# Enhancing Cardiovascular Disease Diagnosis Through Deep Learning Analysis Of Retinal Images

Pranav Gujjar - 100443924

A dissertation submitted to
The School of Computing Sciences of the University of East Anglia
in partial fulfilment of the requirements for the degree of
**MASTER OF SCIENCE.**

AUGUST, 2024

# SUPERVISOR(S), MARKERS/CHECKER AND ORGANISER

The undersigned hereby certify that the markers have independently marked the dissertation entitled "Enhancing Cardiovascular Disease Diagnosis Through Deep Learning Analysis Of Retinal Images" by Dr. Daniel Paredes-Soto, and the external examiner has checked the marking, in accordance with the marking criteria and the requirements for the degree of Master of Science.

**Supervisor** **:** Dr. Daniel Paredes-Soto

**Markers** **:**

Marker 1: Dr. Daniel Paredes-Soto

Marker 2: Dr. Steven Hayward

**External Examiner:**

Checker/Moderator

**Moderator** **:** Dr. Ben Milner

# DISSERTATION INFORMATION AND STATEMENT

Dissertation Submission Date: August, 2024

**Student:** Pranav Gujjar

**Title:** Enhancing Cardiovascular Disease Diagnosis Through Deep Learning Analysis Of Retinal Images

**School:** Computing Sciences

**Course:** Data Science

**Degree:** MSc.

**Duration:** 2023–2024

**Organiser:** Dr. Ben Milner

**STATEMENT:**
Unless otherwise noted or referenced in the text, the work described in this dissertation is, to the best of my knowledge and belief, my own work. It has not been submitted, either in whole or in part for any degree at this or any other academic or professional institution. Subject to confidentiality restriction if stated, permission is herewith granted to the University of East Anglia to circulate and to have copied for non-commercial purposes, at its discretion, the above title upon the request of individuals or institutions.

Pranav Gujjar

Signature of Student

# ABSTRACT

Cardiovascular diseases (CVDs) remain the primary cause of death worldwide, emphasizing the critical need for new diagnostic tools. This dissertation investigates the use of deep learning techniques, specifically the U-Net and Dense U-Net models, to evaluate retinal images for early CVD identification. The study's aim is to produce a non-invasive, efficient, and cost-effective diagnostic tool using retinal imaging, which detects microvascular abnormalities suggestive of systemic disorders. Existing literature highlights the promise of AI in improving diagnostic accuracy and efficiency, with deep learning models outperforming traditional methods in medical image processing.

This research uses Python and the TensorFlow framework to create and train deep learning models on the DRIVE dataset. Preprocessing steps include image normalization, contrast enhancement, green channel extraction, and feature extraction with OpenCV and scikit-image libraries. Data augmentation techniques such as flipping and rotating are used to increase model adaptability. The U-Net model has an encoder-decoder architecture with layers designed to capture both fine details and global context, whereas the Dense U-Net uses dense blocks to improve feature propagation. Important parameters include a learning rate of 0.0001, a batch size of 8, a patch size of 128x128, number of patches , normalization is true and the use of binary cross-entropy with Dice loss for training. To train a model, the dataset is divided into training and validation sets. After training, the saved model is used to evaluate the test dataset. This process ensures generalization and reliability.

The Proposed Framework shows that the U-Net model beats the Dense U-Net model with a Sensitivity of 79.72%, an F1 score of 80.25%,a Jaccard index of 67.01% , an AUC of 97.34%. Dice coefficient of 77.84%, and an accuracy of 95.71%. These measures demonstrate great segmentation quality and diagnosis accuracy, supporting the usefulness of the proposed technique. The Dense U-Net model also performed well, supporting the use of deep learning in retinal image processing for early CVD identification and highlighting its potential for real-world clinical applications.

# ACKNOWLEDGEMENTS

I would like to express my deepest gratitude to everyone who has supported and guided me throughout this dissertation.

First and foremost, I am incredibly grateful to my supervisor, Dr. Daniel Paredes-Soto, for his unwavering support, insightful feedback, and constant encouragement. Your expertise and guidance have been invaluable in shaping this research and bringing it to fruition.

I am deeply appreciative of the great support from my family, especially my wife, parents, in-laws and friends. Their encouragement and understanding have been a constant source of strength throughout this journey.

I would also like to acknowledge the contributions of my colleagues and peers who provided me with helpful discussions and feedback. Your camaraderie and intellectual exchange have been essential to this academic endeavor.

Additionally, I would like to thank the faculty and staff of the Department of Computing Science at the University of East Anglia for providing the resources and a conducive environment for my research.

Thank you all for your invaluable contributions and support. This dissertation would not have been possible without you.

Pranav Gujjar

Norwich, UK.

# Contents

# 1 Introduction

This dissertation investigates the development and execution of deep learning methods to improve retinal image analysis-based cardiovascular disease (CVD) detection. Since cardiovascular diseases are the world's largest cause of mortality, more effective, non-invasive, and affordable diagnostic methods must be developed. Retinal imaging presents a promising approach for early detection of CVD, as it can reveal microvascular abnormalities indicative of systemic diseases. This study focuses on using deep learning models—more especially, U-Net and Dense U-Net architectures—to analyze retinal images and increase the accuracy and efficiency of CVD diagnosis. Metrics like the Dice coefficient and Jaccard index are used in the study to assess model performance, which is done through training and evaluation on the DRIVE dataset. The ultimate goal is to show how these AI models can be used clinically to transform non-invasive cardiovascular diagnosis.
The dissertation consists of into many essential sections, beginning with an introduction that describes the study's motivation, aim, and objectives. Following that, a literature review analyzes existing research on retinal imaging, deep learning in medical diagnostics, and current methodologies' challenges and limits. The methodology part describes the preprocessing methods, dataset selection, and deep learning model design, as well as technical aspects such as U-Net and Dense U-Net architectures. Experimental data are subsequently shown, demonstrating the models' performance in various configurations. The Performance comparison with similar work section compares the study's findings to previous research, highlighting the work's contributions to the field. Furthermore, the dissertation wraps up with a summary of the findings, implications for clinical practice, and recommendations for future study, noting the potential of AI-enhanced retinal imaging to transform cardiovascular disease detection and medical treatment.

## 1.1 Motivation

Cardiovascular diseases (CVDs) are the leading cause of death worldwide, accounting for 17.9 million deaths per year, including conditions such as coronary heart disease and stroke WHO (2024a). Unhealthy diets, physical inactivity, smoking, and excessive alcohol use all contribute to high blood pressure and cholesterol levels, highlighting the importance of lifestyle changes and effective health policies WHO (2024b). The World Health Organization promotes for worldwide initiatives to reduce CVD incidence and mortality, with a focus on early identification, access to healthcare, and risk factor treatment, particularly in low and middle-income countries with the largest burden WHO (2024b). Traditional techniques of CVD screening are time-consuming and error-prone. However, artificial intelligence (AI), which uses machine learning and image processing, provides more accurate and consistent medical image analysis, increasing assessment reproducibility and improving early detection and patient outcomes Luan Xue-Ning and et al. (2024); Rao G. M. and et al. (2024).
In addition, integrating AI into healthcare systems can dramatically reduce medical practitioners' workloads by automating common methods and freeing them up to focus on more challenging cases. This technology innovation not only improves healthcare delivery efficiency, but it also assures that patients obtain fast and precise diagnoses, which are critical for the proper treatment and management of cardiovascular diseases.
The combination of retinal imaging and AI promises a game-changing approach to cardiovascular disease diagnosis. This approach detects small changes in the retina that indicate systemic diseases, making it a non-invasive, cost-effective, and rapid screening tool. Early diagnosis using retinal imaging can lead to quick intervention, dramatically improving patient outcomes and lowering healthcare expenditures. Likewise this new technique has the potential to democratize access to high-quality medical diagnostics, particularly in underserved areas, so addressing global health inequities and contributing to more equitable healthcare outcomes.

### 1.2 Aim

Design, develop and validate a deep learning model for automatically analysing retinal images to support the diagnosis and monitoring of a constrained vascular-related diseases.

### 1.3 Objectives

- Examine and select appropriate datasets that are available to the public for the model's training and testing ( Available Datasets like DRIVE DRIVE (2024), STARE Hoover et al. (2000), CHASE DB1 Fraz et al. (2012)).
- Examine modern deep learning approaches and models for image segmentation analysis to choose the best model for this project.
- Design a deep learning model to segment retinal image vascular networks using the TensorFlow framework.
- Use performance metrics such as Accuracy, DICE and Jaccard indices to evaluate the model's accuracy and efficiency.

## 2 Literature Review

This literature review explores the relationship between retinal imaging and cardiovascular disease diagnosis, focusing on the use of cutting-edge technologies such as deep learning. It discusses the ethical issues surrounding computerized healthcare solutions, evaluates the precision and efficiency of current diagnostic tools, and scores the performance of AI models. The review highlights retinal imaging's great potential for improving the early detection and management of cardiovascular disorders. It also analyzes current obstacles, explores their implications for clinical practice, and outlines future research possibilities, such as the integration of multimodal data and the establishment of standardized protocols.

### 2.1 Retinal Imaging And Cardiovascular Disease

As blood vessels are clearly visible in the retina, retinal imaging is critical in the diagnosis of systemic diseases, particularly cardiovascular diseases. Fundus photography and optical coherence tomography are techniques that provide precise views of the retina's microvascular structure, assisting in the early diagnosis of disorders such as hypertension and diabetic retinopathy (DR) Salz David A. (2015). These imaging techniques allow clinicians to examine and study the fine characteristics of retinal blood vessels, revealing underlying cardiovascular health concerns. AI-enhanced image analysis improves this diagnostic power by enabling for exact identification of small vascular modifications, which is critical for prevention in cardiovascular diseases. AI can detect minute irregularities that human observation alone would miss, enhancing diagnostic accuracy and timeliness Abràmoff Michael D. and et al. (2010). This technological development promotes proactive cardiovascular health care, emphasizing the need of combining artificial intelligence with standard retinal imaging techniques.

### 2.2 Uses Of Retinal Images In Medical Diagnostics

Retinal images are becoming more popular in medical diagnostics to detect and control a wide range of ocular and systemic diseases. Imaging advances such as optical coherence tomography (OCT) and fundus photography enable detailed viewing of retinal structures, allowing for the diagnosis of disorders such as diabetic retinopathy, age-related macular degeneration, and glaucoma. For example, Abràmoff et al. (2018) revealed that an AI-based diagnostic system could

diagnose diabetic retinopathy with a sensitivity of 87.2% and a specificity of 90.7%, allowing for timely referral and treatment. Furthermore, retinal imaging is being investigated for the detection of systemic diseases such as cardiovascular problems, as changes in retinal microvasculature can indicate overall vascular health lui Cheung et al. (2017).

## 2.3 Clinical Integration And Real-World Deployment

Furthermore, early diagnosis and treatment results for individuals with retinal diseases may be improved by incorporating AI models into clinical procedures. An AI system, for example, may detect diabetic retinopathy(DR) with high accuracy and recommend particular referral actions, potentially relieving ophthalmologists of some of their duty De Fauw et al. (2018). To guarantee continued performance and dependability in a variety of clinical contexts, real-world deployment of these models necessitates careful consideration of elements including data unpredictability, ethical issues, and the requirement for ongoing model validation and upgrades.

## 2.4 Deep Learning Approaches In Retinal Image Analysis

The identification of cardiovascular disease through retinal image processing has been transformed by deep learning, namely with the use of convolutional neural networks (CNNs). CNNs are useful for detecting microaneurysms or hemorrhages associated with diseases like hypertension and diabetic retinopathy because they can detect minute changes in vascular architecture through complex pattern analysis Hoque et al. (2019). In terms of accuracy and dependability, these models beat conventional machine learning techniques and even certain ophthalmologists Hoque and Kipli (2021).Cutting-edge models such as U-Net, Dense U-Net and DeepLab have greatly improved image segmentation performance, allowing for early CVD treatment and faster clinical diagnostic procedures Al-Absi Hamada R. H. and et al. (2022). With prompt intervention, these deep learning algorithms may improve patient outcomes by enabling more accurate and efficient examination of retinal images.

### 2.4.1 Comparison With Conventional Machine Learning Methods

When examining retinal images for the detection of cardiovascular disease (CVD), deep learning (DL) algorithms perform significantly differently from conventional machine learning (ML) methods. A support vector machine (SVM) method was found by Poplin et al. (2018) to be 75% accurate in classifying retinal pictures for the purpose of detecting CVD. Gulshan et al. (2016), on the other hand, showed the potential for high accuracy in related CVD applications by demonstrating that a deep convolutional neural network (CNN) could diagnose referable diabetic retinopathy with an accuracy of 96%.

### 2.4.2 Recent Advances In Cardiovascular Risk Prediction

Deep learning models have made substantial progress in predicting cardiovascular diseases (CVD) using retinal images, according to recent studies using extensive performance measures. For example, a systematic evaluation of several research by Li et al. (2024) discovered that deep learning models reliably attain sensitivity of 85-90%, specificity of 80-85%, and area under the curve (AUC) of 0.90-0.95 for detecting several cardiovascular risk indicators. Deep learning is an essential tool for modern medical diagnostics because of its high AUC, sensitivity, and specificity rates, which highlight its potential as a tool for early detection and preventative healthcare methods.Using deep learning for direct cardiovascular risk prediction from retinal images, a recent study by Grangeat et al. (2020) achieved a predictive accuracy of 78%, which is a notable improvement over traditional ML methods, whose accuracy ranged between 65-80% Abràmoff et al. (2016).

## 2.5 Performance Evaluation Of Artificial Intelligence Models Using Retinal Images

Artificial intelligence (AI) has made significant advances recently in the processing of retinal images, especially in the identification of eye conditions including age-related macular degeneration (AMD) and diabetic retinopathy (DR). For example, Gulshan et al. (2016) found that a deep learning algorithm was able to detect referable DR with a sensitivity of 90.3% and a specificity of 98.1%, which is in close agreement with human ophthalmologists' performance. Similar results were reported by Gargeya and Leng (2017), who found that their convolutional neural network (CNN) was able to detect DR from retinal images with a sensitivity of 94% and a specificity of 98%.

### 2.5.1 Comparative Analyses Of AI Model Effectiveness

Comparative analyses show that, depending on their training datasets and architectures, various AI models display differing degrees of efficacy. When Ting et al. (2017) tested the effectiveness of multiple AI systems, for instance, they discovered that an ensemble model performed better than a single CNN model, with an AUC (area under the curve) of 0.936 for DR detection. A hierarchical deep learning model, on the other hand, surpasses conventional machine learning techniques, as demonstrated by a study by Li et al. (2018), which found that it could diagnose AMD with a sensitivity of 95.5% and specificity of 92.6%. These comparison studies highlight how crucial it is to use the best model designs and training strategies in order to maximize diagnostic accuracy.

## 2.6 Challenges And Limitations In Actual Medical Image Analysis Research

The requirement for large, high-quality annotated datasets is one of the main obstacles facing modern AI research, especially in the field of medical imaging. The amount and caliber of training data have a significant impact on how well deep learning models perform, as stated by Esteva et al. (2017). However, Privacy concerns, limited availability, and the need for expert annotations—which are costly and time-consuming—often make accessing such datasets difficult in the medical area. The creation and verification of reliable AI models may be complicated by this limitation.

The capacity of AI models to be generalized presents another important difficulty. According to research by Zech et al. (2018), models that are trained on certain datasets frequently exhibit performance issues when applied to data from other sources or populations. Variations in imaging methods, tools, and patient demographics among various treatment environments give rise to this generalizability problem. As a result, AI models may perform less accurately and consistently in various medical environments, which would restrict their general applicability and usefulness.

Furthermore, a major worry with AI models is their interpretability. A lot of AI systems, especially deep learning models, function as "black boxes," offering little information about how they make particular judgments. According to Lipton (2018), this lack of openness might be problematic in clinical settings because it is essential for clinicians to trust and use AI tools properly to grasp the reasons behind a diagnosis. Although they haven't yet been fully incorporated into common applications, efforts are still being made to provide more comprehensible models or methods for explaining AI judgments.

## 2.7 Ethical And Regulatory Considerations

The use of Artificial Intelligence in the healthcare sector is also severely constrained by ethical and legal issues. It is imperative to tackle concerns like consent, data privacy, and bias in AI

algorithms to guarantee the fair and secure utilization of these technologies. The significance of creating extensive regulatory frameworks that can adjust to the quickly changing landscape of artificial intelligence in healthcare is emphasized by a study by Gerke et al. (2020). Without these frameworks, there may be major challenges in integrating AI into clinical practice, which could postpone the advantages these technologies have to offer. This dissertation does not collect any data directly; instead, it depends entirely on publicly available data. As a result, no further ethical considerations are required in the circumstance of this study.

# 3 Methodology:

Figures 1(p.13) and 2(p.14) show a comprehensive method for utilizing deep learning to evaluate medical images using vascular network segmentation. Retinal imaging is especially valuable for detecting cardiovascular abnormalities.

## 3.1 Preprocessing

Preprocessing is an important step in preparing data for deep learning research. It begins with selecting appropriate tools, such as programming languages and publicly available datasets like DRIVE DRIVE (2024), followed by the use of essential libraries. Key processes include reading and normalizing images, extracting key features, and performing patch extraction to identify regions of interest. Furthermore, data augmentation techniques, mask application, and visualization of extracted features are used to enhance the dataset, ensuring that it is ready for robust model training and precise analysis.

### 3.1.1 Programming Language And Libraries

Advanced machine learning models are frequently developed using Python along with the deep learning framework TensorFlow Abadi et al. (2015). For tasks like image segmentation, natural language processing, and predictive analytics, TensorFlow is one of the best options since it provides all the tools needed to create, train, and run neural networks. A scalable and effective method for developing deep learning applications is to use TensorFlow combined with Python. Strong and effective machine learning solutions can be created with TensorFlow's capabilities together with Python's many libraries and adaptability Abadi et al. (2016).
**The following libraries are necessary to create deep learning models in image processing and segmentation tasks:** `matplotlib.pyplot` Hunter (2007a) for charting, `os` for file operations, `cv2` (OpenCV) Bradski (2000) for image processing, `numpy` Harris et al. (2020) for numerical computations, and `pandas` McKinney et al. (2010) for data handling. Functions for color space conversion, feature extraction, and exposure control are available in the `skimage` Van der Walt et al. (2014) library. `random` Van Rossum (2020) helps to generate random numbers, and `imageio` Klein (2018) manages image I/O (Input/Output). Data splitting and model evaluation are supported by `sklearn.model_selection` Pedregosa et al. (2011) and `sklearn.metrics`, respectively. `Tensorflow.keras` Abadi et al. (2015), a component of the Tensorflow framework, provides tools for configuring and training neural networks, while `scipy.ndimage` Virtanen et al. (2020) handles image processing tasks including morphological operations.

### 3.1.2 Selection Of Publicly Available DRIVE Dataset

The DRIVE DRIVE (2024) dataset was chosen above STARE Hoover et al. (2000) and CHASE DB1 Fraz et al. (2012)because of its standard format, balanced amount of training and testing images, and extensive annotations, which provide a consistent benchmark for evaluating

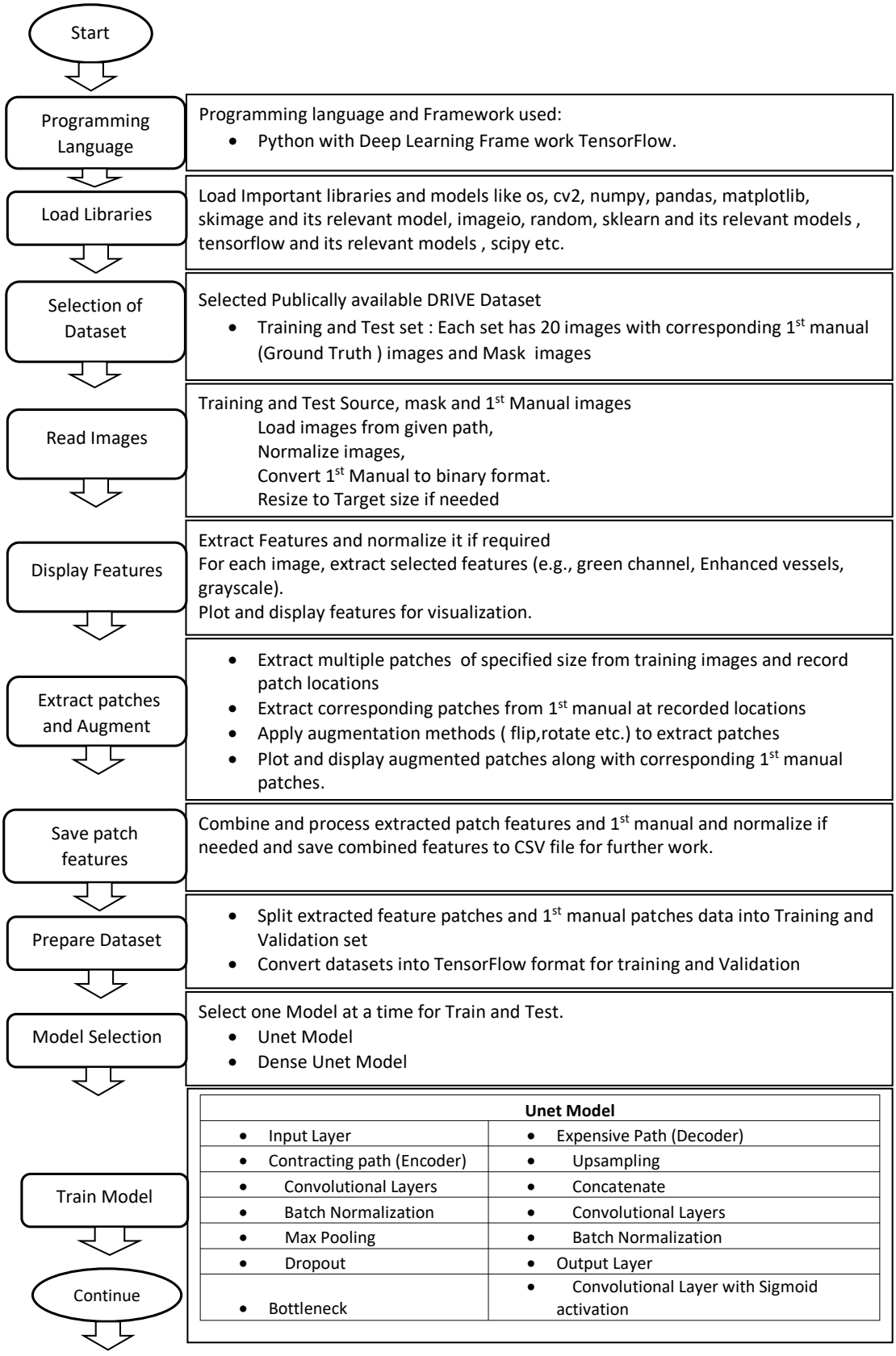



Figure 1: Project Methodology part 1

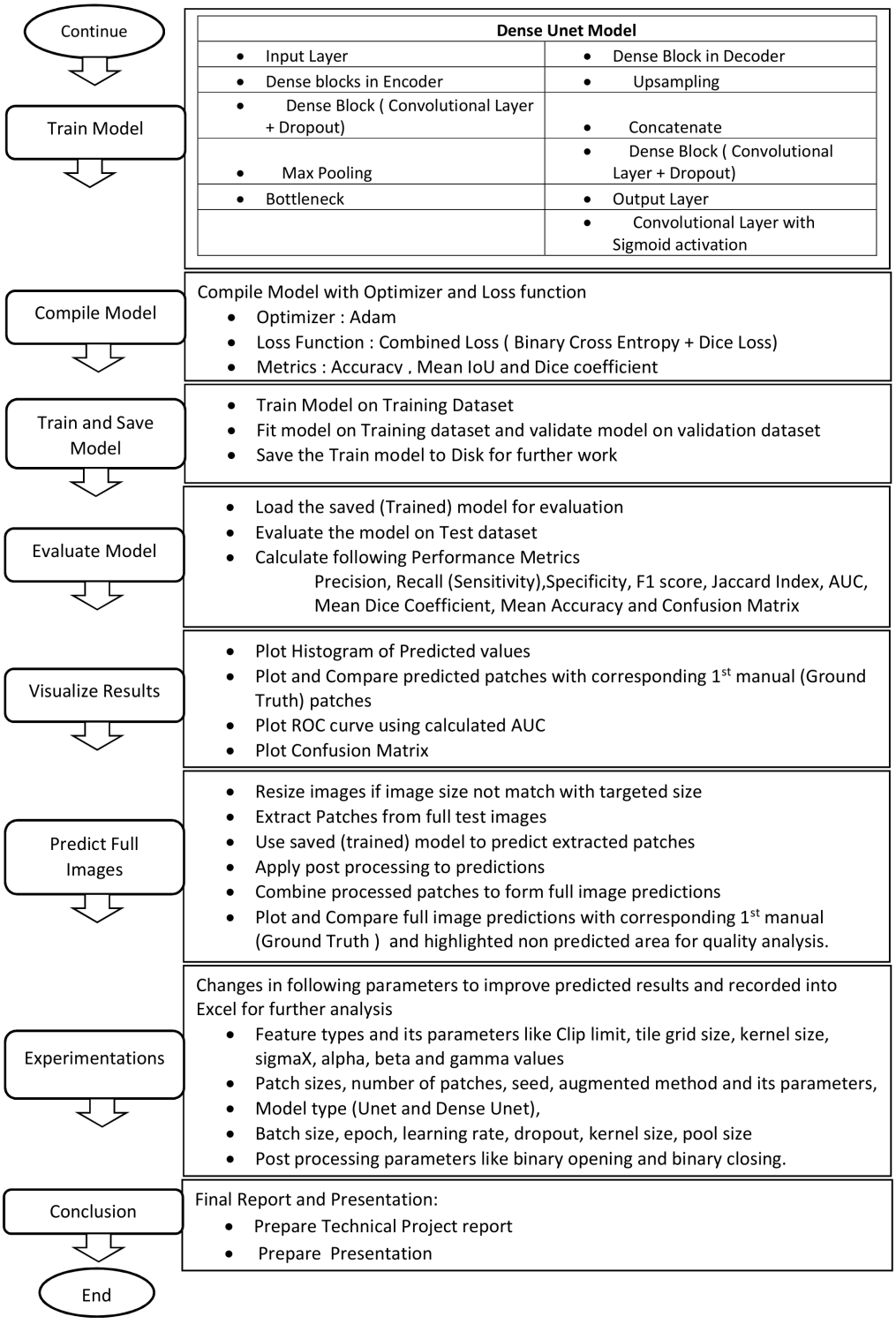



Figure 2: Project Methodology part 2

segmentation algorithms. Furthermore, DRIVE's well-documented and freely accessible dataset promotes reliability and progress in retinal image analysis Staal et al. (2004).The DRIVE (Digital Retinal Images for Vascular Extraction) dataset is frequently used in retinal vascular segmentation studies. It contains 20 training and 20 test images, each with the 1st manual (Ground Truth) annotations and mask images. High-resolution images and properly documented ground truth are key aspects required for accurate model training and validation. Ground truth annotations improve the training process by giving accurate labels for supervised learning.For illustration purpose of training and test image samples in Figure 3(p.16).

### 3.1.3 Read Images

Preprocessing and loading images from designated folders will be performed using an approach that ensures image standardization for consistent model training and evaluation. This method supports both conventional image formats like .tif and GIFs, reading images and organizing them to ensure alignment with manual annotations. Images are normalized to a pixel value range of [0, 1], resized to the necessary dimensions, and transformed from BGR to RGB color space. Binary format conversion is an option. A list of the processed images and their filenames is the output, which is necessary to provide a consistent dataset for deep learning model training, which calls for consistent input dimensions. The preprocessing methods supported by Ronneberger et al. (2015) , who stated the value of consistent image sizes and formats in enhancing model performance and training consistency, are in line with this methodology. Their study shows how important it is to standardize input images through processes like scaling, normalization, and binary conversion in order to facilitate efficient training of deep learning models. Such standardization ensures that models can correctly identify between regions of interest and background in tasks such as image segmentation Ronneberger et al. (2015).

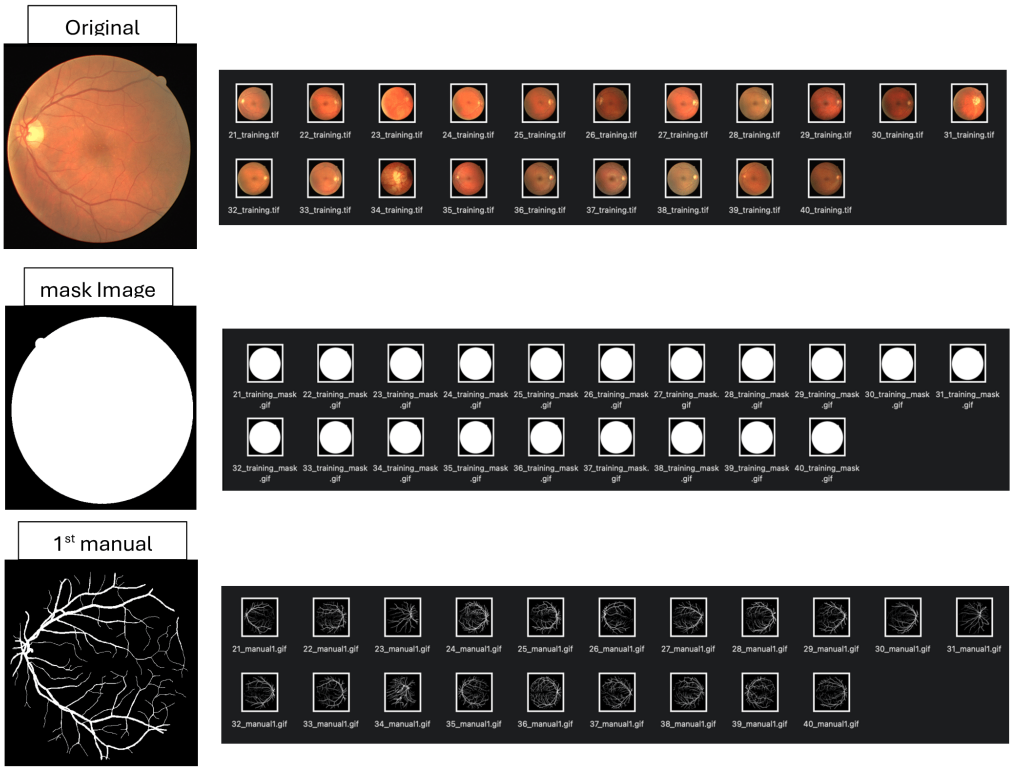
### 3.1.4 Feature Extraction And Normalization

Based on the specified feature type,extracts particular features from the image, focusing important structures for segmentation. Various feature extractions can enhance model performance by offering a variety of data representations. This feature activates by:

**1. Green Channel Extraction:** This method uses image[:, :, 1] to isolate the green channel, which frequently improves contrast for retinal blood vessels Zhao et al. (2015).

**2. Enhanced Vessels:** This technique enhances blood vessels in retinal images by initially extracting the green channel, which provides high vessel contrast. It then uses CLAHE (Contrast Limited Adaptive Histogram Equalization) Pizer et al. (1990) to boost local contrast, followed by Gaussian blurring to smooth out noise. Finally, weighted addition is used to combine the expanded and blurred images, highlighting vessel edges. This method prevents noise through restricting contrast amplification, dividing the image into small tiles for local enhancement, smoothing noise while keeping fine structures, and automatically calculating the standard deviation for blurring. These stages produce a more detailed image of blood vessels for subsequent analysis. Various research studies have backed the strategy of increasing retinal blood vessels with CLAHE (Contrast Limited Adaptive Histogram Equalization) and Gaussian blur. For example, a study found that applying CLAHE combined with Gaussian filters improved the appearance and segmentation of blood vessels in retinal images, making them more distinct while lowering noise Sonali et al. (2019). Another study highlights the use of CLAHE to improve contrast in the green channel, which is critical for retinal image analysis Orlando and Blaschko (2014).

Training Dataset contain original images, mask images and 1st manual (Ground Truth) images




Test Dataset contain original images, mask images and 1st manual (Ground Truth) images

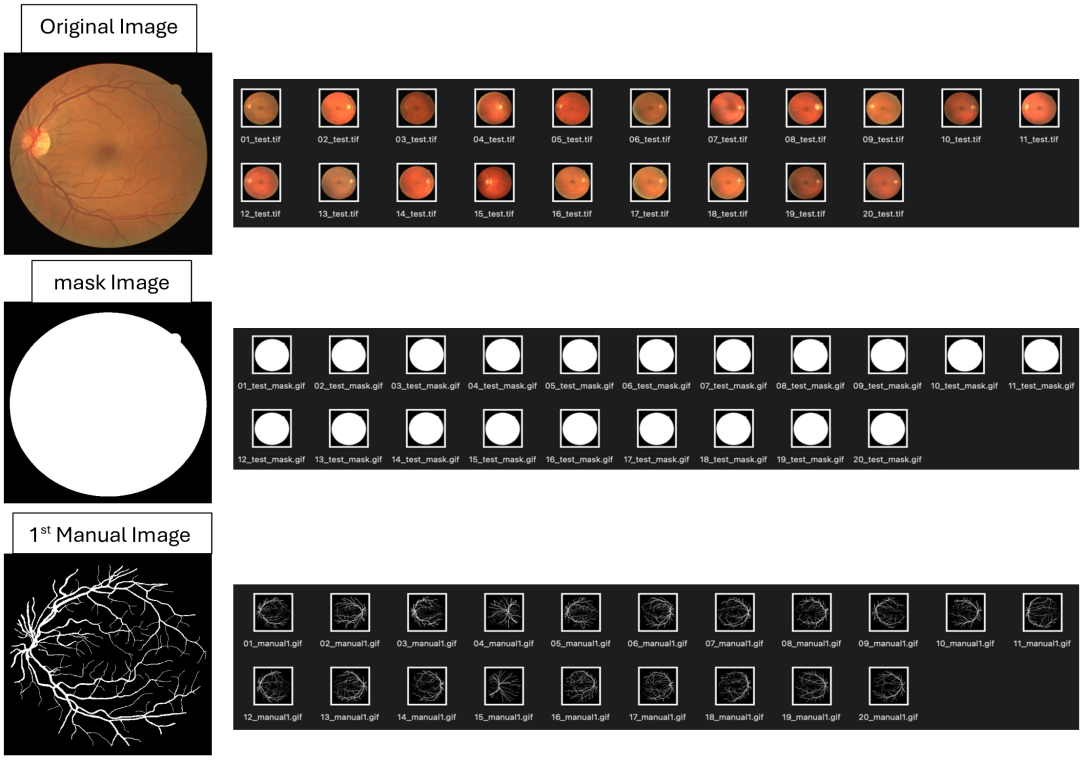



Figure 3: Training and Test image samples

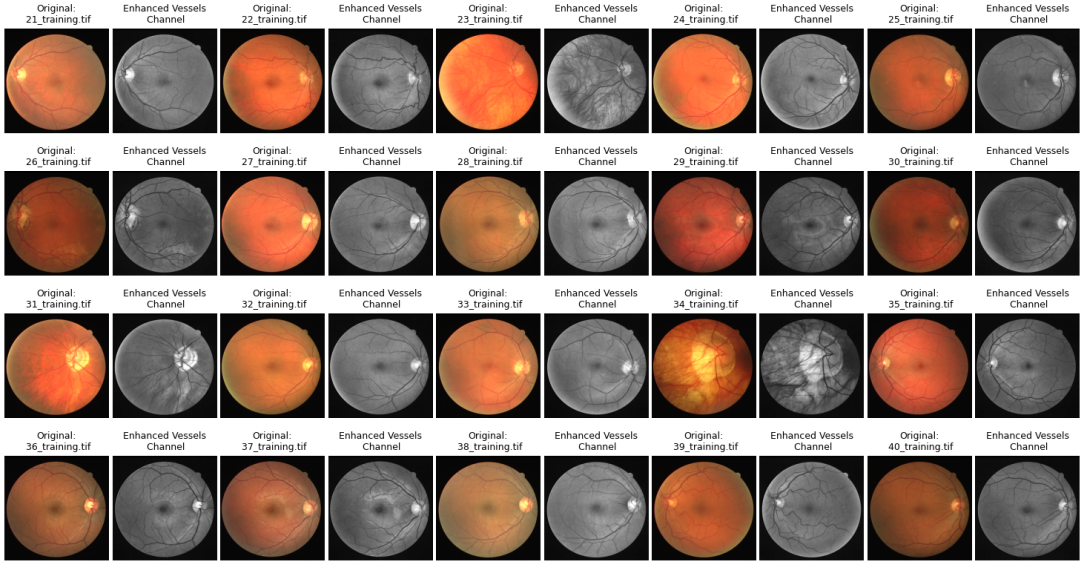



Figure 4: Source Images and Enhanced Version of Images For Comparative Analysis

**3. HSV Color Space Conversion:** This technique separates chromatic content from intensity by converting an RGB image to HSV using rgb2hsv.

**4. Local Binary Pattern (LBP):** This method extracts local texture patterns by converting the image to grayscale and computing LBP.

**5. Contrast Enhancement :** To improve contrast, the image is converted to grayscale and its intensity is rescaled for further use.

**6. Grayscale Conversion:** Using rgb2gray, converts the RGB image to grayscale.

The extracted feature is the output, which has been altered in accordance with the designated feature type. This technique in line with the procedures reported by Zhu et al. (2017), who improved the accuracy of retinal vascular segmentation by employing various feature extraction methods, including LBP, HSV conversion, and contrast enhancement. Neural network training is stabilized and enhanced when extracted features are normalized to a constant range of [0, 1]. Normalized features ensure consistent input scales, thereby preventing large gradient updates and promoting a more stable training process. By applying min-max scaling, the minimum and maximum values of a feature are determined, and the feature is then normalized by subtracting the minimum value and dividing by the range (max - min). This approach is consistent with the normalization strategies discussed in the context of Batch Normalization by Ioffe and Szegedy (2015), who highlight the importance of normalizing inputs to accelerate and stabilize deep neural network training.

### 3.1.5 Source Images With The Extracted Feature Image For Comparative Analysis

To confirm the effectiveness of feature extraction methodologies, a visualization method is employed to compare Source images with their extracted features. This process involves utilizing a feature extraction methodology to derive features from each image, normalizing these features

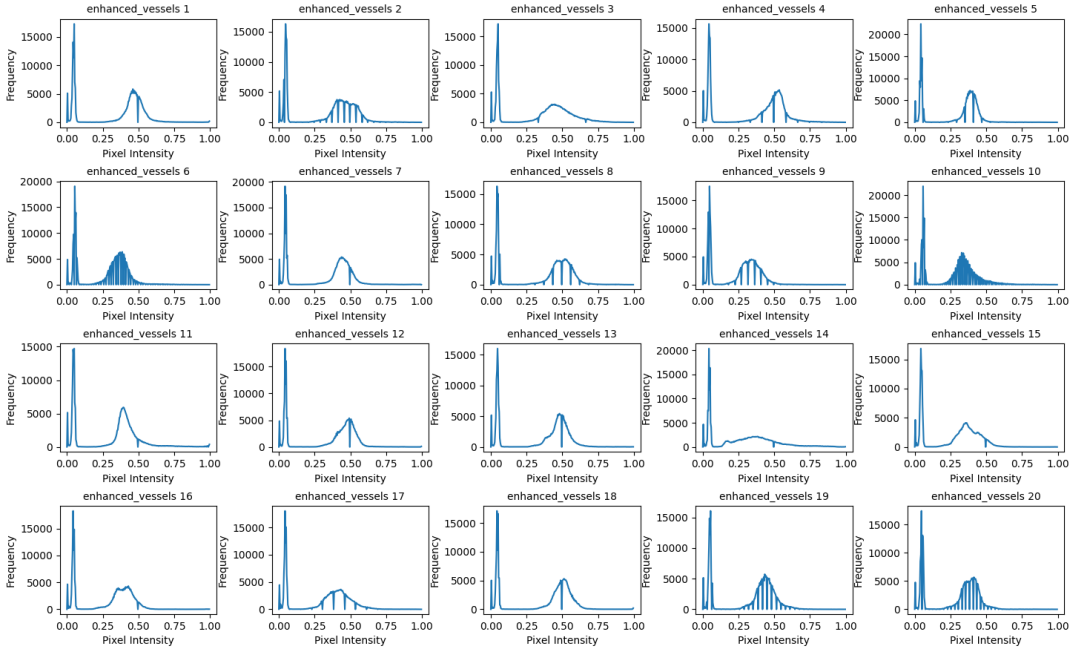



Figure 5: Histogram of Extracted feature Images

if necessary, and then using plotting tool to display the source image alongside its extracted features. This visual comparison, as shown in Figure 4(p.17), helps ensure that the important features in the images are appropriately highlighted by the feature extraction methodologies. This approach is similar to that used by Zhu et al. (2017), who emphasized the importance of visually confirming the effectiveness of feature extraction techniques in improving the accuracy of retinal vascular segmentation. By displaying both the source images and their extracted features, researchers can verify that the feature extraction process is functioning as intended Zhu et al. (2017).

### 3.1.6 Pixel Intensity Distribution

A methodology is applied to plot histograms of pixel intensity values for particular features, which helps with feature distribution analysis. This method provides a clear perspective of the distribution of pixel intensities by calculating and visualizing the histograms. Important information about the features and qualities of each feature is provided by this illustration. As seen in Figure 5, the generated histograms indicate the distribution of pixel values across various features, supporting the reliability of the feature extraction techniques. This strategy is in line with the techniques van der Walt et al. (2014) outline in their `scikitimage` work, in which plotting histograms is utilized to assess image feature distributions and improve comprehension of image attributes. Visualizing pixel intensity distributions allows researchers to confirm that the feature extraction technique effectively highlights relevant features in the images. From the Figure 5,The pixel intensity distributions for 20 improved retinal vascular images are represented by this set of histograms. The frequency of pixel intensities, which range from 0 to 1, is plotted in each histogram and is calculated with 256 bins. The majority of histograms exhibit a general trend with two unique peaks: a secondary peak around 0.25 to 0.50, which corresponds to the enlarged vascular structures, and a dominant peak near 0, which indicates a high frequency of low-intensity (background) pixels. These histograms show the normal patterns of intensity distribution within the improved vessel images, demonstrating how well the segmentation and enhancement procedure separated the vessels from the surrounding area.

### 3.1.7 Patch Extraction

This method involves extracting multiple patches of a specified size from randomly selected areas within an image to enhance data diversity and reduce overfitting. To help train a strong model, this strategy provides many views of the same image and grows the amount of the dataset. The technique ensures consistency by using a random seed and random sampling to set top-left coordinates for patches within image bounds. A list of patches and their coordinates is produced when patches are retrieved using array slicing based on these coordinates. These coordinates can then be aligned with matching manual annotations.This method is consistent with the strategies covered by Shorten and Khoshgoftaar (2019), who emphasize the value of data augmentation in deep learning tasks to enhance model generalization and avoid overfitting. Examples of such augmentations include random cropping and patch extraction. These techniques greatly increase the deep learning models' accuracy and robustness by adding variability to the training set.
By extracting image patches from designated places, the method ensures alignment between image patches and matching ground truth patches. When the model is learning from input-label pairs in supervised learning tasks, this alignment is essential. This technique generates a list of extracted patches corresponding to each specified location using array slicing to extract patches depending on the given patch size from the given locations. This method is similar to that shown by Tajbakhsh et al. (2016), who underlined the significance of effective segmentation jobs requiring accurate patch extraction from input images and associated annotations. In supervised learning contexts, ensuring alignment between input patches and ground truth labels greatly improves the model's performance.

### 3.1.8 Data Augmentation

The augmentation methodology applies particular, randomized, and uniformly distributed augmentation methods to images, therefore increasing the quantity and variety of the training dataset. This technique increases the reliability and generalization of the model by artificially expanding the dataset through the use of transformed images. Using a rotation matrix, the images are then rotated, flipped vertically or horizontally depending on random conditions, and then combined through more complicated transformations like "complex flip rotate," "horizontal flip rotate," and "vertical flip rotate."
The following uniformly distributed and randomized augmentation parameters are used in this methodology:

- **Rotation:** Images are rotated within a specified range of angles, [-45, 45] degrees, with a uniformly and randomly chosen rotation angle.
- **Flip:** An equally produced random value is used to execute either horizontal or vertical flipping. Applying the corresponding flip (horizontal or vertical) occurs if this random value is greater than 0.5.
- **Complex Flip Rotate:** To perform more complicated transformations, images are first randomly flipped (horizontally or vertically, depending on whether the uniformly generated random value is greater than 0.5). After that, the rotation angle is uniformly selected from the range of [-45, 45] degrees to rotate the images within that range. This verifies that when the randomized requirements are satisfied, flipping and rotation are executed in that order.

This method adds controlled variability to the training dataset by the use of random and uniform distribution during the augmentation phase, resulting in the model's improved generalization to new data. The approaches presented by Perez and Wang (2017), who highlight the significance of data augmentation techniques like rotation, flipping, and mixed transformations, are in accordance with the methodology. It is important that the methodology contains these uniformly

distributed and randomized conditions in order to enhance model robustness and efficiently expand the number of training datasets.

#### 3.1.9 Masking Methodology For Enhanced Segmentation In Deep Learning Models

The masking methodology isolates the region of interest and concentrates the model's attention on significant areas by applying a binary mask to a patch. By ensuring that the model learns to focus on particular regions of interest while ignoring irrelevant areas, this process improves segmentation accuracy. By multiplying the patch element-wise by the mask using array operations, the approach produces a masked patch that keeps just the regions indicated by the mask. This method is comparable to the Fully Convolutional Networks (FCNs) approach described by Long et al. (2015), in which regions of interest are isolated using the mask-applied input patches. This technique is essential for enhancing segmentation performance in deep learning models since it provides that only meaningful regions are taken into account during training. Binary masks can be transformed using a particular conversion technique into a floating-point format in order to ensure compatibility with neural network operations. Neural networks normally function with floating-point numbers, therefore this translation is necessary for training and evaluating models. The process ensures that the mask is in a format that can be used for neural network computations by converting mask values to a floating-point data type using array operations. According to Jia et al. (2014), this procedure is consistent with the approach used by the Caffe deep learning framework, which preprocesses input data and converts it to a floating-point format to ensure compatibility and best performance during training and inference.

#### 3.1.10 Visualization Of Feature Extracted Patches, Ground Truth, Augmented Feature Extracted And Augmented Ground Truth

A visualization method is used to compare extracted feature patches and enhanced patches with their corresponding manual annotations, thereby confirming the usefulness and correctness of augmentation methodologies. This method, which involves extracting, augmenting, and normalizing patches before utilizing plotting tools to display them side by side, makes sure that the augmentation approaches are implemented accurately and successfully, as seen in Figure 6(p.21). Plots displaying the initial, enhanced, and matching manual patches comprise the output. This approach is comparable to those used in image processing studies, such those covered by van der Walt et al. (2014), where verifying the correct application and effectiveness of data augmentation techniques depends on the ability to view augmented images.A visual representation of these impacts is critical for confirming the effectiveness of augmentation approaches in enhancing model performance.

#### 3.1.11 Storing Patch Feature

To generate a structured dataset for training and evaluation, a methodology is used that processes image patches, extracts features, and stores them in a CSV file. Structured datasets with extracted features are essential for training and analyzing machine learning models, and storing the data in CSV format allows for easy retrieval and manipulation. The method begins with image extraction, followed by manual annotations and masking. If enabled, particular augmentations are subsequently implemented. Following that, features are retrieved and normalized from the patches, merged, and saved to a CSV file via tabular data operations.This methodology is consistent with previous methodologies used in research, such as those by Simonyan and Zisserman (2015), who extract and analyze image patches to provide structured datasets for deep learning model training. Data augmentation, feature extraction, and normalization are standard processes for preparing datasets for accurate model training. These processes ensure that the dataset is properly organized and ready for effective model training and evaluation.

This illustration shows the locations of extracted patches from the original image, alongside feature-extracted images. The patches are highlighted with colored border boxes for clarity: purple, green, and blue.

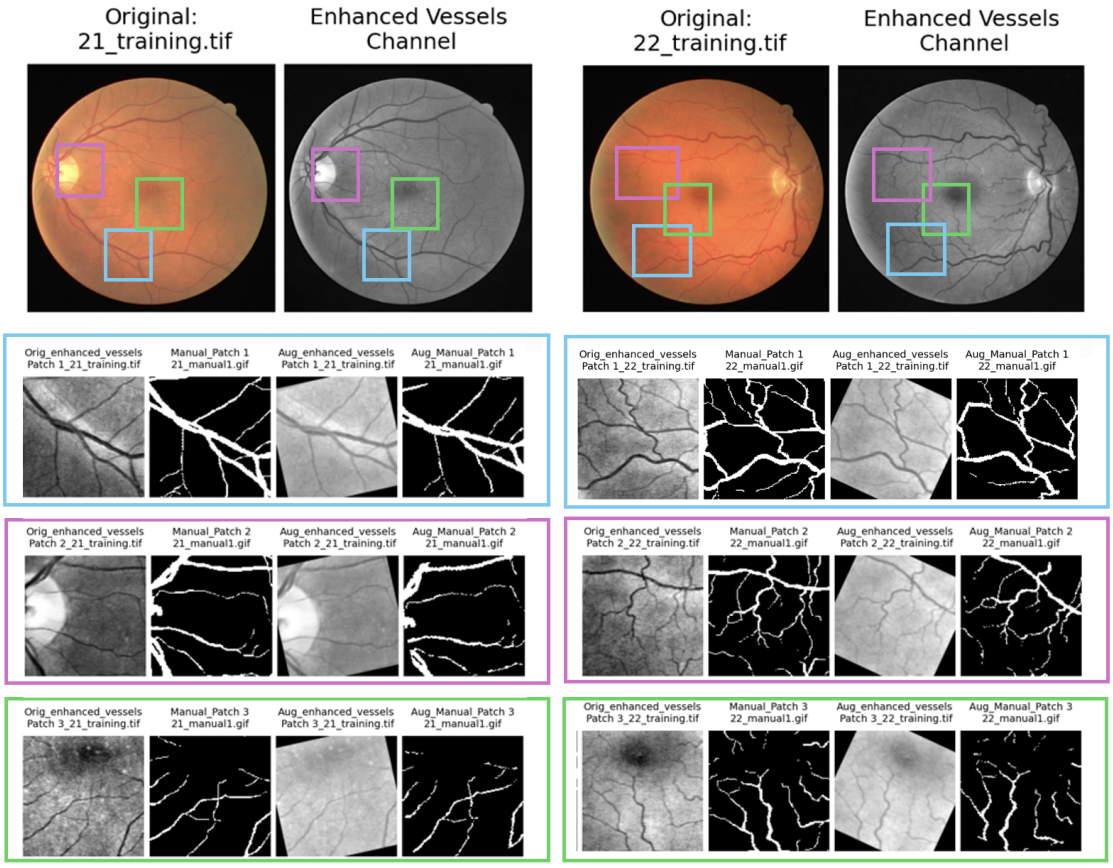



Figure 6: For illustration purpose: Display Feature extracted and Ground truth patches with augmentation and location identification from source and feature extracted images.

### 3.1.12 Dataset Preparation For Deep Learning Model

The process for creating training and validation datasets includes extracting patches and their accompanying labels from images, as well as manual annotations. This step is necessary for training machine learning models, which require paired datasets of inputs and labels. The methodology extracts patches from images and manual annotations, gets features from these patches, and normalizes them as needed. The output consists of datasets in the form of NumPy arrays (X and Y), which are ready to train machine learning models. This method is similar to that utilized by Kather et al. (2016) in their study on the histology of colorectal cancer, in which patches are isolated from entire slide images, features are computed, and datasets are created for model training. This arranged approach ensures that the datasets are properly prepared for machine learning model training and validation.

## 4 Deep Learning Models And Description:

Each architecture in a segmentation framework that uses a variable to switch between the normal U-Net and Dense U-Net models is customised to specific requirements. The regular U-Net was chosen for its efficient encoder-decoder structure, which is capable of collecting both fine features

and global context, making it suited for a wide range of segmentation problems. This model uses simple convolutional layers with batch normalization and incorporates dropout largely in the encoder to facilitate effective feature reduction while simplifying feature expansion in the decoder.
When the Dense U-Net option is selected, dense blocks are used in both the encoder and decoder stages, but batch normalization is not used. These dense blocks are typical of the DenseNet design and feature recurrent convolutions with continuous dropout all over, which strengthens the connections between each layer inside a block. This architecture improves feature transmission and reuse, potentially increasing the network's ability to deal with complicated patterns. The Dense U-Net works to preserve strong feature handling and generalisation by implementing dropout across both the encoder and decoder without batch normalization, making it especially effective for complex segmentation tasks that need comprehensive feature extraction and advanced pattern recognition. This arrangement produces a model that is strong to overfitting and computationally demanding, but capable of improved performance in challenging segmentation scenarios.

## 4.1 U-Net Model Architecture

This method involves constructing a U-Net architecture specifically for image segmentation, capable of capturing both fine details and global context. This architecture is especially useful for medical image segmentation applications such as retinal vascular segmentation. Figure 7(p.23) illustrates the U-Net model's architecture.
The design conforms to the framework suggested by Ronneberger et al. (2015) in their groundbreaking study on biological image segmentation. Because it can capture fine features while retaining the general context of the image, the U-Net design has shown usefulness in a variety of segmentation applications, especially in medical imaging.

### 4.1.1 Input Layer

The model begins with an input layer that takes images with a size of (128, 128,1).

### 4.1.2 Encoder Path

Four convolutional blocks make up the encoder. Two convolutional layers with batch normalization and ReLU activation are included in each block, and a max-pooling layer reduces spatial dimensions after that. The use of dropout layers prevents overfitting. The number of filters rises from 64 to 128 to 256 to 512 as the data moves through these blocks, enabling the model to capture more complex features at deeper levels.

### 4.1.3 Decoder Path

Using concatenation layers to combine feature maps from the connected encoder blocks (skip connections) and upsampling layers to increase spatial dimensions, the decoder reassembles the image. The small features from the previous layers are preserved thanks to this structure. Two convolutional layers with batch normalization and ReLU activation come after each upsampling stage. As the image is reassembled, the number of filters drops from 512 back to 256, 128 and 64.

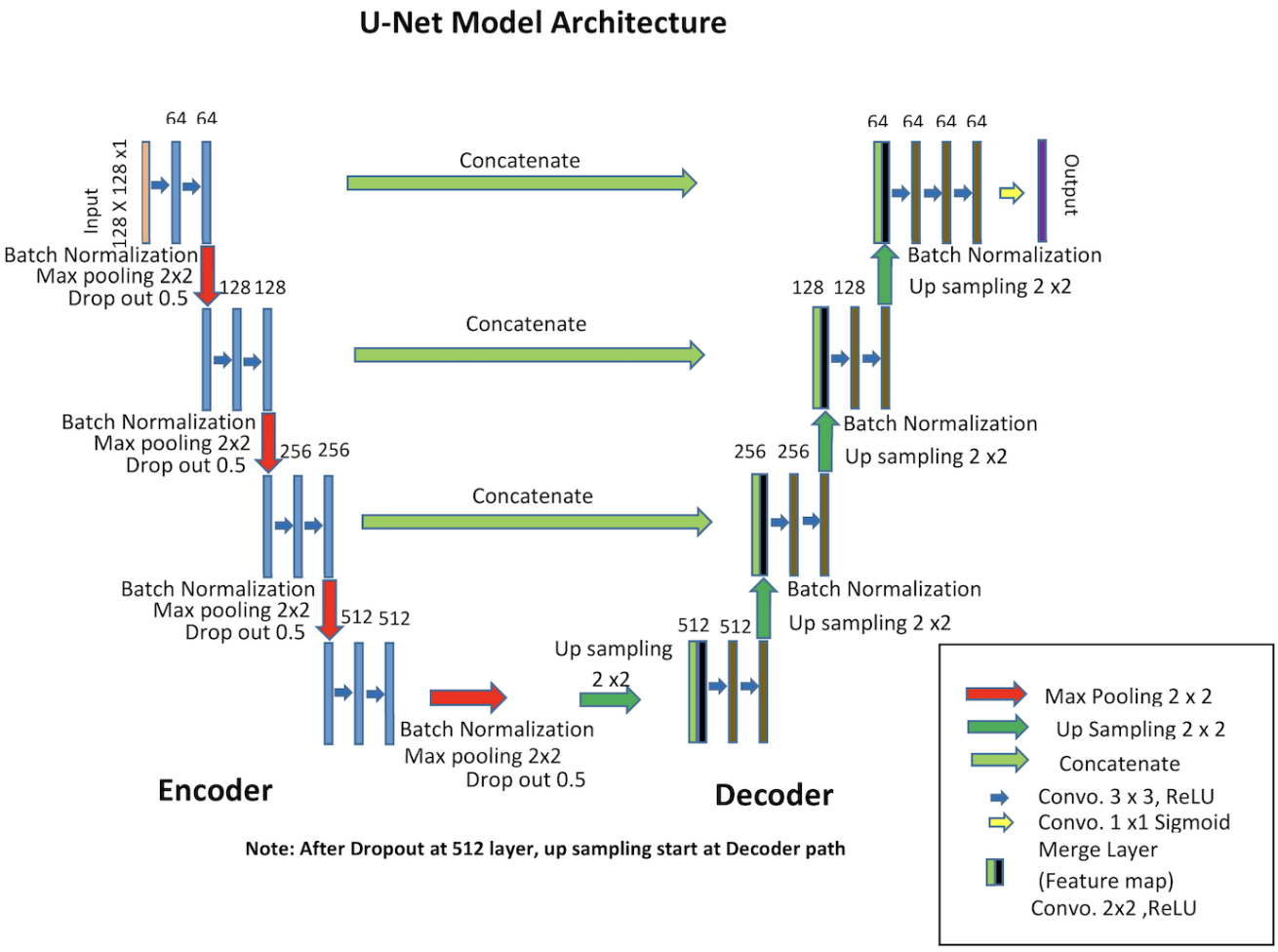



Figure 7: Unet Model architecture for illustration purpose Ronneberger et al. (2015)

#### 4.1.4 Output Layer

A single-channel output image with the same spatial dimensions as the input is produced by the convolutional layer, which is the last layer, using a sigmoid activation function. The probability map for every pixel in this output is shown, together with the probability that it belongs to the specified class.

#### 4.1.5 Model Compilation

To solve class imbalance and confirm consistent segmentation, the model is composed with the Adam optimizer and a combination loss function (binary cross-entropy and dice coefficient). The evaluation measures are Dice coefficient, Mean Intersection over Union (Mean IoU), and accuracy.

### 4.2 Dense U-Net Model Architecture

For image segmentation, The method entails building a Dense U-Net architecture that improves feature propagation and learning efficiency. By utilizing extensive connectivity within blocks, this architecture enhances feature reuse and gradient flow, improving learning performance and efficiency.Figure 8(p.24) illustrates the Dense U-Net model's architecture.
**For the Dense U-Net model**, dense blocks are defined and these blocks improve gradient

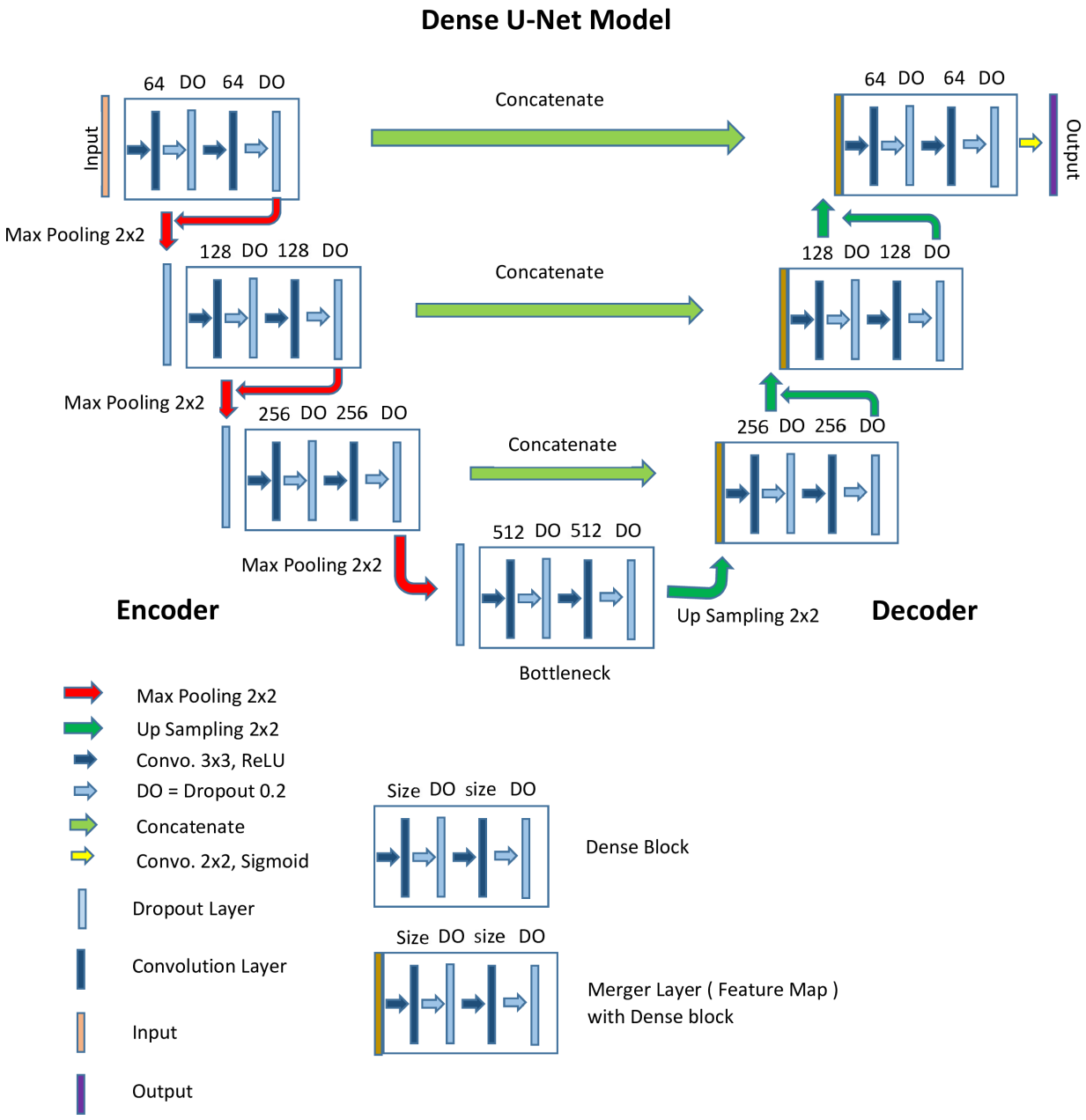



Figure 8: Dense Unet model Architecture for illustration purpose Huang et al. (2016)

flow and feature reuse by feed-forwardly connecting each layer to all other layers. Convolution, dropout, and activation layers are applied successively in each dense block, with input from all earlier levels in the block going toward each layer. The model's performance and learning efficiency are improved by this architecture, which encourages effective gradient flow and greater feature use.
The Dense U-Net model design used in this work which incorporates the DenseNet concepts as outlined by Huang et al. (2016) into the U-Net framework. This improves the model's overall performance as well as the transmission of features and reuse. Improved segmentation outcomes are the consequence of dense connection, which ensures effective gradient flow and higher feature use. Dense connection in combination with the U-Net framework allows it to capture tiny details as well as global context, which makes it especially useful for biomedical image segmentation tasks.

#### 4.2.1 Dense Blocks

Convolutional, dropout, and activation layers combine in a feed-forward method to form dense blocks. Because every layer in the block receives information from every other layer, feature reuse and propagation are improved. The model's capacity to learn complex properties is enhanced by this connection structure, which provides optimal information flow across layers and helps in maintaining a strong gradient.

#### 4.2.2 Encoder Path

The encoder path records information at different sizes while progressively decreasing the input image's spatial dimensions. Dense blocks precede max-pooling layers, which reduce the feature map sample size. Convolutions, dropout, and ReLU activation functions are applied to each dense block. By succeeding blocks, the number of filters rises from 64 to 512, enabling the model to capture more complex information at deeper levels.

#### 4.2.3 Decoder Path

By upsampling the feature maps to the original image size, the decoder path reconstructs the image. To expand the spatial dimensions, it employs upsampling layers. Then, using skip connections, it combines these upsampled features with equivalent feature maps from the encoder path. This method enables the retention of minute features and the spatial context of previous layers. As the feature maps are upsampled, the decoder uses dense blocks to refine the feature, gradually bringing the number of filters down from 512 to 64.

#### 4.2.4 Output Layer

Using a sigmoid activation function, a convolutional layer is the last layer that creates a single-channel output image with the same spatial dimensions as the input. The probability map for every pixel in this output is shown, together with the probability that it belongs to the targeted class.

#### 4.2.5 Model Compilation

The The model is built using the Adam optimizer and a learning rate of 0.001. It uses a binary cross-entropy and dice coefficient loss function coupled, which works especially well for managing class imbalance in segmentation tasks. Accuracy, Mean Intersection over Union (Mean IoU), and dice coefficient are among the evaluation measures that are essential for evaluating segmentation performance.

### 4.3 Loss And Metrics

In image segmentation, the Dice coefficient and Dice Loss techniques quantifies the overlap between predicted and actual segments, making it important for tasks that require accurate region matching. The Combined Loss approach combines Dice Loss and Binary Cross-Entropy to balance overlap accuracy with pixel-wise classification, improving the model's overall performance.

#### 4.3.1 Dice Coefficient And Dice Loss

Dice Loss method computes the overlap between the expected and 1st manual (ground truth) segments is the focus of the metric known as dice loss. With dice loss, segmentation quality may be robustly measured, especially in tasks where maximizing the overlap between the actual and projected regions is important.

**The Dice Loss can be calculated as follows**
**1. Flatten Labels:** The true and predicted labels are flattened to compute their intersection and union.
**2. Calculate Dice Coefficient:** The Dice coefficient is calculated using the equation Parsad (2018):

$$\textbf{Dice Coefficient} = \frac{\textbf{2} \times \textbf{True Positive}}{\textbf{2} \times \textbf{True Positive} + \textbf{False Positive} + \textbf{False Negative}}$$

This formula highlights the balance between the correctly identified positive samples and the errors in both the predicted positive and missed positive samples.
**3. Dice Loss:** The Dice loss is calculated using the equation Yeung et al. (2023):

$$\textbf{Dice Loss} = \textbf{1} - \textbf{Dice Coefficient}$$

This method is consistent with the methodology employed by Milletari et al. (2016) in their work on V-Net, which showed how Dice coefficient and Dice loss performs well in volumetric image segmentation and highlights its relevance to work requiring precise overlap measurement.

#### 4.3.2 Combined Loss

In image segmentation tasks, the combined loss approach balances overlap and pixel-wise accuracy by integrating binary cross-entropy Goodfellow et al. (2016a) and dice loss. Whereas binary cross-entropy evaluates the accuracy at the pixel level, dice loss highlights maximization of the overlap between segments that are predicted and those that are actual. The combined loss, which is obtained by adding these two losses, takes advantage of their respective strengths and offers a reliable metric for segmentation model training. As proven by Sudre et al. (2017), combining binary cross-entropy with dice loss has been shown to increase segmentation adaptability and accuracy. By balancing the advantages of both loss techniques, this method improves the model's ability to handle tasks involving overlap and pixel-by-pixel classification.

Combined Loss equation:

$$\textbf{Combined Loss} = \textbf{Binary Cross-Entropy} + \textbf{Dice Loss}$$

Binary Cross-Entropy equation Goodfellow et al. (2016a):

$$\textbf{Binary Cross-Entropy} = -\frac{\textbf{1}}{\textbf{N}} \sum_{i=1}^{\textbf{N}} [\mathbf{y_i} \log(\mathbf{p_i}) + (\mathbf{1} - \mathbf{y_i}) \log(\mathbf{1} - \mathbf{p_i})]$$

**where:**
$y_i$ is the true label.
$p_i$ is the predicted probability.
$N$ is the number of pixels.

Dice Loss equation Yeung et al. (2023):

$$\textbf{Dice Loss} = 1 - \frac{2 \times \textbf{True Positive}}{2 \times \textbf{True Positive} + \textbf{False Positive} + \textbf{False Negative}}$$

## 4.4 Training And Evaluation

The dataset is divided into training and validation sets for image segmentation tasks in order to prevent overfitting and evaluate model performance. Afterwards, the model is evaluated on unobserved data to confirm its generalizability after being trained and optimized using the training set. At last, the model that has been trained is stored for further use and examination.

### 4.4.1 Dataset Split Into Training And Validation

To separate the dataset into training and validation sets, which is necessary to avoid overfitting and evaluate model performance, use the `train_test_split` method from `sklearn.model_selection`. By dividing the dataset at random, it retains the distribution and consistency of the data by ensuring that both subsets are representative samples. By splitting the dataset, the model's generalization ability can be validated on data that hasn't been seen before. This strategy is consistent with methods outlined by Kohavi (1995), who highlighted the significance of appropriately assessing model performance and generalization using unseen data.

### 4.4.2 Fitting Models

The model is trained using the model.fit technique, which minimizes the loss function by optimizing the model's parameters through gradient descent and backpropagation utilizing the training dataset. This method iteratively updates the weights based on the training data, improving the model's accuracy in image segmentation. During training, the model receives data in batches so that it can modify the weights to minimize loss. Overfitting is avoided and progress is tracked by keeping an eye on performance on both training and validation sets. The training history, which contains measures like accuracy and loss over the training epochs, is included in the output.This method is consistent with deep learning best practices, as explained by Goodfellow et al. (2016b), who emphasize the value of performance monitoring and iterative optimization in neural network training.

### 4.4.3 Evaluation

The model's adaptability and capacity for generalization are evaluated by the model.evaluate technique, which also evaluates the model's performance on test and validation datasets. This procedure is crucial for evaluating the model's performance on hypothetical data and confirming its suitability for practical applications. The validation or test data is fed into the model by the function, which then computes important metrics like loss, accuracy, Mean IoU, and dice coefficient. With the help of these indicators, the model's performance can be thoroughly assessed, identifying both its advantages and disadvantages. In machine learning, evaluating on unseen data is common procedure and essential to verify the model's performance outside of training data.Studies like the one conducted by He et al. (2015), which highlight the significance

of evaluating model performance on separate validation and test datasets to ensure generality and robustness, lend support to this strategy.

#### 4.4.4 Saving Model

The model.save method saves the trained model, including architecture, weights, and training information, to disk. This enables future use without retraining, simplifying deployment and further evaluation. Saving models is a common method in machine learning to ensure consistency and efficiency. It enables the deployment and usage of trained models in a variety of applications without the requirement for retraining, as detailed by Goodfellow et al. (2016b).

## 5 Post-processing:

By using morphological methods, it improves the quality of expected segmentation masks. Using a threshold, this method turns predictions into binary masks, which are subsequently subjected to binary opening and closing processes. Binary closing fills in tiny gaps while binary opening eliminates small items from the foreground, producing segmentation masks that are smoother and more precise. This approach is consistent with image segmentation best practices, like those that Ronneberger et al. (2015) emphasized in their U-Net architecture. Refinement of segmentation masks and correct performance evaluation depend on these post-processing techniques.

## 6 Performance Metrics:

When measuring model performance, accuracy is measured as the percentage of properly detected pixels, offering a straightforward assessment. Precision, recall, and F1 score provide a detailed analysis of true positive and false positive predictions, which is essential when evaluating model performance. Specificity, which measures the ability to correctly identify background pixels, is critical for ensuring that non-target areas are not incorrectly classified as vessels. Furthermore, metrics such as the Jaccard Index (IoU) and AUC-ROC score indicate overlap and classification performance, respectively, while the confusion matrix gives a detailed breakdown of prediction errors, aiding in model optimization.

### 6.1 Accuracy

In a segmentation task,calculates the percentage of correctly identified pixels, offering a simple way to evaluate model performance. To find the total accuracy number, it compares the true and predicted labels and calculates the mean of the correctly categorized pixels.The accuracy is computed using the equation Buhl (2022):

$$\textbf{Accuracy} = \frac{\textbf{Number of Correctly Predicted Pixels}}{\textbf{Total Number of Pixels}}$$

When evaluating segmentation models, accuracy is frequently used together with metrics such as the Dice coefficient and Intersection over Union (IoU). As noted by Ronneberger et al. (2015) in their U-Net study, these metrics provide a thorough assessment of model quality, highlighting their significance in segmentation tasks.

### 6.2 Precision, Recall And F1 Score

A comprehensive performance breakdown for machine learning models is provided by the technique, which computes precision, recall, and F1 score. These measures are essential for comprehending how true positive and false positive predictions are balanced in binary classification

tasks like as image segmentation.
Precision is defined as the ratio of true positives to the total of true and false positives Gupta (2021):

$$\textbf{Precision} = \frac{\textbf{True Positives}}{\textbf{True Positives} + \textbf{False Positives}}$$

Recall (Sensitivity) is defined as the ratio of true positives to the total of true positives and false negatives Gupta (2021):

$$\textbf{Recall ( Sensitivity)} = \frac{\textbf{True Positives}}{\textbf{True Positives} + \textbf{False Negatives}}$$

The F1 score is the harmonic mean of precision and recall Gupta (2021):

$$\textbf{F1 Score} = \frac{2 \times (\textbf{Precision} \times \textbf{Recall})}{\textbf{Precision} + \textbf{Recall}}$$

Powers (2020) pointed out that these indicators provide a thorough evaluation of model performance.

### 6.3 Specificity:

In image segmentation tasks that differentiate background from vessels, specificity assesses the model's accuracy in identifying background pixels Taha and Hanbury (2015). This metric is crucial for ensuring that background areas are not mistakenly labeled as vessels. The formula for calculating specificity in this context is:

$$\textbf{Specificity} = \frac{\textbf{True Negatives}}{\textbf{True Negatives} + \textbf{False Positives}}$$

where true negatives are the background pixels correctly identified, and false positives are those incorrectly labeled as vessels .

### 6.4 Jaccard Index Score

The Jaccard index, which quantifies how similar segments are to predictions. This index, often called Intersection over Union, or IoU, compares the overlap between the ground truth and anticipated masks to provide a reliable indicator of segmentation performance. The Jaccard index (IoU) calculated by following equation Sanjaya et al. (2020):

$$\textbf{Jaccard Index (IoU)} = \frac{\textbf{True Positives}}{\textbf{True Positives} + \textbf{False Positives} + \textbf{False Negatives}}$$

In image segmentation problems, the Jaccard index is commonly utilized, as the PASCAL VOC challenge shows. By analyzing the intersection over the union of the predicted and ground truth segments, it provides a precise indicator of the model's accuracy Everingham et al. (2010).

### 6.5 AUC-ROC Score

The AUC-ROC score, which evaluates the model's capacity for categorization. This score is a useful metric for assessing classification tasks since it offers a unified assessment of the model's performance across all categorization criteria. Since the AUC-ROC score measures the model's performance in a comprehensive manner, it is frequently used in classification tasks, such as

image segmentation. It computes the area under the curve (AUC) after computing the true positive rate (TPR) and false positive rate (FPR) and adjusting the threshold to create the ROC curve.
The true positive rate (TPR) and false positive rate (FPR) are mentioned as below Boesch (2024):

$$\textbf{True Positive Rate (TPR)} = \frac{\textbf{True Positives}}{\textbf{True Positives} + \textbf{False Negatives}}$$

$$\textbf{False Positive Rate (FPR)} = \frac{\textbf{False Positives}}{\textbf{False Positives} + \textbf{True Negatives}}$$

According to Fawcett (2006), this approach is very good at producing a reliable assessment of model performance.

### 6.6 Confusion Matrix

With the use of counts of true positives, true negatives, false positives, and false negatives, the method creates a confusion matrix that offers a thorough analysis of the model's performance. This facilitates the understanding of model performance and the diagnosis of categorization problems. A common method to measure model accuracy in classification tasks, such as image segmentation, is the confusion matrix. It helps in understanding the distribution of various error categories, which is essential for optimizing model performance Powers (2020).

## 7 Predicted Results:

The following portion looked deeply into the segmentation model's predicted outcomes.The evaluation includes a histogram of predicted values, comparisons of predicted patches to ground truth, a ROC curve, and a graphic representation of the confusion matrix. These visual and quantitative assessments aid in evaluating the model's performance, recognizing strengths, and outlining potential areas for development. These metrics and visualizations provide insights into the model's accuracy, confidence, and overall performance in segmentation tasks.

### 7.1 Histogram Of Predicted Values

In order to analyze the distribution of predicted probability, a method is utilized to plot prediction value histograms. It helps in the identification of biases or calibration concerns, as well as the evaluation of the model's confidence in its predictions. The method computes a histogram of prediction values and then plots it to provide information about the model's dependability and performance. This method supports in determining how well the model's predictions match predicted outcomes, providing a visual representation of potential areas for improvement. These kinds of visualization methods are frequently applied in machine learning to assess calibration of models and prediction distributions Hunter (2007b). The histogram in Figure 9(p.31) represents the distribution of the model's prediction values, with the x-axis indicating prediction values and the y-axis representing frequency. The majority of predictions cluster around 0, indicating a significant bias towards classifying pixels as background, most likely due to a class imbalance in the dataset. A smaller peak at 1 indicates that the model occasionally predicts the vessel class, but with much lower frequency. The lack of intermediate prediction values shows that the model is extremely confident in its classifications, raising the possibility that it will underperform in detecting vessels due to the imbalance.

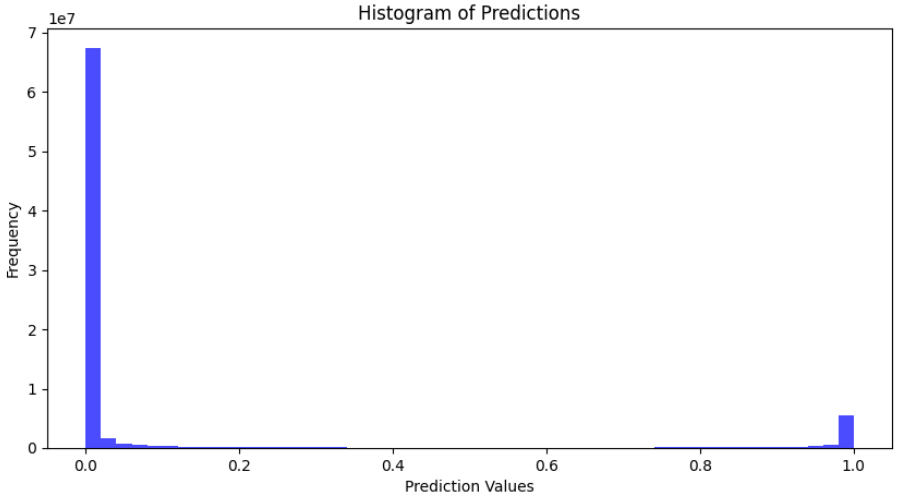



Figure 9: Histogram of Predicted Values

## 7.2 Comparison Of Predicted Patches With Ground Truth And Feature Type Patches

Figure 10 shows a grid of up to 20 test images that compare the source feature type patches, ground truth annotations, and the model's predicted segments. The visualization shows the source image, ground truth, and prediction side by side, allowing for a direct comparison of the model's effectiveness in segmenting retinal vessels. The model performs well in capturing larger vessels, but has limitations in identifying finer, more complicated vessel structures. This side-by-side comparison is useful for identifying places where the model's predictions differ from the ground truth, providing critical insights for further modification to improve segmentation accuracy, especially for smaller vessels.

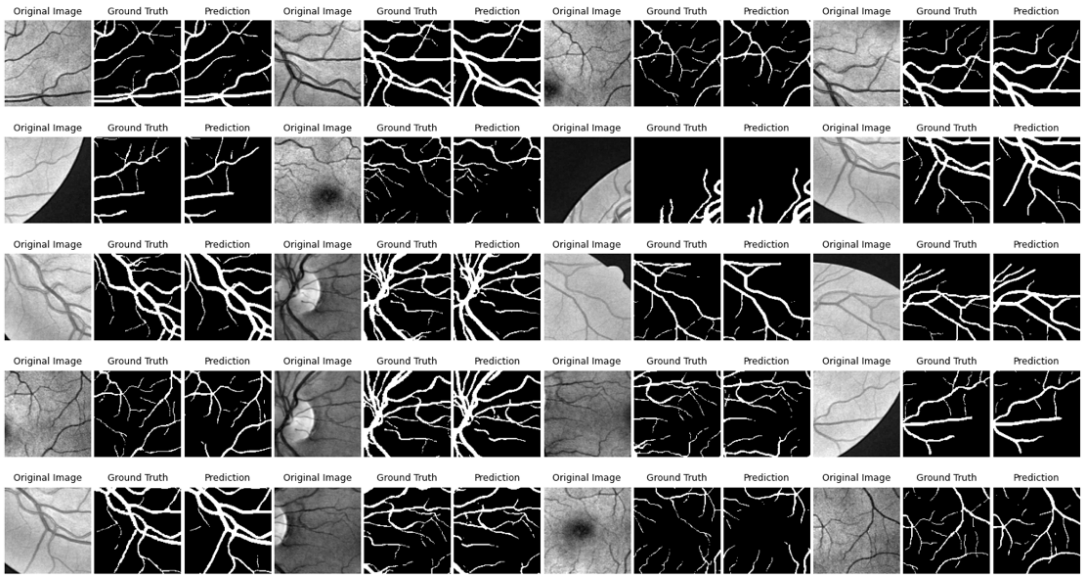
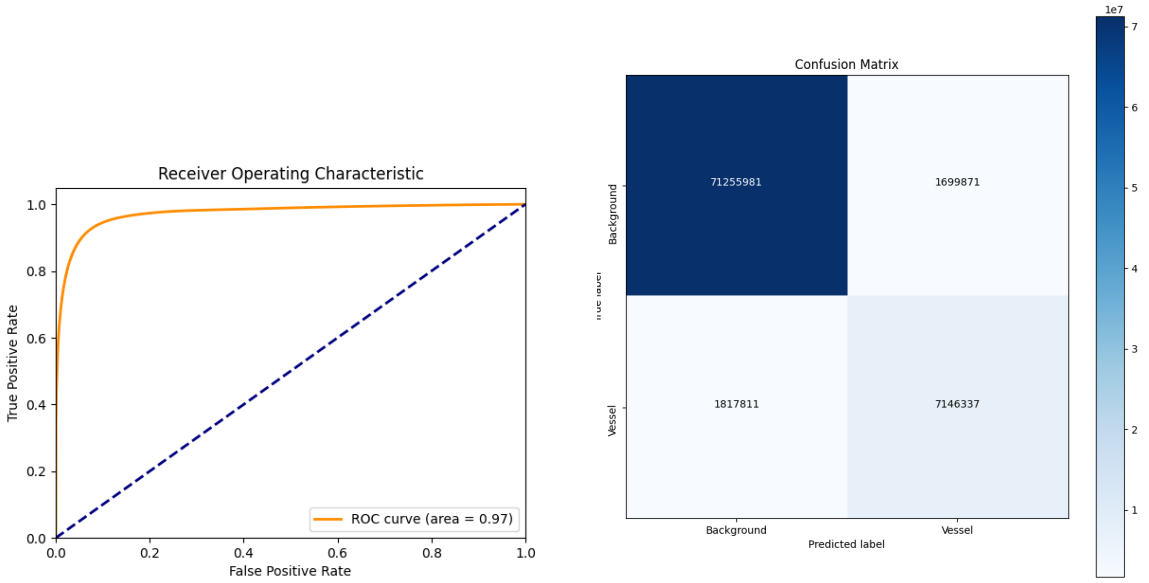



Figure 10: Comparison of Predicted Patches with Ground Truth and Feature Type Patches

## 7.3 ROC Curve

The ROC curve helps determine the optimal threshold for classification tasks by comparing the true positive rate (TPR) and false positive rate (FPR) for different thresholds. This method provides an intuitive graphic representation of the model's performance by computing TPR and FPR at various thresholds and plotting the results. As Fawcett (2006) points out, ROC curves are important evaluation tools for binary classifiers. Figure 11a presents the ROC curve, which measures the performance of a binary classification model. The y-axis represents the True Positive Rate (TPR), while the x-axis represents the False Positive Rate. The plot compares the TPR to the FPR at different threshold values. The orange line demonstrates that the TPR increases as the FPR increases. A perfect model has an area under the curve (AUC) of one, but an AUC of 0.5 shows that the model lacks discriminatory power. In this instance, the AUC is 0.97, suggesting strong model performance.




(a) ROC curve

(b) Confusion Matrix

Figure 11: Comparison of ROC curve and Confusion Matrix

## 7.4 Graphical Representation Of Confusion Matrix

Using a color gradient ranging from blue to white, the method uses Matplotlib to visualize a confusion matrix and display categorization results. 'Background' and 'Vessel' labels are rotated and placed on the y-axis (true) and x-axis (predicted), respectively. To ensure readability, cell values are annotated with black text otherwise and white text for values greater than half of the matrix's maximum. This image, which displays the proportion of true positive, true negative, false positive, and false negative predictions, effectively highlights the model's performance.
Figure 11b highlights how a model for predicting background and vascular pixels in retinal images is assessed using a confusion matrix. True negatives (background pixels correctly recognized) are shown in the matrix as 71,255,981, and true positives (vessel pixels correctly detected) as 7,146,337. There are 1,699,871 false positives (background pixels mistakenly recognized as vessels) and 1,817,811 false negatives (vessel pixels mistakenly classified as background). Although there are still a sizable number of misclassifications, the model is effective in correctly classifying both classes, as indicated by the high values in the true negative and true positive cells.

# 8 Image Prediction And Visualization:

The method for loading a stored model receives its architecture, weights, and training data from disk. This enables the reuse of previously trained models without the requirement for retraining, resulting in more efficient deployment and evaluation. Loading saved models is a standard method in machine learning to ensure that models may be used and evaluated reliably across multiple tasks or environments.This technique is consistent with best practices for model management and deployment Goodfellow et al. (2016b).The model compilation method prepares the loaded model for training or evaluation by specifying the optimizer, loss function, and evaluation metrics. This step is critical to ensuring that the model is configured correctly for its intended application. Compiling the model with the same parameters as those used during training ensures consistency in both performance and evaluation. Chollet (2017) highlights how this strategy is commonly employed in deep learning processes to ensure model integrity.

## 8.1 Image Segmentation

For predicting full images involves resizing if needed, dividing significant images into smaller patches, making predictions for each patch, and then integrating the findings to generate predictions for the entire image. This technique ensures that large images are processed efficiently while keeping constant input dimensions for the model. This method is widely used in image processing, particularly for high-resolution imaging applications like medical imaging. Various image segmentation studies have showed that this strategy efficiently minimizes memory use and improves model performance by breaking down images, extracting patches, making predictions, and then recombining them, as reported by Ronneberger et al. (2015).

## 8.2 Full Image Comparison With Highlights

To identifying areas for improvement is visually comparing the source images, ground truth annotations, and model predictions, with a focus on highlighting differences. This visualization, illustrated in Figure 12(p.34), helps assess the accuracy of the model's predictions by displaying the images side by side using Matplotlib. The red highlights in the "Prd vs GT" (Prd = Predicted, GT = Ground Truth)images indicate where the model's predictions separate from the ground truth, such as false positives or missing vessel segments. In image segmentation tasks, it is common practice to view whole image comparisons with highlighted differences in order to evaluate the model's performance and indicate areas that need to be refined. This approach, used by Ronneberger et al. (2015) in their U-Net study, is essential for visual assessment and interpretation of segmentation results.

# 9 Experimentation:

The U-Net model experiment began with the parameters listed in Table 1(p.35). This initial setup provided a structured method for evaluating model performance and making adjustments.

- **Target Image Size (512 x 512):** Specifies the resolution to which all input images are resized before processing. This standardization is critical for maintaining consistency in input data and ensuring that the neural network receives the same data format across different images.
- **Feature Type (Enhanced vessel):** Identifies the specific feature type the model is trained to enhance or detect, focusing on vascular structures. This specificity is crucial for applications like medical imaging, where precise tissue differentiation is necessary.

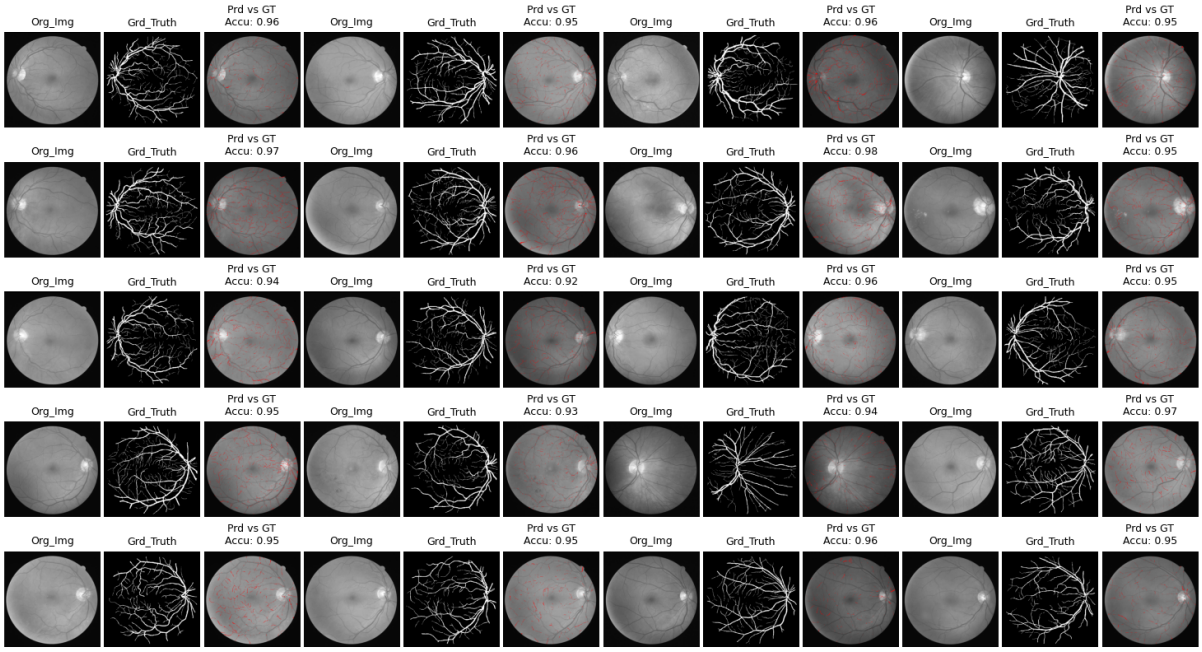



Figure 12: Full Image Compare with Highlights

- **Data Augmentation Method (Flip and Rotate [-45, 45]):** Augmentation techniques such as flipping images horizontally or vertically and rotating them within a range of -45 to 45 degrees are employed to artificially expand the dataset. This enhances the model's generalization capabilities by simulating different orientations and perspectives of the same objects.
- **Patch Size (128x128):** Defines the size of the image patches (128x128 pixels) extracted for processing, which decreases calculating load and allows the model to concentrate on specific features.
- **Number of Patches (100):** The amount of patches collected from every image for training ensures that the model has enough data points and variety within each image for a strong learning process.
- **Normalization (True):** Indicates that each input image's pixel values are normalized, typically to a range that facilitates more effective neural network training.
- **Learning Rate (0.0001):** The learning rate determines the magnitude of weight adjustments in relation to the loss gradient during training, impacting how quickly the model learns.
- **Batch Size (8):** Specifies how many training samples are handled prior to the model's internal variables are updated, balancing processing speed and learning process stability.
- **Epoch (5):** The number of full loops through all of the training dataset, which influences the training time and the model's capacity to gain knowledge from the dataset comprehensively.
- **Seed (30):** A seed value for random number generation, used to ensure reproducibility of results across different runs by making the random number generation consistent.
- **Threshold Value For Prediction (0.5):** Sets the decision boundary for classifying predictions in binary classification tasks, crucial for determining model sensitivity and specificity.

- **U-Net Model - Input size (128, 128, 1):** Details the required input dimensions and channels for the U-Net model, where '1' denotes a single-channel (grayscale) image.
- **U-Net Model – Layers (64, 128, 256, 512):** Lists the number of filters in each convolutional layer of the U-Net, which helps in extracting progressively complex features at different levels of abstraction.
- **Dropout (0.5):** Specifies the fraction of neurons randomly dropped during training to prevent overfitting, encouraging the network to develop more robust features.
- **U-Net Model Details:** Includes the architectural components like convolution layers, batch normalization for faster convergence, MaxPooling2D for spatial data reduction, and dropout for regularization.
- **Extracted Feature Parameters:** Various settings for image processing, including clip-limit(2) for contrast enhancement, tile grid size(6x6) for localized adjustments, and parameters affecting blur and sharpness.
- **Post-Processing Parameters (Binary Opening (1,1), Binary Closing (3,3)):** Defines morphological operations to refine the model's output, enhancing the clarity and accuracy of binary images.

Table 1: Initial Setup Parameters for U-Net Model Experiment

| **Configuration Description** | **Parameter values** |
|---|---|
| Target image size | 512 x 512 |
| Feature Type | Enhanced_vessel |
| Data Augmentation method | Flip and rotate [-45, 45] |
| Patch size | 128 |
| Number of patches | 100 |
| Normalization | True |
| Learning rate | 0.0001 |
| Batch size | 8 |
| Epoch | 5 |
| Seed | 30 |
| Threshold value for prediction | 0.5 |
| Unet model - Input size | 128, 128, 1 |
| Unet model – layers | 64, 128, 256, 512 |
| Dropout | 0.5 |
| Unet model details | Convolution Layer, Batch Normalization, MaxPooling2D, Dropout |
| Extracted Feature Parameters | Clip-limit: 2, tilegridsize: 6x6, Kernel size: 3x3, sigmaX: 0, alpha: 1.5, blurred: -0.5, gamma: 0 |
| Post Process Parameters | Binary Opening (1,1), Binary Closing (3,3) |

Starting with the parameters listed in Table 1, the number of patches for the U-Net model was adjusted between 100, 150, 200, 250, 300, 350, 400, and 500. The results were recorded in an Excel spreadsheet. Analysis demonstrated that the configurations with 200 and 250 patches had better results in terms of F1 score, Jaccard index, mean Dice coefficient, and mean accuracy, as shown in Table 2(p.36). The 200-patch set up, which had the highest values as seen in Table 2(p.36), was chosen for further experimentations.

Table 2: Unet Model - Comparative Performance Metrics for Number of Patches: 200 vs. 250

| **Description** | **N_P_200** | **N_P_250** |
|---|---|---|
| Validation Loss | 0.3527 | 0.3275 |
| Validation Accuracy | 0.9609 | 0.9626 |
| Validation Mean IoU | 0.6120 | 0.6260 |
| Validation Dice Coefficient | 0.7651 | 0.7894 |
| Test Loss | 0.3846 | 0.3822 |
| Test Accuracy | 0.9568 | 0.9569 |
| Test Mean IoU | 0.6088 | 0.6143 |
| Test Dice Coefficient | 0.7543 | 0.7634 |
| **Precision** | **0.8174** | 0.8148 |
| **Recall (Sensitivity)** | **0.7588** | 0.7567 |
| **Specificity** | **0.9789** | **0.9789** |
| **F1 Score** | **0.7870** | 0.7847 |
| **Jaccard Index** | **0.6488** | 0.6456 |
| **AUC** | **0.9716** | 0.9712 |
| **Mean Dice Coefficient** | **0.7636** | 0.7577 |
| **Mean Accuracy** | 0.9545 | **0.9546** |
| **Confusion Matrix** | 57046024 1230616<br>1751055 5508305 | 71413912 1541940<br>2181240 6782908 |

Table 3: U-Net Model Combined Extract Feature and Post Process Parameters

| **Description** | **Para.-EF1** | **Para.-EF2** | **Para.-EF3** | **Para.-EF4** |
|---|---|---|---|---|
| EF cliplimit | 3 | 1 | 1 | 1 |
| EF tile gridsize | 8,8 | 8,8 | 4,4 | 8,8 |
| EF kernal size (ksize) | 5,5 | 3,3 | 3,3 | 3,3 |
| EF sigma | 0 | 0 | 0 | 0 |
| EF alpha | 2 | 1 | 1 | 1 |
| EF blurred | -0.5 | -0.3 | -0.3 | 0 |
| EF gamma | 0 | 0 | 0 | 0 |
| PP Binary opening | 2,2 | 1,1 | 1,1 | 1,1 |
| PP Binary closing | 2,2 | 1,1 | 1,1 | 1,1 |

Note: EF = Extracted Feature, PP = Post Process.

## 9.1 U-Net Model - Further Experiments And Results

Further experiments were carried out to improve predicted results by changing the extracted feature parameters and post-process parameters shown in Table 3. The changes were implemented systematically, and the results were recorded in the same Excel sheet for further study. The experiments and analyzed results in Table 4(p.37) demonstrate that Recall (sensitivity), F1 score, Jaccard index, and mean Dice coefficient were all improved. As a result, extracted feature parameters such as cliplimit: 1, tilegridsize: 4x4, kernel size: 3x3, sigmaX: 0, alpha: 1, blur edge: -0.3, and gamma: 0, as well as post-process parameters such as binary opening (1,1) and binary closing (1,1), will be taken into consideration for further predicted results improvements.

## 9.2 Unet Model - Final Experimentations And Results

After optimizing the extracted feature parameters and post-process configurations, more experiments were run with learning rates ranging from 0.0001 to 0.001 and 0.01, but no improvement

Table 4: Unet Model - Performance Metrics Across Experimental Configurations

| **Description** | **EF1 N_P_200** | **EF2 N_P_200** | **EF3 N_P_200** | **EF4 N_P_200** |
|---|---|---|---|---|
| Validation Loss | 0.3953 | 0.3528 | 0.3662 | 0.3551 |
| Validation Accuracy | 0.9549 | 0.9602 | 0.9595 | 0.9598 |
| Validation Mean IoU | 0.6373 | 0.6038 | 0.6221 | 0.6106 |
| Validation Dice Coefficient | 0.7582 | 0.7695 | 0.7575 | 0.7749 |
| Test Loss | 0.4258 | 0.3810 | 0.4031 | 0.3836 |
| Test Accuracy | 0.9512 | 0.9566 | 0.9547 | 0.9562 |
| Test Mean IoU | 0.6373 | 0.6000 | 0.6190 | 0.6111 |
| Test Dice Coefficient | 0.7503 | 0.7592 | 0.7440 | 0.7654 |
| **Precision** | 0.7555 | **0.8262** | 0.7867 | 0.8177 |
| **Recall (Sensitivity)** | **0.8205** | 0.7701 | 0.8103 | 0.7781 |
| **Specificity** | 0.9669 | **0.9798** | 0.9726 | 0.9784 |
| **F1 Score** | 0.7867 | 0.7972 | **0.7983** | 0.7974 |
| **Jaccard Index** | 0.6484 | 0.6627 | **0.6643** | 0.6630 |
| **AUC** | **0.9745** | 0.9727 | 0.9730 | 0.9713 |
| **Mean Dice Coefficient** | 0.7643 | 0.7723 | **0.7761** | 0.7752 |
| **Mean Accuracy** | 0.9507 | **0.9566** | 0.9547 | 0.9562 |
| **Confusion Matrix** | [[56349296 1927344] [ 1303044 5956316]] | [[57100615 1176025] [ 1669008 5590352]] | [[56681666 1594974] [ 1377023 5882337]] | [[57017141 1259499] [ 1610989 5648371]] |

in outcomes was found. Further experiments included changing the dropout rate from 0.5 to 0.3, which did not produce better results. Subsequent studies with patch counts ranging from 200 to 230, 250, and 270 revealed that 250 patches resulted in higher F1 score, Jaccard index, AUC, mean Dice coefficient, and mean accuracy, as shown in Table 5(p.38). Further experimentation included eliminating a U-Net model layer with 512 units and comparing the results to the 250-patch arrangement; this change did not result in any improvements, as recorded in Table 5(p.38) for comparison reasons.

### 9.3 U-Net Model - Summary Of Experiments And Results

After extensive experimentation, the configuration using the parameters listed in Table 6(p.38) achieved the highest Jaccard index and the most reliable prediction results. Table 7(p.39) presents the outcomes of these final parameters, as detailed in Table 6(p.38).

### 9.4 Dense U-Net Model Experiments

Final U-Net Model Parameters in Table 6(p.38) are taken into Consideration with necessary changes like Dense unet model details to start Dense U-Net model experiments. In order to find the most suitable patch count for the Dense U-Net model, preliminary experiments were carried out using configurations comprising 200, 250, and 500 patches. The setup including 250 patches produced the greatest results in terms of mean accuracy, mean Dice coefficient, F1 score, Jaccard index, and recall (sensitivity). Table 8(p.40) provides record of these results.

Table 5: Comparison of U-Net Model Performance Across Different Configurations

| **Description** | **N_P_200** | **N_P_250** | **Remove_512 layer_NP200** |
|---|---|---|---|
| Validation Loss | 0.3662 | 0.3183 | 0.3584 |
| Validation Accuracy | 0.9595 | 0.9626 | 0.9599 |
| Validation Mean IoU | 0.6221 | 0.6093 | 0.6385 |
| Validation Dice Coefficient | 0.7575 | 0.7974 | 0.7716 |
| Test Loss | 0.4031 | 0.3696 | 0.3900 |
| Test Accuracy | 0.9547 | 0.9571 | 0.9560 |
| Test Mean IoU | 0.6190 | 0.6010 | 0.6369 |
| Test Dice Coefficient | 0.7440 | 0.7728 | 0.7612 |
| **Precision** | 0.7867 | **0.8078** | 0.8021 |
| **Recall (Sensitivity)** | **0.8103** | 0.7972 | 0.8005 |
| **Specificity** | 0.9726 | **0.9767** | 0.9754 |
| **F1 Score** | 0.7983 | **0.8025** | 0.8013 |
| **Jaccard Index** | 0.6643 | **0.6701** | 0.6685 |
| **AUC** | 0.9730 | **0.9734** | **0.9734** |
| **Mean Dice Coefficient** | 0.7761 | 0.7784 | **0.7798** |
| **Mean Accuracy** | 0.9547 | **0.9571** | 0.9560 |
| **Confusion Matrix** | [[56681666 1594974] [ 1377023 5882337]] | [[71255981 1699871] [ 1817811 7146337]] | [[56842871 1433769] [ 1448219 5811141]] |

Table 6: Final Parameters for U-Net Model Experiment

| **Configuration Description** | **Parameter values** |
|---|---|
| Target image size | 512 x 512 |
| Feature Type | Enhanced_vessel |
| Data Augmentation method | Flip and rotate [-45, 45] |
| Patch size | 128 |
| Number of patches | 250 |
| Normalization | True |
| Learning rate | 0.0001 |
| Batch size | 8 |
| Epoch | 5 |
| Seed | 30 |
| Threshold value for prediction | 0.5 |
| Unet model - Input size | 128, 128, 1 |
| Unet model – layers | 64,128,256,512 |
| Dropout | 0.5 |
| Unet model details | Convolution Layer, Batch Normalization, MaxPooling2D, Dropout |
| Extracted Feature Parameters | Clip-limit: 1, tilegridsize: 4x4, Kernel size: 3x3, sigmaX: 0, alpha: 1, blurred: -0.3, gamma: 0 |
| Post Process Parameters | Binary Opening (1,1), Binary Closing (1,1) |

Table 7: U-Net Model - Final Performance Metrics for N_P_250 Configuration

| **Description** | **N_P_250** |
|---|---|
| Validation Loss | 0.3183 |
| Validation Accuracy | 0.9626 |
| Validation Mean IoU | 0.6093 |
| Validation Dice Coefficient | 0.7974 |
| Test Loss | 0.3696 |
| Test Accuracy | 0.9571 |
| Test Mean IoU | 0.6010 |
| Test Dice Coefficient | 0.7728 |
| **Precision** | **0.8078** |
| **Recall (Sensitivity)** | **0.7972** |
| **Specificity** | **0.9767** |
| **F1 Score** | **0.8025** |
| **Jaccard Index** | **0.6701** |
| **AUC** | **0.9734** |
| **Mean Dice Coefficient** | **0.7784** |
| **Mean Accuracy** | **0.9571** |
| **Confusion Matrix** | [[71255981, 1699871], [1817811, 7146337]] |

### 9.5 Dense U-Net Model - 2nd Experiments

The learning rate was then modified, going from 0.0001 to 0.001. This change, however, had no positive effect on the results for 200 or 250 patches.

### 9.6 Dense U-Net Model -3rd Experiments

In subsequent experiments, the seed value was changed from 30 to 42 while the learning rate remained at 0.001. The 200, 250, and 300 patch sizes were put to the test. For 250 patches, better outcomes were observed for all metrics, confirming the effectiveness of this arrangement. Table 8(p.40) provides more specifics on these results.

### 9.7 Dense U-Net Model - Final Experiments

The following set of changes focused on lowering the dropout rate from 0.5 to 0.2. It performed this modification using patch sizes of 250, 300, and 400. Improvements were seen for every parameter that was analyzed, including AUC for 300 patches. Table 8(p.40) lists all of these enhancements.

### 9.8 Dense U-Net Model - Summary Of Experiments And Results

After conducting multiple experiments with different parameter combinations, the most suitable settings were discovered, as shown in Table 9(p.41). This arrangement proved to be the most successful, providing the best outcomes when compared to other combinations. In Table 8(p.40) highlights column `N_P_300_F_Exp_Res` with bold font has the large performance increases, suggesting a strong fit for the Dense U-Net model.

Table 8: Dense U-Net Model Experiment Results

| Description | N_P_250 1st Exp_Res | N_P_250 3rd Exp_Res | N_P_300 F_Exp_Res |
|---|---|---|---|
| Validation Loss | 0.4469 | 0.4345 | 0.3435 |
| Validation Accuracy | 0.9518 | 0.9521 | 0.9589 |
| Validation Mean IoU | 0.4458 | 0.4440 | 0.4421 |
| Validation Dice Coefficient | 0.7154 | 0.7261 | 0.7829 |
| Test Loss | 0.4506 | 0.4366 | 0.3938 |
| Test Accuracy | 0.9515 | 0.9509 | 0.9532 |
| Test Mean IoU | 0.4453 | 0.4423 | 0.4420 |
| Test Dice Coefficient | 0.7149 | 0.7282 | 0.7574 |
| **Precision** | 0.8782 | **0.8792** | 0.8414 |
| **Recall (Sensitivity)** | 0.6462 | 0.6662 | **0.7355** |
| **Specificity** | **0.9890** | 0.9881 | 0.9818 |
| **F1 Score** | 0.7445 | 0.7580 | **0.7849** |
| **Jaccard Index** | 0.5930 | 0.6103 | **0.6459** |
| **AUC** | 0.9695 | 0.9695 | **0.9712** |
| **Mean Dice Coefficient** | 0.7202 | 0.7286 | **0.7562** |
| **Mean Accuracy** | 0.9515 | 0.9509 | **0.9532** |
| **Confusion Matrix** | [[71255981 1699871] [1817811 7146337]] | [[71597782 865861] [1817811 7146337]] | [[85315262 1581578] [3017197 83899963]] |

# 10 Achievement Of Aim And Objectives

The aim of this dissertation was to design, develop, and evaluate a deep learning model for automating the analysis of retinal images, specifically using the DRIVE dataset, to help in the diagnosis and monitoring of vascular diseases. The aim was achieved by developing an effective algorithm that analyzed retinal images with high accuracy and reliability, accurately diagnosing and monitoring vascular conditions. The model's performance was thoroughly evaluated against the DRIVE dataset, showing its potential utility in clinical settings for diagnosing and monitoring retinal vascular diseases.

The study accomplished its objectives by selecting the DRIVE dataset for retinal vascular segmentation, comparing U-Net and Dense U-Net models to determine the best fit, then optimizing these models with TensorFlow. U-Net outperformed other methods in terms of accuracy, recall, F1 score, Jaccard Index, and Dice Coefficient, showing its usefulness in retinal image segmentation.

## 10.1 Examine And Select Datasets

The DRIVE dataset was chosen for its public availability and detailed annotations, making it perfect for creating accurate models for vascular segmentation in retinal images.

## 10.2 Deep Learning Models

The U-Net and Dense U-Net models were compared to establish the optimal fit for retinal vascular segmentation. These models were chosen because of their reliability in medical image segmentation tasks.

Table 9: Final Setup Parameters for Dense U-Net Model Experiment

| **Configuration Description** | **Parameter values** |
|---|---|
| Target image size | 512 x 512 |
| Feature Type | Enhanced_vessel |
| Data Augmentation method | Flip and rotate [-45 45] |
| Patch size | 128 |
| Number of patches | 300 |
| Normalization | True |
| Learning rate | 0.0001 |
| Batch size | 8 |
| Epoch | 5 |
| Seed | 42 |
| Threshold value for prediction | 0.5 |
| Dense Unet model - Input size | 128, 128, 1 |
| Dense Unet model – layers | 64,128,256,512 |
| Dropout | 0.2 |
| Unet model details | Convolution Layer, MaxPooling2D, Dropout |
| Extracted Feature Parameters | Clip-limit: 1, tilegridsize: 8x8, Kernel size: 3x3, sigmaX: 0, alpha: 1, blurred: 0, gamma: 0 |
| Post Process Parameters | Binary Opening (1,1), Binary Closing (1,1) |

## 10.3 Design And Development Of A Deep Learning Model

The TensorFlow framework was used to optimize both U-Net and Dense U-Net models for vascular network segmentation in the DRIVE dataset.

## 10.4 Model Evaluation Using Performance Metrics

Performance indicators show U-Net's higher capabilities compared to Dense U-net.

- **Test Accuracy**: U-Net reached 95.71%, higher than Dense U-Net's 95.32%.
- **Sensitivity (Recall)**: U-Net reported 79.72%, better than Dense U-Net's 73.55%, highlighting its efficiency in identifying positives.
- **Specificity**: U-Net recorded 97.67%, slightly below Dense U-Net's 98.18%, showing a trade-off with sensitivity.
- **F1 Score**: U-Net's 80.25% surpassed Dense U-Net's 78.49%, indicating a balanced performance between precision and recall.
- **Jaccard Index**: U-Net scored 67.01%, higher than Dense U-Net's 64.59%, suggesting better overall segmentation quality.
- **AUC (Area Under the Curve)**: U-Net achieved 97.34% compared to Dense U-Net's 97.12%, indicating superior model performance across all classification thresholds.
- **Mean Dice Coefficient**: U-Net's 77.84% was superior to Dense U-Net's 75.62%, reflecting better overlap between the predicted and actual segmentation.
- **Mean Accuracy**: U-Net also led with 95.71% over Dense U-Net's 95.32%, showcasing its overall effectiveness.

# 11 Performance Comparison With Similar Works

| Reference No. | Author Name | Sensitivity | Specificity | Accuracy |
|---|---|---|---|---|
| | Proposed Approach Performance | **0.7972** | 0.9762 | **0.9571** |
| 1 | Bukenya and Kalema (2022) | 0.7490 | 0.9740 | 0.9540 |
| 2 | Staal et al. (2004) | 0.7194 | 0.9773 | 0.9442 |
| 3 | Soares et al. (2006) | 0.7230 | 0.9762 | 0.9446 |
| 4 | Mendonça and Campilho (2006) | 0.7344 | 0.9764 | 0.9452 |
| 5 | You et al. (2011) | 0.7410 | 0.9750 | 0.9430 |
| 6 | Zhao et al. (2015) | 0.7440 | 0.9780 | 0.9530 |
| 7 | Orlando and Blaschko (2014) | 0.7850 | 0.9670 | - |
| 8 | Azzopardi et al. (2015) | 0.7660 | 0.9700 | 0.9440 |
| 9 | Wang et al. (2023) | 0.7380 | 0.9703 | 0.9403 |
| 10 | Zou et al. (2020) | 0.7761 | 0.9792 | 0.9519 |
| 11 | Du et al. (2021) | 0.7814 | **0.9810** | 0.9556 |
| 12 | Al-Diri et al. (2009) | 2009 | 0.7280 | 0.9550 |

Table 10: Performance Comparison:Sensitivity, Specificity, and Accuracy with similar work

**Table 10** shows a comparison of **sensitivity, specificity, and accuracy** across investigations, highlighting the suggested method's competitive performance. The results had the maximum sensitivity (0.7972) and accuracy (0.9571), demonstrating greater skill in identifying positive cases and overall segmentation correctness. Specificity (0.9762) was again among the highest results, with Du et al. (2021) slightly outperforming at 0.9810, suggesting strong performance in properly recognizing negative situations. When compared to previous methods, such as those by Staal et al. (2004) and Soares et al. (2006), the suggested method shows notable improvements in sensitivity and accuracy, indicating the usefulness of the U-Net-based approach with increased preprocessing and optimization methods. This confirms the approach's robustness and reproducibility in retinal vascular segmentation.

The performance comparison **Table 11** compares the effectiveness of a proposed approach to existing approaches using the **Jaccard Index and Dice Coefficient**, two essential metrics in image segmentation accuracy assessment. The proposed approach outperforms the benchmarks, with a Jaccard Index of 0.6701 and a Dice Coefficient of 0.7784, suggesting a stronger integration between predicted and real segmentations. Notably, Dash and Bhoi (2018) have the next greatest score, with a Jaccard Index of 0.666 and a Dice Coefficient of 0.724, indicating effective, if somewhat less perfect, segmentation capabilities. Orujov et al. (2020) had the lowest performance metrics at 0.550 and 0.380, which could indicate overfitting or insufficient model training.Thanh et al. (2019) achieve a modest performance with a Jaccard Index of 0.5736 and a Dice Coefficient of 0.72807, which is closer to the median benchmark in this dataset. This extensive analysis not only demonstrates the Proposed Approach's robustness in feature distinction, but it also provides a comparative view on the methodological strengths and flaws of the mentioned studies, which will help guide future image segmentation algorithm improvements.

| Reference No. | Author name | Jaccard Index | Dice Coefficient |
|---|---|---|---|
| | Proposed Approach Performance | **0.6701** | **0.7784** |
| 1 | Toptaş and Hanbay (2021) | 0.6148 | 0.7609 |
| 2 | Orujov et al. (2020) | 0.550 | 0.380 |
| 3 | Dash and Bhoi (2018) | 0.666 | 0.724 |
| 4 | Thanh et al. (2019) | 0.5736 | 0.72807 |

Table 11: Performance comparison of Jaccard Index and Dice coefficent

# 12 WorkPlan And Gantt Chart

Incorporating platforms such as Trello Trello (2024) and JIRA JIRA (2024) within the dissertation management framework gives a complete and flexible approach to dissertation management. Trello's Trello (2024) simple and user-friendly layout makes it easier to organize and distribute jobs efficiently. JIRA JIRA (2024) , on the other hand, has powerful features for monitoring dissertation milestones and building detailed Gantt charts, both of which are required for a clear grasp of dissertation time-frames and development stages. By utilizing these capabilities, dissertation supervisor may provide real-time supervision and management of dissertation from anywhere. This connection not only improves supervision but also increases productivity by allowing for seamless access via both mobile and desktop applications, meeting the dynamic needs of modern dissertation team.

## 12.1 Trello Board

Figure 13 represents a well structured execution method with Trello board Trello (2024) for a dissertation studying the use of deep learning in the diagnosis of cardiovascular disorders. It effectively divides tasks into 'To Do,' 'Doing,' and 'Done' parts, allowing for an easily understood overview of dissertation progress while maintaining an organized approach from the first proposal to the final report and presentation.

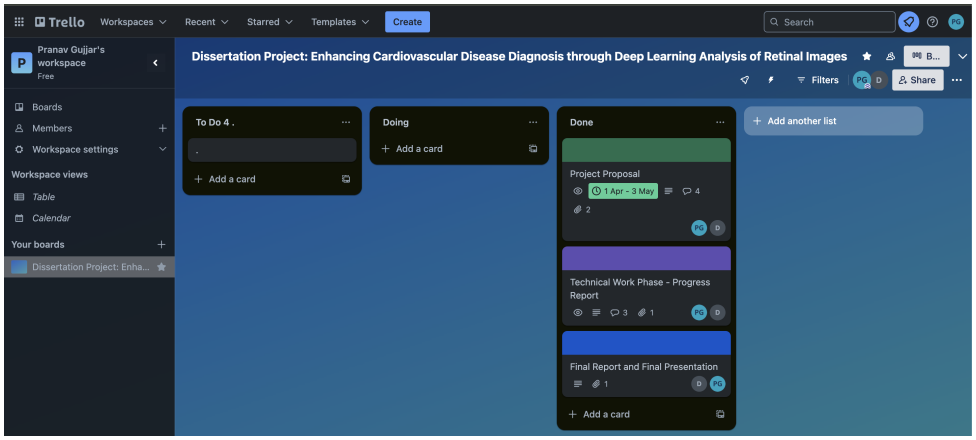



Figure 13: Project Management Tool: Trello Board

## 12.2 Gantt Chart For Dissertation

The dissertation is organized into three major steps, with each focused on one specific component of the research and development method. Here's a quick summary of the steps:

### 12.2.1 Step 1: Project Proposal

The first step is to formulate a dissertation proposal that outlines the study's objectives, parameters, and approach. This includes creating research questions or hypotheses, doing a thorough literature study, writing a proposal, and creating a methodology diagram in consultation with the project supervisor.The first step is identified as DPP (Dissertation Project Proposal) in Gantt Chart Figure 14(p.44).

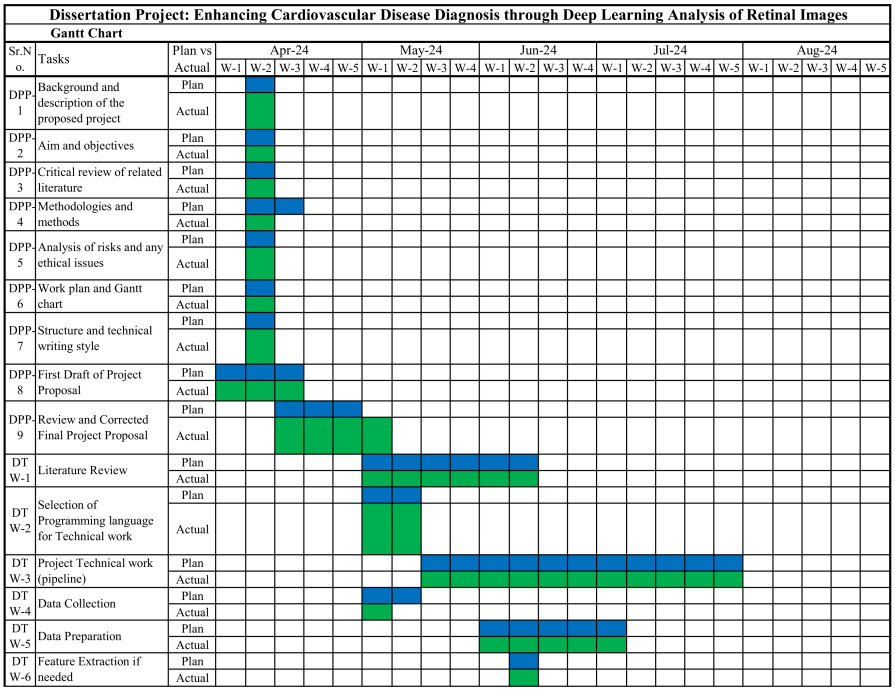



Figure 14: GANTT CHART-part1

### 12.2.2 Step 2: Technical Work

The second step includes the execution of the dissertation's technological aspects. Data collection, programming, experimentations and result analysis are all part of the study aim and Objective. It means sticking to the timeline and milestones mentioned in the dissertation proposal while putting the technique into action. The second step is denoted as DTW (Dissertation Technical Work) on Gantt Chart Figures 14(p.44) and 15(p.45).

### 12.2.3 Step 3: Final Report And Presentation

In the final step, integrate results, observations, and insights into a complete report and prepare for dissertation presentation. This includes integrating technical study findings, developing conclusions, and highlighting recommendations or complications. Create a presentation to effectively share findings with supervisor and examiners. The third step is identified as DRP (Dissertation Report and Presentation) in Gantt Chart Figure 15(p.45).

## 13 Conclusion

In this dissertation, the U-Net model outperformed the Dense U-Net model on key metrics when tested on the DRIVE dataset. Specifically, the U-Net model has an accuracy of 95.71%, a Jaccard index of 67.01%, an F1 score of 80.25%, a sensitivity of 79.72%, and a specificity of 97.34%. These measures have importance for accurately segmenting retinal images, which is required for the early detection and monitoring of vascular diseases. The findings highlight the U-Net model's clinical application potential, as it provides precise and thorough image analysis, which is important in recognizing cardiovascular problems. The experiments highlighted that the U-Net model is extremely good at processing retinal images, making it a promising tool for

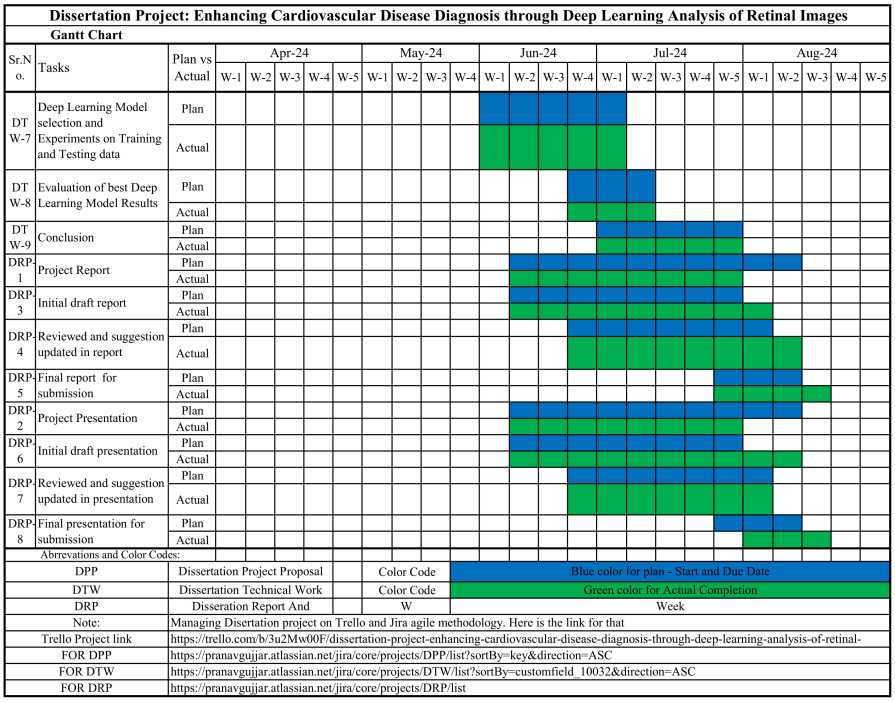



Figure 15: GANTT CHART-part2

non-invasive, accurate cardiovascular disease diagnosis.This study highlights the transformative impact of modern deep learning techniques in improving medical diagnostics, resulting in better disease management and patient outcomes via earlier and more accurate identification.

# 14 Future Scope Of Work

More advanced deep learning techniques, such as attention mechanisms and transformer-based architectures, can help improve the U-Net model's prediction outcomes. These strategies improve the model's capacity to identify relevant elements in retinal images. Another technique is to use ensemble learning, which includes training numerous models and combining their predictions to potentially improve overall performance. Furthermore, fine-tuning hyperparameters by extensive cross-validation and using approaches such as focus loss, which provides more weight to difficult-to-classify samples, can assist alleviate class imbalances and enhance segmentation accuracy.

A multi-scale method can be useful in identifying very thin vessels that the present U-Net model does not detect. This entails processing the images at various resolutions to capture finer detail. Implementing a hybrid model that combines convolutional neural networks (CNNs) and recurrent neural networks (RNNs) can also improve the recognition of very thin vessels by taking into account spatial relationships in images. Advanced data augmentation techniques, such as elastic deformations and synthetic data synthesis, can contribute to more diverse training samples. Furthermore, fine-tuning the loss function to correct missed very thin vasculature and using higher-resolution images during training can improve the model's capacity to detect extremely thin structures.

## References


Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G. S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I., Harp, A., Irving, G., Isard, M., Jia, Y., Jozefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P., Vanhoucke, V., Vasudevan, V., Viégas, F., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., and Zheng, X. (2015). TensorFlow: Large-scale machine learning on heterogeneous systems. Software available from tensorflow.org.

Abadi, M., Barham, P., Chen, J., Chen, Z., Davis, A., Dean, J., Devin, M., Ghemawat, S., Irving, G., Isard, M., Kudlur, M., Levenberg, J., Monga, R., Moore, S., Murray, D. G., Steiner, B., Tucker, P., Vasudevan, V., Warden, P., Wicke, M., Yu, Y., and Zheng, X. (2016). Tensorflow: A system for large-scale machine learning.

Abràmoff, M. D., Lou, Y., Erginay, A., Clarida, W., Amelon, R., Folk, J. C., and Niemeijer, M. (2016). Improved automated detection of diabetic retinopathy on a publicly available dataset through integration of deep learning. *Investigative ophthalmology & visual science*, 57(13):5200–5206.

Abràmoff, M. D., Lavin, P. T., Birch, M., Shah, N., and Folk, J. C. (2018). Pivotal trial of an autonomous ai-based diagnostic system for detection of diabetic retinopathy in primary care offices. *npj Digital Medicine*, 1(1):39.

Abràmoff Michael D., Garvin Mona K., S. M. and et al. (2010). Retinal imaging and image analysis. *IEEE Reviews in Biomedical Engineering*, 3:169–208.

Al-Absi Hamada R. H., Islam Mohammad Tariqul, R. M. A. C. M. E. H. A. T. and et al. (2022). Cardiovascular disease diagnosis from dxa scan and retinal images using deep learning. *Sensors*, 22(12).

Al-Diri, B., Hunter, A., and Steel, D. (2009). An active contour model for segmenting and measuring retinal vessels. *IEEE Transactions on Medical Imaging*, 28:1488–1497.

Azzopardi, G., Strisciuglio, N., Vento, M., and Petkov, N. (2015). Trainable cosfire filters for vessel delineation with application to retinal images. *Medical Image Analysis*, 19:46–57.

Boesch, G. (2024). Precision vs. recall – full guide to understanding model output. *Viso.ai*.

Bradski, G. (2000). The OpenCV Library. *Dr. Dobb's Journal of Software Tools*.

Buhl, N. (2022). Guide to image segmentation in computer vision: Best practices. *Encord Blog*.

Bukenya, F. and Kalema, A. K. (2022). A hybrid multi-scale approach for blood vessel segmentation in retina fundus images. Available at SSRN 4010366.

Chollet, F. (2017). *Deep Learning with Python*. Manning Publications Co., USA, 1st edition.

Dash, J. and Bhoi, N. (2018). An unsupervised approach for extraction of blood vessels from fundus images. *Journal of digital imaging*, 31(6):857–868.

De Fauw, J., Ledsam, J. R., Romera-Paredes, B., et al. (2018). Clinically applicable deep learning for diagnosis and referral in retinal disease. *Nature Medicine*, 24:1342–1350.

DRIVE (2024). Drive: Digital retinal images for vessel extraction. https://drive.grand-challenge.org/DRIVE/.

Du, X.-F., Wang, J.-S., and Sun, W.-Z. (2021). Unet retinal blood vessel segmentation algorithm based on improved pyramid pooling method and attention mechanism. *Physics in Medicine Biology*, 66:175013.

Esteva, A., Kuprel, B., Novoa, R., et al. (2017). Dermatologist-level classification of skin cancer with deep neural networks. *Nature*, 542:115–118.

Everingham, M., Van Gool, L., Williams, C. K. I., et al. (2010). The pascal visual object classes (voc) challenge. *International Journal of Computer Vision*, 88:303–338.

Fawcett, T. (2006). An introduction to roc analysis. *Pattern Recognition Letters*, 27(8):861–874. ROC Analysis in Pattern Recognition.

Fraz, M. M., Remagnino, P., Hoppe, A., Uyyanonvara, B., Rudnicka, A. R., Owen, C. G., and Barman, S. A. (2012). Chase db1: Retinal vessel reference dataset.

Gargeya, R. and Leng, T. (2017). Automated identification of diabetic retinopathy using deep learning. *Ophthalmology*, 124.

Gerke, S., Minssen, T., and Cohen, G. (2020). Chapter 12 - ethical and legal challenges of artificial intelligence-driven healthcare. In Bohr, A. and Memarzadeh, K., editors, *Artificial Intelligence in Healthcare*, pages 295–336. Academic Press.

Goodfellow, I., Bengio, Y., and Courville, A. (2016a). *Deep Learning*. MIT Press. Chapter 6 discusses loss functions, including binary cross-entropy.

Goodfellow, I., Bengio, Y., and Courville, A. (2016b). *Deep Learning*. MIT Press. http://www.deeplearningbook.org.

Grangeat, P., Gharbi, S., Koenig, A., Comşa, M.-P., Accensi, M., Grateau, H., Ghaith, A., Chacaroun, S., Doutreleau, S., and Vergès, S. (2020). Evaluation in healthy subjects of a transcutaneous carbon dioxide monitoring wristband during hypo and hypercapnia conditions. In *2020 42nd Annual International Conference of the IEEE Engineering in Medicine Biology Society (EMBC)*, pages 4640–4643.

Gulshan, V., Peng, L., Coram, M., Stumpe, M. C., Wu, D., Narayanaswamy, A., Venugopalan, S., Widner, K., Madams, T., Cuadros, J., Kim, R., Raman, R., Nelson, P. C., Mega, J. L., and Webster, D. R. (2016). Development and validation of a deep learning algorithm for detection of diabetic retinopathy in retinal fundus photographs. *JAMA*, 316(22):2402–2410.

Gupta, J. (2021). Sometimes accuracy is not the right metric. *Connect Jaya*.

Harris, C. R., Millman, K. J., van der Walt, S. J., Gommers, R., Virtanen, P., Cournapeau, D., Wieser, E., Taylor, J., Berg, S., Smith, N. J., Kern, R., Picus, M., Hoyer, S., van Kerkwijk, M. H., Brett, M., Haldane, A., del Río, J. F., Wiebe, M., Peterson, P., Gérard-Marchant, P., Sheppard, K., Reddy, T., Weckesser, W., Abbasi, H., Gohlke, C., and Oliphant, T. E. (2020). Array programming with NumPy. *Nature*, 585(7825):357–362.

He, K., Zhang, X., Ren, S., and Sun, J. (2015). Deep residual learning for image recognition. *CoRR*, abs/1512.03385.

Hoover, A., Kouznetsova, V., and Goldbaum, M. (2000). Stare: Structured analysis of the retina dataset.

Hoque, M. and Kipli, K. (2021). Deep learning in retinal image segmentation and feature extraction: A review. *International Journal of Online and Biomedical Engineering (iJOE)*, 17.

Hoque, M. E., Kipli, K., Zulcaffle, T. M. A., Mat, D. A. A., Joseph, A., Zamhari, N., Sapawi, R., and Arafat, M. Y. (2019). Segmentation of retinal microvasculature based on iterative self-organizing data analysis technique (isodata). In *2019 International UNIMAS STEM 12th Engineering Conference (EnCon)*, pages 59–64.

Huang, G., Liu, Z., and Weinberger, K. Q. (2016). Densely connected convolutional networks. *CoRR*, abs/1608.06993.

Hunter, J. D. (2007a). Matplotlib: A 2d graphics environment. *Computing in Science & Engineering*, 9(3):90–95.

Hunter, J. D. (2007b). Matplotlib: A 2d graphics environment. *Computing in Science Engineering*, 9(3):90–95.

Ioffe, S. and Szegedy, C. (2015). Batch normalization: Accelerating deep network training by reducing internal covariate shift. In Bach, F. and Blei, D., editors, *Proceedings of the 32nd International Conference on Machine Learning*, volume 37 of *Proceedings of Machine Learning Research*, pages 448–456, Lille, France. PMLR.

Jia, Y., Shelhamer, E., Donahue, J., Karayev, S., Long, J., Girshick, R. B., Guadarrama, S., and Darrell, T. (2014). Caffe: Convolutional architecture for fast feature embedding. *CoRR*, abs/1408.5093.

JIRA (2024). Jira software. https://www.atlassian.com/software/jira. Accessed: 2024-08-21.

Kather, J., Weis, C. A., Bianconi, F., et al. (2016). Multi-class texture analysis in colorectal cancer histology. *Scientific Reports*, 6:27988.

Klein, A. (2018). imageio/imageio: V2.4.1.

Kohavi, R. (1995). A study of cross-validation and bootstrap for accuracy estimation and model selection. In *International Joint Conference on Artificial Intelligence*.

Li, L. Y., Isaksen, A. A., Lebiecka-Johansen, B., Funck, K., Thambawita, V., Byberg, S., Andersen, T. H., Norgaard, O., and Hulman, A. (2024). Prediction of cardiovascular markers and diseases using retinal fundus images and deep learning: A systematic scoping review. *medRxiv*.

Li, Z., He, Y., Keel, S., Meng, W., Chang, R., and He, M. (2018). Efficacy of a deep learning system for detecting glaucomatous optic neuropathy based on color fundus photographs. *Ophthalmology*, 125.

Lipton, Z. C. (2018). The mythos of model interpretability: In machine learning, the concept of interpretability is both important and slippery. *Queue*, 16(3):31–57.

Long, J., Shelhamer, E., and Darrell, T. (2015). Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Luan Xue-Ning, Zhu Jie, M. D.-G. and et al. (2024). Letter: Artificial intelligence still has a long way to go in the medical field. *Angiology*. Published online April 3.

lui Cheung, C. Y., Ikram, M. K., Chen, C., and Wong, T. Y. (2017). Imaging retina to study dementia and stroke. *Progress in Retinal and Eye Research*, 57:89–107.

McKinney, W. et al. (2010). Data structures for statistical computing in python. In *Proceedings of the 9th Python in Science Conference*, volume 445, pages 51–56. Austin, TX.

Mendonça, A. M. and Campilho, A. (2006). Segmentation of retinal blood vessels by combining the detection of centerlines and morphological reconstruction. *IEEE Transactions on Medical Imaging*, 25:1200–1213.

Milletari, F., Navab, N., and Ahmadi, S. (2016). V-net: Fully convolutional neural networks for volumetric medical image segmentation. *CoRR*, abs/1606.04797.

Orlando, J. and Blaschko, M. (2014). Learning fully-connected crfs for blood vessel segmentation in retinal images. In *Medical Image Computing and Computer-Assisted Intervention (MICCAI)*, pages 634–641.

Orujov, F., Maskeliūnas, R., Damaševičius, R., and Wei, W. (2020). Fuzzy based image edge detection algorithm for blood vessel detection in retinal images. *Applied Soft Computing*, 94:106452.

Parsad, N. M. (2018). Deep learning in medical imaging. *DataDrivenInvestor*.

Pedregosa, F., Varoquaux, G., Gramfort, A., Michel, V., Thirion, B., Grisel, O., Blondel, M., Prettenhofer, P., Weiss, R., Dubourg, V., Vanderplas, J., Passos, A., Cournapeau, D., Brucher, M., Perrot, M., and Duchesnay, E. (2011). Scikit-learn: Machine learning in Python. *Journal of Machine Learning Research*, 12:2825–2830.

Perez, L. and Wang, J. (2017). The effectiveness of data augmentation in image classification using deep learning. *CoRR*, abs/1712.04621.

Pizer, S., Johnston, R., Ericksen, J., Yankaskas, B., and Muller, K. (1990). Contrast-limited adaptive histogram equalization: speed and effectiveness. In *[1990] Proceedings of the First Conference on Visualization in Biomedical Computing*, pages 337–345.

Poplin, R., Varadarajan, A. V., Blumer, K., and et al. (2018). Prediction of cardiovascular risk factors from retinal fundus photographs via deep learning. *Nature biomedical engineering*, 2:158–164.

Powers, D. M. W. (2020). Evaluation: from precision, recall and f-measure to roc, informedness, markedness and correlation. *CoRR*, abs/2010.16061.

Rao G. M., Ramesh D., S. V.-o. and et al. (2024). AttGRU-HMSI: enhancing heart disease diagnosis using hybrid deep learning approach. *Scientific Reports*, 14:7833.

Ronneberger, O., Fischer, P., and Brox, T. (2015). U-net: Convolutional networks for biomedical image segmentation. *CoRR*, abs/1505.04597.

Salz David A., W. A. J. (2015). Imaging in diabetic retinopathy. *Middle East African Journal of Ophthalmology*, 22(2):145–150.

Sanjaya, Y., Gunawan, A., and Irwansyah, E. (2020). Semantic segmentation for aerial images: A literature review. *Engineering, MAthematics and Computer Science (EMACS) Journal*, 2:133–139.

Shorten, C. and Khoshgoftaar, T. M. (2019). A survey on image data augmentation for deep learning. *Journal of Big Data*, 6:60.

Simonyan, K. and Zisserman, A. (2015). Very deep convolutional networks for large-scale image recognition.

Soares, J. V. B., Leandro, J. J. G., Cesar Jr., R. M., Jelinek, H. F., and Cree, M. J. (2006). Retinal vessel segmentation using the 2d gabor wavelet and supervised classification. *IEEE Transactions on Medical Imaging*, 25(9):1214–1222.

Sonali, Sahu, S., Singh, A. K., Ghrera, S., and Elhoseny, M. (2019). An approach for de-noising and contrast enhancement of retinal fundus image using clahe. *Optics Laser Technology*, 110:87–98. Special Issue: Optical Imaging for Extreme Environment.

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

Sudre, C. H., Li, W., Vercauteren, T., Ourselin, S., and Cardoso, M. J. (2017). Generalised dice overlap as a deep learning loss function for highly unbalanced segmentations. *CoRR*, abs/1707.03237.

Taha, A. A. and Hanbury, A. (2015). Metrics for evaluating 3d medical image segmentation: analysis, selection, and tool. *BMC Medical Imaging*, 15:29.

Tajbakhsh, N., Shin, J. Y., Gurudu, S. R., Hurst, R. T., Kendall, C. B., Gotway, M. B., and Liang, J. (2016). Convolutional neural networks for medical image analysis: Full training or fine tuning? *IEEE Transactions on Medical Imaging*, 35(5):1299–1312.

Thanh, D. N., Sergey, D., Surya Prasath, V., and Hai, N. H. (2019). Blood vessels segmentation method for retinal fundus images based on adaptive principal curvature and image derivative operators. *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, 42:211–218.

Ting, D. S. W., Cheung, C. Y.-L., Lim, G., Tan, G. S. W., Quang, N. D., Gan, A., Hamzah, H., Garcia-Franco, R., San Yeo, I. Y., Lee, S. Y., Wong, E. Y. M., Sabanayagam, C., Baskaran, M., Ibrahim, F., Tan, N. C., Finkelstein, E. A., Lamoureux, E. L., Wong, I. Y., Bressler, N. M., Sivaprasad, S., Varma, R., Jonas, J. B., He, M. G., Cheng, C.-Y., Cheung, G. C. M., Aung, T., Hsu, W., Lee, M. L., and Wong, T. Y. (2017). Development and Validation of a Deep Learning System for Diabetic Retinopathy and Related Eye Diseases Using Retinal Images From Multiethnic Populations With Diabetes. *JAMA*, 318(22):2211–2223.

Toptaş, B. and Hanbay, D. (2021). Retinal blood vessel segmentation using pixel-based feature vector. *Biomedical Signal Processing and Control*, 70:103053.

Trello (2024). Dissertation project: Enhancing cardiovascular disease diagnosis through deep learning analysis of retinal images. https://trello.com/b/3u2Mw00F/dissertation-project-enhancing-cardiovascular-disease-diagnosis-through-deep-l Accessed: 2024-08-21.

Van der Walt, S., Schönberger, J. L., Nunez-Iglesias, J., Boulogne, F., Warner, J. D., Yager, N., Gouillart, E., and Yu, T. (2014). scikit-image: image processing in python. *PeerJ*, 2:e453.

van der Walt, S., Schönberger, J. L., Nunez-Iglesias, J., Boulogne, F., Warner, J. D., Yager, N., Gouillart, E., and Yu, T. (2014). scikit-image: Image processing in python. *CoRR*, abs/1407.6245.

Van Rossum, G. (2020). *The Python Library Reference, release 3.8.2*. Python Software Foundation.

Virtanen, P., Gommers, R., Oliphant, T. E., Haberland, M., Reddy, T., Cournapeau, D., Burovski, E., Peterson, P., Weckesser, W., Bright, J., van der Walt, S. J., Brett, M., Wilson, J., Millman, K. J., Mayorov, N., Nelson, A. R. J., Jones, E., Kern, R., Larson, E., Carey, C. J., Polat, İ., Feng, Y., Moore, E. W., VanderPlas, J., Laxalde, D., Perktold, J., Cimrman, R., Henriksen, I., Quintero, E. A., Harris, C. R., Archibald, A. M., Ribeiro, A. H., Pedregosa, F., van Mulbregt, P., and SciPy 1.0 Contributors (2020). SciPy 1.0: Fundamental Algorithms for Scientific Computing in Python. *Nature Methods*, 17:261–272.

Wang, N., Li, K., Zhang, G., Zhu, Z., and Wang, P. (2023). Improvement of retinal vessel segmentation method based on u-net. *Electronics*, 12:262.

WHO (2024a). Cardiovascular diseases (cvds).

WHO (2024b). Cardiovascular diseases (cvds).

Yeung, M., Rundo, L., Nan, Y., et al. (2023). Calibrating the dice loss to handle neural network overconfidence for biomedical image segmentation. *Journal of Digital Imaging*, 36:739–752.

You, X., Peng, Q., Yuan, Y., Cheung, Y., and Lei, J. (2011). Segmentation of retinal blood vessels using the radial projection and semi-supervised approach. *Pattern Recognition*, 44:2314–2324.

Zech, J. R., Badgeley, M. A., Liu, M., Costa, A. B., Titano, J. J., and Oermann, E. K. (2018). Variable generalization performance of a deep learning model to detect pneumonia in chest radiographs: A cross-sectional study. *PLoS Medicine*, 15(11):e1002683.

Zhao, Y., Liu, Y., Wu, X., Harding, S. P., and Zheng, Y. (2015). Performance of different segmentation methods, in terms of sensitivity (se), specificity (sp), accuracy (acc) area under the curve (auc), on the drive and stare datasets. *PLOS ONE*. Dataset.

Zhu, C., Zou, B., Zhao, R., Cui, J., Duan, X., Chen, Z., and Liang, Y. (2017). Retinal vessel segmentation in colour fundus images using extreme learning machine. *Computerized Medical Imaging and Graphics*, 55:68–77. Special Issue on Ophthalmic Medical Image Analysis.

Zou, B. et al. (2020). Multi-label classification scheme based on local regression for retinal vessel segmentation. *IEEE/ACM Transactions on Computational Biology and Bioinformatics*, 18:2586–2597.

# 15 Abbreviations

WHO = World Health Organization
CVD = Cardiovascular Disease
AI = Artificial Intelligence
DL = Deep Learning
ML = Machine Learninig
CNNs = Convolutional neural networks
SVM = Support Vector machine
AUC = Area under curve
VGG = Visual Geometry Group
DPP = Dissertation Project Proposal
DTW = Dissertation Technical Work
DRP = Dissertation Report and Presentation
DR = diabetic retinopathy
AMD = age-related macular degeneration
DRIVE = Digital Retinal Images for Vessel Extraction
I/O = Input / Output
STARE = Structured Analysis of the Retina
CHASE DB1 = Child Heart and Health Study in England Dataset (retinal vessel reference dataset)