at SIAM Annual Meeting — July 2024
Spokane, WA

- "Improving Digital Subtraction Angiography Image Segmentation with CNNs"

SIAM Imaging Science Conference — May 2024
Atlanta, GA

- "Improving Digital Subtraction Angiography Image Segmentation with CNNs"

Predictive Modeling in Biology and Medicine Conference — November 2023
UC Riverside, CA

- "Region Growing Algorithm on Digital Subtraction Angiography Images"

Leadership and Service

Applied Mathematics Graduate Student Council — 2023–2024
UC Merced

SIAM Student Chapter — 2023–2026
UC Merced

ABSTRACT OF THE DISSERTATION

Vascular Image Segmentation Architecture for 2D and 3D Biomedical Images

by Asees Kaur
Doctor of Philosophy in Applied Mathematics

University of California Merced, 2026

Committee Chair: Dr. Erica Rutter

Vascular image segmentation is critical to biomedical image analysis, yet remains challenging due to the complex, thin, branching nature of blood vessels. Accurate segmentation is essential for diagnosing cerebrovascular diseases, surgical planning, and disease monitoring. Most approaches make pixel-wise predictions, which often fail to preserve structural contiguity, producing fragmented vessel segments. Medical imaging datasets are also often small, making it difficult to train deep learning models that require large amounts of data. This thesis introduces three pipelines combining local patch-based prediction with a region growing algorithm to produce accurate, connectivity-preserving vascular segmentations under limited data availability.

First, VISA (Vascular Image Segmentation Architecture), a two-phase pipeline combining local patch-based prediction with a Region Growing Algorithm (RGA), segments 2D Digital Subtraction Angiography (DSA) images while maintaining vascular connectivity. By extracting densely overlapping snaps from each image, VISA generates many training samples from a limited dataset. VISA demonstrates competitive performance against a U-Net baseline, with stronger Dice and clDice scores across most folds, indicating better preservation of topological structure, though at the cost of more computation time due to the iterative region growing process.

Second, UI-VISA (UNet Initialized VISA) replaces VISA's random seed initialization with informed seeds from U-Net predictions, improving accuracy and efficiency. Since these seeds focus on foreground regions, the RGA starts from meaningful vessel areas rather than the entire image, reducing the iterations needed for convergence. UI-VISA outperforms both standalone VISA and U-Net in four of five folds in Dice and clDice, while reducing inference time three to five times compared to VISA.

Third, 3D-VISA extends VISA to three-dimensional volumetric data, applying the same two-phase design to segment dense vascular structures in 3D biomedical images. By extracting overlapping 40x40x40 volumetric snaps centered on every voxel, 3D-VISA generates many training samples from a single annotated volume. As a proof-of-concept

on a synthetic vascular dataset, 3D-VISA achieves near-perfect Dice, Jaccard, and clDice scores, demonstrating the potential for volumetric segmentation under limited data.

Together, these three pipelines demonstrate that combining local patch-based prediction with region growing algorithms is a promising direction for accurate, connectivity-preserving vascular segmentation in both 2D and 3D medical images.

Chapter 1

Introduction

Accurately segmenting biomedical images presents a significant challenge due to their complexity, compounded by the labor-intensive nature of manual segmentation and the issue of insufficient data in the healthcare industry. To tackle these challenges, my work focuses on leveraging machine learning techniques, specifically Convolutional Neural Networks (CNNs), to address data scarcity while preserving the characteristics of biomedical images. Chapter 1 provides essential background on biomedical image segmentation and the machine learning techniques used to segment biomedical images. Building on this foundation, Chapter 2 presents a real-world application of region-growing algorithms applied to Digital Subtraction Angiography (DSA) images. Using cross-validation to evaluate the accuracy of these methods is assessed, offering insights into their efficacy for biomedical image segmentation tasks. Finally, Chapter 3 outlines future directions, including extending the algorithm built in Chapter 2 to three-dimensional biomedical imaging data. Expanding to include more data types, this study aims to enhance biomedical image segmentation techniques, benefiting clinical decision-making and patient care.

1.1 Introduction and Background for Biomedical Image Segmentation

Image segmentation is the process of dividing an image into multiple regions, facilitating a deeper understanding and analysis of digital images [3]. Each pixel in an image is categorized into its respective class through pixel-to-pixel prediction [32]. There

are two broad categories of image segmentation: semantic segmentation and instance segmentation. Semantic segmentation categorizes an image into foreground and background, assigning the same class to all similar objects within the image, thereby grouping together parts belonging to the same category [32]. In contrast to this, instance segmentation not only categorizes individual objects within the foreground but also performs semantic segmentation simultaneously, thereby categorizing each object distinctly. Figure 1.1a displays an example of a microscopy image of Glioblastoma-astrocytoma U373 cells on a polyacrylamide substrate from the ISBI Cell Tracking Dataset [2, 25]. In Figure 1.1b, shows the semantic segmentation of the cell image shown in fig. 1.1a and in fig. 1.1c shows the instance segmentation of the same cell image.



(a) Cell Image (b) Semantic Segmentation (c) Instance Segmentation

Figure 1.1: Example of semantic and instance image segmentation.

Image segmentation is applicable across various image types, with significant implications in biomedical imaging. With advancements in medical imaging techniques such as X-rays, ultrasound, CT scans, and MRI, computer-aided image segmentation can extract specific regions of interest (RoIs) for further evaluation, such as organ identification and tumor detection [12]. Medical imaging is essential in diagnosing illnesses, planning treatments, and monitoring progress. It allows healthcare providers to see inside the body and identify any irregularities. Additionally, analyzing medical images can assist in various tasks such as diagnosing diseases, planning treatments, localization of areas of interest, and even aiding in drug development [30].

Biomedical image segmentation, which involves partitioning images into meaningful segments, is particularly significant in both research and clinical settings [48, 41]. By segmenting medical images, clinicians can identify and isolate regions of interest, such as organs or lesions, from complex CT and MRI scans[58]. This process not only facilitates the extraction of critical information regarding the shape, size, or volume of regions like

tumors but also aids in accurate diagnosis and treatment planning. However, manual segmentation can prove tedious, expensive, and prone to human error, highlighting the pressing need for automated and reliable segmentation techniques to enhance efficiency and precision in medical imaging analysis.

Despite numerous successful approaches, image segmentation remains one of the most challenging aspects of computer vision. This difficulty stems from the challenge of effectively representing features, especially in medical images which often suffer from issues like blur, noise, and low contrast. While traditional segmentation techniques such as edge detection and thresholding have historically been employed in medical imaging, the arrival of deep learning has revolutionized image segmentation across various domains, including biomedicine [63]. Deep learning techniques, particularly convolutional neural networks (CNNs) offer advantages over traditional methods, particularly in handling complex tasks with large datasets. CNNs excel at hierarchical feature representation, making them a prominent research focus in image processing and computer vision, and providing excellent segmentation results even in the presence of image noise, blur, or low contrast. [62, 58]

1.1.1 What are Convolutional Neural Networks?

Convolutional Neural Networks (CNNs) are widely used for image analysis tasks, such as image classification, image segmentation, object detection, facial recognition, and much more. For example, CNNs have achieved remarkable accuracy in image classification tasks, particularly with architectures like AlexNet [27], which significantly advanced the field. In image segmentation, the U-Net [44] architecture has been especially effective for biomedical applications. CNNs are also essential in object detection frameworks like YOLO (You Only Look Once) [42] and facial recognition systems such as FaceNet [47], demonstrating their versatility and powerful performance in various image analysis domains.

CNNs were first introduced in 1980 as Neocognitron that laid some foundational ideas in the context of pattern recognition and image processing. [11]. The concept of convolution, which involves sliding a filter (or kernel) over an input image and computing element-wise multiplications, is central to CNNs. This operation allows the network to extract features from the input data, mimicking the human brain's ability to recognize patterns. In a CNN, the convolutional layer applies multiple filters to the input image, each filter extracting different features. These filters, also known as kernels, are

essentially arrays of learnable parameters that are adjusted during the training process to enhance the network's ability to recognize specific patterns or features. Images, being two-dimensional grids of pixel values, are well-suited for convolutional operations. As the filters slide over the input image, they compute the element-wise product between their weights (parameters) and the corresponding region of the image, producing feature maps that represent different aspects of the input. Overall, the convolutional layer plays a crucial role in feature extraction, enabling CNNs to learn hierarchical representations of the input data, ultimately leading to effective pattern recognition, just as the human brain does. [61]

1.1.2 CNNs for Biomedical image segmentation

Deep Learning, particulary Convolutional Neural Networks(CNNs) plays a crucial role in Biomedical Image Segmentation due to their ability to learn complex features adapt to the variability inherent in medical images [55]. Biomedical images, derived from sources such as CT scans, MRIs, X-rays, and ultrasounds, offer valuable insights into various anatomical structures, including organs, blood vessels, and bones [22].

CNNs excel at automatically learning intricate patterns and representations directly from image data, alleviating the time-consuming process of manual interpretation. Among deep learning methods, CNNs are commonly employed for feature extraction in medical image analysis [22, 13]. Notably, the U-Net architecture has emerged as a prominent choice for biomedical image segmentation [44].

The U-Net architecture comprises of an encoder-decoder network, as illustrated in Figure 1.2. The encoder, located on the contracting path (left side), consists of convolutional layers followed by ReLU activation functions and max-pooling operations that aids in feature extraction. On the other hand, the decoder, situated on the expansive path (right side), aims to upsample feature maps obtained from the encoder to preserve contextual information obtained from the encoder at higher resolutions. Crucially, skip connections (seen as arrows in fig. 1.2) facilitate the transfer of information between corresponding encoder and decoder layers, aiding in precise segmentation [44]. The output U-Net is probability maps of the likelihood of each pixel belonging to a particular class.

U-Net has demonstrated remarkable success in medical image segmentation, attributed not only to its capacity to capture contextual information efficiently but also its

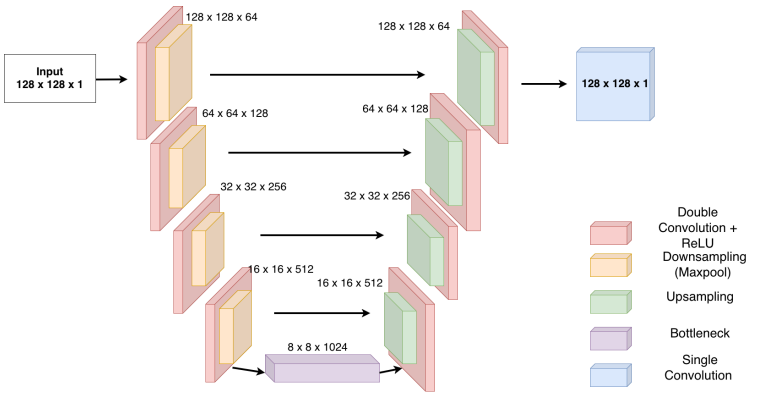


Figure 1.2: Unet architecture

ability to learn from minimal annotated data, often through a tile-based approach [31, 15]. U-Net, while powerful for biomedical image segmentation, faces several challenges. It often produces patchy probability maps , leading to inconsistent and fragmented segmentations that lack smooth and contiguous boundaries. Additionally, U-Net requires extensive annotated training data to generalize effectively, which is labor-intensive and resource-demanding to obtain [44].

Patch-based methods help solve the issue of fragmented segmentation by breaking down large images into smaller, manageable patches. These patches are individual segments of the larger image that the model processes independently, which creates more diverse training data. This approach allows the model to learn detailed features better, leading to smoother and more accurate segmentations. In response to these challenges, my research focuses on a novel patch-based technique, the region growing algorithm [18], to address segmentation complexities in medical images.

1.2 Dissertation Structure

The remainder of this thesis is organized as follows. Chapter 2 introduces VISA (Vascular Image Segmentation Architecture), a two-phase pipeline to address the challenges of segmenting vascular biomedical images. This chapter discusses the training and testing of VISA in detail, compares the quantitative and qualitative results of VISA to a modified U-Net, and discusses the shortcomings and future directions of the approach. Chapter 3 introduces a more informed pipeline UI-VISA (U-Net Initialized Vascular Image Segmentation Architecture), that uses the foreground predictions from the gold standard U-Net as informed seed points for VISA. This chapter discusses the training of both U-Net and VISA architecture and then describes in detail the proposed approach that forms the phase two of the pipeline by using U-Net predictions as informed start point for Region Growing Algorithm. Further, the chapter compares the results of the UI-VISA with standalone VISA and U-Net, and then discusses a more informed approach for future work. Chapter 4 introduces 3D-VISA, an extension of the original VISA pipeline to three-dimensional volumetric data. As most biomedical image data is inherently three-dimensional, this chapter presents a proof of concept of 3D-VISA on a synthetic vascular dataset, demonstrating remarkable quantitative and qualitative results, and discusses evaluating the pipeline on real clinical datasets as future work. Finally, Chapter 5 summarizes the key contributions and findings across all three pipelines and outlines directions for future work.

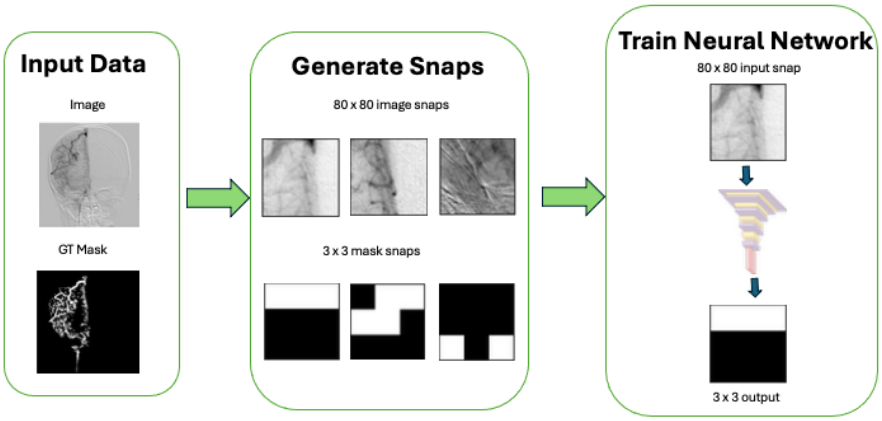
Chapter 2

VISA: Vascular Image Segmentation Architecture for Two-Dimensional Digital Subtraction Angiography Images

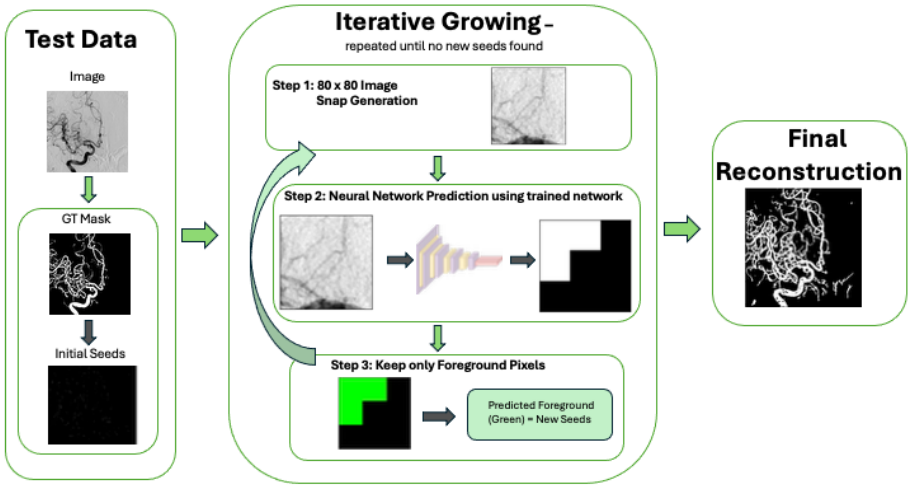
Chapter Overview

This chapter presents a novel segmentation pipeline designed to enhance the connectivity of segmented regions in Digital Subtraction Angiography (DSA) images, which are crucial for visualizing cerebral blood vessels [65]. The pipeline effectively combines the strengths of overlapping region-based methods and the region-growing algorithm to address the inherent limitations of standard U-Net architectures, particularly in segmenting intricate vascular structures while maintaining their connectivity. Connectivity in the context of blood vessel segmentation refers to the ability to represent the vascular network as continuous and uninterrupted structures. This is vital for ensuring that the segmented blood vessels accurately reflect the true anatomy, which is essential for reliable diagnosis and treatment planning. Traditional U-Net architectures, while powerful, often produce patchy probability maps when processing entire images at once, leading to fragmented vessel segments that obscure the true vascular structure.

This chapter presents Vascular Image Segmentation Architecture(VISA), a novel two-phase segmentation pipeline designed to address these challenges, as illustrated in Figure 2.1. In the first phase, shown in Figure 2.1(a), we employ a convolutional neural network (CNN) inspired by the U-Net encoder architecture to predict 3×3 output masks from 80×80 input regions. Rather than processing the entire image as a single unit, we



(a) Vascular Image Segmentation Architecture (VISA) training pipeline



(b) Vascular Image Segmentation Architecture (VISA) testing pipeline: Region Growing Algorithm (RGA)

Figure 2.1: Proposed Vascular Image Segmentation Architecture (VISA) Framework: (a) Training Pipeline Using 80×80 Image Snaps and 3×3 Ground Truth(GT) Mask Snaps for Neural Network Training, and (b) Testing Pipeline with Iterative Seed Expansion and Network Prediction for Vessel Segmentation

extract overlapping regions centered on each pixel— that we call "snaps"—from the DSA images. Each 80×80 snap provides the network with sufficient contextual information to learn relevant features and recognize local patterns in the vascular structures, and the corresponding 3×3 output captures the local neighborhood around each pixel. This approach generates diverse and detailed training data from limited medical images, which is essential for accurately capturing the fine structures of blood vessels. The predicted 3×3 masks provide localized context and serve as seed points for the next phase of the pipeline.

In the second phase of VISA (Figure 2.1(b)), the Region Growing Algorithm (RGA) takes over, using these predictions as seed points to iteratively expand the segmented regions. Importantly, the expansion only occurs in the direction of predicted foreground pixels, which correspond to the blood vessels in the DSA images. At each iteration, the algorithm extracts an 80×80 snap centered on each candidate pixel neighboring the previously identified foreground regions, passes them through the trained network to obtain 3×3 predictions, and continues expanding only around pixels predicted as foreground. This iterative process ensures smooth and contiguous boundaries for the blood vessels, effectively mitigating the boundary artifacts and inconsistencies typically introduced by processing methods that treat pixels independently. By maintaining connectivity throughout the segmentation process, the algorithm ensures that the vascular network is represented as continuous structures rather than fragmented segments. Together, these two phases of VISA produce accurate, cohesive, and clinically valuable segmentations of vascular structures.

This combined approach results in accurate, cohesive, and clinically valuable segmentations of blood vessels. By addressing the limitations of traditional U-Net architectures through localized prediction and connectivity-preserving expansion, this pipeline ensures that the segmented vascular structures are continuous and reflective of true anatomical connections, which is crucial for effective medical imaging analysis and ultimately improves diagnostic and treatment outcomes for patients with cerebrovascular diseases.

2.1 Introduction and Motivation

Cerebrovascular diseases (CVDs) are conditions that affect blood flow and blood vessels in the brain, encompassing aneurysms, strokes, arteriovenous malformations, and

arteriovenous fistulas. Patients with CVDs are at risk of significant morbidity and mortality, making these diseases one of the leading causes of death today [65]. The motivation for this project stems from the challenges faced by stroke patients. A stroke is when blood flow to the brain is blocked or reduced, it leads to permanent neurological damage, disability, or even death[1]. Symptoms of strokes include dizziness, sudden confusion, and weakness in the arms or legs. Strokes are broadly categorized into two types: ischemic and hemorrhagic[10]. Ischemic strokes, the most common type, are caused by the blockage of an artery due to clogged blood vessels[7]. Hemorrhagic strokes occur when a blood vessel in the brain leaks and bleeds into or around the brain.

Even though Computed Tomography (CT) and Magnetic Resonance (MR) imaging techniques are used to detect strokes [19], Digital Subtraction Angiography (DSA) is a particularly valuable imaging method for diagnosing and guiding treatment in cases of strokes and other CVDs [49]. First described in 1935 and digitized in the 1970s [16], DSA is a fluoroscopy technique used in interventional radiology to visualize blood vessels clearly in a bony or dense soft tissue environment [49]. This involves inserting a small catheter, usually in an artery in the leg, and passing it up to the blood vessels into the brain [49]. Images are produced using a contrast medium by subtracting a "pre-contrast image" or mask from subsequent images once the contrast medium has been introduced into the system, hence the term "digital subtraction angiography" [49].

Image segmentation, defined as the process of partitioning a digital image into multiple segments to isolate specific anatomical structures, is essential for clearly visualizing blood vessels in Digital Subtraction Angiography (DSA) images. This extraction allows clinicians to make objective, data-driven decisions regarding vascular health. However, as illustrated in fig. 2.2 manually tracing vessel boundaries is a tedious and error prone process, and is highly susceptible to inter- and intra-observer variability. To address these inconsistencies, machine learning methods—specifically Convolutional Neural Networks (CNNs)—have become the standard for biomedical tasks. Architectures such as the U-Net [44] excel in this domain because they automatically learn the intricate patterns and morphological variations inherent in complex vascular structures.

However, the traditional U-Net has some drawbacks. Since it processes the entire image at once, it can blur the local information of the blood vessels and their connections. This results in patchy probability maps, which are characterized by discontinuous vessel segments and isolated artifacts [23, 15]. In addition to these architectural issues, medical image analysis often faces a shortage of labeled training data. To counteract this problem, patch-based methods can be used to create many training image-label pairs from a single

image. This process of creating patches helps preserve localized information and handles different image resolutions. Most importantly, it addresses the data scarcity problem by generating a larger dataset from limited original samples.

This project aims to use an existing patch-based method known as Region Growing Algorithm (RGA) [18] on DSA images. This approach preserves local information, handles different image resolutions, and addresses the issue of data sparcity by creating more training samples from a single image. We demonstrate that this approach offers the potential to improve segmentation accuracy by focusing on local regions of interest within the images, thus enhancing the overall performance of automated segmentation techniques in the context of cerebrovascular diseases.

The remainder of this chapter is organized as follows: Section 2.2 describes the DSA dataset (Section 2.2.1), preprocessing and training steps (Section 2.2.2) for VISA, and Section 2.2.3 explains the region-growing algorithm. Section 2.3 presents quantitative and qualitative results comparing our architecture to standard U-Net segmentation and Section 2.4 discusses challenges and future directions of this project.

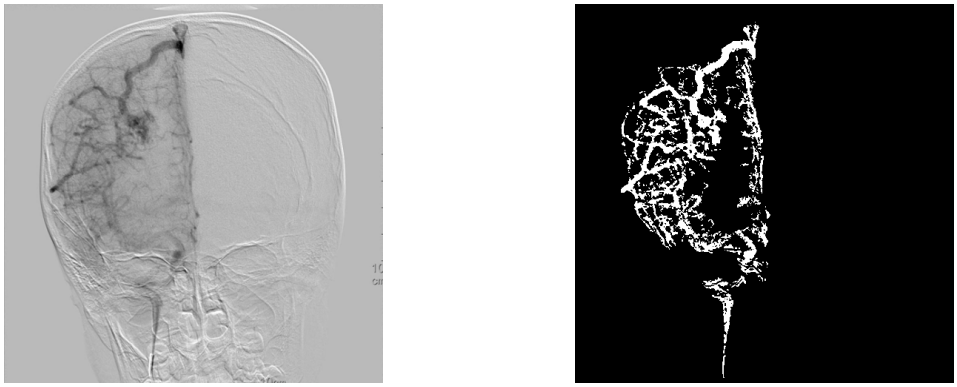
2.2 Methods

In this section, we describe the full pipeline VISA for training and evaluating the RGA on DSA images, as illustrated in Figure 2.1. We begin by introducing the dataset and cross-validation strategy used to ensure fair and generalizable evaluation. We then detail the training pipeline (Figure 2.1(a)), which encompasses snap extraction and class balancing, followed by the CNN architecture and training procedure. Finally, testing pipeline (Figure 2.1(b)), which applies a Region Growing Algorithm (RGA) to iteratively generate full-image vessel segmentations from the trained model.

2.2.1 Data

The dataset consists of 26 DSA images and their manually traced masks [65] (see Figure 2.2 for an example of a typical DSA image and its corresponding mask) from a group of patients. There is only one DSA image per patient and we use the manually traced mask as the ground truth(GT). The contrast agent was iodixanol (Visipaque, General Electric), administered by intra-arterial power injection at 5 ml/s for a total amount of 8 ml into the cervical carotid arteries, or at 3 ml/s for a total amount of 5 ml

into the cervical vertebral arteries. The patients that were included in this study had either pathological abnormalities from arteriovenous malformations, arteriovenous fistula, and cerebrovascular aneurysm, or normal DSA findings [65].



(a) DSA Image (b) DSA Mask

Figure 2.2: Left (a) is an example of a typical DSA Image, acquired during angiography, and on the right (b) is its corresponding manually traced mask we use as ground truth.

In what follows we will detail the 5-fold cross-validation method used in this work. The 26 image-mask pairs were randomly split into 5 folds, each containing 21 pairs for training and 5 pairs for testing. The 26 image-mask pairs were randomly split into 5 folds. Since 26 does not divide evenly by 5, one fold contains 6 images for testing while the remaining four folds each contain 5, resulting in 20 or 21 images for training per fold respectively. The training set was further divided into training and validation subsets using an 80-20 ratio. The model was trained on 80% of the training data and validated on the remaining 20%, with the validation set used to select the best model weights and determine the threshold. The trained models were then evaluated on the corresponding held-out test sets using the region growing algorithm described in Section 2.2.3. After splitting the dataset into five folds, each image–mask pair in the training set was balanced as described in Section 2.2.2, ensuring that all classes were represented evenly before snaps from all training images were pooled into a single training set.

2.2.2 Training

Having described the dataset and cross-validation strategy, we now detail the training pipeline illustrated in Figure 2.1(a). This involves three steps: extracting overlapping snaps from each image, addressing class imbalance through down-sampling, and training the CNN on the resulting balanced dataset.

Data Preparation and Snap Extraction

Each image and its corresponding mask is of size 512×512 pixels. We process each image in a patch-based fashion — by extracting small rectangular regions centered on every pixel in the image. We call these regions "snaps" to distinguish them from the non-overlapping "patches" used in standard tiling approaches. Unlike patches, snaps overlap heavily, which is central to our method's ability to preserve vascular connectivity. For pre-processing, we normalized the pixel intensities by dividing each pixel value by the maximum pixel value in that image. Medical imaging datasets are often small, and models trained on limited data risk overfitting to the training set. To increase the diversity of training samples available to the model, we applied data augmentation to each training image and its corresponding mask. Specifically, we applied random 90° rotations and horizontal and vertical flips, each with a probability of 0.5. Each transformation that was applied produced an additional image-mask pair, meaning each original training image could yield up to four versions (the original plus up to three augmented copies). To handle boundary effects during data extraction, we apply zero-padding of 40 pixels on all four sides of both the images and masks. This increases the dimensions from 512×512 to 592×592, ensuring that regions centered near the borders of the original image remain fully defined after padding. Next we construct snaps, let (i, j) denote the coordinates of a pixel in the original image, where $i, j \in \{0, \ldots, 511\}$ under zero-based indexing. The corresponding location in the padded image is given by

$$(i + 40,\ j + 40).$$

For each such pixel, we extract two aligned regions that we refer to as *snaps*: an 80×80 snap from each padded image, centered at $(i + 40, j + 40)$, and the corresponding 3×3 snap from each padded mask, aligned at the same center, this process is show in Figure 2.3 and Figure 2.4. Each snap provides a localized view of its surroundings, capturing both the pixel of interest and its immediate context.

This procedure converts each original image–mask pair into a collection of paired snaps. Specifically, for each image of size $(512, 512)$, a total of 262,144 image–label pairs are generated—one for every pixel in the image—where each 80×80 input snap corresponds to a local 3×3 output snap. The training dataset is constructed from these snaps. Figure 2.3 illustrates the creation of 80×80 snaps from a training image. The left panel shows the original image, the middle panel shows a zoomed-in area from the original image, and the right panel shows six examples of the corresponding 80×80 snaps for that zoomed-in section. Similarly, we constructed the testing dataset by forming snaps from the testing images.

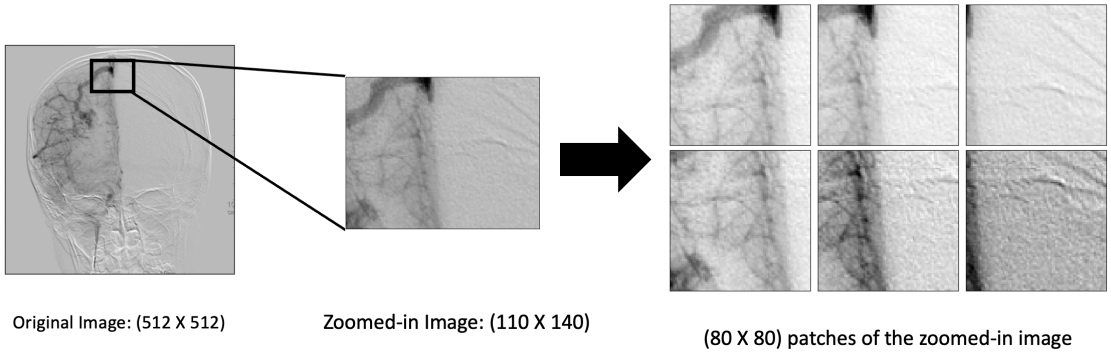


Figure 2.3: Example illustrating the creation of snaps for training images. The left panel displays a sample training image, while the middle panel offers a close-up view of a referenced section from the left image. The right panel demonstrates the generation of 80 x 80 snaps from the zoomed-in section shown in the middle panel.

The training masks were composed by taking a $(3, 3)$ snap around each pixel for each corresponding mask. A mask is an image the same size as the original image, where each pixel indicates the class or object of the corresponding pixel in the original image. Masks are used in image segmentation tasks to classify each pixel in the image. This 3×3 snap captures the local neighborhood around each pixel, providing context for the central pixel, which is crucial for segmentation tasks. An example of how this was implemented is shown in Figure 2.4. The left panel displays a sample training mask, the middle panel offers a close-up view of a referenced section from the mask in the left panel, and the right panel demonstrates the generation of 3×3 snaps from the zoomed-in section shown in the middle panel.

Addressing Class Imbalance

It is evident from the DSA image and mask shown in Figure 2.2 that there are more background pixels (zeros) than foreground pixels (ones) in the image. Because the images represent blood vessels, the background pixels vastly outnumber the foreground pixels. This class imbalance problem can influence what is learned and prioritized in the network, and thus we balanced the data. Since directly assessing the class imbalance from the images could be complex, we opted to analyze the training masks. Each mask snap is of size 3×3 and contains only binary values (0 for background, 1 for foreground). For each snap, we compute the sum of its entries, which ranges from 0 to 9. A value of 0 indicates that the snap is entirely background, while a value of 9 indicates

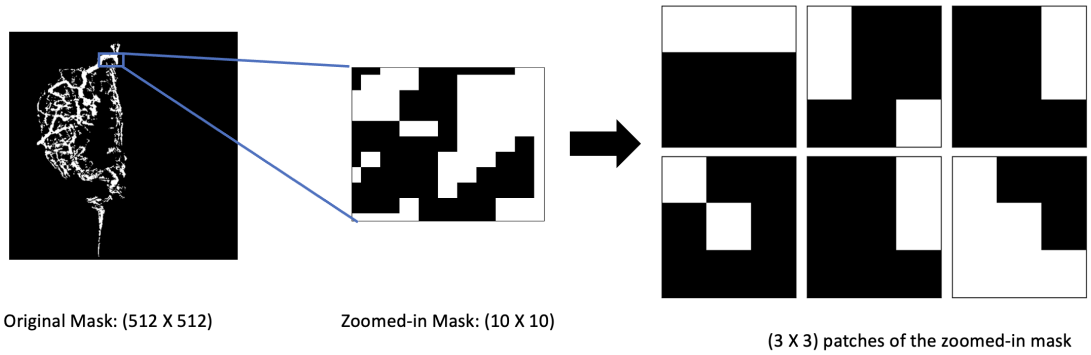


Figure 2.4: Example illustrating the creation of snaps for training masks. The left panel displays a sample training mask, while the middle panel offers a close-up view of a referenced section from mask in the left panel. The right panel demonstrates the generation of 3×3 snaps from the zoomed-in section shown in the middle panel.

that the snap is entirely foreground.

Sum	Number of snaps per sum
0	235316
1	3803
2	3209
3	3212
4	2404
5	2186
6	2634
7	2131
8	2018
9	5231

Table 2.1: Distribution of Training Mask Snap Sums: Table showing the counts of training image/mask patches for each sum, from 0 to 9, highlighting a predominance of background patches (sum 0) and the fewest patches at sum 8.

Table 2.1 shows the distribution of sums of training mask snaps, ranging from 0 to 9 in the left column, with the right column showing the number of training image-mask snaps for each corresponding sum for one of the training images. The vast majority of patches

belong to the category of sum 0, which means most snaps are from the background of the image. Analyzing the table further, we notice that the lowest number of snaps belong to sum 8, with approximately 2,000 snaps. To mitigate this class imbalance issue, we down-sampled the data. This means we deliberately reduced the number of training samples by selecting a subset of snaps, ensuring a more balanced representation of foreground and background classes. We identified the class with the fewest samples (in our case, sum 8 with approximately 2,000 snaps) and randomly sampled 2,000 snaps from each category, along with their corresponding 80×80 image snaps. By doing so, we ensured that the model receives sufficient training data from each class to learn effectively without being biased towards the majority class.

Architecture and Training

Having prepared and balanced our training data through the procedures described above, we now describe the neural network architecture used for the vessel segmentation approach.

A 25-layer Neural Network that resembles the encoder part of U-Net[44] was used; there are 5 double convolutional layers activated by ReLu and each double convolutional layer is followed by a maxpooling layer, the last double convolution layer is followed by a single convolution to reshape into (3,3,1) - where 1 represents if a pixel is in foreground or background. The architecture is exhibited in Figure 2.5. This model was trained on the training data created earlier in the data preprocessing step. During training, the model was optimized using the Adam optimizer [24] and the Binary Cross Entropy loss function. Training was conducted on a GPU available on the MERCED Cluster. The training process spanned 50 epochs, with a batch size of 50 images image-label pairs for the training data. To increase the diversity of training data seen by the model at each epoch, the balanced snap extraction procedure described in Section 2.2 was re-run at the start of every epoch. This means a different random subset of snaps was sampled from each category for each epoch, rather than reusing the same fixed set of patches throughout training.

For comparison purposes, we also implemented a modified U-Net model shown in Figure 2.6 following the architecture and data preparation method described by Zhang et al. [65]. For U-Net training, we extracted 128×128 pixel image patches with randomly varying overlap from the original 512×512 pixel DSA images, following their methodology. Each 512×512 pixel image yielded approximately 200 patches. The U-Net was trained to predict full 128×128 segmentation masks from these 128×128

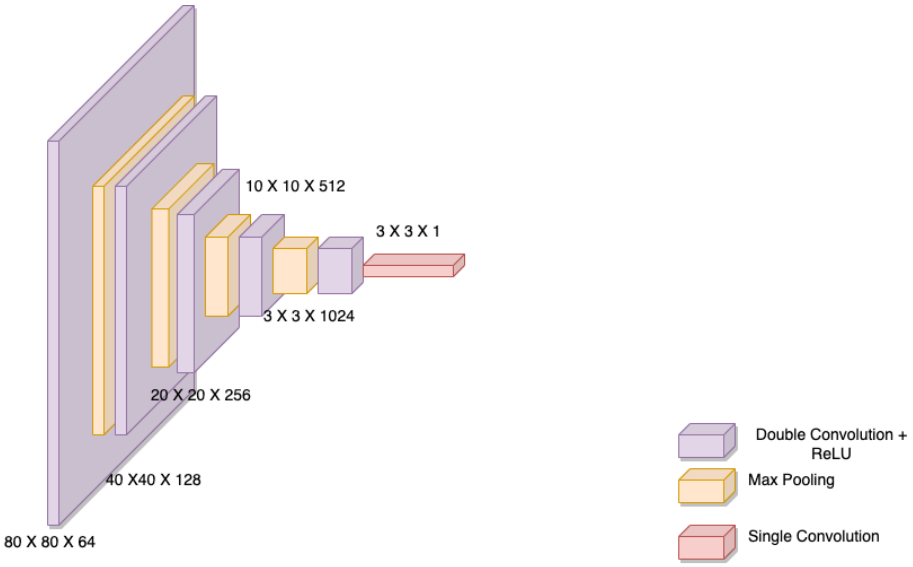


Figure 2.5: Network Architecture: Convolutional Neural Network (CNN) architecture consisting of layers with double convolution and ReLU (purple), max pooling (yellow), and a final single convolution (red). The network takes an $80 \times 80 \times 1$ grayscale snap as input. The dimensions labeled on each layer reflect the output of that layer, starting from $80 \times 80 \times 64$ after the first double convolution and ending at $3 \times 3 \times 1$.

input patches. The implemented U-Net follows a standard encoder–decoder architecture with skip connections. The encoder consists of four downsampling stages, each composed of two 3×3 convolutional layers with ReLU activation and dropout, followed by 2×2 max pooling, progressively increasing the number of feature channels from 64 to 512. A bottleneck layer with 1024 feature channels connects the encoder and decoder. The decoder mirrors the encoder structure using transposed convolutions for upsampling, concatenation with corresponding encoder feature maps via skip connections, and double convolutional blocks to refine spatial detail. A final 1×1 convolution produces a single-channel segmentation output. Both the RGC model and U-Net used the same 5-fold cross-validation splits (21 images for training, 5 for testing per fold) to ensure a fair comparison. The key difference between the two approaches is in both data preparation and prediction strategy: RGC uses densely overlapping 80×80 snaps with 3×3 outputs combined with iterative region growing, while U-Net uses randomly extracted 128×128 patches with full-size 128×128 outputs processed independently.

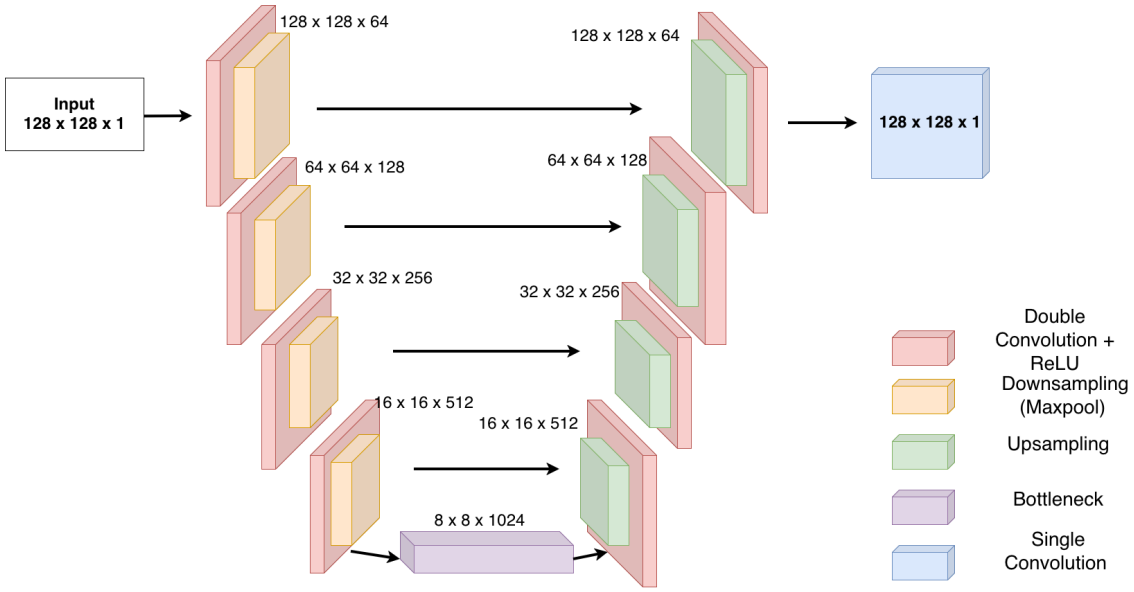


Figure 2.6: The modified U-Net architecture used

Convolutional Neural Networks can be designed with various input-output size relationships depending on the specific architecture and segmentation strategy [44]. In addition, independent pixel-wise predictions often result in spatial inconsistencies, producing fragmented or discontinuous vessel segments. The 80×80 input snap provides sufficient contextual information for the network to learn relevant features and recognize

local patterns in vascular structures. The output size 3×3 was specifically chosen to predict a small neighborhood of pixels rather than a single isolated pixel. This approach ensures that each predicted pixel is evaluated in relation to its immediate neighbors, which helps maintain local spatial consistency and reduces artifacts such as jagged edges or isolated predictions. By predicting contiguous 3×3 regions, the network promotes spatial coherence within local neighborhoods, which is important for maintaining smooth vessel boundaries. This design choice is particularly important for cerebrovascular segmentation, where preserving vascular connectivity is crucial. Blood vessels form continuous, tree-like networks with well-defined topological structure. The 3×3 output ensures that neighboring predictions are derived from overlapping contextual information, helping to maintain the connectivity and contiguity of the segmented vascular structures. While the extreme input-output size ratio does help with feature extraction, the primary advantage of the Region Growing Connectivity approach is its ability to preserve the structural integrity of the vascular structures.

2.2.3 Testing

Having trained the model using the procedure described above, we now detail the testing pipeline, which is the phase 2 of VISA as illustrated in Figure 2.1(b). This section describes the Region Growing Algorithm (RGA) used to generate full-image segmentation predictions from the trained network, followed by the post-processing steps applied to convert the network's probability outputs into final binary masks suitable for evaluation.

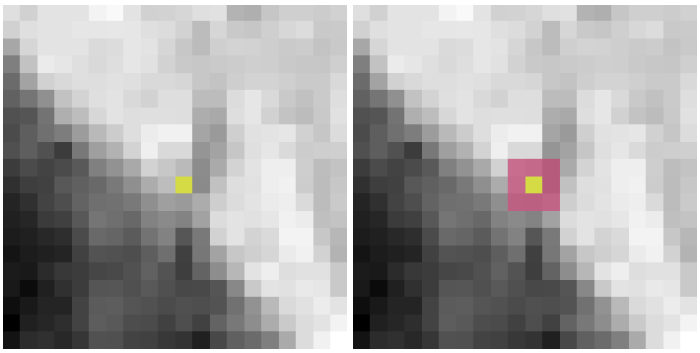
Region Growing Algorithm

In the testing phase of our architecture, we utilize the Region Growing Algorithm(RGA) for efficient segmentation. Unlike the training phase, where the focus is on parameter optimization, testing aims for accurate results within a reasonable timeframe. Region growing is a segmentation technique that constructs regions by starting from a set of selected seed pixels and iteratively incorporating neighboring pixels that satisfy a similarity criterion. In our approach, an 80×80 image snap is extracted around each seed pixel and passed through the trained network to predict a 3×3 segmentation mask centered at the seed location. The foreground predictions within this mask are added to the growing segmented region. The algorithm then examines the neighboring pixels of the newly added foreground pixels, extracts the corresponding snaps, and repeats the prediction process. This iterative procedure continues until no

additional pixels satisfy the inclusion criteria, at which point the region growing process terminates. This design addresses both computational efficiency and structural coherence in vessel segmentation. A straightforward pixel-wise approach would require generating predictions for every pixel location in a DSA image. In one of our images, this approach generated more than 260,000 snaps that would need to be processed through the network. Even with sub-sampling strategies, such as predicting every third pixel, the computational cost would still be too high to be practical. This iterative approach, expanding around foreground pixels, ensures contiguous regions without processing the entire image. This targeted strategy significantly reduces the processing time compared to comprehensive image analysis. This algorithm grows iteratively around each foreground pixel towards the direction of foreground pixels to ensure the contiguity of the predicted region.

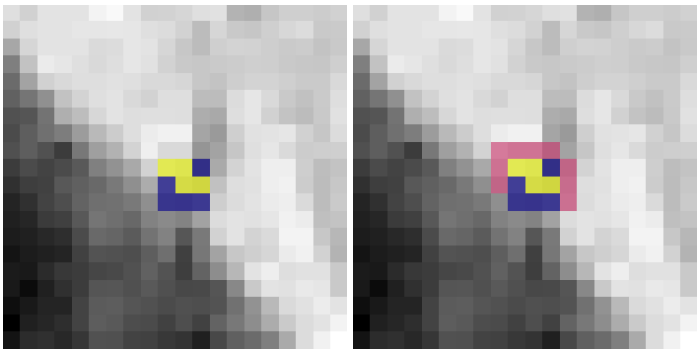
Figure 2.7 shows the working of the region growing algorithm in detailed steps, representing one full iteration of the algorithm. In all of the images, yellow represents predicted foreground pixels, purple represents predicted background pixels, and red represents the pixels that will be predicted in the next iteration of the algorithm. For example, Figure 2.7a shows one foreground pixel on the testing image in yellow. Figure 2.7b shows the candidate pixels that will be predicted in the next iteration. To determine these candidates, we extract an 80×80 snap around each candidate pixel and pass it through the trained network. The network outputs a 3×3 probability mask for each candidate pixel, indicating the likelihood of each pixel being classified as foreground or background. The algorithm makes a prediction based on the trained model, which can be seen in Figure 2.7c, which shows the predictions from the above candidate pixels. The pixels in yellow were categorized as foreground and the pixels in purple were categorized as the background. To avoid unnecessary computation, the algorithm only expands in directions of foreground pixels and discards the background ones. Thus, in the next iteration of the algorithm, the pixels that will be predicted into foreground and background (Figure 2.7d, red), only surround the centroids which were predicted as foreground in the previous iteration (Figure 2.7c, yellow). We can see in Figure 2.7d that the next set of candidate pixels in red are either adjacent or diagonal to the foreground pixels. The resulting probability outputs are then thresholded to produce binary predictions, as described in Section 2.2.3.

In practice, however, we do not start with one candidate initial pixel, we instead start with many to ensure computational speed. Instead, we start with 500 initial randomly selected pixels in the image. We keep track of the indices whose prediction is 1 i.e. the foreground pixels and only those indices go into the next iteration and the others are removed. We follow this procedure until there are no new indices are left to run.



(a) Starting with a single initial foreground pixel in yellow

(b) Next candidate pixels to pass through the trained network in red



(c) Foreground(yellow) and background(purple) predicted pixels

(d) Next iteration of candidate pixels(red) to pass through the network

Figure 2.7: Example illustrating the region growing method. The process begins with a single initial foreground pixel (yellow) as shown in the first panel. The second panel identifies the next candidate pixels (red) that will be evaluated by the trained network. The third panel shows the predicted classification of pixels into foreground (yellow) and background (purple). Finally, the fourth panel highlights the next set of candidate pixels (red) for the subsequent iteration of the network evaluation. Yellow indicates foreground pixels, purple indicates background pixels, and red indicates the next candidate pixels for classification.

Algorithm 1 Region Growing Algorithm

Require: Trained network f, test image I, threshold τ
Ensure: Predicted segmentation mask $\hat{Y}$

Normalize I by dividing by its maximum pixel value
Pad I with 40 zeros on all sides to obtain I_{pad}
Initialize $\hat{Y}(i,j) \leftarrow 0$ for all pixels (i,j)
Initialize pixel score dictionary $\mathcal{P} \leftarrow \{\}$
Initialize visited set $S \leftarrow \emptyset$
Sample 500 random pixels from I to form initial candidate set $\mathcal{C}_0$
$t \leftarrow 0$
while $\mathcal{C}_t \neq \emptyset$ **do**
    for all $(i,j) \in \mathcal{C}_t$ **do**
        Extract 80×80 snap $\mathcal{S}(i,j)$ from I_{pad} centered at (i,j)
        Obtain 3×3 prediction mask $P \leftarrow f(\mathcal{S}(i,j))$
        for all $(r,c) \in P$ **do**
            Let $(x,y) \leftarrow (i+r-1, j+c-1)$
            Append $P[r,c]$ to $\mathcal{P}[(x,y)]$
        end for
        $S \leftarrow S \cup \{(i,j)\}$
    end for
    $\mathcal{F}_t \leftarrow \{(i,j) \in \mathcal{P} : \text{mean}(\mathcal{P}[(i,j)]) \geq \tau\}$
    $\mathcal{C}_{t+1} \leftarrow \{(i,j) \notin S : (i,j) \text{ in } \mathcal{F}_t\}$
    $t \leftarrow t+1$
end while
Reconstruct mask:
for all $(i,j) \in \mathcal{P}$ **do**
    if $\text{mean}(\mathcal{P}[(i,j)]) \geq \tau$ **then**
        $\hat{Y}(i,j) \leftarrow 1$
    end if
end for
Remove padding by cropping rows and columns by 40 of $\hat{Y}$ to recover the original 512×512 mask
return $\hat{Y}$

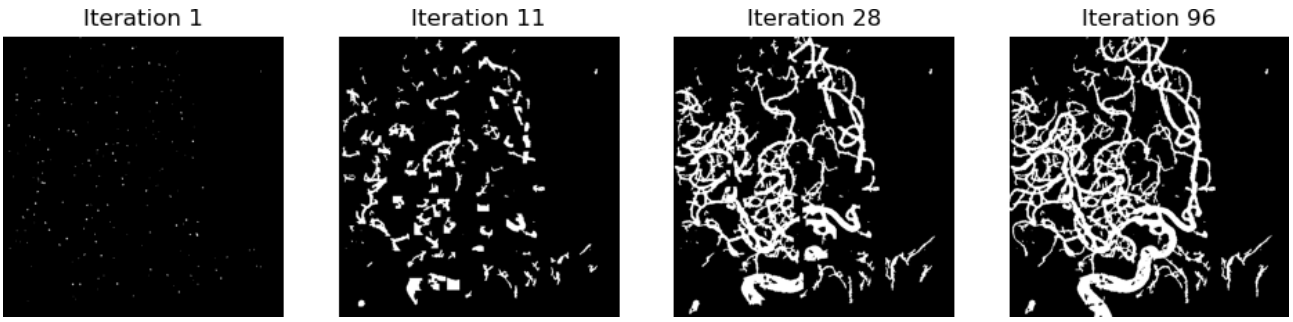


Figure 2.8: Reconstruction process of the predicted test image. The series of subplots illustrates the progressive refinement and development of the image as the algorithm iteratively updates the predictions.

Figure 2.8 displays the construction of the predicted test image.

Post-processing

After training both the VISA and U-Net architectures using the procedures described above, we evaluated their performance on the test sets from each of the five folds. For the VISA, we applied the region growing algorithm described in Section 2.2.3 to generate full-image predictions, starting with 500 randomly selected initial seed pixels and iteratively expanding around foreground predictions. For the U-Net model, predictions were generated by extracting 128×128 patches from each test image and combining the resulting segmentation masks. We then compared the segmentation results from both methods through qualitative visualization and quantitative metrics to assess their relative performance in capturing vascular structures while maintaining connectivity. In biomedical image segmentation, Dice score has become the most popular measures of evaluation. The Dice score is calculated as:

$$\text{Dice Score(A,B)} = \frac{2|A \cap B|}{|A| + |B|} \tag{2.1}$$

The closer the score is to 100%, the more similar the predictions are to the original mask.

While the Dice score measures the overall overlap between the predicted segmentation and the ground truth, it does not explicitly account for the connectivity or topology of thin, elongated structures such as blood vessels. As a result, segmentations with similar Dice scores may differ substantially in terms of vascular continuity, with

breaks or disconnections that are clinically relevant. To address this limitation, we also evaluate segmentation performance using centerline Dice (clDice) [52], a metric specifically designed for vascular structures. The clDice metric assesses how well the centerlines of the predicted and ground truth segmentations align with each other, thereby emphasizing connectivity and structural correctness rather than area overlap alone. It is computed by first extracting the centerlines (skeletons) of both the predicted segmentation and the ground truth mask. The clDice score is then defined as the harmonic mean of the fraction of predicted centerline pixels lying within the ground truth segmentation and the fraction of ground truth centerline pixels lying within the predicted segmentation and is calculated as:

$$\text{clDice(skeleton)} = \frac{2 \times |\text{Skeleton}_{\text{seg}} \cap \text{Skeleton}_{\text{gt}}|}{|\text{Skeleton}_{\text{seg}}| + |\text{Skeleton}_{\text{gt}}|} \tag{2.2}$$

A higher clDice value indicates better preservation of vessel continuity and topology. Here, $\text{Skeleton}_{\text{seg}}$ represents the skeleton of the segmented image, and $\text{Skeleton}_{\text{gt}}$ represents the skeleton of the ground truth mask. Skeletonization is a process that reduces vascular structures to their medial axes while maintaining their connectivity.

To convert the network's continuous probability outputs into binary segmentation masks, we first averaged the predicted probabilities across all overlapping 3×3 predictions for each pixel, since each pixel may be predicted multiple times due to the overlapping nature of the snaps. The averaged probability was then thresholded to produce the final binary mask. Pixels with an averaged predicted probability greater than 0.5 were classified as foreground, while the remainder were classified as background. This threshold was selected based on experiments conducted on the validation set, where values of 0.1, 0.2, 0.3, 0.4, 0.5, 0.6, 0.7, 0.8, 0.9, and 1.0 were evaluated using the average Dice score computed against the ground truth masks. Thresholds between 0.3 and 0.5 achieved the highest average Dice score and were therefore used for all final evaluations on the test set.

2.3 Results and Analysis

Figure 2.9 shows a qualitative comparison between U-Net and VISA predictions across three representative examples. The images use a color-coded scheme to visualize prediction accuracy: white represents true positives (correctly identified vessel pixels), black represents true negatives (correctly identified background), red represents false

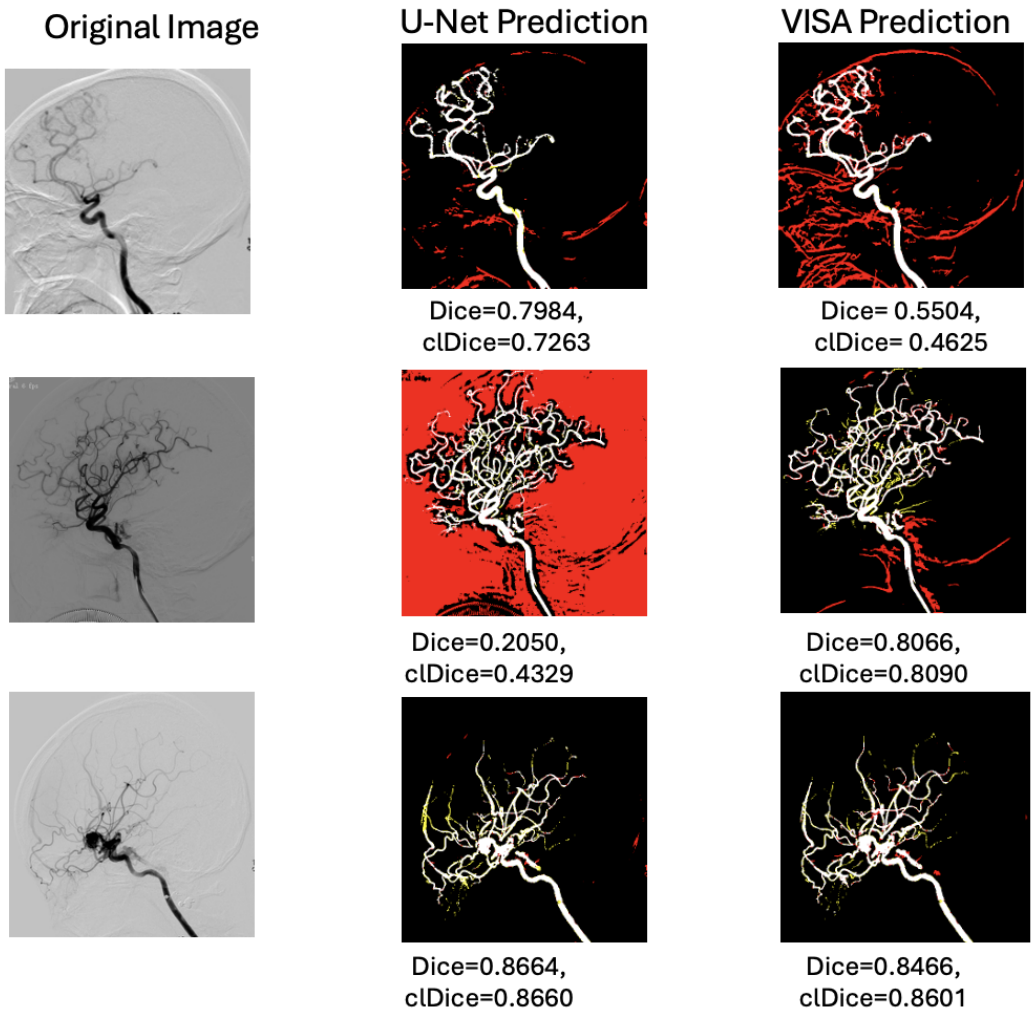


Figure 2.9: Qualitative comparison between U-Net and VISA predictions across three representative test cases. Each row shows an original DSA image (left), U-Net prediction (middle), and VISA prediction (right), with their respective Dice and clDice scores displayed below each prediction. The color-coded scheme indicates: white (true positives), black (true negatives), red (false positives), and yellow (false negatives). Row 1 (Image 21, Fold 4): U-Net performs better. Row 2 (Image 08, Fold 2): VISA performs better. Row 3 (Image 20, Fold 1): Both methods perform similarly.

positives (background incorrectly classified as vessel), and yellow represents false negatives (vessel pixels incorrectly classified as background). In the first row, we observe a case where U-Net achieves slightly better performance, with scores of 0.7984 and 0.7263 for Dice and clDice respectively, compared to VISA's 0.5504 and 0.4625. The second row demonstrates a case where VISA significantly outperforms U-Net, achieving Dice and clDice scores of 0.8066 and 0.8090 compared to U-Net's 0.2050 and 0.4329. The third row shows a case where both methods perform similarly, with similar Dice scores (VISA: 0.8466, U-Net: 0.8664).

Across all 26 images, VISA demonstrates superior overall performance without augmentations, achieving an average Dice of 0.7755 and clDice of 0.7580, and with augmentations, an average Dice of 0.7803 and clDice of 0.7602. In comparison, U-Net achieves an average Dice of 0.7565 and clDice of 0.7222 without augmentations, and 0.7789 and 0.7584 respectively with augmentations. Upon examining the color-coded results, the prevalence of yellow regions (false negatives) versus red regions (false positives) varies between methods and images. In general, both algorithms show some tendency to under-predict vessel structures in certain regions while over-predicting in others. The balance between false positives and false negatives is important because it reflects the trade-off between sensitivity (detecting all actual vessel structures) and specificity (correctly identifying background regions).

Table 2.2 summarizes the cross-validation results for both VISA and U-Net models across all five folds without any augmentations. The VISA method consistently achieves higher average Dice scores across most folds, ranging from 0.7438 to 0.8119. Similarly, VISA demonstrates more consistent clDice scores, ranging from 0.7351 to 0.8049, while U-Net clDice scores vary between 0.6382 and 0.8054. Notably, U-Net's performance in Fold 2 shows significant degradation with a Dice score of 0.6382 ± 0.0464, indicating challenges with specific cases in that fold. It is worth noting that the without-augmentation condition refers to models trained without any data augmentation or resampling for both VISA and U-Net. The with-augmentation condition differs between the two methods: for U-Net, augmentations consist of the geometric transformations described in [65], while for VISA, augmentations include the same geometric transformations alongside epoch-wise resampling of training snaps, in which a different random subset of snaps is drawn from each class at every epoch.

Table 2.3 shows the results when augmentations are applied during training. With augmentation and resampling, U-Net outperforms VISA in Folds 1, 3, and 4, while VISA achieves higher scores in Folds 0 and 2. Notably, VISA demonstrates more consistent

Table 2.2: Summary of Cross-Validation Results for U-Net and VISA Models without any augmentations

	U-Net		VISA	
Fold	**Dice**	**clDice**	**Dice**	**clDice**
0	0.8054 ± 0.0609	0.7747 ± 0.0920	$\mathbf{0.8119 \pm 0.0333}$	$\mathbf{0.8049 \pm 0.0686}$
1	$\mathbf{0.7979 \pm 0.0585}$	$\mathbf{0.7742 \pm 0.0662}$	0.7917 ± 0.6080	0.7726 ± 0.0593
2	0.6382 ± 0.0464	0.5796 ± 0.0646	$\mathbf{0.7506 \pm 0.0435}$	$\mathbf{0.7383 \pm 0.0494}$
3	$\mathbf{0.7783 \pm 0.0756}$	$\mathbf{0.7603 \pm 0.0646}$	0.7438 ± 0.0676	0.7351 ± 0.0484
4	0.7446 ± 0.0746	0.7098 ± 0.0835	$\mathbf{0.7590 \pm 0.1102}$	$\mathbf{0.7428 \pm 0.1317}$

performance across folds, with lower variance in Dice scores compared to U-Net, which shows higher variability particularly in Fold 2 (0.6682 ± 0.2381). The clDice scores in Table 2.2 and Table 2.3 suggest that VISA generally maintains better topological structure and connectivity of the vascular networks compared to U-Net, particularly in cases where U-Net performance degrades.

Table 2.3: Summary of Cross-Validation Results for U-Net (aug) and VISA (aug + resampling) Models

	U-Net (aug)		VISA (aug + resamp)	
Fold	**Dice**	**clDice**	**Dice**	**clDice**
0	0.8048 ± 0.0708	0.7632 ± 0.1070	$\mathbf{0.8126 \pm 0.0290}$	$\mathbf{0.7991 \pm 0.0597}$
1	$\mathbf{0.8137 \pm 0.0485}$	$\mathbf{0.7925 \pm 0.0591}$	0.8133 ± 0.0431	0.7909 ± 0.0494
2	0.6682 ± 0.2381	0.6866 ± 0.1479	$\mathbf{0.7433 \pm 0.0406}$	$\mathbf{0.7178 \pm 0.0569}$
3	$\mathbf{0.7882 \pm 0.0696}$	$\mathbf{0.7719 \pm 0.0595}$	0.7795 ± 0.0692	0.7714 ± 0.0466
4	$\mathbf{0.8089 \pm 0.0484}$	$\mathbf{0.7817 \pm 0.0589}$	0.7585 ± 0.1347	0.7358 ± 0.1604

It is important to note that our U-Net implementation follows the architecture and training procedure described by Zhang et al. [65]. However, there is a key difference in how metrics are calculated. Zhang et al. use a relaxed evaluation criterion: for each pixel predicted as part of a blood vessel, if any pixel in a 3×3 window centered at that position in the ground truth is on a blood vessel, they consider it a correct prediction. This more lenient approach results in higher reported metrics in their paper (average Dice coefficient of 0.8268 ± 0.052).

Table 2.4: Summary of Cross-Validation Results with Relaxed 3×3 Window Evaluation for U-Net and VISA Models without augmentations

	U-Net		VISA	
Fold	**Dice**	**clDice**	**Dice**	**clDice**
0	$\mathbf{0.8351 \pm 0.0731}$	0.8325 ± 0.1159	0.8328 ± 0.0371	$\mathbf{0.8407 \pm 0.0816}$
1	$\mathbf{0.8409 \pm 0.0486}$	$\mathbf{0.8321 \pm 0.0505}$	0.8220 ± 0.0595	0.8184 ± 0.0574
2	0.7417 ± 0.1056	0.7146 ± 0.1657	$\mathbf{0.8183 \pm 0.0448}$	$\mathbf{0.8310 \pm 0.0622}$
3	$\mathbf{0.8028 \pm 0.0652}$	0.7934 ± 0.0598	0.7902 ± 0.0548	$\mathbf{0.8014 \pm 0.0543}$
4	0.8256 ± 0.0976	0.8340 ± 0.1149	$\mathbf{0.8672 \pm 0.0420}$	$\mathbf{0.8968 \pm 0.0429}$

Table 2.5: Summary of Cross-Validation Results with Relaxed 3×3 Window Evaluation for U-Net (aug) and VISA (aug + resampling) Models

	U-Net (aug)		VISA (aug + resamp)	
Fold	**Dice**	**clDice**	**Dice**	**clDice**
0	$\mathbf{0.8392 \pm 0.0847}$	$\mathbf{0.8344 \pm 0.1294}$	0.8304 ± 0.0359	0.8298 ± 0.0730
1	$\mathbf{0.8545 \pm 0.0407}$	$\mathbf{0.8504 \pm 0.0408}$	0.8453 ± 0.0382	0.8343 ± 0.0404
2	$\mathbf{0.8196 \pm 0.0636}$	$\mathbf{0.8240 \pm 0.1105}$	0.8030 ± 0.0529	0.7975 ± 0.0789
3	$\mathbf{0.8196 \pm 0.0632}$	0.8202 ± 0.0644	0.8155 ± 0.0612	$\mathbf{0.8258 \pm 0.0532}$
4	$\mathbf{0.8636 \pm 0.0583}$	0.8733 ± 0.0762	0.8628 ± 0.0417	$\mathbf{0.8856 \pm 0.0625}$

Table 2.4 and Table 2.5 present the cross-validation results for U-Net and VISA under the relaxed 3×3 window evaluation criterion, without and with augmentations respectively. Without augmentations (Table 2.4), the two models perform comparably overall, though U-Net achieves higher Dice scores in folds 0, 1, and 3, while VISA outperforms U-Net in folds 2 and 4, most notably in fold 4 where VISA reaches a Dice of 0.8672 and clDice of 0.8968 compared to U-Net's 0.8256 and 0.8340. U-Net also struggles in fold 2 without augmentation, where its Dice drops to 0.7417, well below VISA's 0.8183 on the same fold. As shown in Table 2.5, adding augmentation helps U-Net considerably, with it achieving higher or comparable Dice scores across all five folds. This is most evident in fold 2, where its Dice recovers to 0.8196, narrowing the gap with VISA (aug + resampling) at 0.8030. VISA with augmentation and resampling is more consistent across folds than its non-augmented counterpart, though it does not surpass augmented U-Net in Dice. In terms of clDice, VISA (aug + resampling) performs better in folds 3 and 4, suggesting it better preserves the topological connectivity of

vessel structures in those cases.

2.4 Discussion

In this work, we presented VISA (Vascular Image Segmentation Architecture), a two-phase segmentation pipeline for segmenting two-dimensional Digital Subtraction Angiography (DSA) images, combining local patch-based prediction with a region growing algorithm. By combining densely overlapping 80×80 image *snaps* with 3×3 output predictions and an iterative region growing algorithm, our method addresses several challenges inherent to traditional segmentation techniques, including fragmented vessel predictions, class imbalance, and limited labeled training images. While U-Net achieves higher or comparable Dice scores in several folds, particularly with augmentation, these numbers do not capture the full picture. In fold 2, U-Net even with augmentations suffers a near to complete failure on image 08 (Dice: 0.2050), pulling its fold average down to 0.6682 ± 0.2381, whereas VISA (with augmentation and resampling) maintains consistent performance of 0.7433 ± 0.0406 on the same fold. This highlights a key advantage of VISA that it is more stable and robust across varying image characteristics, which is critical in a medical imaging setting where reliability is just as important as peak performance.

More importantly, Dice alone does not tell the full story for vascular segmentation. Blood vessels form continuous, tree like networks where connectivity is clinically essential. A segmentation with slightly lower Dice but intact vessel continuity is far more useful than one with higher overlap but broken vessel structures. The clDice metric captures this directly, and VISA consistently achieves competitive or superior clDice scores compared to U-Net, particularly in folds 0, 3, and 4 with augmentation. This is expected by design: since the region growing algorithm can only expand along connected foreground regions, fragmented predictions are structurally less likely, making VISA a more natural fit for vascular segmentation tasks where topological correctness matters.

It is also worth noting that VISA's quantitative performance is tied to the choice of initial seed pixels, which are currently selected randomly. Since the region growing algorithm expands outward from these seeds, poor or sparse initial seeds can lead to incomplete segmentations, directly affecting Dice and clDice scores. This is an inherent limitation of the current setup rather than of the region growing framework itself, and addressing smarter seed selection is the focus of the next chapter, where we aim to improve VISA's quantitative performance while retaining its connectivity preserving

properties. The snap based design allows the network to learn localized features effectively from a limited number of training images by generating a large number of overlapping training samples. Combined with epoch-wise resampling to address class imbalance, this approach helps the model avoid being biased towards the majority background class. Overall, VISA represents a promising direction for vascular segmentation in DSA images, where preserving the continuity and connectivity of blood vessel structures is just as important as maximizing overlap with the ground truth.

2.5 Acknowledgments

I would like to thank Xiaoyin Xu for providing the DSA images used in this study [65].

2.6 Supplementary Material

Table 2.6: VISA without augmentations Results - Fold 0

Image	Accuracy	clDice	Dice	Jaccard	Sensitivity	Specificity
15	0.9656	0.7922	0.8020	0.6694	0.7415	0.9888
07	0.9847	0.8505	0.8480	0.7361	0.7621	0.9979
11	0.9724	0.6555	0.7646	0.6189	0.6548	0.9957
22	0.9762	0.8317	0.7825	0.6427	0.7041	0.9938
23	0.9812	0.8760	0.8675	0.7661	0.9099	0.9864
18	0.9662	0.8060	0.7976	0.6634	0.8662	0.9745
Average Dice: 0.8104 ± 0.0360			**Average clDice:** 0.8020 ± 0.0710			

Table 2.7: VISA with augmentations and resampling Results - Fold 0

Image	Accuracy	clDice	Dice	Jaccard	Sensitivity	Specificity
15	0.9696	0.8045	0.8238	0.7004	0.7582	0.9915
07	0.9843	0.8370	0.8429	0.7284	0.7503	0.9982
11	0.9741	0.6712	0.7803	0.6398	0.6713	0.9964
22	0.9766	0.8325	0.7830	0.6434	0.6961	0.9947
23	0.9799	0.8484	0.8536	0.7446	0.8666	0.9881
18	0.9660	0.8009	0.7922	0.6560	0.8433	0.9762
Average Dice: 0.8126 ± 0.0290			**Average clDice:** 0.7991 ± 0.0597			

Table 2.8: U-Net without augmentations Results - Fold 0

Image	Accuracy	clDice	Dice	Jaccard	Sensitivity	Specificity
15	0.9716	0.8427	0.8411	0.7258	0.8016	0.9892
07	0.9841	0.7823	0.8545	0.7460	0.8347	0.9929
11	0.9703	0.5993	0.7368	0.5833	0.6075	0.9970
22	0.9722	0.7523	0.7230	0.5661	0.5978	0.9964
23	0.9844	0.8966	0.8880	0.7986	0.9155	0.9894
18	0.9644	0.7751	0.7888	0.6512	0.8633	0.9729
Average Dice: 0.8054 ± 0.0609			**Average clDice:** 0.7747 ± 0.0920			

Table 2.9: U-Net with augmentations Results - Fold 0

Image	Accuracy	clDice	Dice	Jaccard	Sensitivity	Specificity
15	0.9767	0.8742	0.8697	0.7695	0.8301	0.9918
07	0.9799	0.7074	0.8285	0.7072	0.8649	0.9868
11	0.9704	0.5888	0.7346	0.5806	0.5978	0.9978
22	0.9708	0.7266	0.7039	0.5431	0.5714	0.9966
23	0.9866	0.9096	0.9042	0.8252	0.9380	0.9901
18	0.9630	0.7723	0.7880	0.6502	0.8943	0.9687
Average Dice: 0.8048 ± 0.0708			**Average clDice:** 0.7632 ± 0.1070			

Table 2.10: VISA without augmentations Results - Fold 1

Image	Accuracy	clDice	Dice	Jaccard	Sensitivity	Specificity
16	0.9832	0.7847	0.7851	0.6462	0.7170	0.9951
04	0.9569	0.6581	0.6643	0.4973	0.5409	0.9925
20	0.9818	0.8519	0.8566	0.7491	0.8030	0.9948
24	0.9772	0.7820	0.8154	0.6883	0.8785	0.9832
19	0.9699	0.7566	0.8122	0.6838	0.7484	0.9910
Average Dice: 0.7867 ± 0.0653			**Average clDice:** 0.7666 ± 0.0628			

Table 2.11: VISA with augmentations and resampling Results - Fold 1

Image	Accuracy	clDice	Dice	Jaccard	Sensitivity	Specificity
16	0.9820	0.7485	0.7444	0.5929	0.6130	0.9985
04	0.9606	0.6968	0.7141	0.5553	0.6233	0.9895
20	0.9814	0.8601	0.8466	0.7340	0.7610	0.9973
24	0.9813	0.7811	0.8303	0.7099	0.8013	0.9922
19	0.9705	0.7814	0.8158	0.6889	0.7521	0.9913
Average Dice: 0.7903 ± 0.0516			**Average clDice:** 0.7736 ± 0.0532			

Table 2.12: U-Net without augmentations Results - Fold 1

Image	Accuracy	clDice	Dice	Jaccard	Sensitivity	Specificity
16	0.9854	0.8259	0.8179	0.6919	0.7650	0.9953
04	0.9550	0.6627	0.6868	0.5230	0.6252	0.9833
20	0.9822	0.8542	0.8597	0.7539	0.8079	0.9948
24	0.9783	0.7735	0.8192	0.6938	0.8609	0.9854
19	0.9692	0.7547	0.8060	0.6750	0.7356	0.9914
Average Dice: 0.7979 ± 0.0585			**Average clDice:** 0.7742 ± 0.0662			

Table 2.13: U-Net with augmentations Results - Fold 1

Image	Accuracy	clDice	Dice	Jaccard	Sensitivity	Specificity
16	0.9866	0.8470	0.8368	0.7194	0.7994	0.9950
04	0.9597	0.7006	0.7225	0.5656	0.6657	0.9848
20	0.9832	0.8660	0.8664	0.7644	0.8090	0.9958
24	0.9784	0.7781	0.8251	0.7023	0.8894	0.9838
19	0.9705	0.7709	0.8175	0.6913	0.7597	0.9906
Average Dice: 0.8137 ± 0.0485			**Average clDice:** 0.7925 ± 0.0591			

Table 2.14: VISA without augmentations Results - Fold 2

Image	Accuracy	clDice	Dice	Jaccard	Sensitivity	Specificity
14	0.9689	0.7998	0.8209	0.6962	0.8497	0.9798
01	0.9588	0.6997	0.7182	0.5603	0.9214	0.9611
05	0.9485	0.7291	0.7506	0.6008	0.8537	0.9580
08	0.9597	0.8019	0.7978	0.6637	0.8293	0.9735
25	0.9634	0.6789	0.7237	0.5670	0.6508	0.9883
Average Dice: 0.7622 ± 0.0407			**Average clDice:** 0.7419 ± 0.0508			

Table 2.15: RGC with augmentations and resampling Results - Fold 2

Image	Accuracy	clDice	Dice	Jaccard	Sensitivity	Specificity
14	0.9625	0.7283	0.7572	0.6093	0.6971	0.9868
01	0.9587	0.6813	0.7078	0.5477	0.8785	0.9635
05	0.9607	0.7408	0.7773	0.6357	0.7551	0.9812
08	0.9617	0.8090	0.8066	0.6759	0.8328	0.9754
25	0.9628	0.6410	0.7006	0.5391	0.5919	0.9922
Average Dice: 0.7499 ± 0.0406			**Average clDice:** 0.7201 ± 0.0569			

Table 2.16: U-Net without augmentations Results - Fold 2

Image	Accuracy	clDice	Dice	Jaccard	Sensitivity	Specificity
14	0.9617	0.6668	0.7144	0.5557	0.5717	0.9974
01	0.9506	0.6272	0.6592	0.4917	0.8380	0.9574
05	0.9471	0.5819	0.6343	0.4645	0.5054	0.9912
08	0.8739	0.5393	0.5826	0.4110	0.9167	0.8694
25	0.9565	0.4827	0.6007	0.4293	0.4448	0.9971
Average Dice: 0.6382 ± 0.0464			**Average clDice:** 0.5796 ± 0.0646			

Table 2.17: U-Net with augmentations Results - Fold 2

Image	Accuracy	clDice	Dice	Jaccard	Sensitivity	Specificity
14	0.9779	0.8312	0.8566	0.7492	0.7877	0.9953
01	0.9711	0.7666	0.7804	0.6399	0.9025	0.9752
05	0.9673	0.7940	0.8107	0.6817	0.7709	0.9869
08	0.3036	0.4329	0.2050	0.1142	0.9354	0.2366
25	0.9628	0.6082	0.6882	0.5247	0.5586	0.9949
Average Dice: 0.6682 ± 0.2381			**Average clDice:** 0.6866 ± 0.1479			

Table 2.18: VISA without augmentations Results - Fold 3

Image	Accuracy	clDice	Dice	Jaccard	Sensitivity	Specificity
17	0.9827	0.7526	0.7633	0.6172	0.9105	0.9850
06	0.9719	0.6719	0.7075	0.5474	0.6215	0.9922
12	0.9692	0.7808	0.8267	0.7046	0.7423	0.9941
13	0.9686	0.7788	0.7775	0.6360	0.6633	0.9961
26	0.9705	0.6738	0.6425	0.4733	0.6375	0.9850
Average Dice: 0.7435 ± 0.0632			**Average clDice:** 0.7316 ± 0.0490			

Table 2.19: VISA with augmentations and resampling Results - Fold 3

Image	Accuracy	clDice	Dice	Jaccard	Sensitivity	Specificity
17	0.9881	0.8037	0.8219	0.6976	0.8919	0.9912
06	0.9792	0.7527	0.7940	0.6584	0.7325	0.9935
12	0.9697	0.7791	0.8276	0.7059	0.7368	0.9952
13	0.9718	0.8138	0.8027	0.6705	0.6948	0.9967
26	0.9728	0.6811	0.6465	0.4776	0.5998	0.9889
Average Dice: 0.7785 ± 0.0672			**Average clDice:** 0.7661 ± 0.0475			

Table 2.20: U-Net without augmentations Results - Fold 3

Image	Accuracy	clDice	Dice	Jaccard	Sensitivity	Specificity
17	0.9924	0.8607	0.8752	0.7781	0.8722	0.9962
06	0.9757	0.6954	0.7466	0.5957	0.6555	0.9942
12	0.9678	0.7577	0.8227	0.6988	0.7560	0.9910
13	0.9705	0.7987	0.7951	0.6599	0.6921	0.9956
26	0.9709	0.6892	0.6521	0.4838	0.6564	0.9845
Average Dice: 0.7783 ± 0.0756			**Average clDice:** 0.7603 ± 0.0646			

Table 2.21: U-Net with augmentations Results - Fold 3

Image	Accuracy	clDice	Dice	Jaccard	Sensitivity	Specificity
17	0.9905	0.8377	0.8570	0.7497	0.9272	0.9925
06	0.9769	0.7222	0.7709	0.6272	0.7116	0.9922
12	0.9687	0.7623	0.8254	0.7027	0.7496	0.9927
13	0.9740	0.8422	0.8274	0.7056	0.7543	0.9938
26	0.9716	0.6952	0.6606	0.4932	0.6651	0.9849
Average Dice: 0.7882 ± 0.0696			**Average clDice:** 0.7719 ± 0.0595			

Table 2.22: VISA without augmentations Results - Fold 4

Image	Accuracy	clDice	Dice	Jaccard	Sensitivity	Specificity
03	0.9447	0.6923	0.6927	0.5298	0.8856	0.9491
02	0.9813	0.8821	0.8588	0.7525	0.7648	0.9987
10	0.9458	0.8164	0.8195	0.6942	0.9262	0.9488
21	0.9356	0.5297	0.6131	0.4421	0.9317	0.9358
09	0.9763	0.8225	0.8548	0.7464	0.8257	0.9902
Average Dice: 0.7678 ± 0.0981			**Average clDice:** 0.7486 ± 0.1257			

Table 2.23: VISA with augmentations and resampling Results - Fold 4

Image	Accuracy	clDice	Dice	Jaccard	Sensitivity	Specificity
03	0.9596	0.7172	0.7299	0.5747	0.7748	0.9736
02	0.9807	0.8798	0.8573	0.7503	0.7807	0.9968
10	0.9587	0.8593	0.8563	0.7488	0.9266	0.9636
21	0.9138	0.4625	0.5504	0.3797	0.9633	0.9110
09	0.9755	0.7888	0.8424	0.7278	0.7747	0.9940
Average Dice: 0.7673 ± 0.1184			**Average clDice:** 0.7415 ± 0.1508			

Table 2.24: U-Net without augmentations Results - Fold 4

Image	Accuracy	clDice	Dice	Jaccard	Sensitivity	Specificity
03	0.9453	0.6815	0.6954	0.5330	0.8865	0.9498
02	0.9625	0.6913	0.6654	0.4986	0.5023	0.9994
10	0.9621	0.8639	0.8679	0.7666	0.9379	0.9658
21	0.9565	0.6103	0.7028	0.5418	0.9387	0.9576
09	0.9700	0.7019	0.7913	0.6547	0.6720	0.9976
Average Dice: 0.7446 ± 0.0746			**Average clDice:** 0.7098 ± 0.0835			

Table 2.25: U-Net with augmentations Results - Fold 4

Image	Accuracy	clDice	Dice	Jaccard	Sensitivity	Specificity
03	0.9541	0.7294	0.7354	0.5815	0.9050	0.9579
02	0.9743	0.8201	0.7936	0.6579	0.6663	0.9990
10	0.9661	0.8783	0.8812	0.7876	0.9462	0.9692
21	0.9738	0.7263	0.7984	0.6644	0.9453	0.9755
09	0.9752	0.7544	0.8359	0.7180	0.7451	0.9965
Average Dice: 0.8089 ± 0.0484			**Average clDice:** 0.7817 ± 0.0589			

Chapter 3

UI-VISA: U-Net Initialized Vascular Image Segmentation Architecture

3.1 Introduction and Motivation

Medical image segmentation, particularly of vascular structures such as those in Digital Subtraction Angiography (DSA) images, remains a challenging task due to the complex, thin, and branching nature of blood vessels. In the previous chapter, we evaluated two approaches for DSA image segmentation: a modified U-Net [65] and VISA(Vascular Image Segmentation Architecture), a region growing based pipeline. While U-Net has established itself as a strong baseline in biomedical image segmentation owing to its encoder-decoder architecture and skip connections, it struggles with the fine, elongated structures characteristic of vascular imagery. VISA, on the other hand, excels at preserving local connectivity and accurately tracing vascular structures through RGA(Region Growing Algorithm). However, as discussed in the previous chapter, its performance is sensitive to the choice of initial seed points, which were selected randomly.

Region growing algorithms have been widely adopted in vascular image segmentation because of their ability to preserve spatial connectivity and trace elongated, branching structures that pixel-wise classifiers tend to fragment [35]. The core idea is that the algorithm starts from a set of initial seed pixels and grows iteratively outward, incorporating neighboring pixels into the segmentation based on a predefined similarity criterion. This neighborhood-aware expansion is particularly critical for thin structures like blood vessels, where the foreground region is almost entirely boundary, meaning that

unlike thicker structures such as cells, not all neighbors of a foreground pixel can be assumed to also be foreground. Unlike pixel-wise approaches such as U-Net, which classify each pixel independently, region growing explicitly preserves spatial context and enforces the topological connectivity that is biologically expected of vascular structures. However, region growing algorithms are sensitive to the choice of initial seed points, as poor initialization can cause the algorithm to miss vessel branches or grow into irrelevant background regions [35].

To address these limitations, this chapter proposes UI-VISA (UNet Initialized Vascular Image Segmentation Architecture), a hybrid pipeline that combines the strengths of both approaches. In the previous chapter, we introduced VISA, a two-phase pipeline where a CNN predicts local 3×3 masks from 80×80 snaps which then serve as seed points for the Region Growing Algorithm (RGA). In UI-VISA, we only use the second phase of VISA, the RGA, and replace the first phase with a U-Net segmentation. The modified U-Net first performs a segmentation, leveraging its encoder-decoder structure to learn localized image features and its skip connections to retain spatial information lost during downsampling, and the resulting foreground predictions are then used as seed points for the RGA. It is important to note that the RGA still uses the CNN trained in the first phase of VISA to evaluate each candidate pixel during region growing. For each seed pixel, the RGA extracts an 80×80 snap which is passed through this trained CNN to produce a 3×3 output mask, refining the initial segmentation by enforcing local connectivity and recovering fine structural details that U-Net may have missed or incorrectly predicted, while also reducing unnecessary computation and thus saving time. The high-level testing pipeline for UI-VISA is illustrated in Figure 3.1.

The remainder of this chapter is organized as follows: Section 3.2 describes the DSA dataset, training of both methods in detail U-Net and region growing, Section 3.2.4 explains the how region-growing algorithm is used for testing on the U-Net predicted segmentations, Section 3.3 presents quantitative and qualitative results comparing our hybrid approach to standard U-Net segmentation and standalone 2D VISA segmentation, and Section 3.4 discusses the limitation of this approach and future directions of this project.

3.2 Methods

In this section, we describe the full pipeline of UI-VISA, illustrated in Figure 3.1. We begin by describing the dataset, followed by the training of the modified U-Net and the

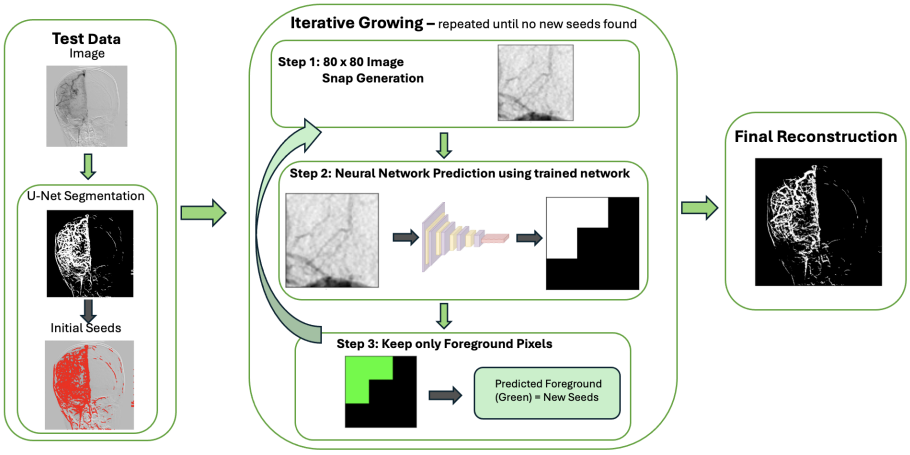


Figure 3.1: Proposed UI-VISA framework for vessel segmentation. The left panel shows a test image segmented by U-Net, with predicted foreground pixels (red) used as initial seeds. The middle panel illustrates the iterative seed expansion pipeline: 80×80 image snaps centered on current seeds are fed into a trained CNN to predict 3×3 mask snaps, with newly predicted foreground pixels added as seeds and the process repeated until convergence. The right panel shows the final reconstructed vessel segmentation.

VISA CNN, and conclude with the testing phase in which the two methods are combined into the proposed hybrid approach. The U-Net is trained as a standard segmentation network, but within the UI-VISA pipeline its predictions serve solely as initial seed points for the region growing stage rather than as the final segmentation. The VISA CNN is similarly trained as a standalone network, with region growing only employed at test time when the trained network is used to iteratively expand from the U-Net-derived seeds.

3.2.1 Data

The dataset consists of 26 DSA images and their manually traced masks [65] (see Figure 2.2 for an example of a typical DSA image and its corresponding mask) from a group of patients. There is only one DSA image per patient. The contrast agent was iodixanol (Visipaque, General Electric), administered by intra-arterial power injection at 5 ml/s for a total amount of 8 ml into the cervical carotid arteries, or at 3 ml/s for a total amount of 5 ml into the cervical vertebral arteries. The patients included in this study had either pathological abnormalities from arteriovenous malformations, arteriovenous fistula, and cerebrovascular aneurysm, or normal DSA findings [65]. The same 5-fold

cross-validation procedure described in Section 2.2.1 was followed, where the 26 image-mask pairs were randomly split into 5 folds with the same training, validation, and test splits. Both the U-Net and the VISA CNN were evaluated on the same held-out test sets to ensure a fair comparison.

3.2.2 Data preparation

In this section, we describe the data preparation procedures for both models used in UI-VISA, the U-Net and the VISA CNN, as their inputs and training strategies differ.

U-Net

To address the limited training data problem, data augmentation was applied to the training set. The dataset consists of 26 DSA images, each of size 512×512 pixels, which is insufficient to train a deep neural network without overfitting. Following Zhang et al. [65], patches of size 128×128 pixels were extracted as inputs to the network. This patch size is well-suited to DSA vessel segmentation, as blood vessels in DSA images appear as thin, fine structures whose local appearance and contrast are contained within a small spatial region. Training on patches rather than full images allows the network to focus on these fine-scale vessel features. To increase the effective size of the training set, 200 patches were randomly sampled from each image. These patches were also extracted from the validation set in the same manner but were not subjected to any augmentation.

Each training patch was subjected to a series of transformations to artificially increase the diversity of the training set. With only 26 DSA images available, augmentation is essential to prevent overfitting and encourage the network to learn vessel features that are invariant to orientation and scale, reflecting the natural variation in how blood vessels appear across different patients and imaging conditions. Random rotations in increments of $90°$ (i.e., $0°, 90°, 180°,$ and $270°$) were applied to add rotational invariance, which is well-motivated in vascular imaging since blood vessels do not have a canonical orientation. Random horizontal and vertical flips were applied for similar reasons; a horizontal flip mirrors the image left to right, while a vertical flip mirrors it along the horizontal axis, helping the model become robust to varied vessel orientations. Finally, random scaling with a factor sampled uniformly from [0.8,1.2] was applied to simulate variability in to make the model robust to differences in vessel size across patients. Following scaling, patches were center-cropped if larger than 128×128 pixels, or padded

back to 128×128 pixels if smaller, to maintain the fixed input size required by the network. Reflection padding was applied to the image and the mask. Each transformation was applied independently with a probability of 0.5, and all transformations were applied identically to both the image patch and its corresponding ground truth mask to maintain consistency between the image patch and its ground truth mask.

VISA CNN

The VISA CNN follows the same data preparation procedure described in Section 2.2.2, where 80×80 image snaps and corresponding 3×3 mask snaps are extracted for every pixel in the training images, and class imbalance is addressed through epoch-wise resampling.

3.2.3 Training

This subsection details the training of the two models used in the hybrid pipeline: the modified U-Net and the VISA CNN.

Training Modified U-Net

The modified U-Net used in this work follows the architecture described in [65] and is illustrated in Figure 3.2. The network consists of two main paths: a contracting path (encoder) that extracts features through downsampling, and an expanding path (decoder) that restores spatial resolution through upsampling, with skip connections linking the two to recover spatial detail lost during downsampling. The encoder consists of four double 3×3 convolutional blocks with ReLU activations and dropout of 0.2, with feature channels progressing from 64 to 512, followed by a bottleneck block at 1024 channels. The decoder mirrors the encoder structure, progressively restoring spatial resolution while reducing feature channels from 512 back to 64, before a final 1×1 convolution produces a single channel output where each pixel value represents the probability of belonging to the vessel class.

The model was trained using stochastic gradient descent (SGD) with a momentum of 0.9, an initial learning rate of 0.01, and a batch size of 32 patches. Binary cross-entropy loss was applied directly to the raw logits for numerical stability. For each epoch, the model was trained on all training patches and subsequently evaluated on the validation set without gradient computation. Training incorporated early stopping with a patience of 25

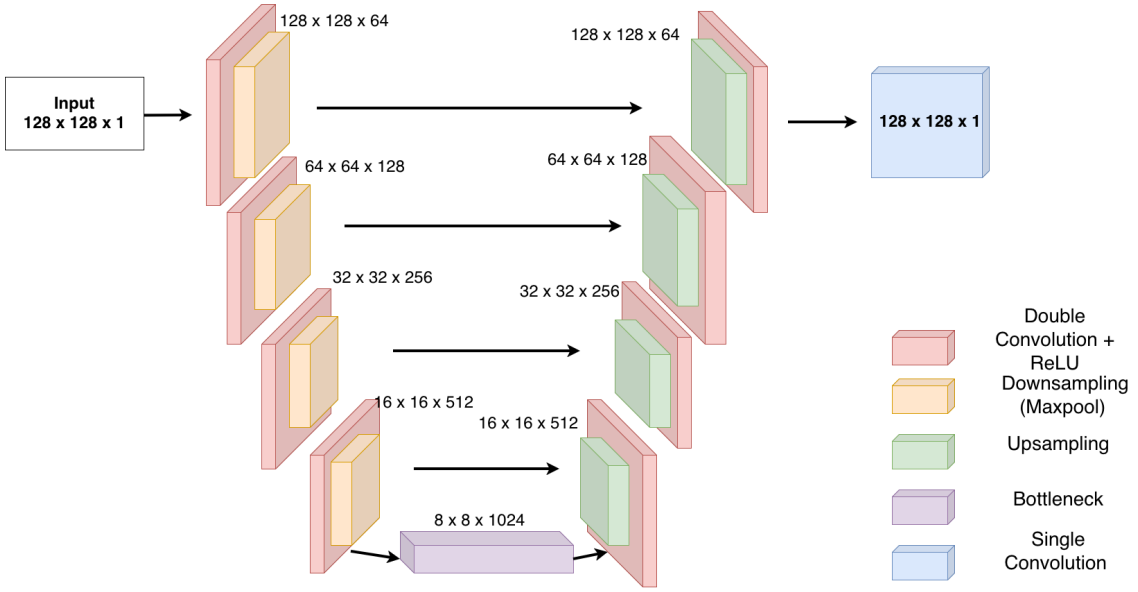


Figure 3.2: Architecture of the modified U-Net. The encoder (left) consists of four double convolution blocks with ReLU activations, each followed by max pooling, progressively reducing the spatial dimensions from 128×128 to 8×8 while increasing the feature channels from 64 to 512. A bottleneck block at the deepest level expands the channels to 1024. The decoder (right) mirrors the encoder, restoring spatial resolution through upsampling and concatenating encoder feature maps via skip connections (horizontal arrows). A final 1×1 convolution produces the output segmentation mask of size $128 \times 128 \times 1$. (Note that this figure is identical to Figure 2.6 from Chapter 2.)

epochs monitored on the validation loss to prevent overfitting, and the model checkpoint corresponding to the lowest validation loss was retained as the final model for each fold.

Training VISA CNN

The Region Growing CNN (RGC) model used in this work was trained as described in Section 2.2.2. Briefly, the model takes 80×80 image snaps as input and predicts a 3×3 segmentation mask centered at each pixel, capturing the local neighborhood rather than an isolated pixel prediction. The architecture resembles the encoder path of a U-Net, consisting of five double convolutional blocks activated by ReLU, each followed by max pooling, with a final 1×1 convolution that reshapes the output to $3 \times 3 \times 1$. This architecture is shown in Figure 3.3. To handle boundary effects during snap extraction,

zero-padding of 40 pixels was applied to all sides of each image, increasing the dimensions from 512×512 to 592×592 and ensuring that snaps centered near the image borders remain fully defined. To increase the diversity of training samples, random $90°$ rotations and horizontal and vertical flips were applied to each training image and its corresponding mask prior to snap extraction, each with a probability of 0.5.

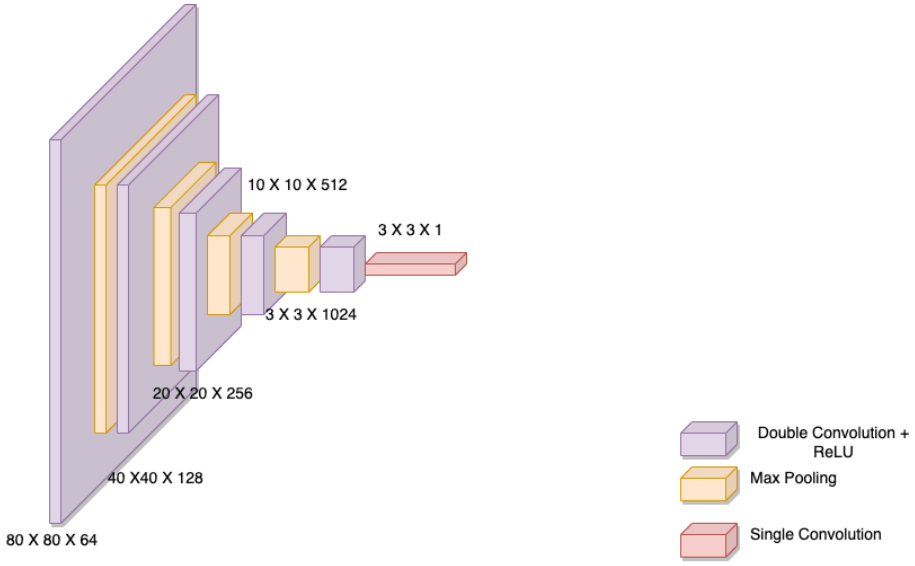


Figure 3.3: VISA CNN Network Architecture: The architecture consists of layers with double convolution and ReLU (purple), max pooling (yellow), and a final single convolution (red). The network takes an $80 \times 80 \times 1$ grayscale snap as input. The dimensions labeled on each layer reflect the output of that layer, starting from $80 \times 80 \times 64$ after the first double convolution and ending at $3 \times 3 \times 1$. (Note that this figure is identical to Figure 2.5 from Chapter 2

.)

The DSA images contain far more background pixels than foreground, since blood vessels occupy only a small fraction of the image area, and this imbalance is clearly reflected in the extracted snaps. To address this, the mask snaps were categorized by the sum of their 3×3 entries, ranging from 0 (entirely background) to 9 (entirely foreground), and downsampled to match the size of the smallest category. This resampling was repeated at the start of every epoch so that the model saw a different random subset of snaps each time, further increasing training diversity. The model was trained using the Adam optimizer [24] with binary cross-entropy loss on densely

overlapping snaps extracted from the same 26 DSA images. The same 5-fold cross-validation splits were used to ensure a fair comparison between the hybrid approach and the standalone methods.

3.2.4 Testing

This section describes the proposed hybrid approach, UI-VISA, where the U-Net predictions are used as seed points for the Region Growing Algorithm (RGA) to produce the final vessel segmentation.

Proposed Hybrid Approach

As illustrated in Figure 3.1, the UI-VISA testing pipeline begins with the U-Net producing a full segmentation of the test image, as shown in the left panel of the figure. The U-Net output is thresholded at 0.5, where predictions greater than 0.5 are classified as foreground and those less than 0.5 are classified as background. The resulting foreground pixels, shown in red at bottom of the left panel, are then used as the initial seeds for the RGA, the second phase of VISA, replacing the random initialization used in the previous chapter. The RGA then proceeds iteratively from these seeds, as shown in the middle panel, outlined in the Algorithm 2. At each iteration, an 80×80 snap is extracted around each current seed pixel and passed through the trained VISA CNN (Section 3.2.3), which outputs a 3×3 segmentation mask centered at that location. The foreground predictions within each 3×3 mask are added to the growing segmentation, and the neighboring pixels surrounding these newly added foreground pixels become the candidate seeds for the next iteration. The algorithm only expands in the direction of foreground predictions and discards background predictions, ensuring that the segmentation grows along vessel structures rather than into the background. This process repeats until no new foreground pixels are found and the segmentation has converged, producing the final reconstructed vessel segmentation shown in the right panel. Since each pixel may be predicted multiple times due to the overlapping nature of the snaps, the predicted probabilities are averaged across all overlapping predictions before the final threshold is applied.

It is important to note that the final segmentation is produced entirely by the RGA, not the U-Net. The U-Net predictions serve solely as an informed starting point, replacing the 500 randomly selected seeds used in Chapter 2. The RGA then refines this initialization by expanding around the seed pixels, recovering fine vessel details and

enforcing local connectivity in regions where the U-Net may have produced fragmented or incomplete predictions. This refinement step is critical for vascular structures, where connectivity and continuity are clinically important. Following the same procedure established in Chapter 2, the tolerance threshold applied to the RGC predictions was selected per fold based on validation set experiments, ranging from 0.4 to 0.6, where pixels with a predicted probability greater than the threshold are classified as foreground. This per-fold threshold selection is kept consistent with the standalone VISA method to ensure a fair comparison between the hybrid UI-VISA and the baseline VISA.



(a) Raw DSA Image (b) U-Net Prediction (c) Initial Seeds (red)

Figure 3.4: Left: raw DSA test image. Middle: U-Net foreground predictions (white) thresholded at 0.5. Right: U-Net foreground predictions shown in red, used as initial seeds for the Region Growing Algorithm.

Figure 3.4(a) shows the raw DSA test image, and Figure 3.4(b) shows the corresponding U-Net prediction with full augmentations. We can see that the U-Net predictions are thicker and fragmented rather than being vascular and contiguous. The white in Figure 3.4(b) represents the foreground predictions made with a threshold of 0.5 and black represents the background predictions. This fragmented prediction is a limitation of pixel-wise prediction approaches like U-Net, which do not explicitly enforce structural continuity and can over-predict in regions where vessel boundaries are ambiguous. Figure 3.4(c) shows the foreground predictions made by U-Net in red, which are then taken as initial seeds for the region growing algorithm. Rather than discarding these imperfect predictions, the hybrid approach uses them as an informed starting point for the RGA, the second phase of VISA, which then refines the segmentation by enforcing local connectivity and recovering the fine vessel details that the U-Net missed or over-predicted.

Algorithm 2 UI-VISA testing Algorithm

Require: Trained U-Net g, trained VISA CNN f, test image I, threshold τ
Ensure: Predicted segmentation mask $\hat{Y}$

Normalize I by dividing by its maximum pixel value
Pad I with 40 zeros on all sides to obtain I_{pad}
Initialize $\hat{Y}(i,j) \leftarrow 0$ for all pixels (i,j)
Initialize pixel score dictionary $\mathcal{P} \leftarrow \{\}$
Initialize visited set $S \leftarrow \emptyset$
Obtain U-Net foreground predictions: $\mathcal{C}_0 \leftarrow \{(i,j) : g(I)(i,j) \geq 0.5\}$
$t \leftarrow 0$
while $\mathcal{C}_t \neq \emptyset$ **do**
    for all $(i,j) \in \mathcal{C}_t$ **do**
        Extract 80×80 snap $\mathcal{S}(i,j)$ from I_{pad} centered at (i,j)
        Obtain 3×3 prediction mask $P \leftarrow f(\mathcal{S}(i,j))$
        for all $(r,c) \in P$ **do**
            Let $(x,y) \leftarrow (i+r-1, j+c-1)$
            Append $P[r,c]$ to $\mathcal{P}[(x,y)]$
        end for
        $S \leftarrow S \cup \{(i,j)\}$
    end for
    $\mathcal{F}_t \leftarrow \{(i,j) \in \mathcal{P} : \text{mean}(\mathcal{P}[(i,j)]) \geq \tau\}$
    $\mathcal{C}_{t+1} \leftarrow \{(i,j) \notin S : (i,j) \in \mathcal{F}_t\}$
    $t \leftarrow t+1$
end while
Reconstruct mask:
for all $(i,j) \in \mathcal{P}$ **do**
    if $\text{mean}(\mathcal{P}[(i,j)]) \geq \tau$ **then**
        $\hat{Y}(i,j) \leftarrow 1$
    end if
end for
Remove padding by cropping rows and columns by 40 of $\hat{Y}$ to recover the original 512×512 mask
return $\hat{Y}$

3.2.5 Post-processing

Segmentation performance is evaluated using two complementary metrics: the Dice score and the centerline Dice (clDice) [52]. The Dice score measures the pixel-level overlap between the predicted segmentation and the ground truth mask, and is defined as:

$$\text{Dice Score}(A, B) = \frac{2|A \cap B|}{|A| + |B|} \tag{3.1}$$

where A and B denote the predicted segmentation and ground truth mask respectively. A Dice score of 1 indicates perfect overlap, while a score of 0 indicates no overlap.

While the Dice score captures overall segmentation accuracy, it does not explicitly account for the connectivity or topology of thin, elongated structures such as blood vessels. Two segmentations with similar Dice scores may differ substantially in terms of vascular continuity, with breaks or disconnections that are clinically significant. To address this, we also evaluate using clDice [52], a metric designed specifically for tubular structures. clDice first extracts the centerlines (skeletons) of both the predicted segmentation and the ground truth mask. Skeletonization is a morphological operation that reduces a binary segmentation to its medial axis, a one-pixel-wide representation that preserves the topology and connectivity of the original structure while discarding its thickness. clDice is defined as:

$$\text{clDice}(skeleton) = \frac{2 \times |\text{Skeleton}_{\text{seg}} \cap \text{Skeleton}_{\text{gt}}|}{|\text{Skeleton}_{\text{seg}}| + |\text{Skeleton}_{\text{gt}}|} \tag{3.2}$$

where $\text{Skeleton}_{\text{seg}}$ and $\text{Skeleton}_{\text{gt}}$ denote the skeletons of the predicted segmentation and ground truth mask respectively. A higher clDice value indicates better preservation of vessel continuity and topology. Together, Dice and clDice provide a comprehensive assessment of both pixel-level accuracy and structural connectivity, which are both essential for reliable vascular segmentation.

3.3 Results and Analysis

Figure 3.5 presents a qualitative comparison of the three methods across three representative test images. The images use a color-coded scheme to visualize prediction accuracy: white represents true positives (correctly identified vessel pixels), black represents true negatives (correctly identified background), red represents false positives

(background incorrectly classified as vessel), and yellow represents false negatives (vessel pixels incorrectly classified as background). In the first row, VISA achieves a strong Dice of 0.7925, while U-Net performs poorly with a Dice of only 0.3216, likely due to the model instability observed in Fold 2. The hybrid approach recovers well with a Dice of 0.7694, demonstrating that even when U-Net predictions are fragmented, the RGA refinement stage produces a coherent segmentation. In the second row, U-Net achieves the best Dice of 0.7920, while both VISA (0.5102) and the hybrid (0.5963) struggle. This is consistent with the quantitative results for Fold 4, where U-Net outperforms the other methods on average. In the third row, all three methods perform similarly, with Dice scores of 0.6519, 0.6577, and 0.6559 for VISA, U-Net, and UI-VISA respectively, suggesting that this particular image is equally challenging for all approaches.

Table 3.1: Summary of Cross-Validation Results for VISA, U-Net, and UI-VISA Models

	VISA		**U-Net**		**UI-VISA**	
Fold	**Dice**	**clDice**	**Dice**	**clDice**	**Dice**	**clDice**
0	0.8126 ± 0.0290	0.7991 ± 0.0597	0.8165 ± 0.0597	0.7898 ± 0.0796	$\mathbf{0.8227 \pm 0.0363}$	$\mathbf{0.8084 \pm 0.0519}$
1	0.8133 ± 0.0431	0.7909 ± 0.0494	0.8106 ± 0.0507	0.7898 ± 0.0612	$\mathbf{0.8225 \pm 0.0438}$	$\mathbf{0.8074 \pm 0.0568}$
2	0.7433 ± 0.0406	0.7178 ± 0.0569	0.6905 ± 0.1910	0.6655 ± 0.1754	$\mathbf{0.7695 \pm 0.0471}$	$\mathbf{0.7600 \pm 0.0567}$
3	0.7795 ± 0.0692	0.7714 ± 0.0466	0.7747 ± 0.0687	0.7596 ± 0.0600	$\mathbf{0.7829 \pm 0.0724}$	$\mathbf{0.7823 \pm 0.0522}$
4	0.7585 ± 0.1347	0.7358 ± 0.1604	$\mathbf{0.7942 \pm 0.0530}$	$\mathbf{0.7634 \pm 0.0586}$	0.7759 ± 0.1041	0.7587 ± 0.1169

Table 3.1 summarizes the cross-validation results for the UI-VISA, U-Net, and VISA models across all five folds. The UI-VISA approach achieves the highest Dice and clDice scores in four out of five folds, demonstrating that replacing random seed initialization with informed U-Net predictions consistently improves segmentation performance. Across folds 0–3, the hybrid model outperforms both standalone methods in both metrics, with Dice scores ranging from 0.7695 to 0.8227 and clDice scores ranging from 0.7600 to 0.8084. The only exception is Fold 4, where U-Net achieves the highest Dice (0.7942) and clDice (0.7634), while the hybrid model scores 0.7759 and 0.7587 respectively. This is also reflected qualitatively in the second row of Figure 3.5, which shows a test image from Fold 4 where U-Net produces a stronger segmentation than the hybrid approach. Detailed per-image metrics including Accuracy, Dice, clDice, Jaccard, Sensitivity, and Specificity for each fold for UI-VISA approach and U-Net are provided in Section 3.5, and for VISA refer to Section 2.6.

Notably, Fold 2 highlights the advantage of the UI-VISA approach most clearly. U-Net alone achieves a Dice score of only 0.6905 ± 0.1910, reflecting significant

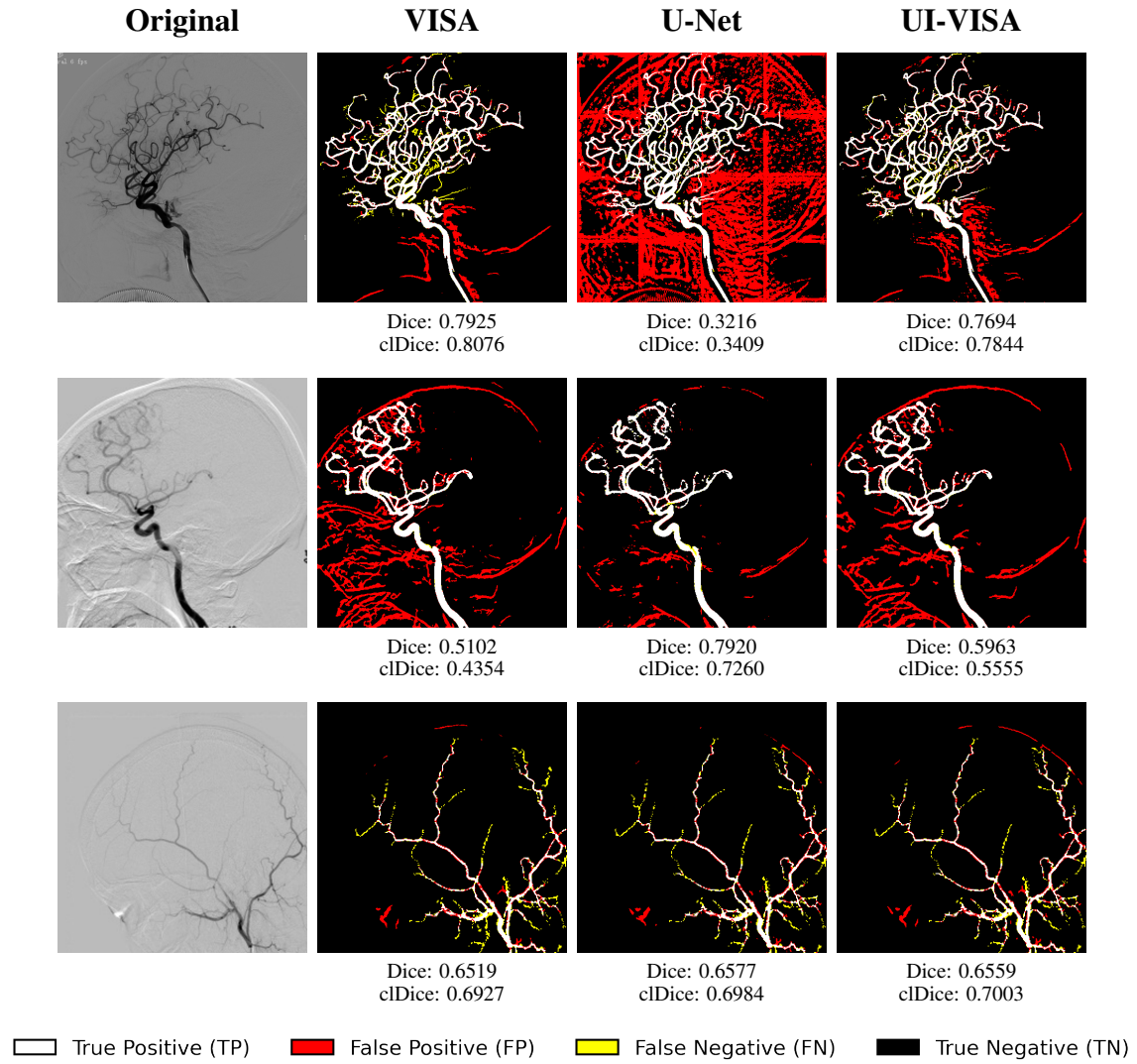


Figure 3.5: Qualitative comparison of segmentation outputs from VISA, U-Net, and UI-VISA approaches across three test images. Dice and clDice scores are reported under each prediction. White: true positives, Black: true negatives, Red: false positives, Yellow: false negatives.

instability across the test images in that fold. The standalone VISA method improves upon this with a Dice of 0.7433, while the hybrid approach achieves 0.7695, demonstrating that informed seed initialization allows the region growing algorithm to recover meaningful vessel structure even when the U-Net predictions are unreliable. The

improvement in clDice from 0.6655 (U-Net) to 0.7600 (UI-VISA) in Fold 2 further confirms that the hybrid approach better preserves vascular connectivity and topological structure. This is particularly evident in the first row of Figure 3.5, which shows Image 08 from Fold 2. The U-Net prediction is severely degraded with a Dice of only 0.3216, yet the hybrid approach recovers a coherent vessel segmentation with a Dice of 0.7694, demonstrating the robustness of the region growing refinement stage even when initialized from poor U-Net seeds.

Table 3.2: Testing Time per Fold for VISA, U-Net, and UI-VISA Models

Fold	**VISA (s)**	**U-Net (s)**	**UI-VISA (s)**
0	3074.99	38.60	1043.67
1	2222.99	32.70	695.17
2	4420.55	39.95	1004.08
3	2436.47	32.53	592.30
4	9341.06	33.26	1811.65

Table 3.2 reports the inference time per fold for each method. U-Net is by far the fastest, completing inference in under 40 seconds per fold, since it only requires a forward pass over extracted patches without any iterative expansion. However, speed alone does not reflect segmentation quality for vascular structures, where connectivity and continuity are biologically essential properties. As demonstrated in Table 3.1, U-Net consistently produces lower clDice scores than the hybrid approach across most folds, indicating that its pixel-wise predictions fail to preserve the topological structure of the vascular network, a critical limitation for clinical applications where vessel continuity directly informs diagnosis and treatment planning.

3.3.1 Statistical Inference

To validate the UI-VISA results, we performed a statistical hypothesis test comparing per-image performance against the U-Net baseline:

$$H_0 : \text{UI-VISA and U-Net perform equally}$$
$$H_1 : \text{UI-VISA outperforms U-Net}$$

Because each of the 26 images appears in exactly one test fold, we obtained a single Dice and clDice score per image for both methods. For each metric, we computed the

per-image difference between UI-VISA and U-Net and applied the Shapiro-Wilk test to assess whether these differences were normally distributed. Using a significance level of $\alpha = 0.001$. For both metrics the p-value was below 0.001 (Dice:p = 0.000003; clDice:p = 0.000007) so we used the Wilcoxon signed-rank test rather than a paired t-test.

The Wilcoxon signed-rank test is a nonparametric test for paired data. It ranks the per-image differences between the two methods by size and checks whether they tend to fall on one side of zero, which would indicate that one method consistently outperforms the other. [59] We applied the test separately to the Dice and clDice scores. For each metric, we computed the per-image difference between UI-VISA and U-Net across all 26 test images and used the one-sided Wilcoxon signed-rank test to evaluate our hypothesis that UI-VISA outperforms U-Net, at a significance level of $\alpha = 0.05$.

Table 3.3 shows the per-image Dice and clDice scores for both UI-VISA and U-Net across all images, along with the computed difference d. UI-VISA scores higher on 17 of the 26 images for Dice and 18 of 26 for clDice. Each of the differences d was ranked by absolute value, and the ranks with positive sign were summed. Under the null hypothesis this sum would be close to the value expected if the signs were assigned at random. The p-value is the probability of observing a sum at least this large if the two methods performed equally. Using a significance level of $\alpha = 0.05$, the improvement in clDice was statistically significant (p = 0.023), while the improvement in Dice was not (p = 0.104). This is consistent with what UI-VISA is designed to do. clDice measures how well the vessel structure is preserved, while Dice measures pixel overlap, so an improvement in clDice suggests the region growing step recovers connectivity that U-Net misses.

3.4 Discussion

In this work, we presented UI-VISA (UNet Initialized Vascular Image Segmentation Architecture), a hybrid pipeline that combines U-Net predictions with the Region Growing Algorithm (RGA) of VISA to improve seed initialization and produce more accurate and connected vessel segmentations. The key distinction between the standalone VISA and UI-VISA lies in seed initialization. In standalone VISA, 500 seeds are selected randomly from the image, meaning they can fall anywhere in the image including background regions. This results in a large number of initial candidate pixels that must be processed and discarded before the algorithm converges on meaningful vessel structures, driving up computational cost. In UI-VISA, seeds are drawn entirely from U-Net

Table 3.3: Per-image Dice and clDice scores for U-Net and UI-VISA on the 26 test images. Each image appears in exactly one test fold. The difference d is computed as UI-VISA minus U-Net, so positive values favour UI-VISA.

	Dice			clDice		
Image	U-Net	UI-VISA	d	U-Net	UI-VISA	d
01	0.785	0.712	-0.074	0.761	0.707	-0.054
02	0.785	0.871	$+0.085$	0.802	0.889	$+0.087$
03	0.704	0.721	$+0.017$	0.695	0.712	$+0.016$
04	0.716	0.752	$+0.035$	0.691	0.733	$+0.042$
05	0.789	0.794	$+0.005$	0.763	0.778	$+0.015$
06	0.739	0.795	$+0.056$	0.690	0.764	$+0.074$
07	0.853	0.824	-0.029	0.765	0.764	-0.001
08	0.322	0.769	$+0.448$	0.341	0.784	$+0.443$
09	0.825	0.855	$+0.031$	0.736	0.801	$+0.065$
10	0.864	0.836	-0.029	0.858	0.837	-0.021
11	0.779	0.824	$+0.045$	0.665	0.745	$+0.081$
12	0.822	0.857	$+0.034$	0.755	0.818	$+0.063$
13	0.807	0.846	$+0.039$	0.816	0.855	$+0.038$
14	0.857	0.844	-0.013	0.834	0.844	$+0.010$
15	0.862	0.867	$+0.004$	0.869	0.874	$+0.005$
16	0.829	0.832	$+0.003$	0.843	0.843	-0.000
17	0.847	0.761	-0.087	0.838	0.775	-0.064
18	0.775	0.776	$+0.002$	0.763	0.767	$+0.003$
19	0.816	0.837	$+0.021$	0.768	0.805	$+0.037$
20	0.869	0.886	$+0.017$	0.865	0.893	$+0.028$
21	0.792	0.596	-0.196	0.726	0.556	-0.170
22	0.729	0.779	$+0.050$	0.767	0.831	$+0.064$
23	0.901	0.867	-0.035	0.910	0.869	-0.040
24	0.823	0.806	-0.017	0.782	0.763	-0.019
25	0.700	0.729	$+0.029$	0.628	0.687	$+0.059$
26	0.658	0.656	-0.002	0.698	0.700	$+0.002$

foreground predictions, meaning every seed is already located within a predicted vessel region. This informed initialization eliminates unnecessary expansion into background regions and significantly reduces the total number of iterations required for convergence. As a result, UI-VISA is substantially faster than standalone VISA, ranging from approximately 592 to 1812 seconds per fold compared to 2223 to 9341 seconds for standalone VISA, a reduction of roughly 3 to 5 times.

Taken together, these results demonstrate that UI-VISA gives the most favorable balance among the three methods: it is more accurate and better connected than U-Net alone, more computationally efficient than standalone VISA, and consistently preserves the vascular connectivity that is essential for reliable DSA image segmentation. The reduction in inference time relative to standalone VISA, combined with improved or comparable segmentation accuracy, supports UI-VISA as a practical and effective approach for DSA vessel segmentation.

Despite the computational savings achieved through informed seed initialization, UI-VISA remains substantially slower than standalone U-Net inference. A promising direction for future work is to further reduce computational cost while preserving the connectivity advantages of the region growing approach. Rather than initializing seeds from all U-Net foreground predictions, a more targeted strategy would be to restrict seed selection to pixels lying on the boundary between predicted foreground and background regions in the U-Net output. Additionally, to avoid unnecessary computation, the RGA could be augmented with an early termination criterion: if the VISA CNN prediction at a given location agrees with the U-Net prediction, expansion in that direction is halted. This boundary-aware initialization strategy has the potential to significantly reduce the number of iterations required for convergence, bringing the computational cost of UI-VISA closer to that of standalone U-Net while retaining the topological correctness that makes region growing essential for vascular segmentation.

3.5 Supplementary Material

Table 3.4: UI-VISA Results - Fold 0

Image	Accuracy	clDice	Dice	Jaccard	Sensitivity	Specificity
15	0.9754	0.8740	0.8665	0.7645	0.8505	0.9884
07	0.9814	0.7639	0.8236	0.7001	0.7755	0.9936
11	0.9781	0.7455	0.8244	0.7012	0.7502	0.9949
22	0.9758	0.8309	0.7787	0.6376	0.7007	0.9936
23	0.9806	0.8693	0.8665	0.7645	0.9323	0.9841
18	0.9604	0.7669	0.7765	0.6346	0.8941	0.9659
Average Dice: 0.8227 ± 0.0363			**Average clDice:** 0.8084 ± 0.0519			

Table 3.5: U-Net with Augmentations Results - Fold 0

Image	Accuracy	clDice	Dice	Jaccard	Sensitivity	Specificity
15	0.9753	0.8686	0.8620	0.7575	0.8210	0.9913
07	0.9835	0.7649	0.8526	0.7431	0.8527	0.9912
11	0.9743	0.6648	0.7791	0.6382	0.6620	0.9973
22	0.9726	0.7673	0.7291	0.5737	0.6074	0.9962
23	0.9862	0.9097	0.9015	0.8207	0.9340	0.9900
18	0.9605	0.7635	0.7748	0.6323	0.8832	0.9669
Average Dice: 0.8165 ± 0.0597			**Average clDice:** 0.7898 ± 0.0796			

Table 3.6: UI-VISA Results - Fold 1

Image	Accuracy	clDice	Dice	Jaccard	Sensitivity	Specificity
16	0.9864	0.8433	0.8324	0.7129	0.7884	0.9953
04	0.9609	0.7328	0.7515	0.6019	0.7500	0.9789
20	0.9849	0.8927	0.8856	0.7947	0.8678	0.9933
24	0.9751	0.7628	0.8061	0.6752	0.9055	0.9793
19	0.9715	0.8053	0.8371	0.7198	0.8430	0.9837
Average Dice: 0.8225 ± 0.0438			**Average clDice:** 0.8074 ± 0.0568			

Table 3.7: U-Net with Augmentations Results - Fold 1

Image	Accuracy	clDice	Dice	Jaccard	Sensitivity	Specificity
16	0.9856	0.8433	0.8293	0.7084	0.8167	0.9932
04	0.9580	0.6910	0.7161	0.5577	0.6712	0.9826
20	0.9831	0.8646	0.8688	0.7680	0.8283	0.9943
24	0.9782	0.7821	0.8228	0.6989	0.8848	0.9839
19	0.9702	0.7681	0.8160	0.6891	0.7605	0.9901
Average Dice: 0.8106 ± 0.0507			**Average clDice:** 0.7898 ± 0.0612			

Table 3.8: UI-VISA Results - Fold 2

Image	Accuracy	clDice	Dice	Jaccard	Sensitivity	Specificity
14	0.9732	0.8439	0.8436	0.7296	0.8640	0.9831
01	0.9568	0.7073	0.7118	0.5525	0.9357	0.9581
05	0.9604	0.7779	0.7939	0.6583	0.8397	0.9725
08	0.9476	0.7844	0.7694	0.6252	0.9111	0.9515
25	0.9645	0.6868	0.7289	0.5735	0.6500	0.9894
Average Dice: 0.7695 ± 0.0471			**Average clDice:** 0.7600 ± 0.0567			

Table 3.9: U-Net with Augmentations Results - Fold 2

Image	Accuracy	clDice	Dice	Jaccard	Sensitivity	Specificity
14	0.9778	0.8339	0.8566	0.7492	0.7920	0.9948
01	0.9720	0.7614	0.7853	0.6465	0.9000	0.9763
05	0.9659	0.7633	0.7889	0.6514	0.7015	0.9923
08	0.6131	0.3409	0.3216	0.1916	0.9557	0.5767
25	0.9636	0.6278	0.6998	0.5383	0.5771	0.9943
Average Dice: 0.6905 ± 0.1910			**Average clDice:** 0.6655 ± 0.1754			

Table 3.10: UI-VISA Results - Fold 3

Image	Accuracy	clDice	Dice	Jaccard	Sensitivity	Specificity
17	0.9814	0.7745	0.7607	0.6138	0.9664	0.9818
06	0.9767	0.7640	0.7950	0.6598	0.8266	0.9854
12	0.9724	0.8182	0.8566	0.7492	0.8342	0.9876
13	0.9758	0.8546	0.8460	0.7331	0.8051	0.9912
26	0.9702	0.7003	0.6559	0.4880	0.6848	0.9825
Average Dice: 0.7829 ± 0.0724			**Average clDice:** 0.7823 ± 0.0522			

Table 3.11: U-Net with Augmentations Results - Fold 3

Image	Accuracy	clDice	Dice	Jaccard	Sensitivity	Specificity
17	0.9899	0.8381	0.8474	0.7352	0.9119	0.9924
06	0.9742	0.6902	0.7389	0.5859	0.6664	0.9921
12	0.9684	0.7551	0.8223	0.6982	0.7396	0.9935
13	0.9714	0.8164	0.8074	0.6770	0.7254	0.9936
26	0.9714	0.6984	0.6577	0.4900	0.6611	0.9849
Average Dice: 0.7747 ± 0.0687			**Average clDice:** 0.7596 ± 0.0600			

Table 3.12: UI-VISA Results - Fold 4

Image	Accuracy	clDice	Dice	Jaccard	Sensitivity	Specificity
03	0.9528	0.7116	0.7211	0.5639	0.8673	0.9592
02	0.9824	0.8886	0.8708	0.7712	0.8008	0.9969
10	0.9499	0.8369	0.8358	0.7180	0.9594	0.9485
21	0.9274	0.5555	0.5963	0.4248	0.9786	0.9245
09	0.9772	0.8010	0.8555	0.7475	0.7961	0.9940
Average Dice: 0.7759 ± 0.1041			**Average clDice:** 0.7587 ± 0.1169			

Table 3.13: U-Net with Augmentations Results - Fold 4

Image	Accuracy	clDice	Dice	Jaccard	Sensitivity	Specificity
03	0.9446	0.6953	0.7041	0.5433	0.9366	0.9452
02	0.9734	0.8018	0.7854	0.6467	0.6559	0.9989
10	0.9602	0.8579	0.8644	0.7611	0.9547	0.9610
21	0.9722	0.7260	0.7920	0.6557	0.9658	0.9726
09	0.9737	0.7360	0.8249	0.7020	0.7313	0.9961
Average Dice: 0.7942 ± 0.0530			**Average clDice:** 0.7634 ± 0.0586			

Chapter 4

3D-VISA: Three dimensional Vacular Image Segmentation Architecture

4.1 Introduction and Motivation

Medical segmentation is the process of identifying and delineating specific anatomical structures or pathological regions within three-dimensional medical images. It plays a crucial role in modern clinical practice and directly impacts patient outcomes as it helps clinicians for example access the accurate size of a tumor or to precisely measure whether the cancer is growing or shrinking in response to a treatment. For example in surgical planning the accurate segmentation is not only important for technical reasons but also for the safety of a patient and efficient treatment process as it allows surgeons to distinguish healthy tissue from diseased tissue and plan interventions that maximize treatment benefit while minimizing damage to surrounding structures [67, 68]. Specifically in three-dimensional image segmentation, unsegmented images frequently show overlapping tissues which obscure target structures, making the precise delineation of boundaries between regions of interest, such as tumors and surrounding tissues essential for accurate diagnosis, treatment planning, and monitoring of disease progression [64].

A 3D medical image is structured as a volumetric grid of size $(w \times h \times d)$, where each discrete element, *or voxel*, represents a specific physical coordinate in space. In this framework, binary segmentation is defined as a classification task that partitions the volume by assigning a label $L \in \{0, 1\}$ to each voxel, identifying whether it belongs to a target anatomical structure, such as a brain lesion or a specific organ. The success of this

process is determined through segmentation evaluation, a comparative analysis that quantifies the spatial correspondence or distance between the predicted labels and a gold-standard ground truth. [56]

In the medical imaging domain, creating a sufficiently large 3D dataset is highly challenging due to the invasive nature of some imaging techniques (e.g. CT scans), the extended duration required for imaging, and the laborious process of manually annotating 3D volumetric data [6]. These limitations create a significant barrier to training deep learning models, which typically require large amounts of labeled data to achieve robust performance. To address this data scarcity challenge, we leverage the idea of using patch based methods and Region Growing algorithm. Unlike conventional approaches that require extensive datasets, patch-based methods can generate training data from a single annotated image-mask pair. The extracted patches can then be used to train a 3D convolutional neural network (CNN) model.

Early approaches to segmenting 3D medical volumes relied on applying 2D convolutional neural networks slice by slice, treating each 2D cross-section of the volume independently before stitching the predictions back into a 3D segmentation [14]. While this approach leveraged the success of well-established 2D architectures such as U-Net [44], it suffered from a fundamental limitation of processing each slice independently. Because of that these methods failed to exploit the spatial context and continuity between the slices, which is critical for accurately delineating 3D structures such as vessels, tumors, and organs [14, 60].

To address the spatial limitation, fully 3D CNNs were introduced that processed the entire 3D volume together instead of the 2D slices. Architectures such as 3D UNet [8] and 3D VNet [37] replaced the traditional 2D layers with their 3D counterparts such as 3D convolutions, pooling, etc. to have the network better extract the 3D spatial features from all three directions of the image volumes. The 3D Vnet [37] took a step further by introducing a 3D optimization framework utilizing a dice-coefficient based loss function, specifically engineered to handle the extreme class imbalances common in localized medical contexts. Despite their success to process whole 3D volumes, 3D CNNs face a critical challenge of utilizing a substantial amount of computational resources as the computational complexity grows cubically [14]. On top of this, these models do require large amounts of 3D annotated data which might get difficult to obtain in the medical domain due to the invasive nature of some imaging modalities, the time-consuming acquisition process, and the expertise required for manual annotation of volumetric data [6, 14].

To overcome this obstacle of inability to access large amounts of data and save computational resources, patch based methods were introduced. A patch is a sub volume extracted from the larger volume and processed independently, which significantly reduces memory and computation cost [20]. Patch based methods also help generate more training data thus helping solve the data scarcity problem in medical domain. A notable example of a successful patch-based approach is DeepMedic [20], which used small 3D patches extracted from the larger volume and achieved strong performance on brain lesion segmentation, showing that patch-based methods can work well even with limited available data. But since standard patch based methods predict locally, at times they can miss the contiguity of the data specifically in the vascular structures such as vessels as breaks in connectivity can disrupt the topological structure of the vascular network.

In this chapter, we introduce 3D-VISA, an extension of VISA (Vascular Image Segmentation Architecture) to three-dimensional volumetric data. Following the same two-phase structure as VISA introduced in Chapter 2, 3D-VISA trains a 3D CNN on overlapping $40 \times 40 \times 40$ snaps, each producing a localized $3 \times 3 \times 3$ prediction in the first phase. The resulting predictions then guide a 3D Region Growing Algorithm (RGA) in the second phase, which expands from the 3D CNN predictions by iteratively checking neighboring voxels to produce a complete, contiguous volumetric segmentation. By extracting these overlapping snaps centered at every voxel, 3D VISA generates large number of training samples from a single volumetric image, addressing the data scarcity problem common in medical imaging. Furthermore, unlike standard patch-based methods that predict locally, the 3D RGA ensures that the final segmentation is spatially contiguous by iteratively expanding only around foreground predictions, preserving the topological connectivity of the vascular network.

The remainder of this chapter is organized as follows: Section 4.2 described the 3D dataset used in this work, Section 4.3 details the data preparation, addressing the class imbalance problem and the 3D CNN architecture used to train the data, Section 4.4 explains the 3D Region Growing Algorithm in detail, then Section 4.5 and Section 4.6 show the results and discuss future work of this project.

4.2 Data

The dataset used in this work is a synthetic dataset taken from [57]. The dataset consists of synthetic 3D vascular images, generated by simulating physiologically

plausible vascular trees using a computational angiogenesis model. The dataset was generated using a simulation method based on [46, 45], it consists of 136 volumetric images of size $325 \times 304 \times 600$ voxels, each accompanied by ground truth labels for vessel segmentation, centerlines, and bifurcation detection. Vessel intensities were sampled from $[128, 255]$ and non-vessel intensities from $[0, 100]$, with Gaussian noise applied to each volume. The dataset shows significant class imbalance, with vessel, centerline, and bifurcation labels occupying approximately $2.1\%, 0.2\%$, and 0.05% of the total voxel intensities, respectively.

4.3 Training

In this section, we detail the training pipeline illustrated in Figure 4.1, which involves three steps: extracting overlapping snaps from each image volume, addressing class imbalance through down-sampling, and training the 3D CNN on the resulting balanced dataset.

The top part of Figure 4.1 illustrates the "snap" extraction process. Rather than processing the entire volume as a single unit, we extract overlapping patches centered on each voxel, which we refer to as "snaps" from the volumetric images and their corresponding masks. Specifically, we extract a $40 \times 40 \times 40$ volumetric snap from the image volume and a corresponding $3 \times 3 \times 3$ volumetric snap from the mask volume. Each $40 \times 40 \times 40$ snap provides the network with sufficient contextual information to learn relevant features and recognize local patterns in the vascular structures, while the $3 \times 3 \times 3$ mask snap captures the local neighborhood label around each voxel. This strategy generates diverse and detailed training samples from a limited number of volumes, which is essential for accurately capturing fine vascular structures. The bottom part of Figure 4.1 shows the per-snap inference step where each $40 \times 40 \times 40$ volumetric image snap is passed through a 3D CNN, which outputs a predicted $3 \times 3 \times 3$ volumetric mask snap.

4.3.1 Data Preparation and Snap Extraction

For preprocessing, we start with three total images: one image-mask pair each for training, validation and testing. As described in Section 4.2 the dimensions of each image are $325 \times 304 \times 600$ voxels. Each volume is padded with $20 \times 20 \times 20$ zeros. Below, we discuss each aspect of training our neural network. We extract *snaps* centered at each

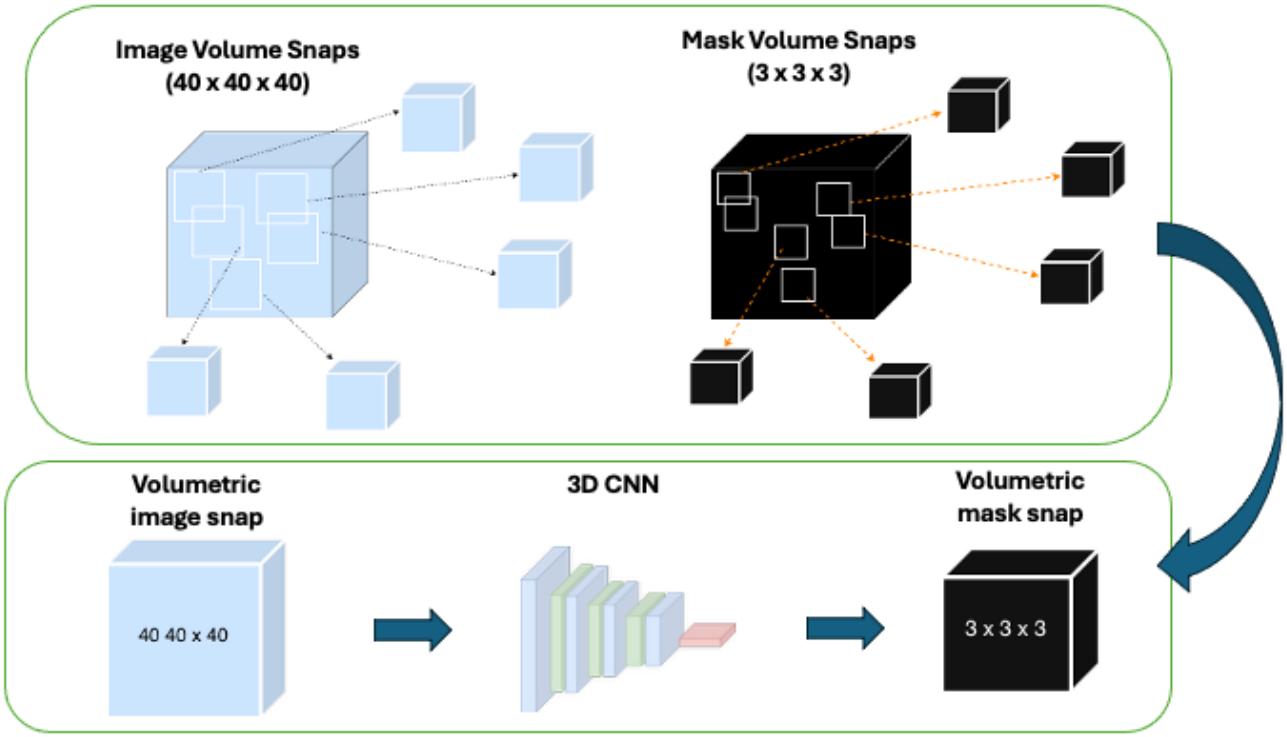


Figure 4.1: Overview of training phase of 3D-VISA. *(Top)* Shows the creation of volumetric snaps that are extracted from the full image volume ($40 \times 40 \times 40$) and its corresponding mask volume($3 \times 3 \times 3$). *(Bottom)* Each $40 \times 40 \times 40$ volumetric image snap is passed through a 3D CNN, producing a predicted $3 \times 3 \times 3$ volumetric mask snap as output.

voxel position in the volume, then balance the dataset to ensure equal representation across different foreground density levels.

Each image snap has a size of $40 \times 40 \times 40$ voxels. The corresponding mask snaps are $3 \times 3 \times 3$ voxels, which allows the network to make predictions for a small, focused neighborhood around the chosen voxel rather than predicting each voxel independently. This ensures that the model considers the local context when deciding if a voxel belongs to a structure of interest. Using this approach, roughly 59 million training snaps are generated per image. Since these snaps are overlapping, they also allow each voxel to be seen in multiple contexts, which helps the network generalize better despite the limited number of annotated volumes.

4.3.2 Addressing Class Imbalance

Initial training on all 59 million training patches gave poor results. The network predicted almost all zeros for the mask patches, completely missing the vessel structures. This suggested that the network was not learning the foreground structures. We suspected the problem was due to class imbalance, so we looked more closely at the mask patches. To understand the data distribution, we calculated the sum of voxel values in each $3 \times 3 \times 3$ mask patch. These sums range from 0 (no foreground voxels) to 27 (all foreground voxels). Table 4.1 shows the distribution of patches across these sum categories. The data is severely imbalanced and we see that nearly 95% of patches have sum 0 (no foreground), while the smallest category (sum 25) contains only $28,520$ samples. To address this imbalance, we down-sample all categories to match the smallest one. We randomly select $28,520$ patches from each sum category, creating a balanced training set of $28,520 \times 28 = 798,560$ total samples. While this significantly reduces the dataset size, it ensures equal representation across all foreground density levels.

4.3.3 Network Architecture and Training

The network has a 9 layer 3D CNN that follows an encoder style architecture consisting of four double convolutional blocks, each comprising two convolutional layers with batch normalization and ReLU activation, followed by max-pooling layers of sizes $2 \times 2 \times 2$, $3 \times 3 \times 3$, and $2 \times 2 \times 2$ respectively, as shown in Figure 4.3. These operations progressively reduce spatial resolution while increasing feature depth, enabling the extraction of hierarchical representations. The last layer consists of a single convolution to produce the required output of $3 \times 3 \times 3 \times 1$, where 1 represents if a voxel is in foreground or background. The network processes a localized 3D input snap of size

Sum Category	Number of Snaps	Sum Category	Number of Snaps
0.0	56,282,257	14.0	80,553
1.0	279,921	15.0	81,819
2.0	294,986	16.0	66,946
3.0	227,623	17.0	67,062
4.0	156,374	18.0	82,555
5.0	152,968	19.0	41,604
6.0	167,460	20.0	44,272
7.0	127,363	21.0	38,630
8.0	140,272	22.0	49,535
9.0	133,108	23.0	38,126
10.0	86,574	24.0	55,865
11.0	79,352	25.0	28,520
12.0	83,876	26.0	50,712
13.0	79,858	27.0	261,809
		Total	**59,029,359**

Table 4.1: Number of training snaps per sum category before balancing

$40 \times 40 \times 40$ and maps to an output mask of size $3 \times 3 \times 3$ centered on the voxel of interest, capturing all contextual information around each voxel and ensuring consistency. By predicting a $3 \times 3 \times 3$ region, the network ensures spatial consistency within this smaller volume, helping to maintain smooth transitions and reduce jagged edges in the segmentation map, which is particularly important for preserving the connectivity and contiguity of vascular structures. This model was trained using the Adam optimizer [24] with Binary Cross Entropy with logits loss function on a GPU available on the Pinnacles Cluster Section 4.7, for 30 epochs as shown in Figure 4.2.

4.4 Testing

This section describes the testing phase of 3D-VISA, where the 3D RGA iteratively extracts $40 \times 40 \times 40$ snaps from the test volume, passes them through the trained 3D CNN, and expands around foreground predictions to produce the final volumetric segmentation.

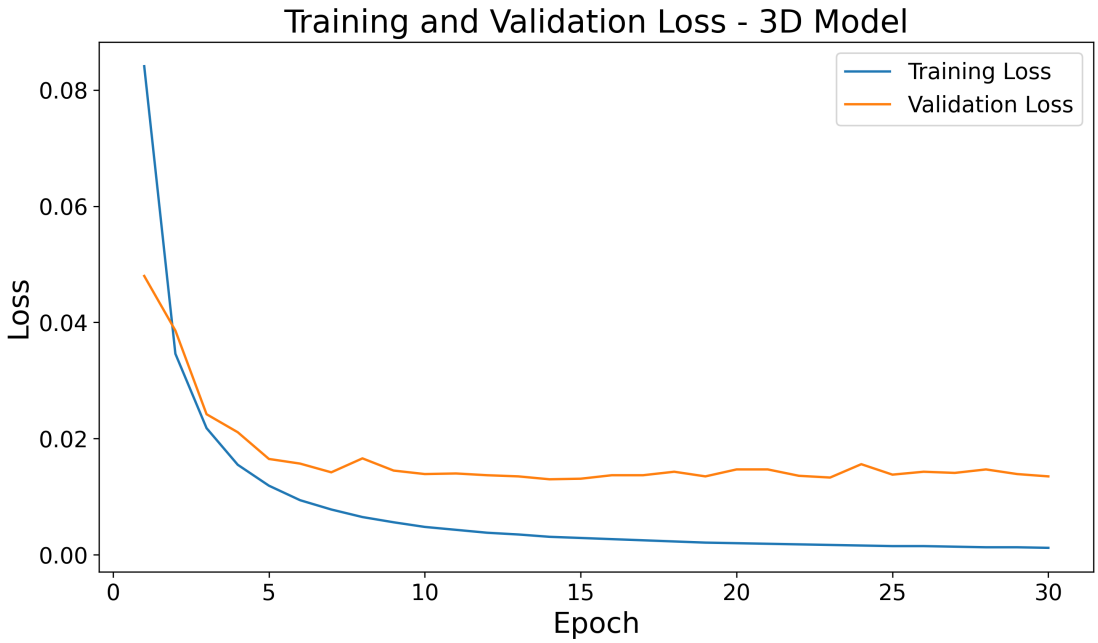


Figure 4.2: Training and validation loss curve for the 3D CNN model over 30 epochs; blue curve shows the training loss and orange corve shows the validation loss

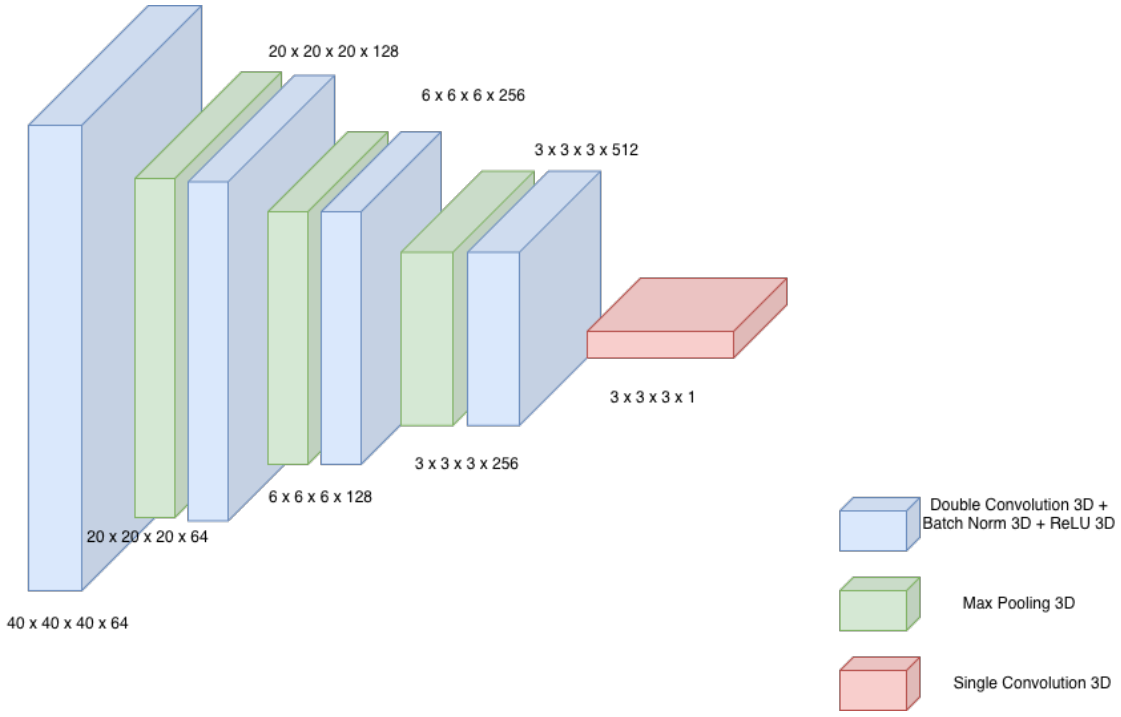


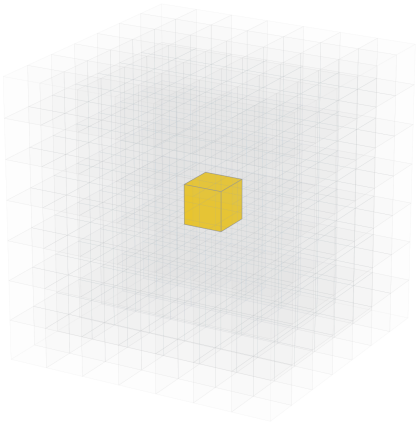
Figure 4.3: 3D Network Architecture. Convolutional Neural Network (CNN) architecture consisting of layers with double 3D convolution and 3D ReLU (blue), 3D max pooling (green), and a final single 3D convolution (red). The network takes a $40 \times 40 \times 40$ grayscale patch as input. The dimensions labeled on each layer reflect the output of that layer, starting from $40 \times 40 \times 40 \times 64$ after the first double convolution and ending at $3 \times 3 \times 3 \times 1$.

4.4.1 3D Region Growing Algorithm (3D RGA)

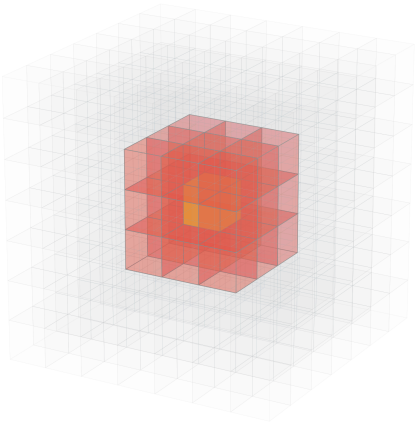
In the testing phase of our network evaluation, we utilize the 3D RGA, the second phase of 3D-VISA for efficient segmentation. Unlike the training phase, where the focus is on parameter optimization, testing aims for accurate results within a reasonable time-frame. The main idea for the algorithm is that we start by generating $40 \times 40 \times 40$ snaps from the 3D volume and pass those through the trained 3D CNN from Section 4.3 that produces $3 \times 3 \times 3$ volume masks. Then the region growing algorithm iteratively expands around foreground voxel predictions and discards the background predictions. This approach ensures contiguous regions without processing the entire 3D volume. This targeted strategy significantly reduces processing time compared to comprehensive image analysis.

Figure 4.4 shows the working of the region growing algorithm in detailed steps, representing one full iteration of the algorithm. In all of the images, yellow represents predicted foreground voxels, purple represents predicted background voxels, and red represents the voxels that will be predicted in the next iteration of the algorithm. For example, Figure 4.4a shows one foreground voxel on the testing image in yellow. Figure 4.4b shows the candidate voxels that will be predicted in the next iteration. To determine these candidates, we extract an $40 \times 40 \times 40$ snap around each candidate voxel and pass it through the trained network. The network outputs a $3 \times 3 \times 3$ probability mask for each candidate voxel, indicating the likelihood of each voxel being classified as foreground or background. The algorithm makes a prediction based on the trained model, which can be seen in Figure 4.4c, which shows the predictions from the above candidate voxels. The voxels in yellow were categorized as foreground and the voxels in purple were categorized as the background. To avoid unnecessary computation, the algorithm only expands in directions of foreground voxels and discards the background ones. Thus, in the next iteration of the algorithm, the voxels that will be predicted into foreground and background (Figure 4.4d, red), only surround the centroids which were predicted as foreground in the previous iteration (Figure 4.4c, yellow). We can see in Figure 4.4d that the next set of candidate voxels in red are either adjacent or diagonal to the foreground voxels.

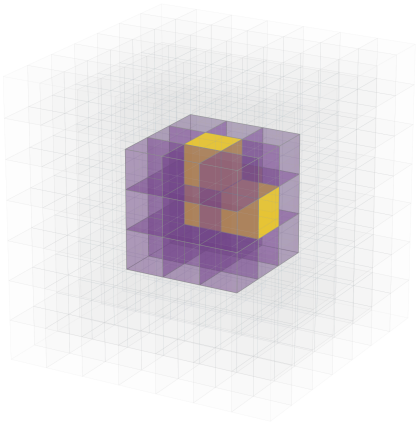
We start by picking 500 initial voxels from the test volume, referred to as centroids. Around each centroid, we extract a small 3D snaps of size $40 \times 40 \times 40$ as explained in Section 4.3.1 with respect to a center voxel. These snaps are then passed through the trained model, which outputs corresponding $3 \times 3 \times 3$ masks. From these initial predictions, we identify which voxels are foreground and which are background. In the next iteration, we ignore the background voxels and only keep the foreground ones as the



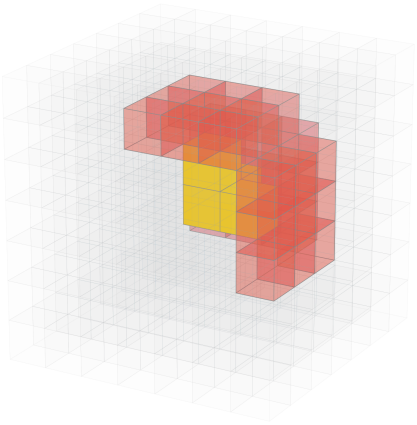
(a) Starting with single initial voxel in yellow



(b) Next set of voxels to pass through the trained network in red



(c) Candidates classified into foreground (yellow) and background (purple)



(d) Next set of candidate voxels (red) identified

Figure 4.4: Example illustrating the 3D region growing method. The process begins with a single initial foreground voxel (yellow) as shown in the first panel. The second panel identifies the next candidate voxels (red) that will be evaluated by the trained network. The third panel shows the predicted classification of voxels into foreground (yellow) and background (purple). Finally, the fourth panel highlights the next set of candidate voxels (red) for the subsequent iteration of the network evaluation. Yellow indicates foreground voxels, purple indicates background voxels, and red indicates the next candidate voxels for classification.

Algorithm 3 3D Region Growing Algorithm

Require: Trained network f, test volume I, threshold τ
Ensure: Predicted segmentation mask volume $\hat{Y}$

Normalize I by dividing by its maximum voxel value
Pad I with 20 zeros on all sides to obtain I_{pad}
Initialize $\hat{Y}(i, j, k) \leftarrow 0$ for all voxels (i, j, k)
Initialize voxel score dictionary $\mathcal{P} \leftarrow \{\}$
Initialize visited set $S \leftarrow \emptyset$
Sample 500 random voxels from I to form initial candidate set $\mathcal{C}_0$
$t \leftarrow 0$
while $\mathcal{C}_t \neq \emptyset$ **do**
    for all $(i, j, k) \in \mathcal{C}_t$ **do**
        Extract $40 \times 40 \times 40$ volumetric snap $\mathcal{S}(i, j, k)$ from I_{pad} centered at (i, j, k)
        Obtain $3 \times 3 \times 3$ prediction mask volume $P \leftarrow f(\mathcal{S}(i, j, k))$
        for all $(r, c, b) \in P$ **do**
            Let $(x, y, z) \leftarrow (i + r - 1, j + c - 1, k + b - 1)$
            Append $P[r, c, b]$ to $\mathcal{P}[(x, y, z)]$
        end for
        $S \leftarrow S \cup \{(i, j, k)\}$
    end for
    $\mathcal{F}_t \leftarrow \{(i, j, k) \in \mathcal{P} : \text{mean}(\mathcal{P}[(i, j)]) \geq \tau\}$
    $\mathcal{C}_{t+1} \leftarrow \{(i, j, k) \notin S : (i, j, k) \text{ in } \mathcal{F}_t\}$
    $t \leftarrow t + 1$
end while
Reconstruct mask:
for all $(i, j) \in \mathcal{P}$ **do**
    if $\text{mean}(\mathcal{P}[(i, j, k)]) \geq \tau$ **then**
        $\hat{Y}(i, j, k) \leftarrow 1$
    end if
end for
Remove padding by cropping in each direction by 20 of $\hat{Y}$ to recover the original mask volume
return $\hat{Y}$

new set of centroids. The same process repeats — for each new centroid, we take a snap, run it through the model, and update the mask predictions. This continues until there are no new voxels being added to the growing region. Throughout the process, we keep track of which voxels have already been visited to make sure each voxel is processed only once. If multiple predictions overlap at the same voxel, we simply average their prediction values.

4.4.2 Post Processing

To evaluate the performance of the proposed segmentation approach, we adopt a set of standard metrics for 3D medical image segmentation. Following the framework proposed by [56], the evaluation is based on four fundamental cardinalities: True Positives (TP) which is voxels correctly predicted as vessel, False Positives (FP) which is voxels incorrectly predicted as vessel, True Negatives (TN) which is voxels correctly predicted as background, and False Negatives (FN) which is voxels incorrectly predicted as background.

The Dice Coefficient [56] is the most widely used metric for validating medical volume segmentations. It measures the spatial overlap between the predicted segmentation and the ground truth. A Dice score of 1 indicates perfect overlap, while a score of 0 indicates no overlap and it is defined as:

$$\text{Dice} = \frac{2 \cdot \text{TP}}{2 \cdot \text{TP} + \text{FP} + \text{FN}} \tag{4.1}$$

The Jaccard Index [56], also known as Intersection over Union (IoU), is defined as the intersection between the predicted and ground truth segmentations divided by their union:

$$\text{Jaccard} = \frac{\text{TP}}{\text{TP} + \text{FP} + \text{FN}} \tag{4.2}$$

While the Dice coefficient measures the volumetric overlap, it does not access the connectivity of the vascular structures, as it does not weigh the vessels of varying radii [52, 60]. This is particularly important here when segmenting vascular structures for preserving topology and connectivity. To address this, the Centerline Dice (clDice) metric was proposed by [52], which measures the overlap between the predicted segmentation mask and the vessel centerline, ensuring the topological continuity is captures in the evaluation. Let V_P and V_L denote the predicted and ground truth segmentation masks. Skeletonization is a process that reduces each mask to 1 voxel thin

centerline representation that preserves the topological structure of the original shape such as connectivity while discarding its thickness. Let S_P and S_L denote their respective skeletons, extracted via a skeletonization algorithm. The clDice is computed as the harmonic mean of two overlap measures [52]:

$$T_{\text{prec}}(S_P, V_L) = \frac{|S_P \cap V_L|}{|S_P|} \tag{4.3}$$

$$T_{\text{sens}}(S_L, V_P) = \frac{|S_L \cap V_P|}{|S_L|} \tag{4.4}$$

$$\text{clDice}(V_P, V_L) = \frac{2 \times T_{\text{prec}}(S_P, V_L) \times T_{\text{sens}}(S_L, V_P)}{T_{\text{prec}}(S_P, V_L) + T_{\text{sens}}(S_L, V_P)} \tag{4.5}$$

where T_{prec} measures the proportion of the predicted skeleton that lies within the ground truth mask, and T_{sens} measures the proportion of the ground truth skeleton that lies within the predicted mask. A clDice score of 1 indicates perfect topological overlap, while a score of 0 indicates complete failure to capture the vessel centrelines. While clDice is commonly computed on 2D slices in 3D settings, the vascular structures in our dataset exhibit connectivity in 3D rather than within individual 2D slices. We therefore perform skeletonization directly on the full 3D volume to capture the topological continuity of the vascular structure.

4.5 Results and Analysis

This section presents the quantitative and qualitative evaluation of the proposed 3D region growing segmentation algorithm on a single test images from the synthetic vascular dataset. Table 4.2 reports a Dice score of 0.9985, Jaccard index of 0.9971, and clDice of 0.9989, all of which are close to 1, indicating near-perfect segmentation performance. These high scores are expected given the synthetic nature of the dataset. However, a closer inspection of the predictions reveals that the algorithm incorrectly classified an entire vessel as background, as illustrated in Figure 4.5.

Figure 4.5 presents the qualitative results of the 3D region growing algorithm. The dataset described in Section 4.2 consists of dense synthetic vascular structures that are connected in 3D but not within individual 2D planes. As the dataset is synthetic, the predicted segmentations are close to perfect, as reflected in Table 4.2, making missed vessels difficult to visualise directly. To better highlight segmentation errors, all volumes

Metric	Score
Dice	0.9985
Jaccard	0.9971
clDice	0.9989

Table 4.2: Quantitative evaluation results of the proposed 3D region growing segmentation approach on test volume.

in Figure 4.5 are cropped to the z-range of the largest false negative region and rendered in the YZ plane. From left to right: the first panel shows the cropped raw image volume; the second panel shows the corresponding ground truth mask; the third panel shows the predicted segmentation overlaid with false negatives (yellow) and false positives (red); and the fourth panel shows a zoomed-in view of the largest missed vessel.

Figure 4.6 presents the confusion matrix of the segmentation results. The algorithm correctly identified the vast majority of vessel voxels, achieving a true positive rate of 9.98×10^{-1} (1,185,596 voxels) and a true negative rate of 9.9998×10^{-1} (58,090,941 voxels), indicating near-perfect classification of both vessel and background. The false negative rate of 1.73×10^{-3} (2,060 voxels), combined with a false positive rate of 2.42×10^{-5} (1,403 voxels), confirms that the algorithm rarely misclassified background as vessel.

4.6 Discussion

In this work, we presented 3D-VISA, an extension of VISA to three-dimensional volumetric data for segmenting dense vascular structures in volumetric images. Following the same principle as its 2D counterpart, the approach extracts overlapping volumetric snaps of size $40 \times 40 \times 40$ from the image volume, providing the network with sufficient local context to capture the fine structure of thin but densely connected vessels. Each snap is passed through a trained 3D CNN, which produces a corresponding $3 \times 3 \times 3$ output from the mask volume. The 3D RGA then expands from seed voxels in the direction of foreground predictions, discarding background voxels and reducing computation time. This design ensures that the network learns not only voxel-level predictions but also the surrounding spatial context, which is critical for preserving the topological connectivity of vascular structures.

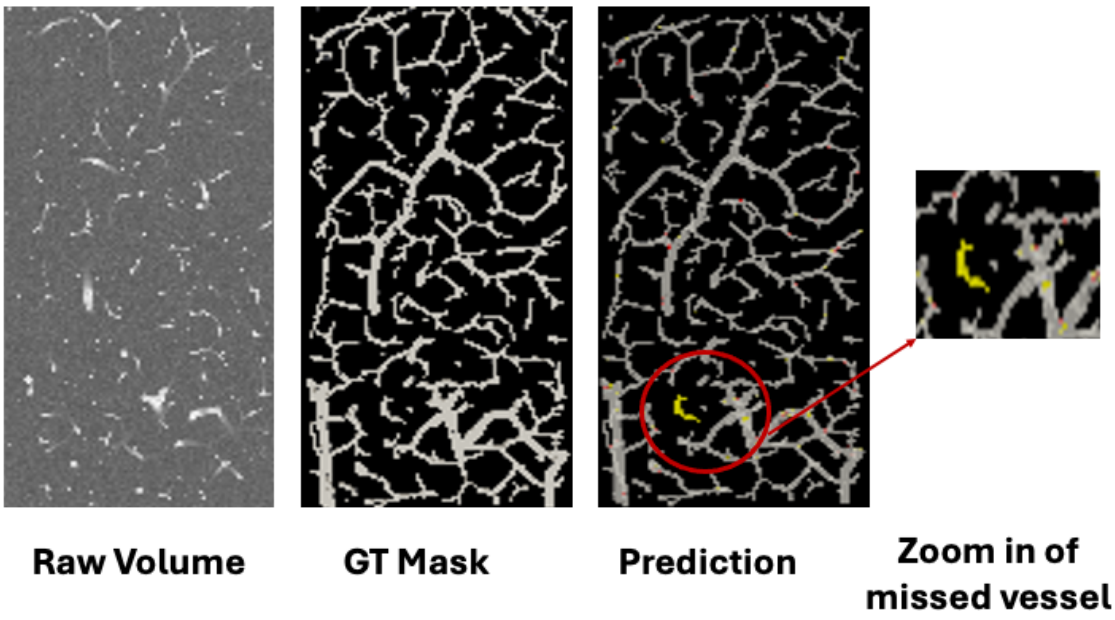


Figure 4.5: Qualitative results of the 3D region growing algorithm. All volumes are cropped to the z-range of the largest missed vessel and rendered in the YZ plane. From left to right: raw image volume, ground truth mask, predicted segmentation with false negatives (yellow) and false positives (red), and a zoomed-in view of the largest missed vessel.

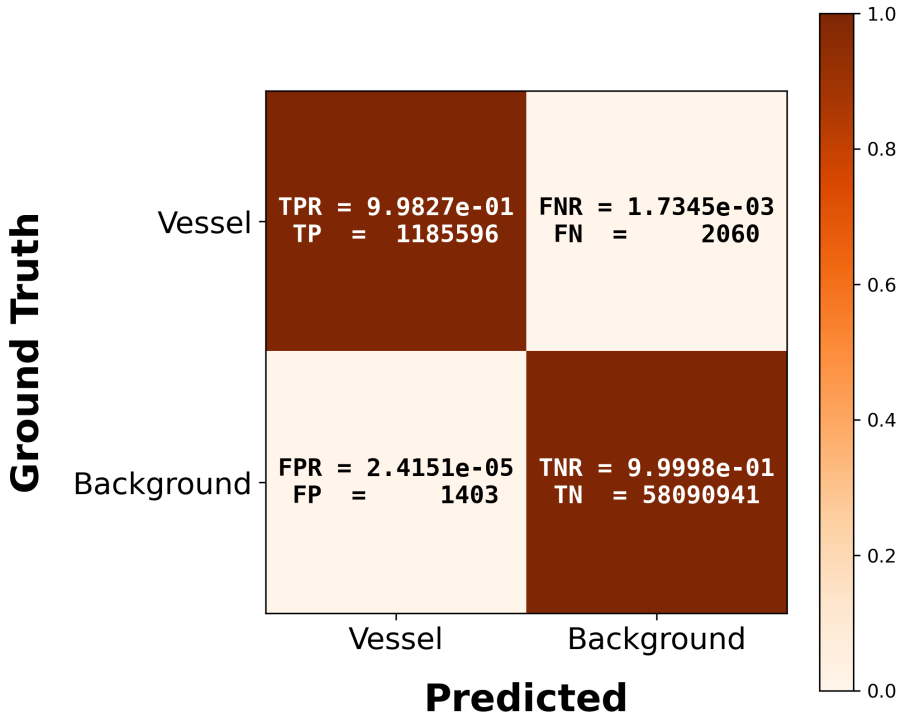


Figure 4.6: Confusion matrix: normalized across the rows; evaluating the proposed 3D vessel segmentation framework against ground truth labels. Quadrants display both absolute voxel counts and normalized rates: True Positive Rate (TPR), False Negative Rate (FNR), False Positive Rate (FPR), and True Negative Rate (TNR).

We present a proof-of-concept for 3D-VISA on a synthetic vascular dataset. The initial results are promising, with near-perfect Dice, Jaccard, and clDice scores while training on a single image. However, the qualitative analysis revealed that the algorithm failed to segment one vessel entirely, highlighting that volumetric overlap metrics alone may not capture all failure modes. Future work will focus on evaluating the algorithm on a larger number of volumes and validating its performance on real clinical datasets, where vascular structures are more variable and imaging conditions are less controlled.

4.7 Acknowledgments

This research [Part of this research] was conducted using Pinnacles (NSF MRI, # 2019144) at the Cyberinfrastructure and Research Technologies (CIRT) at University of California, Merced.

Chapter 5

Conclusions and Future Work

This thesis presented three region growing-based pipelines to address the challenging task of vascular image segmentation. Segmenting tubular structures such as blood vessels remains difficult due to two key problems: the scarcity of labeled biomedical data and the tendency of standard pixel-wise segmentation approaches to produce fragmented predictions that fail to preserve vascular connectivity. The three pipelines introduced in this thesis address both of these challenges by combining local patch-based prediction with a region growing algorithm, generating large numbers of training samples from limited data while ensuring spatially contiguous segmentations.

In the first part of the thesis we introduced VISA (Vascular Image Segmentation Architecture), a two-phase pipeline for segmenting 2D Digital Subtraction Angiography (DSA) images. VISA addresses the data scarcity problem by extracting densely overlapping patches, which we called "snaps", centered on every pixel in the image, generating a large number of training samples from a limited number of annotated images. In the first phase, a 2D CNN is trained on these snaps to predict a 3×3 output mask from an 80×80 input snap, capturing the local neighborhood around each pixel rather than making an isolated pixel-wise prediction. In the second phase, the trained CNN is used within the Region Growing Algorithm (RGA), which iteratively expands only in the direction of foreground predictions, ensuring that the final segmentation is spatially contiguous. While VISA is computationally more expensive than standard U-Net due to the iterative nature of the RGA, and its performance is sensitive to the choice of initial seed points, it consistently demonstrates better preservation of vascular connectivity as measured by clDice scores.

To address these limitations, we introduced UI-VISA (U-Net Initialized Vascular

Image Segmentation Architecture), where a modified U-Net architecture first segments the image and the resulting foreground predictions are then used in the second phase as informed initial seeds for the RGA, replacing the random initialization of standalone VISA. This pipeline showed significant improvement in vascular connectivity compared to both standalone VISA and U-Net, as measured by Dice and clDice scores across most folds, while also reducing computation time compared to standalone VISA.

In the biomedical imaging field, most clinical data is three-dimensional, as imaging techniques such as CT and MRI acquire volumetric data. To extend the capabilities of VISA to this setting, we introduced 3D-VISA, an extension of VISA to three-dimensional volumetric data. As a proof of concept, we started with one pair each of volumetric image and its ground truth mask for training, validation and testing from a synthetic vascular dataset. In the first phase, $40 \times 40 \times 40$ snaps were extracted from the image volume and corresponding $3 \times 3 \times 3$ snaps from the mask volume to train the 3D CNN. In the second phase of 3D-VISA, the trained 3D CNN was used on the test volume along with the 3D RGA to produce the final volumetric segmentation. The results were promising, with near-perfect Dice, Jaccard, and clDice scores, demonstrating that the two-phase VISA design generalizes naturally to three-dimensional data.

Each of the three pipelines introduced in this thesis have room for improvement. For VISA, in phase two, the current approach averages overlapping snap predictions for each pixel, and future work could explore different aggregation strategies for overlapping pixel predictions such as voting schemes to improve segmentation accuracy. For UI-VISA since the seeds are currently initialized from all U-Net foreground predictions, a more targeted strategy such as restricting seed selection to boundary pixels between predicted foreground and background regions could further reduce the number of iterations required for convergence, bringing the computational cost closer to that of standalone U-Net while retaining the connectivity preserving properties of the region growing approach. Future work for 3D-VISA will focus on evaluating the algorithm on a larger number of volumes and validating its performance on real clinical datasets, where vascular structures are more variable and imaging conditions are less controlled than in synthetic data. This will provide a more rigorous assessment of the generalizability of 3D-VISA in realistic clinical settings.