UNIVERSITY OF SOUTHAMPTON
FACULTY OF ENGINEERING AND PHYSICAL SCIENCES
DEPARTMENT OF ELECTRONICS AND COMPUTER SCIENCE

Deep Vessel Segmentation

Author:
Matthew DE VRIES

Supervisor:
Dr Xiaohao CAI
Second Examiner:
Dr Sasan MAHMOODI

September 27, 2022

A dissertation submitted in partial fulfilment
of the degree of Master of Science Artificial Intelligence

in the

Department of Electronics and Computer Science

UNIVERSITY OF SOUTHAMPTON

Abstract

Faculty of Engineering and Physical Sciences
Department of Electronics and Computer Science

Master of Science Artificial Intelligence

Deep Vessel Segmentation

by Matthew DE VRIES

Blood vessel segmentation is a crutial challenge which assists directly in several clinical fields. Many approaches exist from manual to computer-aided automatic segmentation, usually based on some variant of the U-Net. The invention of attention gates has taken the deep learning community by storm, particularly in semantic segmentation tasks with many state-of-the-art incorporating some form of attention gates. Attention is essentially a mechanism within a network which weights features by relative importance to a problem and then uses these features to solve the problem. Soft-attention is used in this project, where the weights of importance are learned through the standard backpropagation algorithm. This project explores vessel segmentation techniques using deep learning through the U-Net and builds on the current state-of-the-art known as the Iternet, to incorporate attention gates into its base module. Pre-processing included CLAHE, and green-channel extraction. A patch-based approach to training was incorporated. Experiments were conducted on the varying publicly available retinal blood vessel datasets; DRIVE, STARE, CHASE_DB1, and HRF which show the significant impact these attention gates can have on the performance of models, specifically in terms of sensitivity. Furthermore, attempts were made to use the trained models on different datasets of vessels from chloroplast in Bienertia chlorenchyma cells. Work will continue on this.

Acknowledgements

Thank you to my supervisor, Dr Xiaohao Cai, for providing extensive guidance and feedback throughout this project. Thanks also to my family, for supporting me, and for providing guidance and a sounding board when required.

Declaration of Authorship

I, Matthew DE VRIES, declare that this thesis titled, "Deep Vessel Segmentation" and the work presented in it are my own. I confirm that:

- I have read and understood the ECS Academic Integrity information and the University's Academic Integrity Guidance for Students.
- I am aware that failure to act in accordance with the Regulations Governing Academic Integrity may lead to the imposition of penalties which, for the most serious cases, may include termination of programme.
- I consent to the University copying and distributing any or all of my work in any form and using third parties (who may be based outside the EU/EEA) to verify whether my work contains plagiarised material, and for quality assurance purposes.
- I have acknowledged all sources, and identified any content taken from elsewhere. GitHub code used is acknowledged on the GitHub repository.
- I have not used any resources produced by anyone else without citation. All resources used by others are cited.
- I did all the work myself, or with my allocated group, and have not helped anyone else.
- The material in the report is genuine, and I have included all my data/code/designs.
- I have not submitted any part of this work for another assessment with an exception to the Project Preparation course which has been agreed upon.
- My work did involve human participants, their cells or data, or animals. However the data collected followed ethical procedures.

Signed: _____

Date: _____

"Abandon the idea that you are ever going to finish. Lose track of the 400 pages and write just one page for each day, it helps. Then when it gets finished you are always surprised."

John Steinbeck

Contents

List of Figures

List of Tables

List of Abbreviations

AC	**AC**curacy
AG	**A**ttention **G**ate
AUC	**A**rea **U**nder receiver operating **C**urve
BCE	**B**inary **C**ross **E**ntopy
CLAHE	**C**ontrast-**L**imited **A**daptive **H**istogram **E**qualisation
CNN	**C**onvolutional **N**eural **N**etwork
COSFIRE	**CO**mbination **S**hifted **F**ilter **RE**sponses
CT	**C**omputer **T**omography
DANN	**D**omain **A**dversarial **N**eural **N**etwork
DC	**D**ice **C**oefficient
DL	**D**ice **L**oss
DNA	**D**eoxyribo**N**ucleic **A**cid **I**maging
ELM	**E**xtreme **L**earning **M**achine
EM	**E**xpectation **M**aximisation
EMOABC	**E**lite-guided **M**ulti-**O**bjective **A**rtificial **B**ee **C**olony
FCCRF	**F**ully **C**onnected **C**onditional **R**andom **F**ield
FCN	**F**ully **C**onvolutional **N**etwork
FL	**F**ocal **L**oss
FN	**F**alse **N**egative
FP	**F**alse **P**ositive
GAN	**G**enerative **A**dversarial **N**etwork
GMM	**G**aussian **M**ixture **M**odel
H&E	**H**ematoxylin & **E**osin
HMRF	**H**idden **M**arkov **R**andom **F**ield
JS	**J**accard **S**imilarity
LSTM	**L**ong **S**hort **T**erm **M**emory
MRA	**M**agnetic **R**esonance **A**ngiography
MRF	**M**arkov **R**andom **F**ield
MRI	**M**agnetic **R**esonance **I**maging
MPP	**M**arkov **P**oint **P**rocess
OCT	**O**ptical **C**oherence **T**omography
RFCN	**R**ecurrent **F**ully **C**onvolutional **N**etwork
RNN	**R**ecurrent **N**eural **N**etwork
SE	**SE**ensitivity
SP	**SP**ecificity
SOM	**S**elf **O**rganising **M**ap
TN	**T**rue **N**egative
TP	**T**rue **P**ositive
W-BCE	**W**eighted-**B**inary **C**ross **E**ntropy

Chapter 1

Introduction

In the medical industry nowadays, imaging is a crucial component in many applications. These applications continue throughout the clinical process in diagnostic settings, as well as an addition in preparation before surgical operations. The developments of imaging techniques such as Magnetic Resonance Imaging (MRI), Computer Tomography (CT), Fundus Photography and Nuclear Medicine offer doctors with high-resolution images necessary to perform an accurate analysis. Segmentation is usually a critical step for the task of processing an excessive amount of medical images with great detail.

1.1 Background

Generally, image segmentation is the process of separating an image into numerous different parts. Instead of looking at the entire data presented in an image altogether, it may be better to centre on a certain region-based semantic object in image segmentation (Sulaiman et al., 2016). The goal of image segmentation is, therefore, to search for the meaningful regions which represent parts of certain objects for more straightforward analysis (Stockman and Shapiro, 2001). Manual segmentation can be an expensive procedure concerning time producing results which lack reproducibility or suffer from inter-observer and intra-observer variability. On the other hand, automatic methods require at least one expert clinician to evaluate the segmentation results as a gold standard. Algorithms used on medical images may require a more reliable application background than those used for standard image processing. Furthermore, high levels of noise usually interfere with medical images (Zaidi and Erwin, 2007); thus algorithms should be complex and robust enough to handle the task.

1.2 Motivation

Blood vessel analysis plays a vital role in numerous clinical fields (Ramakonar et al., 2018; De Momi et al., 2014; Folkman, 1995; McDonald and Baluk, 2002; Miri et al., 2017). Automatic blood vessel segmentation techniques are of significant interest in healthcare research as they could assist clinicians drastically. In the past few years, with the explosions of deep learning, numerous algorithms have been proposed which can perform computer-aided segmentation.

Retinal blood vessels consist of a central retinal artery, vein, as well as their branches. This central artery, also known as the arteria centralis retinæ, penetrates the optic nerve with its accompanying vein where it branches into superior and inferior vessel branches. In a tree-like fashion, the branching process continues creating a network of connected paths varying in size. Nutrients pass from the blood through the thin vessel branches called capillaries. Blockages or leaks in these capillaries cause most problems related to retinal blood vessels. There exists a close relationship between the artery and the retina, making the eyeball a window into one's health. In fact, "it is the only way for doctors to inspect the blood vessel system in the human body *in vivo*"(Li et al., 2019). Direct analysis of retinal blood vessels is highly beneficial in its accuracy and simplicity in terms of its non-invasive nature. Thus segmentation of these vessels is crucial in the diagnostic process of many bodily disorders.

Retinal blood vessel analysis initially found a use for detection of glaucoma, diabetic retinopathy, retinal neoplasms, and other retinal focused issues. Over time, research found that their structures may present significant indicators in many other diagnostic procedures such as that in cardiovascular and cerebrovascular disease, hypertension, and atherosclerosis (Dougherty, 2009). Cheung, Wong, and Hodgson (2009) described how retinal blood vessels might lay out "a lifetime summary measure of genetic and environmental exposure", thus proving their substantial merit as a risk marker for proceeding systemic disease in the patient.

Numerous observable characteristics are present in retinal blood vessels, with

the transparency, colour and diameter carrying meaningful information in diagnostic procedures. In order to measure these, accurate depictions of the vessel boundaries need to be in place, which is done through vessel segmentation. Automatic blood vessel segmentation is a challenging task, but automatic segmentation of retinal blood vessels presents several problems making the task even more arduous. Retinal images have extremely low contrast between background and vessel regions due to the image consisting of mostly red-band pixels. There also exist problems of unbalanced illumination in a single raw image, making it challenging to determine vessel from the background. Furthermore, retinopathy symptoms include colour and shape variations out of the norm, posing a problem causing algorithms missclassifying noise in images.

Another interesting use-case for retinal blood vessel segmentation is presented in biometrics. Vessel maps are unique for each individual, much like a fingerprint or one's DNA. Retinal recognition is built directly off the basis of analysing vessel structures as biometric identifiers. More so, the structure of retinal blood vessels is not entirely genetically determined, allowing identical twins to have different patterns from one another. The retinal vessels are altered through diseases such as glaucoma and diabetes; however, the structure remains unchanged throughout a person's life. The consistent nature allows retinal identification systems to be the most reliable and precise biometric behind DNA.

1.3 Research questions

The main aim of this project is to aid clinical fields in diagnosis, both planning and actual treatment, as well as evaluation and follow up. To this end, this project will explore automatic vessel segmentation techniques which can be broadly used across vessels in different parts of the anatomy and varying imaging modalities. Several actualities inspire this idea.

Firstly, manual segmentation of blood vessels carries an exorbitant price in terms of time. Along with this, it lacks both inter- and intra-operator reproducibility and repeatability. Using a typical deep-learning approach across the board as a tool for

semi-automatic or automatic segmentation of blood vessels will significantly decrease this time as well as lean towards a more robust, and reproducible method gaining the trust of more doctors. Secondly, the importance of analysing retinal blood vessels is made clear from the section above. Thirdly, there exist numerous blood vessel segmentation techniques in the literature; however, these methods are being improved year-on-year with more room for improvement.

With the recent introduction of attention gates and its significant impact on deep learning, especially in semantic segmentation tasks (Li et al., 2018; Fu et al., 2018; Schlemper et al., 2019; Sinha and Dolz, 2019; Guo et al., 2020; Niu et al., 2020; Zhao et al., 2020b), it is believed that incorporating attention into the current models could increase model performance without undertaking too much computational overhead. Essentially, this is achieved through a mechanism where a network weights features by relative importance to a task, and then uses these features to achieve the task. Coefficients, known as attention coefficients, are multiplied to outputs of the layers in a network, allowing them to focus on target regions and ignore irrelevant information. Attention is explain in more detail in Chapter 3.3.2. It is worth noting that Guo et al. (2020) recently proved how the addition of spatial attention mechanisms to the original U-Net could improve the state-of-the-art sensitivity.

Furthermore, existing deep learning models have mostly relied on local appearances learned on the regular image grid, without considering the graphical structure of the vessel shape. Effective use of the strong relationship that exists between vessel neighbourhoods can help improve the vessel segmentation accuracy. We have seen one example in the literature of a method utilising the graphical structure of vessels through graph neural networks. Implementation of this was left as the computational cost significantly outweighed the performance increase.

We consider the following research questions in order to understand which segmentation techniques and deep learning architectures are best suited for segmentation of tubular structures across the board:

1. Can an end-to-end deep-learning vessel segmentation approach for different anatomical regions and various imaging modalities be used?

2. Is there a benefit in incorporating attention into segmentation networks?

3. Can trained knowledge on vessel datasets of one region in the body be transferred to other anatomical regions?

1.4 Structure of the dissertation

The proceeding chapter (Chapter 2) introduces and discusses the relevant breakthroughs in the literature, allowing us to gain a broad understanding of where the research has been and hypothesise where it might be beneficial to explore. Chapter 3 describes the methods used in this project which is aided by Appendix A for completeness sake. Chapter 4 presents the databases used in the experimentation as well as pre-processing steps and the proposed framework of experimentation. Chapter 5 presents the results, with Chapter 6 discussing the results and concluding the project as well as presenting future research prospects.

Chapter 2

Review of Literature

In order to gain an in-depth understanding of the topic of vessel segmentation, one needs to familiarise with what has been done. This allows us to "stand on the shoulders of giants" and push the boundaries in science. Segmentation tasks occur in numerous different fields from retail to self-driving cars. This review will focus on segmentation in medical image analysis and, in particular, those based on deep learning. We briefly introduce the advances made in general medical image segmentation continuing onto specific tasks of vessel segmentation and even more specifically, retinal vessel segmentation.

2.1 General medical image segmentation

General medical image segmentation refers to the automatic segmentation of organs, tumours, or any other structure present in medical images. The most common of these are tumour and brain lesion segmentation.

Region growing segmentation is an iterative process which looks at neighbouring pixels of primary seed points and determines whether or not the pixel neighbours should be added to the region. Afifi et al. (2015) proposed a region growing segmentation algorithm which combined the traditional seed region growing with the local search process to increase performance. The method finds the seed point automatically for region growing and finds a threshold utilising an average of the minimum and the maximum grey value of the image. These algorithms were trialed on real and simulated databases. Ahlem and Layachi (2015) proposed a technique for automatic seed point selection for seed regions growing in mammogram images. They applied a threshold of the image for binary intensity, and divided the

image into black and white regions. Black regions were ignored, while the white were deemed as the suspected region. Original images were then reverted where k-nearest neighbours was applied to determine statistical features of foundation entries and find the seed point.

Segmentation is an extremely customary subject of papers which apply deep learning to medical image analysis, with some examples shown in Figure 2.1, and therefore has also seen an extraordinary diversity in methodology, which includes the construction of unique convolutional neural network (CNN)-based architectures as well as the application of recurrent neural networks (RNN) (Litjens et al., 2017a). The most famous of these novel architectures used in medical imaging is U-Net. Ronneberger, Fischer, and Brox (2015) proposed this network as well as a training strategy which utilises data augmentation in order to use available sample images more effectively and efficiently. The architecture is made up of paths to capture context as well as those that allow accurate localisation. They were able to exhibit that such network can be trained end-to-end from few images, outperforming a sliding-window convolutional network, which was the previous best method. Moreover, U-Net was fast which is essential for practical applications in industry. It is claimed that "the state-of-the-art models for image segmentation consist of variants of the encoder-decoder architecture used in U-Net" (Zhou et al., 2018a).

Çiçek et al. (2016) used a similar approach for 3D data by introducing a network for volumetric segmentation which is able to learn from sparsely annotated volumetric images in order to give a dense 3D segmentation. The architecture extends that of the previous U-Net from Ronneberger, Fischer, and Brox (2015) by replacing 2D operations with 3D equivalents.

Milletari, Navab, and Ahmadi (2016) proposed an extension to (Ronneberger, Fischer, and Brox, 2015) that incorporates ResNet-like residual blocks for 3D image segmentation. Their CNN was trained on MRI volumes of prostate, which learned to segment the whole volume in one go. A novel objective function was introduced, based on Dice coefficient (Dice, 1945), which was optimised during training. They were able to deal with situations which presented a significant imbalance between the number of background and foreground voxels. Data augmentation was done

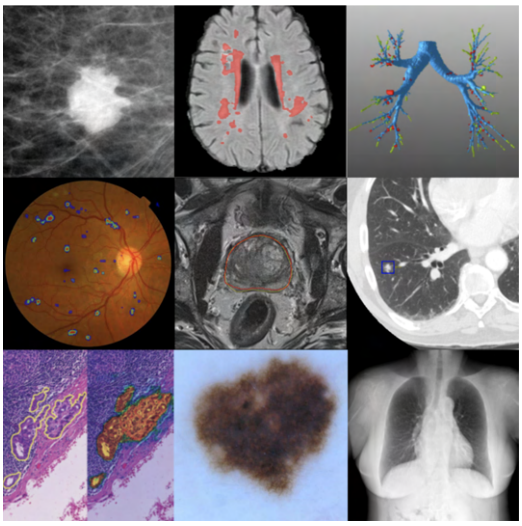


FIGURE 2.1: Collage of some medical imaging applications in which deep learning has achieved state-of-the-art results. From top-left to bottom-right: mammographic mass classification (Kooi et al., 2016), segmentation of lesions in the brain image from (Ghafoorian et al., 2016), leak detection in airway tree segmentation (Charbonnier et al., 2017), diabetic retinopathy classification (Grinsven et al., 2016), prostate segmentation (top rank in PROMISE12 challenge), nodule classification, breast cancer metastases detection in lymph nodes, human expert performance in skin lesion classification (Esteva et al., 2017), and state-of-the-art bone suppression in x-rays, image from (Yang et al., 2017) [Source: (Litjens et al., 2017b)]

in a way which applies random non-linear transformations as well as histogram matching in order to deal with the limited number of annotated volumes available for training. Their approach achieved good performance on difficult data while requiring much less processing time.

Zhou et al. (2018a) presented U-Net++, a newer, more robust network for segmentation of medical images than U-Net. Their architecture is a "deeply-supervised encoder-decoder network where the encoder and decoder sub-networks are connected through a series of nested, dense skip pathways" (Zhou et al., 2018a). The skip pathways aimed at decreasing the gap between the feature maps of the decoder and encoder networks. They were able to show that the optimiser would have an easier learning task when these feature maps similar.

RNNs have become increasingly popular for the problem of segmentation over recent years. For example, Xie et al. (2016) used a spatial clockwork RNN in order to segment H&E-stained histopathology images. This network took prior spatial information of the current patch into account.

Stollenga et al. (2015) first used a 3D Long Short Term Memory (LSTM) network with convolutional layers in multiple directions. They re-arranged the traditional order of computations in multidimensional-LSTM in pyramidal schemes. The following PyraMiD-LSTM was simple to parallelise, which is beneficial for 3D data such as stacks of brain slice images. Their PyraMiD-LSTM achieved state-of-the-art results of pixel-wise brain image segmentation on MRBrainS13 (van Opbroek, van der Lijn, and de Bruijne, 2013).

Andermatt, Pezold, and Cattin (2016) presented an RNN to segment 3D volumes of biomedical images. Their network consisted of multi-dimensional gated recurrent units. They applied an online data augmentation techniques, allowing for accurate estimations with a smaller dataset. Their method performed amongst the state-of-the-art in terms of speed, accuracy and memory efficiency on a popular brain segmentation challenge dataset.

Incorporating both U-Net architectures with RNN would then prove beneficial. Chen et al. (2016) utilised a combination of bi-directional LSTMs with 2D U-Net's to segment regions in 3D microscopy images. This was the first deep learning framework for 3D image segmentation which explicitly leveraged 3D image anisotropism.

Poudel, Lamata, and Montana (2016) proposed a recurrent fully-convolutional network (RFCN) which learnt representations of the image from a full group of 2D slices with the ability to leverage inter-slice spatial dependencies through memory units. Their RFCN simplified the analysis pipeline and enabled real time applications by combining detection and segmentation into a single algorithm.

Oktay et al. (2018) incorporated attention gates (AGs) into the standard U-Net architecture, allowing their model to be robust to various shapes and sizes of target structures. Irrelevant regions were suppressed in the model, allowing more focus to be placed on regions of interest. It was seen that the addition of these AGs added minimal computational overhead accompanied by a significant increase in model sensitivity and predictive accuracy. The main focus of the paper was on pancreatic segmentation, however it was shown how the Attention U-Net could be used across

varying datasets of different sizes to improve performance.

Summarising, segmentation in medical imaging has seen an extreme incorporation of deep learning related methods. Novel architectures have been designed to directly solve segmentation. These have obtained significant results, which often improved those obtained with FCNs (Litjens et al., 2017a).

2.2 Vessel segmentation in various parts of the anatomy

2.2.1 Unsupervised approaches

Hassouna et al. (2006) presented a statistical method for the extraction of 3D blood vessels from time-of-flight Magnetic Resonance Angiography (MRA). The histogram of the pixel intensity was classified as either background noise or blood vessel. The background noise class was modelled by one Rayleigh and two Gaussian distributions, while the blood vessel class was modelled by a single Gaussian distribution. The parameters of the Rayleigh and Gaussian distribution parameters were estimated with use of the Expectation Maximisation (EM) algorithm (Dempster, Laird, and Rubin, 1977). In order to improve segmentation quality in regions of vascular signal loss, spatial constraints through Markov Random Field (MRF) modelling were included. For this reason, MRF modelling was extremely useful when images presented with high levels of noise.

Xu et al. (2010) utilised Gaussian Mixture Models (GMMs) to fit stochastic distributions of brain vessels in MRI images. A pre-processing step through mixed-integer programming was used to reduce the amount of elements mixed. Structural equation modelling estimated the parameters of the GMM. Small branches of the vessels in the brain were able to be found with this technique with fast speeds of convergence.

Goceri, Shah, and Gurcan (2016) utilised K-means clustering to segment liver vessels. Results from initial segmentation are iteratively refined with linear contrast stretching in an attempt to construct a mask image. Regions of the vessel are then reconstructed with the marker image and the mask from the initial segmentation

and refining stages respectively. The technique showed an adaption to varying pixel intensity distributions in the image.

Shalaby et al. (2018) extracted 3D cerebrovascular from time-of-flight MRA. The multi-model nature of MRA meant that through the identification of probability models of voxel intensities, vessel structures can easily segmented through voxel-wise classification. These models were found by approximating the distribution of intensities with linear combinations of discrete Gaussians. Experiments on real and synthetic data showed strong performance.

Ghazal, Al Khalil, and El-Baz (2018) presented a parametric mixture model for cerebrovascular segmentation. Spatial interactions defined by Markov-Gibbs random field models and statistical intensities of Gaussian models were used to refine the segmentation results. The proposed technique made use of a two-stage segmentation with varying sized Gaussian kernels. Strong performance was shown on 2D slices of 3D MRA data.

Gur et al. (2019) developed an unsupervised method for blood vessel segmentation based on the field of active contours. A novel loss function, inspired by the morphological active contours without edges (Chan and Vese, 2001) optimisation technique is introduced. This allows for computational efficiency through morphological curvature operators. Their method outperformed supervised methods 3D and 4D datasets.

Wang et al. (2020) presented VasNet, an unsupervised transfer-learning vasculature-aware technique for pathovascular segmentation from small unlabelled sets of angiography images. The method is based on the Domain Adversarial Neural Network (DANN) (Ganin et al., 2016). The algorithm produces a range of information from vascular structure to blood flow rates.

Fan et al. (2020) automated cerebrovascular segmentation in an unsupervised fashion based on deep neural networks and Hidden Markov Random Fields (HMRF).

The model is trained with labelled HMRF rather than annotated images. The algorithm was tested on time-of-flight magnetic resonance angiography images outperforming common HMRF-based segmentation techniques in terms of Dice coefficient and pixel classification.

2.2.2 Supervised approaches

Nekovei and Ying Sun (1995) were one of the first to utilise machine learning techniques for vessel segmentation. They used a sliding window approach with a feed-forward neural network to classify centre pixels in small windows of angiograms. The window moved pixel by pixel across the entire image. A three-layer network architecture was shown to be sufficient for the job, outperforming other common approaches.

Zeng et al. (2016) utilised extreme learning machines (ELM) to segment liver vessels from CT images. Offset medialness, Sato, and Frangi filters were used in extracting vessel map features. Pre-processing steps involved noise reduction techniques and nonlinear anisotropic diffusion filters to reduce the effect of inhomogeneous backgrounds. The method showed fast computation with leading results among other methods tested on their dataset.

Charbonnier et al. (2016) proposed a technique which improved the segmentation of airways in thoracic CT by detecting and then removing leaks. This leak detection was described as a classification problem, such that a CNN was used perform the classification task. They made use of the fact that several segmentations could be extracted from a single algorithm by "changing the parameters that influence the amount of leaks and the tree length" (Charbonnier et al., 2016). This allowed the segmented airway tree length to be increased.

Smistad and Lovstakken (2016) presented an algorithm which made use of a CNN for detecting blood vessels from B-mode ultrasound images. This method had the ability to dictate the size and location of the vessels in the images in real-time.

Tetteh et al. (2018) presented DeepVesselNet, a network which uses deep learning and is tailored to overcome difficulties faced when extracting vessel networks or trees and corresponding features in 3D angiographic volumes.

Livne et al. (2019) utilised a U-Net architecture for brain vessel segmentation. Their network architecture was very similar to that presented in Ronneberger, Fischer, and Brox (2015) with the only difference in the number of channels being halved in each layer. Two-dimensional patches of different sizes were extracted from the images as a pre-processing step in order to be fed as input to the model. Strong performance was shown on the PEGASUS study with discussion of improvements for small vessel segmentation.

2.3 Retinal vessel segmentation

There have been multiple algorithms proposed right from the preliminary attempts to utilise deep learning for vessel segmentation (Nekovei and Ying Sun, 1995; Soares et al., 2006; Staal et al., 2004a). So far, these algorithms have mostly been applied to retinal images, since there exist several publicly available labelled databases (Moccia et al., 2018a). Again, we explore the deep learning approaches now to retinal vessel segmentation grouped into unsupervised and supervised approaches.

2.3.1 Unsupervised approaches

Bhuiyan et al. (2007) presented a technique for texture based blood vessel segmentation to overcome the complication of massive variations in local contrasts of minor vessels. They used Gabor energy filters to analyse texture features extracted from retinal images in order to construct a feature vector for each pixel. Feature vectors were classified into vessel and non-vessel based on the properties of textures using fuzzy C-means clustering.

Al-Rawi and Karajeh (2007) used genetic algorithms to find optimal parameters of matched filters for vessel segmentation tasks in retinal imaging. Genetic algorithms were used on the testing set of the DRIVE database to compute parameters

optimising sensitivity of the model.

Kande, Subbaiah, and Savithri (2010) proposed a novel automated vasculature segmentation approach in retinal images. The intensity information from green and red channels of an image was used in order to rectify no-uniform illumination in the colour fundus data. Enhanced blood vessels, obtained through match filtering, were segmented using spatially weighted fuzzy C-means clustering. These preserved the spatial structure of the vessel graphical segments. State-of-the-art unsupervised performance was shown on the DRIVE and STARE databases.

Lupaşcu and Tegolo (2011) proposed an unsupervised segmentation methodology of retinal blood vessels. They trained a Self Organising Map (SOM) and clustered the map units into two classes using K-means. The whole image is then, again, used as input to the SOM with the vessel network finally processed using a hill-climbing technique on the connected parts of the segmented output image.

Yu et al. (2012) presented a computationally fast and simple algorithm for vessel segmentation. By computing the eigenvalues of the Gaussian filtered image, a probability map of the vessel was produced. In order to segment this map, local cross-entropy thresholding of the second order was then applied. A post-processing step was then performed in an attempt to reduce false positives. The method showed state-of-the-art results at much higher speeds than competing unsupervised methods of the time on the DRIVE, STARE and HRF databases.

Roychowdhury, Koozekanani, and Parhi (2015) exploited GMMs in a novel three-stage vessel segmentation algorithm. Pre-processing included the extraction of two binary images; one following high-pass filtering, and the other following morphological reconstructed enhanced image for the vessel map. A post-processing stage was included to combine significant portions of blood vessels with classified vessel pixels. Results suggest state-of-the-art comparable accuracies on the DRIVE, STARE and CHASE_DB1 databases.

Strisciuglio et al. (2015) proposed a vessel segmentation method based on Combination of Shifted Filter Responses (COSFIRE) (Azzopardi and Petkov, 2013). Their

Bar-selective COSFIRE (B-COSFIRE) approach uses nonlinear filtering through a combination of difference-of-Gaussian filters. The two B-COSFIRE filters presented respond along vessel and at vessel endings respectively. Through the summation of the response maps, a vessel map is segmented. Promising results with fast computation was achieved on DRIVE, STARE, CHASE_DB1, and HRF databases.

Lahiri et al. (2016) automated blood vessel segmentation through unsupervised hierarchical feature learning through the ensemble of a multi-level stacked auto-encoder. It was seen that training the auto-encoders in an ensemble fashion may diversify the dictionary learnt of visual kernels for the segmentation problem. Highly significant accuracies were shown on the DRIVE dataset.

Zhang et al. (2016) utilised a filter-based approach for segmenting blood vessels in retinal images. Filters were based rotating 3D frames in functions on the Lie-group domain orientation and positions. 3D orientation scores are produced through lifting 2D images through wavelet-transforms. Vessels are enhanced from the lifted domain through Gaussian derivatives of the second order, perpendicular to the line structures. After this multi-scale filtering, outputs are projected to 2D giving enhanced vessel maps from which a binary segmentation is computed by means of thresholding. The method proved extremely computationally efficient with state-of-the-art results on a number of datasets.

Nowińska, Yavuz, and Köse (2017) developed segmentation technique involving both K-means and fuzzy C-means clustering in order to obtain segmented vessel trees. Vessel structures were enhanced through Gaussian, Gabor, and Frangi filtering before utilising a top-hat transform. The output of the clustering algorithms was passed through a post-processing step in an attempt to rid incorrectly segmented regions which were isolated. This method showed diagnosis acceptable performance on the DRIVE and STARE databases.

Lahiri et al. (2018) extended their work in Lahiri et al. (2017) through the addition of an unsupervised loss function and a structured prediction-based architecture. Their method is proposed to learn in a supervised and unsupervised manner under multitask objective settings. Performance was demonstrated on vessel segmentation

tasks on the DRIVE and STARE databases showing strong performance.

Chalakkal and Abdulla (2018) proposed the use of curvelet transform and line operators for the segmentation of small fringe vessels. An isotropic diffusion filtering and adaptive histogram equalisation were used to enhance the vessel map as a pre-processing step. Post-processing included the removal of isolated segmented areas. Results outperformed state-of-the-art supervised and unsupervised techniques in sensitivity and accuracy on the DRVIE dataset.

Bilal Khomri (2018) segmented vessel maps through an unsupervised technique founded upon the elite-guided multi-objective artificial bee colony (EMOABC) (Xiang, Zhou, and Liu, 2015) algorithm. In an attempt to minimise noise response from lesions, the algorithm makes use of an energy curve function which computes optimal thresholding values. Computational efficiency and speed was achieved through a stopping criterion on the EMOAC algorithm's parameters. This technique boasted simplicity and speed while outperforming meta-heuristic algorithms on the DRIVE and STARE databases.

Abbas et al. (2019) utilised conditional patch-based GANs. Previous deep-learning approaches have given equal importance to thick and thin vessels, whereas this method utilised a patch-based generator and discriminator networks conditioned on the sample data as well as including an extra loss function to optimise the learning of thin and thick vessel structures separately. State-of-the-art performance was shown on both the DRIVE and STARE databases.

Liu, Gu, and Lu (2019) presented an unsupervised technique utilising ensembling to combine multiple segmentation results of vessel segmentation in retinal images. The ensemble strategy allows the exploitation of advantages of numerous methods. State-of-the-art performance was shown on DRIVE, STARE, CHASE_DB1 databases.

Shah et al. (2019) presented a simplistic method based on Gabor wavelets and multi-scale line detection to enhance and segment vessel maps in retinal images.

The Gabor wavelet is well suited for directional feature detection as well as enhancing structures with varying widths, such as vessels. Results proved state-of-the-art performance on DRIVE, STARE, and HRF databases albeit with similar techniques.

Li, Comer, and Zerubia (2020) based a novel unsupervised vessel segmentation approach on their previous connected tube marked point process (MPP) model (Li, Comer, and Zerubia, 2018). Following a pre-processing step, the vessel architecture is extracted using thee connected tube MPP. The vessel is then segmented through K-means clustering. Accurate results were seen on both the DRIVE and STARE databases as well as a high G-means score.

Yang et al. (2020) used a combination of deep convolutional adversarial networks with short connection and dense blocks, dubbed SUD-GAN, for retinal vessel segmentation tasks. The GAN was set up such that the generator acquired a U-shape encoder-decoder architecture with gradient vanishing prevented through the inclusion of short connection block between convolutions. The discriminator is fully convolutional except for a single dense connection in the middle part of the network in an attempt to magnify its discriminatory ability. This method outperformed the state-of-the-art on the DRIVE and STARE databases through its ability to more accurately locate the edge of vessels.

Liu et al. (2020) used variational intensity cross channel encoders to segment vessels from Optical Coherence Tomography (OCT) angiography in an unsupervised manner. This approach tracks vessel masks through the exploitation of a common underlying structure shown in two OCT angiography images of the same area but captured with differing devices. Results show their method outperforms commonly used methods.

Zhao et al. (2020a) used general adversarial learning with a large receptive field for vessel segmentation in the context of retinal imaging. The generator outputs a realistic vessel map which the discriminator differentiates between that and samples drawn from database. Optimisation was catalysed by using a residual module in both the generator and discriminator networks. The receptive field was enlarged

dilated convolutions in the generator network which avoided an increase computational overhead. State-of-the-art performance was shown on DRIVE and STARE databases in many metrics.

Rocha et al. (2020) made use of 2D Gabor wavelets and contrast limited adaptive histogram equalisation to segment and enhance blood vessels in retinal images. These methods incorporated both mask and edge detection to perform state-of-the-art unsupervised segmentation on the DRIVE and STARE databases in terms of sensitivity.

2.3.2 Supervised approaches

Staal et al. (2004b) made use of the k-nearest neighbours algorithm to classify feature vectors which made use of properties of line elements extracted from retinal images. The line elements were created through the extraction of image ridges which correspond to the vessel tree. Experiments conducted on two separate datasets show strong performance over other methods of the time.

Soares et al. (2006) developed a method to classify each pixel in retinal images as a vessel or non-vessel on the feature vector for that pixel. Feature vectors were created using 2D Gabor filters and intensity. Simple Bayesian classifiers were used on these feature vectors which had the advantage of fast computation. Strong performance was shown both DRIVE and STARE databases with state-of-the-art results shown on the DRIVE database in terms of AUC.

Rodrigues et al. (2013) segmented retinal vasculature using OCT images. The study begun by extracting a group of 2D fundus reference images from 3D OCT and then using this as input to Support Vector Machines (SVM). The tuning parameters and the kernel of the SVM played a large role in the effectiveness of the method. The method had the ability to effectively segment pathological and healthy vessels. Therefore, this method can be used in the study of disease progression.

Fu et al. (2016) developed a novel deep learning approach to vessel segmentation which made use of boundary detection. The technique combined conditional random fields and multi-scale CNNs to approximate long-range interactions between pixels and learn hierarchical representations respectively. Along with efficient computation, the method produced state-of-the-art performance on DRIVE, STARE and CHASE_DB1 databases.

GeethaRamani and Balasubramanian (2016) used a combination of unsupervised and supervised techniques to segment retinal vasculature in fundus images. Feature vectors are created from images pre-process though Gabor filtering, half-wave rectification and colour transformations. Dimension reduction of these features was established by Principal Component Analysis. Clustering was then used to label pixels as vessels or non-vessels where the non-vessel cluster went through a further step involving ensemble classification by tree-based methods. A combination of the clustering and the ensemble classification gave the final segmentation result. The technique performed well in terms of accuracy on the DRIVE dataset.

Aslani and Sarnel (2016) developed a method which utilised B-COSFIRE filter responses as features for supervised segmentation of retinal blood vessels. A number of other features were combined into a 17D feature vector for each pixel in the image which included Gabor filter responses and "vesselness" measures. Random forest classifiers were then used to classify each pixel based on these features in order to gain a final segmentation. State-of-the-art performance was shown on both DRIVE and STARE databases in both pathological and cross training cases.

Li et al. (2016) remoulded the task of vessel segmentation from pixel-wise classification to a cross-modality data transformation problem with the modalities being the colour retinal image and the vessel map. Novel neural networks were used to learn the mappings from the modalities. Their algorithm boasted the lack of pre- and post-processing techniques with the input being the green channel of the retinal fundus image. State-of-the-art results in terms of specificity, accuracy and sensitivity were achieved on DRIVE, STARE and CHASE_DB1 datasets.

Orlando, Prokofyeva, and Blaschko (2017) proposed a method which was based

upon discriminatively trained fully connected conditional random field (FCCRF) models. SVMs were trained in a supervised manner to learn the model parameters. Fully connected models allowed to near real-time inference where the conditional random fields had the advantage of dealing with elongated and thin structures present in vessels over previous methods. Strong performance was shown on DRIVE, STARE and CHASE_DB1 databases with potential shown for use in segmentation of other tubular structures in medical imaging.

Tetteh et al. (2017) presented a feature extraction method based on inception networks for segmentation tasks through pixel classification. They extracted features through convolutions. Layers of fully convolutional networks were stacked up on the feature extraction layers. The method was tested for segmentation purposes, proving to outperform most existing hand crafted or deterministic feature schemes found in literature. Furthermore, they proposed methods of extending the feature extraction scheme to handle 3D datasets.

Zhu et al. (2017) proposed a method utilising an Extreme Learning Machine (ELM) for retinal vessel segmentation. It began by extracting a set of discriminative feature vectors for each pixel of the fundus image. Next, they constructed a matrix for individual pixels of the training dataset which were based on those feature vectors. This was used as input to the ELM. The ELM classifier produced a binary retinal vascular segmentation output. They then implemented an optimisation or de-noising process in order to remove regions less than 30 pixels which are isolated from the retinal vascular.

Xiao et al. (2018) proposed an architecture built on top of the original U-Net which incorporated weighted attention mechanisms, dubbed weighted Res-UNet. Skip connections based on He et al. (2015) were added to learn more discriminative features. Pre-processing included contrast-limited adaptive histogram equalisation (CLAHE) to enhance contrast in the image. A patch-extraction approach was used to increase the dataset through horizontal and vertical flips. The weighted attention mechanism made use of the circular template describing the region of interest. This allowed the model to only focus on the region of interest while avoiding noisy background. Strong performing segmentation results were shown on the DRVIE and

STARE databases in terms of sensitivity and accuracy.

Jiang et al. (2018a) proposed a method for retinal vessel tree segmentation. This was based on using pre-trained CNNs as feature extractors through transfer learning. This method simplified the common retinal vessel segmentation issue from segmentation of whole slide image to regional vessel element recognition and result merging. The study reported state-of-the-art results.

Liu et al. (2018) utilised a deep CNN to segment the vessel in fundus images. Furthermore, they made use of a dense connection method in series. Here, the final layer of the network may be used as the features of the first layer, in order to prevent the gradient from disappearing. State-of-the-art vessel segmentation results were reported.

Alom et al. (2018) proposed variants of the U-Net which incorporated residual and recurrent blocks. Two models, RU-Net and R2U-Net, which were recurrent CNNs and recurrent residual CNNs respectively incorporated with the U-Net. Advantages from all included techniques were utilised in a way which proved state-of-the-art performance on blood vessel segmentation in retinal images, skin cancer segmentation, and segmentation of lesions in the lung. Models were trained on patches extracted from the raw image and model hyperparameters are reported for easing replication of results.

Zhuang (2018) presented a network architecture inspired by U-Net and its variant, Recurrent Residual convolutional U-Net (R2-UNet), for retinal vessel segmentation. Their multi-branch CNN, LadderNet, included more information paths than their ancestor architecture. The LadderNet can be described as a chain of U-Net architectures with skip connections between the two U-Nets. Their implementation included chains of more than two U-Nets to produce a highly complication segmentation model. Their model proved superior over previous state-of-the-art methods with a downfall of being computationally expensive.

Yan, Yang, and Cheng (2018) introduced a segmentation loss to be used in deep learning-based vessel segmentation algorithms. With the aim of balancing out the

loss calculation in order to learn thin vessel features, they jointly adopted pixel-wise and segment-level losses. Essentially, their contribution added segment-level loss to the widely used pixel-wise loss in the literature. A problem with pixel-wise losses is their thickness inconsistency in the thinner vessels. The segment-level loss attempted to measure this inconsistency of each individual segmented vessel rather than the segmented pixel. In order to construct this loss, manual segmentation results are used to segment vessels from the vessel tree. This loss was built on a pixel-wise loss except with an adaptive weight matrix based on a diameter consistency measurement. Experiments on the DRIVE, STARE, HRF, and CHASE_DB1 datasets show that the incorporation of this loss criterion into current models significantly improved performance.

Hu et al. (2019) proposed variation of U-Net for vessel segmentation in fundus images of the retina. Their Minimal U-Net (Mi-UNet) design reduced the number of parameters in the standard U-Net architecture by almost 450 fold which helped to avoid model over-fitting and a lack of vascular detail in the output image. Furthermore, they proposed a bridge-style architecture based on their Mi-UNet which incorporated a saliency mechanism dubbed Salient U-Net (S-UNet). The significantly reduced number of learnable parameters allowed for real-time segmentation. Moreover, experimental results of the S-UNet show superior performance on the DRIVE and STARE databases with the added benefit of being able to use whole images as input without pre-processing steps.

Luo et al. (2019) presented a technique which incorporated a densely connected network and an attention mechanism into the original U-Net architecture to segment human bulbar conjunctival micro-vessels. Image pre-processing included CLAHE for vessel map enhancement. A patch-extraction procedure was incorporated instead of using whole images as input in an attempt to increase the training size of the datasets used. The dense connection had the effect of significantly reducing the number of parameters in the model which suppressed over-fitting on small datasets. Their Attention-Dense-UNet (AD-UNet) was also tested on retinal fundus images showing strong performance on the DRIVE and STARE datasets.

Fan et al. (2019) presented a method based on U-Net with a similar encoder-decoder type architecture for vessel segmentation in retinal images. Octave convolutions were utilised in an attempt to learn hierarchical multi-frequency features through feature encoder blocks. With the intuition of decoding these feature maps back to an image, an octave transpose convolution was proposed which takes as input these multi-frequency feature vectors to output restored images with spatial details. Similar to the U-Net, skip connections are adopted to feed location information from the encoder to the decoder. Being able to utilise full-image extractions rather than a patch-based approach allowed fast computation. State-of-the-art performance was shown on DRIVE, STARE, CHASE_DB1, and HRF databases.

Jin et al. (2019) proposed a method based off U-Net which exploits local features in retinal vessels. Their Deformable U-Net (DUNet) integrated deformable convolutional layers (Dai et al., 2017) in the common U-Net architecture in an attempt to ensure accurate localisation of segmented regions. Receptive fields are adaptive according to vessel shape and structure in this model allowing an increase in performance. State-of-the-art performance in terms of global accuracy was seen on the DRIVE, STARE and CHASE_DB1 databases. Trained on these public databases, the models were then tested on other datasets proving superior generalisation over other methods.

Yan, Yang, and Cheng (2019) proposed a three-stage learning procedure for segmenting blood vessels in retinal images in order to separate the process of segmenting vessels of different diameters. The stages, which included thick vessel segmentation, thin segmentation and vessel fusion, allowed the negative influence of the highly imbalanced vessel sizes to be minimised substantially. Thick vessels account for nearly 80% of the vessel map, thus separating the process was thought to decrease the overlooking of the thin vessels. In order to train the models differently in each stage, the ground truth images needed a skeletonisation pre-processing method to distinguish thick from thin vessels. The segmentation fusion stage refined the results to remove incorrectly labelled vessel pixels. Fully convolutional networks were used in each stage. State-of-the-art results were seen on the DRIVE, STARE, and CHASE_DB1 databases.

Li et al. (2019) presented what is, at the time of writing this, the state-of-the-art architecture for retinal vessel segmentation. Their method aimed at replicating the step-wise procedure of an expert annotator. Expert annotators first roughly segment the vessel map and then correct and refine this initial segmentation through refinery steps. This iterative procedure is replicated in the Iternet. A base model does the initial segmentation and then smaller refinery models update this initial segmentation. The base model utilised in their paper was the standard U-Net with the refinery models being smaller versions of the U-Net. It was shown how output after each refinery model improved in connectivity of vessels. Furthermore, they introduced a metric which could be highly beneficial in problems of vessel segmentation known as connectivity. The base model was able to accurately segment superior vessel structures with the refinery models segmenting those inferior vessels. It was shown that models were able to be trained on minimal dataset sizes without the need for patch extraction. The model is interesting and simple in its replication of human procedures leading to its significant increase in performance over other methods. It may be beneficial to examine how adding attention to the base model could increase its performance.

Guo et al. (2020) incorporated a method which did not require the availability of numerous training samples. Their Spatial-Attention U-Net (SA-UNet) made use of a spatial-attention mechanism to map attention along the spatial dimension of the original U-Net. Furthermore, they made use of dropout to ensure model generalisation. This method performs better than the Iternet described above on the DRIVE and CHASE_DB1 databases in terms of sensitivity. This further suggests how attention could be added to current methods to improve performance.

Existing CNN architectures have not taken into account the graphical structure of the vessel shape, but have rather relied on the local appearances learned on the usual image grid. Efficient use of the relationship that exists between vessel neighbourhoods may improve the accuracy of vessel segmentation problems. Shin et al. (2019) incorporated a graph neural network into a CNN architecture in order to exploit global vessel structures as well as local appearances. The architecture is extremely useful since it is easily applied to add to any type of CNN-based vessel segmentation method to significantly enhance the performance.

Table 2.1 summarises the reviewed literature. It is evident that the U-Net has formed a backbone for segmentation tasks in medical image analysis, with no exception in vessel segmentation problems. Most models have built on this with the current state-of-the-art being an iterative version of this architecture. There have been attempts to incorporate attention into these segmentation models proving to increase model performance without incurring too large a computational overhead. It may thus be necessary to explore the U-Net, Iternet, and these models with incorporated attention gates in order to understand and improve current performances.

Section	Authors	Region	Imaging Modality	Method
1	Afifi et al. (2015)	Brain	MRI	Region growing
	Ahlem and Layachi (2015)	Breast	Mammogram	Region growing
	Ronneberger, Fischer, and Brox (2015)	Neuronal	EM	U-Net
	Çiçek et al. (2016)	Kidney	CT	3DU-Net
	Milletari, Navab, and Ahmadi (2016)	Prostate	MRI	V-Net
	Zhou et al. (2018a)	Chest, liver, colon	CT	Nested U-Net
	Xie et al. (2016)	Muscle	H&E slides	Spatial CW-RNN
	Stollenga et al. (2015)	Brain	MRI	3D LSTM
	Andermatt, Pezold, and Cattin (2016)	Brain	MRI	RNN
	Chen et al. (2016)	Neuronal	EM recordings	U-Net with RNN
	Poudel, Lamata, and Montana (2016)	Heart	MRI	RCFN
	Oktay et al. (2018)	Pancreas	CT	Attention U-Net
2.1	Hassouna et al. (2006)	Brain	MRA	EM &MRF
	Xu et al. (2010)	Brain	MRI	GMMs
	Bruyninckx et al. (2010)	Liver	CT	Ant colony optimisation
	Oliveira, Feitosa, and Correia (2011)	Liver	CT	GMM
	Descombes et al. (2012)	Brain	Micro CT	Watershed algorithm
	Goceri, Shah, and Gurcan (2016)	Liver	MRI	K-means clustering
	Shalaby et al. (2018)	Brain	MRA	GMMs
	Ghazal, Al Khalil, and El-Baz (2018)	Brain	MRA	MRF
	Gur et al. (2019)	Brain	Microscopy	Field of active contours
	Fan et al. (2020)	Brain	MRA	HMRF
2.2	Nekovei and Ying Sun (1995)	Liver	Cineangiography	MLP
	Zeng et al. (2016)	Liver	MRI	ELM
	Charbonnier et al. (2016)	Thoracic	CT	CNN
	Smistad and Lovstakken (2016)	Heart	Ultrasound	CNN
	Tetteh et al. (2018)	Brain	Angiography	FCN
	Livne et al. (2019)	Brain	CT	U-Net
3.1	Bhuiyan et al. (2007)	Retinal	Fundus	Fuzzy C-means clustering
	Al-Rawi and Karajeh (2007)	Retinal	Fundus	Genetic algorithms
	Lam and Yan (2008)	Retinal	Fundus	Laplacian operators
	Kande, Subbaiah, and Savithri (2010)	Retinal	Fundus	Fuzzy C-means clustering
	Vlachos and Dermatas (2010)	Retinal	Fundus	Multi-scale line-tracking
	Lupaşcu and Tegolo (2011)	Retinal	Fundus	SOM
	Yu et al. (2012)	Retinal	Fundus	Guassian filtering
	Oliveira, Ren, and Cavalcanti (2012a)	Retinal	Fundus	Fuzzy C-means clustering
	Asad, Azar, and Hassanien (2014)	Retinal	Fundus	Ant colony optimisation
	Zhang et al. (2015)	Retinal	Fundus	SOM
	Roychowdhury, Koozekanani, and Parhi (2015)	Retinal	Fundus	GMM
	Strisciuglio et al. (2015)	Retinal	Fundus	B-COSFIRE
	Mapayi, Tapamo, and Viriri (2015)	Retinal	Fundus	Fuzzy C-means
	Sreejini and Govindan (2015)	Retinal	Fundus	Multi-scale matched filtering
	Annunziata et al. (2016)	Retinal	Fundus	Neighbourhood estimating filter
	Lahiri et al. (2016)	Retinal	Fundus	Auto-encoder
	Zhang et al. (2016)	Retinal	Fundus	Filter-based
	Lupascu and Tegolo (2016)	Retinal	Fundus	Clustering SOM
	Neto] et al. (2017)	Retinal	Fundus	Filters and curvature analysis
	Nowińska, Yavuz, and Köse (2017)	Retinal	Fundus	K-& fuzzy C-means clustering
	Tavakoli et al. (2017)	Retinal	Fundus	Radon transforms
	Santhosh Krishna, Gnanasekaran, and Aswini (2018)	Retinal	Fundus	CLAHE, Gaussian kernels
	Lahiri et al. (2018)	Retinal	Fundus	GAN
	Dash and Bhoi (2018)	Retinal	Fundus	Ostu thresholding
	Chalakkal and Abdulla (2018)	Retinal	Fundus	Curvelet transform
	Bilal Khomri (2018)	Retinal	Fundus	EMOABC
	Ali, Wan Zaki, and Hussain (2018)	Retinal	Fundus	K-means clustering
	Abbas et al. (2019)	Retinal	Fundus	GAN
	Memari et al. (2019)	Retinal	Fundus	Spatial fuzzy C-means clustering
	Liu, Gu, and Lu (2019)	Retinal	Fundus	Ensemble of multiple methods
	Shah et al. (2019)	Retinal	Fundus	Gabor wavelet
	Li, Comer, and Zerubia (2020)	Retinal	Fundus	MPP model
	Yang et al. (2020)	Retinal	Fundus	SUD-GAN
	Liu et al. (2020)	Retinal	OCT angiography	Variation auto-encoders
	Rocha et al. (2020)	Retinal	Fundus	Gabor wavelets
	Wang et al. (2020)	Retinal	DSA	DANN
3.2	Staal et al. (2004b)	Retinal	Fundus	KNN
	Soares et al. (2006)	Retinal	Fundus	Bayesian classifier
	Rodrigues et al. (2013)	Retinal	OCT	SVM
	Fu et al. (2016)	Retinal	Fundus	CNN
	GeethaRamani and Balasubramanian (2016)	Retinal	Fundus	Tree-based
	Aslani and Sarnel (2016)	Retinal	Fundus	Random forest
	Li et al. (2016)	Retinal	Fundus	CNN
	Orlando, Prokofyeva, and Blaschko (2017)	Retinal	Fundus	FCCRF
	Tetteh et al. (2017)	Retinal	Fundus	FCN
	Zhu et al. (2017)	Retinal	Fundus	ELM
	Xiao et al. (2018)	Retinal	Fundus	Weighted Res-UNet
	Jiang et al. (2018a)	Retinal	Fundus	Transfer learning
	Liu et al. (2018)	Retinal	Fundus	CNN
	Alom et al. (2018)	Retinal	Fundus	R2U-Net
	Zhuang (2018)	Retinal	Fundus	LadderNet
	Yan, Yang, and Cheng (2018)	Retinal	Fundus	Novel loss function
	Hu et al. (2019)	Retinal	Fundus	S-UNet
	Luo et al. (2019)	Retinal	Fundus	U-Net variant
	Fan et al. (2019)	Retinal	Fundus	U-Net variant
	Jin et al. (2019)	Retinal	Fundus	DUNet
	Yan, Yang, and Cheng (2019)	Retinal	Fundus	FCN
	Li et al. (2019)	Retinal	Fundus	Iternet
	Shin et al. (2019)	Retinal	Fundus	GNN

TABLE 2.1: Summarising the reviewed literature of vessel segmentation. MRI: Magnetic Resonance Imaging, EM: Electron Microscopy, CT: Computed Tomography, H&E: Hematoxylin and Eosin, MRA: Magnetic Resonance Angiography, OCT: Optical Coherence Tomography, DSA: Digital Subtraction Angiography

Chapter 3

Methodology

The task of segmentation of tubular structures such as vessels is shied from the faint-hearted due to its sheer difficulty. This chapter deals with the theoretical considerations used to deal with the task of vessel segmentation.

3.1 A brief discussion of the pre-requisites

This project focuses on the use of deep learning in supervised vessel segmentation problems. As seen in Chapter 2, these deep learning models fall under the category of fully-convolutional networks (FCN). FCNs are particular kinds of CNNs where every layer consists of convolutional layers. Any paper must be fully contained within itself. In an attempt to set up notation as well as allow the reader to have all pre-requisite knowledge available, Appendix A introduces neural networks, convolution, and convolutional neural networks. Furthermore, elementary topics of backpropagation, activation and loss functions, data augmentation and regularisation are presented in great detail. For those more advanced readers, the fundamental ideas which are specifically crucial to this problem are briefly presented.

3.1.1 Loss functions

[1]Semantic segmentation is a pixel-wise classification problem. With those of segmenting a single region from the background, the problem becomes that of binary classification with a pixel being either the region of interest or the background. Vessel segmentation problems are usually binary pixel-wise classification tasks. Other scenarios include segmenting pathological vessels and normal vessels or inferior

[1]The mathematical notation used follows from Jadon (2020)

and superior vessels. In these cases, the problem is no longer binary as there are more than two classes. Classification problems require loss functions different to those used in regression. Loss functions are based on statistical distributions of desired outputs such that Binary Cross Entropy is formed on the basis of the Bernoulli distribution with Categorical Cross Entropy on the Multinoulli distribution (Jadon, 2020). For the task of semantic segmentation, Jadon (2020) outlines numerous loss functions which have proven extremely effective in recent years. Here, those common and interesting (according to myself) are presented.

One of the most popular loss functions used in scenarios such as these is that of Binary Cross Entropy (BCE). Ma Yi-de, Liu Qing, and Qian Zhi-bai (2004) defined this as a difference measure between two distributions, which is given mathematically as:

$$C_{BCE}(y, \hat{y}) = -\sum_{i}(y_i log(\hat{y}_i) + (1 - y_i) log(1 - (\hat{y}_i))), \tag{3.1}$$

where y is the true response and $\hat{y}$ is the predicted response from a model. Variants of this include the Weighted Binary Cross Entropy:

$$C_{W-BCE}(y, \hat{y}) = -\sum_{i}(\beta y_i \log(\hat{y}_i) + (1 - y_i) \log(1 - (\hat{y}_i))), \tag{3.2}$$

where the β is a tuning parameters which trades-off between false negatives and false positives. $\beta > 1$ will decrease false negatives while $\beta < 1$ will decrease false positives. And Balanced Cross Entropy:

$$C_{BalCE}(y, \hat{y}) = -\sum_{i}(\beta y_i \log(\hat{y}) + (1 - \beta)(1 - y_i) \log(1 - (\hat{y}_i))), \tag{3.3}$$

where $\beta = 1 - \frac{y}{Height \times Width}$.

Developed by Lin et al. (2017), Focal Loss is also a variation of BCE. The aim of this loss was to steer the learning process to focus on training examples which are deemed harder to classify. Their goal was to address the issue of large data imbalance between regions of interest and background. This may prove highly beneficial in tasks of vessel segmentation where large vessels are easily segmented, with thinner vessels being completely overlooked. The large majority of vessel images are

also background thus the problem is highly imbalanced. To define the focal loss, we must rewrite BCE as a piece-wise function:

$$C_{BCE}(p, y) = \begin{cases} -\log(p) & \text{if } y = 1 \\ -\log(1-p) & \text{otherwise} \end{cases}, \tag{3.4}$$

where $p \in [0, 1]$ is the models output probability for the class labelled $y = 1$ and $y \in \{\pm 1\}$ is the ground truth. Thus we can write:

$$p_t = \begin{cases} p & \text{if } y = 1 \\ 1-p & \text{otherwise} \end{cases}, \tag{3.5}$$

such that $C_{BCE}(p, y) = C_{BCE}(p_t) = -\log(p_t)$. Now, we can define the focal loss:

$$C_{FL}(p_t) = -(1-p_t)^{\gamma} \log(p_t), \tag{3.6}$$

for some "tunable" focusing parameter $\gamma \geq 0$. Notice how the focal loss is the BCE loss multiplied by $(1-p_t)^{\gamma}$, known as the "modulating factor". Lin et al. (2017) described two important aspects of focal loss:

1. Misclassified examples with small p_t values, will cause the modulating factor to be close to 1, not affecting the loss. Where as p_t approaches 1, this modulating factor approaches 0, effectively down-weighting the loss for well classified examples.

2. The 'focusing parameter', γ, has the ability to adapt the rate at which simpler examples are down-weighted in a smooth manner. This parameter controls the modulating factors effect, where $\gamma = 0$ reduces the loss to BCE .

Lin et al. (2017) found a γ value of 2 to be superior in their experiments. In practice, it may be beneficial to use a weighted focal loss:

$$C_{FL}(p_t) = -\alpha_t (1-p_t)^{\gamma} \log(p_t), \tag{3.7}$$

for some tuning parameter $\alpha > 0$.

Another popular method used in semantic segmentation is known as Dice Loss. The Dice coefficient is used in computer vision problems to determine the equivalence between images (Jadon, 2020). Sudre et al. (2017) adapted this metric in order

to be incorporated into a loss function defined as:

$$C_{DL}(y, \hat{p}) = 1 - \frac{2y\hat{p} + 1}{y + \hat{p} + 1}. \tag{3.8}$$

Finally, a combination of different losses may also be used. An example of this technique is a novel loss function known as Combo Loss:

$$C_{CL}(y, \hat{y}) = \alpha C_{BalCE} - (1 - \alpha) C_{DL}(y, \hat{y}). \tag{3.9}$$

3.1.2 Data augmentation

A.2.4 explains, in more detail, the benefits of data-augmentation in the bias-variance trade-off. Wang et al. (2019a) claimed that "data augmentation is more important than model architectures for retinal vessel segmentation." Their experiments showed how simple U-Net models may outperform state-of-the-art complex architectures through effective augmentation regimes. A major finding was the extreme effect that image patch sampling at numerous orientation angles would have on the generalisation error.

Common data augmentation operations include shifts, rotation and scaling images. With segmentation algorithms, it is essential that if these transformations are undertaken on the inputs, they must be done on the outputs as the gold standard segmentation will change with shift and rotation of the input image. The injection of noise into training images has also proven to increase the performance of vessel segmentation (Shorten and Khoshgoftaar, 2019). When comparing varying learning algorithms or deep learning architectures, it is thus essential to take into account data augmentation. Often this isn't the case as state-of-the-art results shown in papers may be achieved through thorough data pre-processing and augmentation rather than the carefully curated models. To accurately compare models, controlled experiments must be performed.

3.2 An introduction to fully-convolutional networks

Compared to image classification and object detection tasks, semantic segmentation is widely considered as much more difficult (Tsang, 2018). Semantic segmentation

aims to label each and every pixel of an image rather than giving a single label to an entire image. In classification problems, images are downsized into feature maps through convolutional and pooling layers described in Appendix A.2.2 and A.2.3, respectively. These feature maps are then usually fed through fully-connected layers or other classification algorithms such as support vector machines or random forest classifiers, in an attempt to provide a single label for the entire input image. Cireşan et al. (2012) proposed training a CNN through a sliding-window organisation in an attempt to classify each pixel's class label with patches around that pixel as input. Although significantly outperforming every other contestant at the ISBI 2012 EM Segmentation challenge, the technique suffered some major drawbacks. Firstly, requiring the model to run individually for each patch results in ample computational time and redundancy due to patch overlapping. Furthermore, there exists a "trade-off between localisation accuracy and the use of context" (Ronneberger, Fischer, and Brox, 2015). More elegant approaches exist by replacing fully-connected layers in the CNN by 1×1 convolutions, to result in a fully-convolutional network (FCN).

Originally introduced to semantic segmentation by Long, Shelhamer, and Darrell (2014), FCNs have become the trusted architecture for the task of spatially dense prediction. Slight modification to CNNs allow FCNs to appropriately segment images in a pixel-wise manner. *Upsampling* from the *downsampled* feature maps allows FCNs to output images of the same size as the input giving pixel-wise classification, which is the basis for semantic segmentation. Replacing the fully-connected layers with convolutional layers steers the aim of image classification to image context, i.e. where regions of interest are located in the image. Image recognition networks, such as that defined by Krizhevsky, Sutskever, and Hinton (2012) and its successors (Simonyan and Zisserman, 2014; Szegedy et al., 2014) produce output without spatial information, due to this being concealed in the fully-connected layer with fixed input dimension. These fully-connected layers can be seen as convolutional layers, with a kernel the size of the entire input region (Long, Shelhamer, and Darrell, 2014) which will allow input of varying size to output classification maps. This technique has proven both absolutely and asymptotically efficient (Figure 3.1) (Long, Shelhamer, and Darrell, 2014).

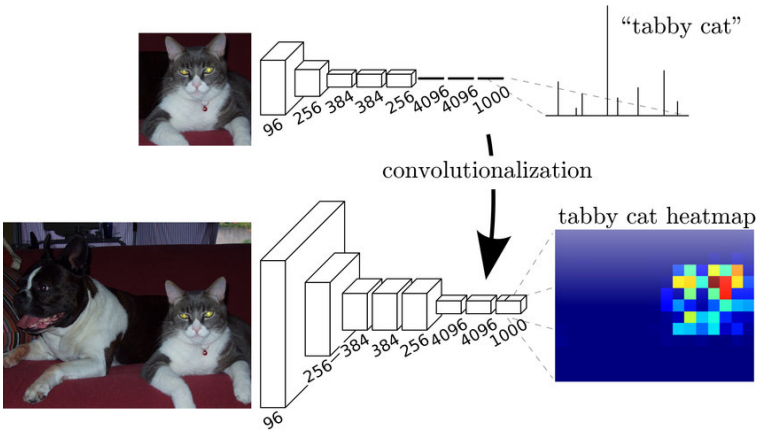


FIGURE 3.1: From CNNs to FCNs: Replacing the fully connected layers in a CNN with convolutional layers. [Source: (Long, Shelhamer, and Darrell, 2014)]

3.2.1 Transpose convolutional layer

Commonly (and mathematically mistakenly) referred to as "deconvolutional layers", transpose convolutional or upsampling layers allow FCNs to produce dense map predictions from downsampled images represented as feature maps. Upsampling with some factor f is equivalent to convolution with fractional stride $1/f$. Upsampling through transpose convolution is straight forward to implement as the forward and backward passes in convolution are simply reversed.

As explained in Section A.2.3, pooling layers reduce the number of learnable parameters by representing activations in a receptive field by some summary statistic. Although extremely efficient in classifcation tasks, spatial context is disorientated during pooling. Proposed by Noh, Hong, and Han (2015), unpooling in transpose convolutional layers reverses the process of pooling for image reconstruction (Figure 3.2).

Unpooling outputs sparse activations which the transpose convolution densifies. Instead of connecting numerous activations in some receptive field to a sole activation, as done in convolutional layers, transpose convolutional layers associate a sole activation to a window of multiple activations. The kernels which are learnt during training in transpose convolutional layers have the effect of shape reconstruction from feature maps. A hierarchical structure captures varying degrees of spatial detail, with higher layers capturing class-specific features and lower layers extracting general shape.

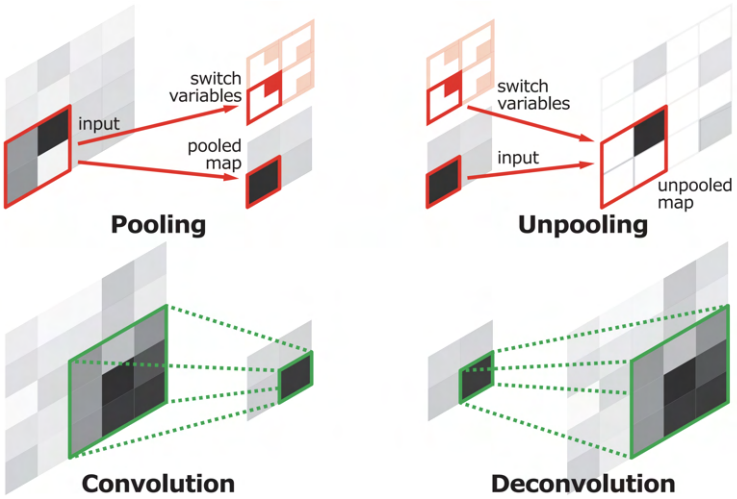


FIGURE 3.2: A graphical representation of pooling, unpooling, convolution, and deconvolution in CNNs. [Source: (Noh, Hong, and Han, 2015)]

3.3 Segmentation Architectures

In order to conduct a thorough analysis of vessel segmentation techniques, a number of architectures which have shaped the direction of current research have been selected. Firstly, due to its extreme breakthrough in segmentation tasks and its specific targeting on medical applications, implementation and analysis of the U-Net was deemed highly necessary. As described in Section 2, the U-Net has been the basis of segmentation architectures since its creation with most state-of-the-art methods consisting of a slight variation to this architecture.

A major research theme of this project was to determine whether the use of attention could improve model performance. This was primarily due to the significant effect that attention has had throughout deep learning applications. Its ease of incorporation into pre-existing models allows for controlled experimentation and a proposed increase in performance requiring minimal effort. Thus, an incorporation of attention into the standard U-Net architecture in an Attention U-Net was considered.

The current state-of-the-art vessel segmentation architecture in the Iternet. This model not only surpasses its competitors in performance among all publicly available datasets, but its simplicity and intuition are easily seen. This architecture acts in such a way which a human annotator would in stages of baseline segmentation

followed by refinery steps.

The baseline segmentation model used in the Iternet paper was the standard U-Net architecture. This U-Net was developed in 2015, with new variations recently surpassing its performance on different segmentation tasks. In an attempt to create a novel architecture and stand on the shoulders of giants such as the Iternet, an iterative architecture which incorporates attention, named the *Attention Iternet* is defined in this project.

3.3.1 U-Net

Introduced by Ronneberger, Fischer, and Brox (2015), the U-Net was the first FCN aimed at tackling the task of biomedical image segmentation. In the biomedical domain, the number of images available to train computational models is minute. U-Net aimed at accurately capturing context and localised information with few training examples yielding explicit segmentations. Traditional CNNs have an extremely large amount of learnable parameters and therefore require large datasets. Due to the size of the common datasets in medical image segmentation, models need to optimise the amount of information which could be learned from a single image. The Encoder-Decoder formation, utilised in networks like U-Net, replaces the fully-connected layers with upward convolutions in the Decoder which significantly decreases the number of learnable parameters in an attempt to decrease capacity and increase learning on unseen data.

Long, Shelhamer, and Darrell (2014)'s primary idea was to replace pooling with upsampling operators to produce output with increased resolution. Ronneberger, Fischer, and Brox (2015) modified this in a way where the upsampling section of the network would, too, contain significant feature channels, allowing the propagation of contextual details layers with higher resolution. This would result in a network in which the contracting and expansive paths are symmetric, producing a u-shaped design (Figure 3.3), hence the name. Features from the encoder or contracting path are concatenated with those from the decoder or expanding path through skip connections. These allow spatial information, which may have been disoriented through multiple convolutions, to be passed on.

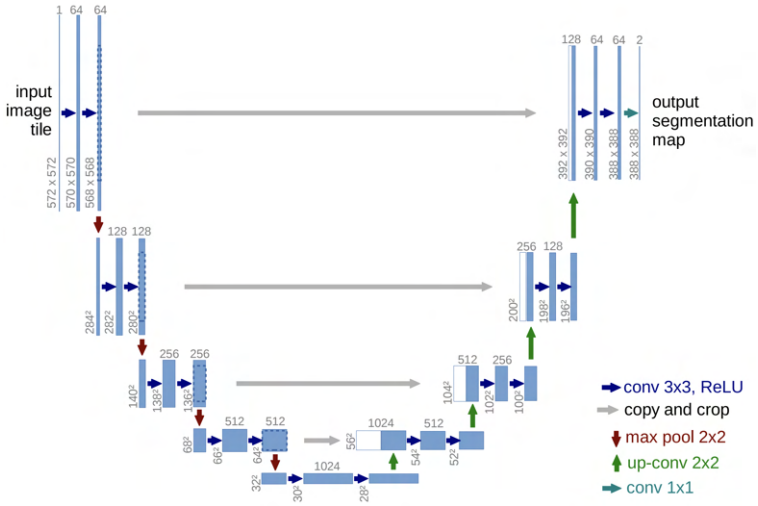


FIGURE 3.3: The U-Net architecture. [Source: (Ronneberger, Fischer, and Brox, 2015)]

Ronneberger, Fischer, and Brox (2015) detail the importance of data augmentation in biomedical segmentation tasks, especially through deformations. Deformations are common in biomedical tissue, and realistic deformations are proven to be easily recreated.

The encoder path keeps to standard CNN architecture with repeated 3×3 unpadded convolutions followed by ReLU activations and 2×2 max-pooling with a stride of 2. The amount of feature channels is doubled in each downsampling step of the contracting path. The decoder or expansive path consists of steps of upsampling followed by a 2×2 convolution, halving the feature channel number. The concatenation of this output and that from corresponding encoder step through the skip connection, are processed through two 3×3 convolutions followed by ReLU activations. A 1×1 convolution is used at the end of the network to map feature channels to the desired number of classes. A total of 23 convolutional layers are used in the network described in the paper.

The model is trained with input images and corresponding GT segmentations through stochastic gradient descent described in (Jia et al., 2014). Ronneberger, Fischer, and Brox (2015) describe the importance of weight initialisation in order to mitigate the effect of some neurons having strong activations with others having close to zero. They recommend initialising weights in such a way that each feature map will have unit variance. Sampling weights from a Gaussian distribution will achieve

this. A combined loss function with cross entropy loss and pixel-wise soft-max was used during training.

The U-Net has proved highly successful in the domain of biomedical segmentation, making it the foundation on which other methods have been built. This project aimed at exploring this architecture and its variants as well as novel variations.

3.3.2 Attention U-Net

Hore and Chatterjee (2019) define attention as "the cognitive process of selectively concentrating on a few relevant things while ignoring others". Initially created by Bahdanau, Cho, and Bengio (2014) and arguably one of the most important ideas in deep learning in recent years, attention mechanisms have changed the way we deal with high-dimensional data in tasks of natural language processing and computer vision.

Attention is a mechanism which works by weighting features by relative importance to a problem and then using these weights to improve performance. This is learned in the training process, and is categorised into hard- and soft-attention. Hard-attention relies on the use of reinforcement learning and is non-differentiable in most cases, e.g. iterative region proposal. On the other hand, soft-attention makes use of back-propagation and is represented in a probabilistic sense.

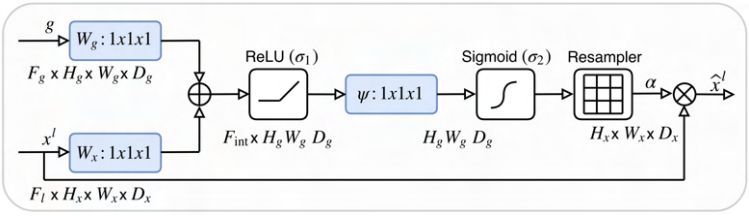


FIGURE 3.4: Additive attention gate diagram. Feature maps (x^l) from outputs of previous convolutions are scaled through learned attention coefficients (α). The spatial area is determined through analysis of both the contextual information through the gating signal (g), which is fetched from the skip connections, and the activations. [Source: (Oktay et al., 2018)]

[2]Attention coefficients, $\alpha_i \in [0,1]$, are used to maintain only those features relevant in the task at hand. Convolutional layers in a networks output high dimensional representations of images. Stacking these convolutional layers separates pixels in an image according to their semantics. We define $\boldsymbol{x_l}$ as the feature maps extracted after convolution, which is then passed through an activation function (usually ReLU): $\sigma_1\left(x_{i,c}^l\right) = \max\left(0, x_{i,c}^l\right)$, where i and c are the spatial and channel dimensions of the feature map. Attention gates (AG) combine these feature maps with attention coefficients through element-wise multiplication, such that: $\hat{\boldsymbol{x}}_{\boldsymbol{i,c}}^{\boldsymbol{l}} = \boldsymbol{x}_{\boldsymbol{i,c}}^{\boldsymbol{l}} \cdot \alpha_i^l$, allowing them to learn to focus on target regions. Attention values, are calculated for every pixel vector $c_i^l \in \mathbb{R}^{F_l}$, with the dimension, F_l, being the feature map count in layer l. For this project, there is only one semantic class thus single-dimensional attention coefficients are learned. Oktay et al. (2018) described how "each attention gate learns to focus on a subset of target structures". The gating vectors $g_i \in \mathbb{R}^{F_g}$ carries contextual information used to determine regions of interest and prune feature responses in lower-levels (shown in Figure 3.4). Additive attention is used in this project and can be defined as:

$$q_{att}^l = \psi^T\left(\sigma_1\left(W_x^T x_i^l + W_g^T g_i + b_g\right)\right) + b_\psi, \tag{3.10}$$

where $\sigma_1\left(x_{i,c}^l\right) = \max\left(0, x_{i,c}^l\right)$, is the ReLU activation function. The coefficients are calculated as:

$$\alpha_i^l = \sigma_2\left(q_{att}^l\left(x_i^l, g_i; \Theta_{att}\right)\right), \tag{3.11}$$

where $\sigma_2\left(x_{i,c}\right) = \frac{1}{1+\exp(-x_{i,c})}$ is the sigmoid activation function which is used over the softmax function due to the issue of the softmax producing more sparse activations at output layers. The set of parameters Θ_{att} which is made up of channel-wise 1x1x1 convolutions as linear transformations $W_x \in \mathbb{R}^{F_l \times F_{int}}, W_g \in \mathbb{R}^{F_g \times F_{int}}$ $\psi \in \mathbb{R}^{F_{int} \times 1}$ and bias terms $b_\psi \in \mathbb{R}, b_g \in \mathbb{R}^{F_{int}}$, defines the AG. These parameters are trained using the standard back-propagation described in Section A.1.4.

AGs may be utilised in standard CNN architectures without requiring much computational overhead while improving sensitivity and accuracy (Schlemper et al., 2018). Oktay et al. (2018) incorporated these AGs in the standard U-Net architecture

[2]The notation and description is largely taken from Oktay et al. (2018)

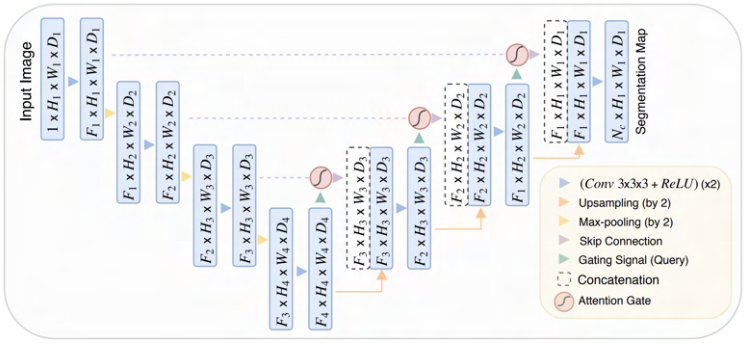


FIGURE 3.5: Attention U-Net. Images are downsampled by a factor of 2 similarly to the contracting part of the standard U-Net. Attention gates are used to filter features arriving at the expanding part through the skip connections. The segmentation output is defined by the number of semantic classes, N_c. The details of the attention gates are shown in Figure 3.4. [Source: (Oktay et al., 2018)]

for segmentation of the pancreas in CT abdominal datasets (Figure 3.5). Salient features are highlighted in the skip connections, where irrelevant information is gated in the AGs. The gating functions are incorporated directly before the concatenation through the skip connections, whereby merging only those activations which are deemed relevant. Gradients arising from regions of background are weighted less during the backward pass of back-propagation, allowing those parameters in the shallow layers of the network to be mostly updated based on spatially relevant regions. Furthermore, the addition of AGs enables models to learn to focus on regions of interest which may vary in shapes and sizes. This is clearly important in tasks of vessel segmentation. The update rule for the network parameters in layer $l-1$ is given by:

$$\frac{\partial\left(\hat{x}_i^l\right)}{\partial\left(\Phi^{l-1}\right)}=\frac{\partial\left(\alpha_i^l f\left(x_i^{l-1};\Phi^{l-1}\right)\right)}{\partial\left(\Phi^{l-1}\right)}=\alpha_i^l\frac{\partial\left(f\left(x_i^{l-1};\Phi^{l-1}\right)\right)}{\partial\left(\Phi^{l-1}\right)}+\frac{\partial\left(\alpha_i^l\right)}{\partial\left(\Phi^{l-1}\right)}x_i^l, \tag{3.12}$$

with $f\left(x^l;\Phi^l\right)=x^{(l+1)}$ being the convolution function in layer l characterised by the kernel weights Φ.

Bahdanau, Cho, and Bengio (2014) proposed the use of a vector containing context in an attempt to "align and target" inputs. This would maintain information from hidden states in RNN encoder paths to align them with current output targets. This enabled the model to "attend to" particularly relevant inputs allowing complex relationships between input and targets to be better learned. Their method proved

highly accurate on neural machine translation tasks while agreeing with human intuition.

Xu et al. (2015) were the first to propose a framework which utilises visual attention. Their method attempted to align images with words. As such, stacked convolutional layers were used as feature extractors with RNNs to align these features through the use of attention (Figure 3.6). Captions generated are "aligned" to specific parts of the image, highlighting relevant information such as objects or areas.

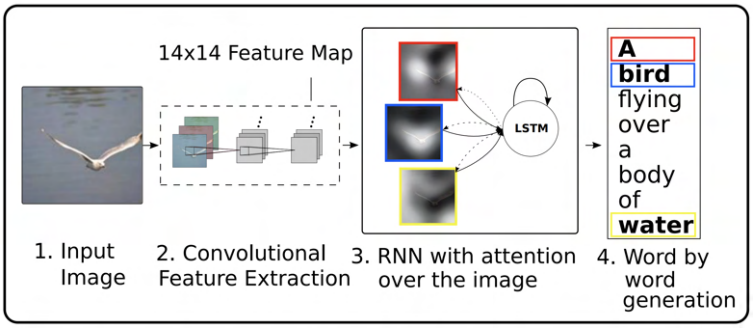


FIGURE 3.6: Using attention gates in image captioning tasks. [Source: (Xu et al., 2015)]

Oktay et al. (2018) then proposed the use of attention gates for medical image segmentation which would learn to attend to relevant structures in images without supervision.

3.3.3 Iternet

Li et al. (2019) designed Iternet for vessel segmentation which kept the process of the human expert in mind through an iterative process, hence the name. Vessel map annotators split the segmentation process into different stages: first drawing a rough segmentation map, and then refining this through intuition and knowledge that vessels are connected. The Iternet uses segmented vessel maps from a base model as input to refining models which learn to make the segmentation better or correct mistakes automatically. These models can be any segmentation models; however, Li et al. (2019) made use of U-Net as the base model and a miniature version of the U-Net (mini-UNet) as the refinery model. Their choice was based on the strong performance which U-Net has shown on multiple medical image segmentation tasks.

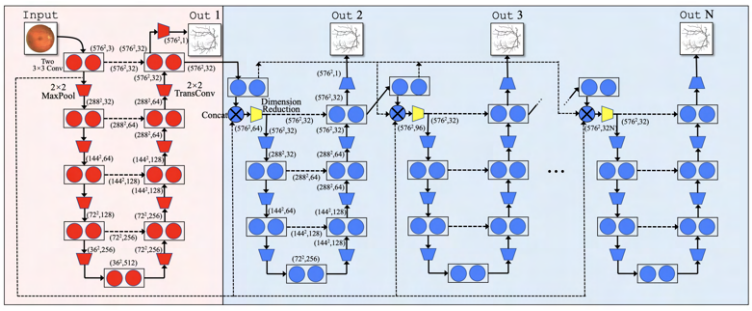


FIGURE 3.7: The Iternet architecture. [Source: (Li et al., 2019)]

U-Net outputs a segmentation map represented by probabilities of pixels being regions of interests. This is represented in a single channel. The mini-UNets, which are the refinery models, use as input the second to last layer of the preceding model. These refinery models are stacked in order to obtain better results through each addition. The use of the 32-channel input rather than the last layer of the preceding network (1-channel), allows more information to be carried to the next refinery stage. Using lightweight mini-UNets as opposed to regular U-Nets proved to improve performance significantly, probably due to over-fitting. The iterative process of the refinery models was proven to connect mistakenly split vessels over time.

Skip-connections were introduced in the Iternet, similar to those used in the U-Net. The idea was that the refinery models should be able to see what the original (raw) image looks like in order to make accurate refinements (much like an expert annotator would). Furthermore, this aimed at avoiding the issue of vanishing gradients which is present in many deep learning models. Three types of skip-connections are present in this architecture. An intra-module connection is the usual one used in U-Nets which connects encoding or contracting paths to the decoding or expanding paths. The next connects the base U-Net to the refinery models. This allows the refinery models to see information very similar to that of the input image. Here, the feature from the first layer of the base U-Net is concatenated with each feature from the first layer of the refinery modules. Finally, there exists a connection amount the refinery modules. The features from the lower modules are concatenated with the upper ones. This architecture is the current state-of-the-art across all mainstream retinal vessel segmentation datasets and is shown in Figure 3.7.

3.3.4 Attention Iternet

Here we propose a novel segmentation technique which utilises attention in the Iternet. The base module of the Iternet described above is replaced with the Attention U-Net. The Iternet is a highly adaptable model with the authors creating an iterative structure of already proposed models rather than a completely new architecture. We expect this adaption to the base module to provide the refinery modules with a better starting point on which to build. We hypothesise that these refinery modules will then be able to focus more on the inferior vessels and the connectivity that most other techniques will miss, with minimal computational overhead.

3.4 Performance Measures

We use several performance measures to analyse the methods used in this study quantitatively. In tasks of semantic segmentation, accuracy alone is not sufficient in presenting the advantages and drawbacks of specific methods. This is especially true when the data is highly imbalanced, i.e. most (c. 85%) of the input image represents background. In these cases, a model which classifies every pixel as background will obtain a highly accurate prediction, however, would prove completely useless.

We evaluate semantic segmentation tasks on a pixel-wise classification basis. In order to thoroughly compare and evaluate the different methods explored in this research, metrics such as accuracy (AC), sensitivity (SE), specificity (SP), Dice coefficient (DC), and Jaccard similarity (JS) are used. With tasks of segmenting a single class from the background in an image, the pixel-wise classification is binary, i.e. each pixel is either the region of interest (vessel) or background (not vessel). Thus, in order to calculate these performance metrics, we must first calculate the values; True Positive (TP), True Negative (TN), False Positive (FP), False Negative (FN).

3.4.1 Accuracy

Although misleading at times, accuracy is by far the most common reported metric in semantic segmentation tasks, including those of blood vessel segmentation. In fact, state-of-the-art is measured on accuracy alone. This reports on the proportion of pixels which are correctly classified by the model as vessel or non-vessel. There

have been two similar methods of calculating accuracy in retinal blood vessel segmentation tasks. The first of which would only take into account those pixels inside the field of view. Those outside the field of view would not be counted no matter how the model classified them. The second includes all the pixels in the image. The first method is not commonly used any more due to some popular datasets not providing the field of view, depriving the ability of fair comparisons on these datasets. This research will consider the second method. The accuracy is calculated as:

$$AC = \frac{TP + TN}{TP + TN + FP + FN}. \tag{3.13}$$

3.4.2 Sensitivity and Specificity

Sensitivity can be described as the *true positive rate* with specificity known as the *true negative rate*. These are commonly used in tasks of binary classification tasks as they rely directly on the computation of TP, TN, FP, and FN. Sensitivity reports on an algorithms ability to segment vessel structures where specificity reports the ability of an algorithm to circumvent noise in images. Sensitivity is calculated as:

$$SE = \frac{TP}{TP + FN}, \tag{3.14}$$

and specificity as:

$$SP = \frac{TN}{TN + FP}. \tag{3.15}$$

Both $SE, SP \in [0, 1]$ with 1 being the best a model can achieve.

3.4.3 Dice Coefficient and Jaccard Similarity

Also known as the Intersection-Over-Union (IoU), the Jaccard similarity index has been used throughout the literature in semantic segmentation tasks. The metric is simple to implement and highly effective. As seen in Figure 3.8, the Jaccard similarity is "the area of overlap between the predicted segmentation and the ground truth divided by the area of union between the predicted segmentation and the ground truth" (Tiu, 2019) and calculated as:

$$JS = \frac{|GT \cap SR|}{|GT \cup SR|}, \tag{3.16}$$

where GT refers to the ground truch and SR refers to the segmentation result from the model.

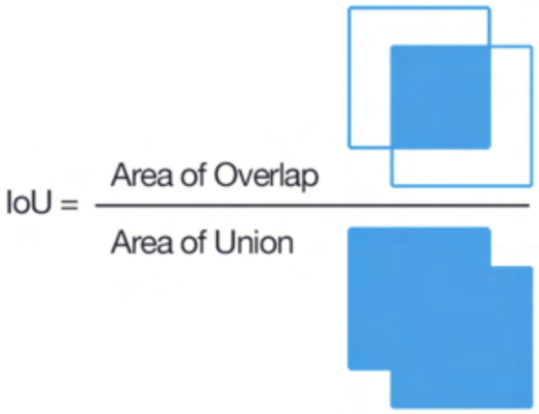


FIGURE 3.8: A graphical representation of the IoU metric. [Source: (Tiu, 2019)]

Also known as the F1-score, the Dice coefficient is the "harmonic mean of the precision and recall" (Sateesh, 2018). The Dice coefficient is positively correlated to the Jaccard similarity in the sense that if the if given two models, A and B, if the Jaccard index says model A is better than model B, then so will the Dice coefficient. Both $DC, JS \in [0, 1]$, with 1 representing a perfect score. The Dice coefficient is calculated as:

$$DC = 2\frac{|GT \cap SR|}{|GT| + |SR|}. \tag{3.17}$$

Chapter 4

Data and Implementation Details

4.1 Retinal blood vessel databases

The availability of data is what has truly driven deep learning in recent years. Publicly available databases allow researchers to develop methods in a competitive manner, constantly creating new state-of-the-art methods and pushing the boundaries further. ImageNet (Deng et al., 2009) has arguably played one of the most influential roles in the development and exposure of the power of deep learning. According to The Economist (2016) when Krizhevsky, Sutskever, and Hinton (2012) won ImageNet with AlexNet, "suddenly people started to pay attention, not just within the AI community but across the technology industry as a whole." These publicly available databases allow for controlled comparison and benchmarks on which to build novel architectures.

The study of blood vessel segmentation has largely focused on that of retinal images due to the availability of databases. These retinal blood vessel databases consist of fundus photography which involves capturing the rear of the eye, known as the fundus. Specialised cameras are used to capture these images which include a microscope and an attached flash. The main structures which are present in these images are the macula, optic disc as well as the central and peripheral retina. An important feature of these databases for deep learning segmentation purposes, is the availability of the ground truth of the vessel structures. These are the expertly annotated vessel maps. There a currently nine different publicly available retinal blood vessel databases, four of which include ground truth annotations. Those which include the ground truth are: DRIVE (Staal et al., 2004c), STARE (Goldbaum, 1975),

CHASE_DB1 (Owen et al., 2009), HRF (Köhler et al., 2013). The following subsections describe the datasets in more detail with Table 4.1.

Databases	Number of images	Resolution
DRIVE	40	768×584
STARE	400 (20 with GT)	605×700
CHASE_DB1	28	1280×960
HRF	45	3504×2336

TABLE 4.1: Description of retinal databases.

4.1.1 DRIVE

The aim of the DRIVE (Digital Retinal Images for Vessel Extraction) database was to allow for comparative studies of blood vessel segmentation techniques in retinal images. Consisting of 40 colour fundus images, the DRIVE database is one of the largest and most commonly used retinal blood vessel segmentation databases. These images were randomly chosen from a larger dataset consisting of 400 subjects between 25-90 years of age. Of the 40 available images, 7 present signs of early diabetic retinopathy while the remaining 33 show no signs. The images are JPEG compressed and obtained using 8-bit colour planes with size 768×584. A field of view is provided for each image which is approximately 540 pixels in diameter.

The database has been split into training and testing sets, both consisting of 20 images each. The training images are accompanied with a single manually annotated segmentation of the vessel map, while the test images are accompanied with two. One of these is considered the gold standard while the other is that from an independent human observer.

4.1.2 STARE

Derived from the STARE (Structured Analysis of the Retina) project, the STARE database is commonly used along with DRIVE in most retinal segmentation papers. The database consists of 400 fundus images of which 20 are accompanied by their ground truth vessel segmentation map from two separate expert annotators. The images were obtained using 8-bit colour planes with size 605×700.

Of those 20 images which are accompanied by their ground truth, 11 present signs of some form of retinal disease while the remaining are considered healthy. This may be viewed as not being representative of the population and may cause difficulties in generalisation of the models built on this database.

4.1.3 CHASE_DB1

The CHASE_DB1 database is a fundus image database which is part of a larger project known as CHASE (Child Heart and Health Study in England). This subset database consists of fundus images taken from both eyes of 14 children subjects which leads to a total of 28 images. The images of resolution 1280×960 were obtained using a 35-degree field-of-view on a handheld NM-200-D fundus camera. Each image is accompanied by two expertly annotated ground truth segmentations. There is no information on the health records of each retinal image, however the images present decent quality with good lighting and contrast.

4.1.4 HRF

The HRF (High Resolution Fundus) database was created to support the study of automatic blood vessel segmentation techniques in retinal images. Obtain through the us of a Canon CR-1 fundus camera with a field of view of 45-degrees, the images consist of the highest resolution among all the dataset with a size of 3504×2336. The dataset consists of a total of 45 retinal images consisting of equal splits of healthy, diabetic retinopathy, and glaucomatous patients. Each raw fundus image is accompanied by a ground truth vessel map which was obtained from a group of experts.

4.2 Pre-processing

Pre-processing steps have consistently played a role in vessel segmentation procedures. Raw fundus images exhibit unbalanced illumination, poor contrast and blurred vessels. Vessel enhancement approaches enable the improvement of vessel map perception. Furthermore, it may be impractical for CNNs to process fundus images in their entirety due to their size. A number of different pre-processing techniques exist which aim to add contrast between vessel and background as well as get images into suitable sizes to work with deep learning models. Moccia et al. (2018a)

details numerous techniques, however this project will focus on those common and efficient in experimentation.

4.2.1 Patch Extraction

Training CNN model from scratch usually requires a large number of images. Many medical imaging datasets, including retinal blood vessel databases, consist of very few images which are simply not enough to train complex models. Data-augmentation (described in Section A.2.4) could be used to help models generalise on unseen data, however a common approach to "increase" the dataset size as well as allowing computational efficiency is to utilise patch-extraction.

Inspired by the works done by Alom et al. (2018), a random patch-extraction process has been utilised in this project. Patches of resolution 48×48 pixels were randomly extracted from each image and its corresponding ground truth. It was guaranteed that the patches would consist of relevant information (in the field-of-view) through discarding those patches outside the field-of-view. Patches which consisted of purely black tiles were skipped and the random extraction was then repeated until a desired number of patches were extracted for each image. Algorithm 1 describes the process in more detail. Figure 4.1 shows a batch of the patches extracted from the raw images and their corresponding ground truth patches.

Random patch extraction was useful in training the models as any predetermined number of patches could be extracted from each image with overlapping patches. However, for visualisation purposes of a built model. It may be necessary to extract non-overlapping patches in a grid-like formation. In this way, we are able to reconstruct the full retinal image by storing the spatial index of each patch.

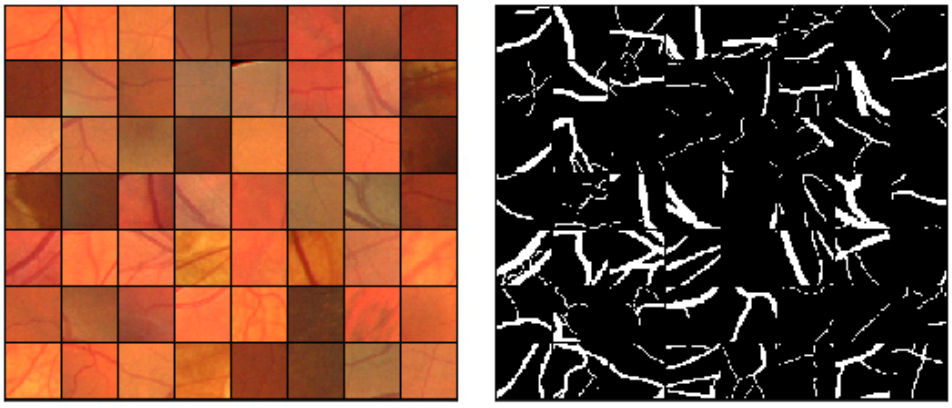
4.2.2 Grey-scale Conversion

Retinal images usually have low contrast between the vessel and the background making it difficult for automatic segmentation algorithms to accurately extract vessel maps. The green channel of these images however, present high contrast with darker regions representing vessels on a lighter background. This allows the vessel structure to be easily seen with a naked eye. Figure 4.2 shows a comparison of the different channels of the retinal image. It is easily seen that the green channel provides the modes contrast between the vessel structure and the background. This

Algorithm 1: Random Patch Extraction

Input: Train images $X \in \mathbb{R}^{N \times C \times H \times W}$, ground truths $G \in \mathbb{R}^{N \times 1 \times H \times W}$
Input: Patch size p
Input: Number of patches per image N_p
Output: Patch train images $I \in \mathbb{R}^{(N \times N_p) \times C \times p \times p}$, ground truths $T \in \mathbb{R}^{(N \times N_p) \times 1 \times p \times p}$

```
iter-tot = 0
for i ← 0 to N do
    k = 0
    while k < N_p do
        Randomly generate centre coordinates of the patch
        Patches I and labels T are extracted from X and G centred on (x, y),
         respectively
        if Not inside FOV then
            continue
        else
            Save patch
            iter − tot+ = 1
            k+ = 1
```



(A) Input images. (B) Ground truth labels.

FIGURE 4.1: An example of patches extracted from the DRIVE dataset.

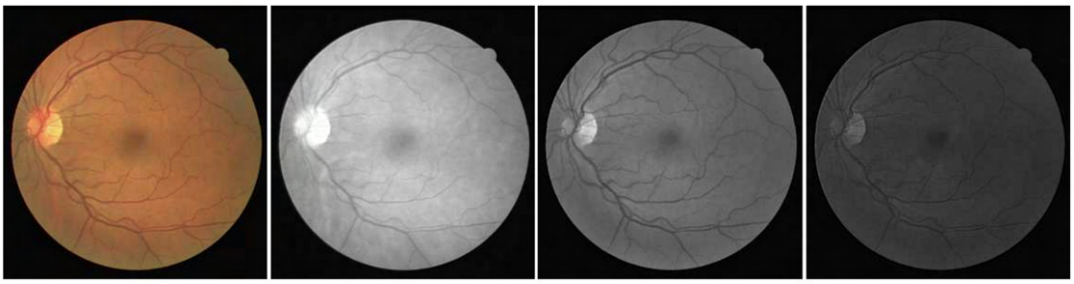


FIGURE 4.2: Comparison of the contrast between red, green and blue channels of a colour retinal fundus image. From left to right: The original colour retinal fundus image. The red channel of the image. The green channel of the image. The blue channel of the image.[Source Szpak and Tapamo (2008)]

is an extremely efficient form of pre-processing and requires the absolute minimum computation. Many techniques (discussed in Section 2) have utilised grey-scale conversion of the retinal image into its green channel only, providing exceptional results when doing so. For this project, I have adopted the technique of using the green-channel of the raw image as it is simple and effective.

4.2.3 Contrast-limited adaptive histogram equalisation

Image enhancement methods commonly include histogram equalisation due to its simplicity and lack of computational requirements. Contrast-limited adaptive histogram equalisation (CLAHE) is a form of histogram equalisation which is commonly used to enhance either colour or green-channel only retinal images. An image histogram is a statistical representation of intensity values. Manipulating this histogram allows enhancement of images. The objecting of histogram manipulation in this context is to obtain a uniform distribution of the intensity.

Adaptive histogram equalisation is used in an attempt to improve the contrast in images. This is different from normal histogram equalisation in the sense that multiple histograms are computed, with each corresponding to different regions of the image. These histograms are then used to redistribute intensity values of the image. Adaptive histogram equalisation, however, has the tendency to over-amplify noise in near-constant sections, which is commonly present in medical images. The variant of this, CLAHE, limits this contrast amplification so as to reduce the issue of noise amplification.

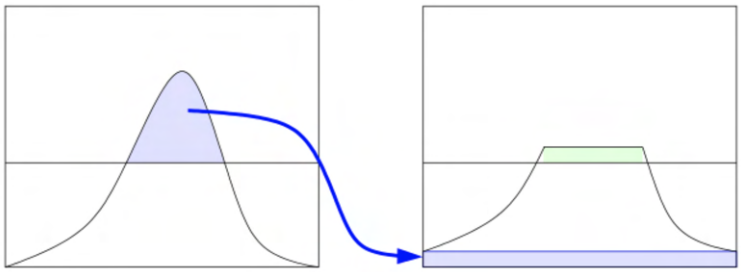


FIGURE 4.3: Histogram redistribution as seen in CLAHE.

The contrast amplification for each pixel in CLAHE is computed from the slope of the transformation function which is proportional to the slope of cumulative distribution function (CDF) of the neighbourhood. This is therefore proportional to the histogram value at that pixel. Amplification limits are defined by clipping the histogram at specific values before the CDF is computed. This in turn limits to the transformation function (Pizer et al., 1987). This clipping limit is usually chosen to be between 3 and 4 times the mean value of the histogram. Th clipped part is not discarded but rather redistributed as shown in Figure 4.3.

CLAHE was implemented using the OpenCV library (Bradski, 2000). Figure 4.4 shows the effect of CLAHE on vessel enhancement. It is easily seen how the vessel structure is visually amplified with smaller vessel which weren't previously visible in the green channel image (Figure 4.4b), coming to light in the image after CLAHE is applied (Figure 4.4c).

(A) Raw image. (B) Green channel. (C) CLAHE of the green channel.

FIGURE 4.4: Showing the effect of CLAHE on green channel retinal images of the DRIVE dataset.

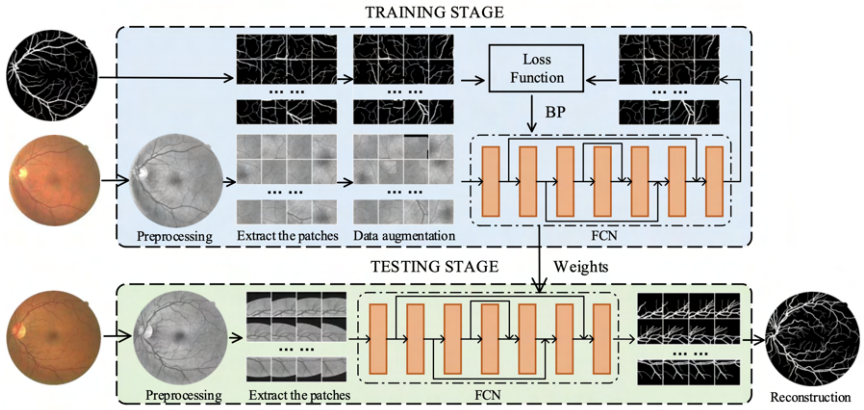


FIGURE 4.5: Overview of the proposed framework. [Source: (Jiang et al., 2019)]

4.2.4 Framework

Due to the random patch extraction process, the amount of data given to the models to train on remained constant throughout the different datasets used. Data of the varying retinal datasets were split into training, validation and testing sets. Utilising a validation set rather than cross-validation was deemed sufficient due the size of the resulting dataset after pre-processing. All models were trained on the pre-processed datasets as described above with the further use of data augmentation. Data augmentation included rotation, as well as horizontal and vertical flipping of the patch input images. In an attempt to improve robustness of the models, noise was injected into the training images. Figure 4.5 describes the proposed framework used in this study.

Chapter 5

Results and Discussion

5.1 Experiment 1: Retinal data sets

[1]The training process of the proposed methods has been conducted using Google Colaboratory (Colab) (Bisong, 2019). Colab is a Google Research hosted Jupyter Notebook product which allows researchers to execute python code with free access to Nvidia K80, T4, P4 and P100 GPUs. Models were built using the `PyTorch` (Paszke et al., 2019) library with image processing helped through the use of `OpenCV` (Bradski, 2000) and `Pillow` (Clark, 2015) libraries.

5.1.1 DRIVE

Initial analysis was done on the DRIVE dataset. Analysis began with the most common type of medical segmentation architectures, the U-Net, following the formulation described in Section 3.3.1 presented by Ronneberger, Fischer, and Brox (2015). This was used as a benchmark to which the other methods are compared. This implementation aimed to gain a thorough understanding of the effect which the hyper-parameters in the model as well as pre-processing would have on its performance. The baseline model was trained on full, unprocessed, colour images which were cropped to 256×256 pixels. The model was trained for 150 epochs which was only feasible due to the small size of the dataset. The model included data augmentation during training.

Figure 5.1 shows the segmentation results of this method. Although the superior vessel branches are segmented correctly from the background and the general structure is present, the result is far from clinically acceptable as most of the inferior

[1]All code is open-sourced and available on https://github.com/DeVriesMatt/Deep-Vessel-Segmentation

vessels are not segmented at all and instead are completely ignored. Section 3.4 discusses how accuracy may be a very misleading metric in this scenario and is clearly shown to be the case when referring to the results in Table 5.2. We see accuracy and specificity greater than those reported by Jin et al., 2019. These results are directly attributed to the fact that if an algorithm, in the case of a highly imbalanced dataset such as this, classifies every pixel as background, then these metrics will be high. Furthermore, specificity is described as the "true negative rate", thus classifying every pixel as background will give a specificity of 1. The Dice coefficient and the sensitivity are alternative and more meaningful metrics to consider here, showing this baseline model's inferiority.

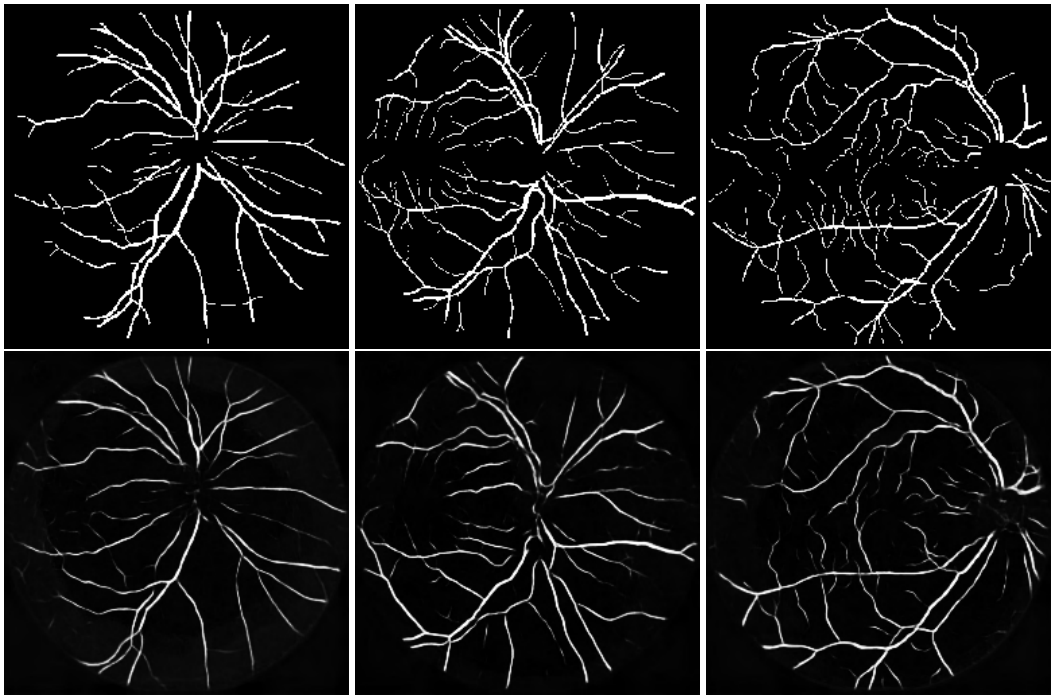


FIGURE 5.1: Segmentation result of the baseline model. The first row shows the expertly annotated ground truth. The second row shows the result from the baseline U-Net model.

Next, the U-Net was then trained on patches of size 48×48 without any further pre-processing. The model was trained for 50 epochs using SGD with Adam optimisation at an initial learning rate of 0.002 with a learning rate schedule where the learning rate decreases by a factor of 10 every 10 epochs. This hyperparameter was chosen through thorough trial and error. The number of epochs chosen seemed sufficient due to convergence of the validation accuracy as well as the limited resources available. As per Kingma and Ba (2014), β_1, the exponential decay rate for the first

moment estimates was chosen to be 0.9 with β_2, the exponential decay rate for the second-moment estimates set at 0.999. Initially, the model did not include dropout and the output was evaluated using a BCE loss function.

In an attempt to increase the performance further, all pre-processing techniques described in Chapter 4.2 were used. That is, 50 000 patch of size 48×48 were extracted from the raw images. These images were passed through CLAHE with only the green channel used. This single-channel image was then normalised and subject to transformations such as horizontal and vertical flipping as well as rotation. Multiple different loss functions were tested thoroughly on each dataset, including all those described in A.1.3. Utilising the Combo Loss with $\alpha = 0.33$ proved most beneficial in this scenario.

Figure 5.2 shows the training and validation accuracies on losses of the different models. We see that although the models may still be learning from the training data, it is not beneficial to continue training further as the validation accuracy has stopped increasing, in fact, the models begins over-fitting after about 25 epochs which is shown by and increase in the validation loss. These plots show superior performance of the models which incorporate attention on the training data. When considering the validation data, an attention-incorporated U-Net clearly outperforms the standard U-Net in terms of patch accuracy. With regards to the iterative models, the attention-incorporated Iternet model and the standard Iternet model seem to perform very similarly. Table 5.1 shows just how similar the results are on the training dataset, however we do see that the iterative models slightly outperform the base models with the Attention Iternet outperforming the rest in most metrics except for the Jaccard similarity.

When analysing the test results in Table 5.2 we see that the models outperform those reported in terms of accuracy. The bold text represents the highest score for that metric across the board, whilst the red coloured text represents the highest score for that metric of those models tested in this study. We see that the Attention Iternet outperforms the other models in terms of sensitivity, specificity, Dice coefficient, and Jaccard similarity. The Iternet is superior, however, in accuracy.

Figure 5.3 shows the segmentation results from the different models. It is clear that the models perform extremely similar, with slight differences only seen on very close inspection. It is evident that the iterative models output a more connected vessel map, with those models which incorporate attention able to segment the finer vessels.

Table 5.3 shows the time taken to train the different models on a dataset of 50 000 patches of size 48×48 using the Nvidia Tesla P100 GPU provided by Google Colab Pro. It is clear that adding attention gates to the models increases computational time. This time is almost doubled for the U-Net, whereas for the Iternet model, adding attention to the base model only increases the training time by roughly 15%.

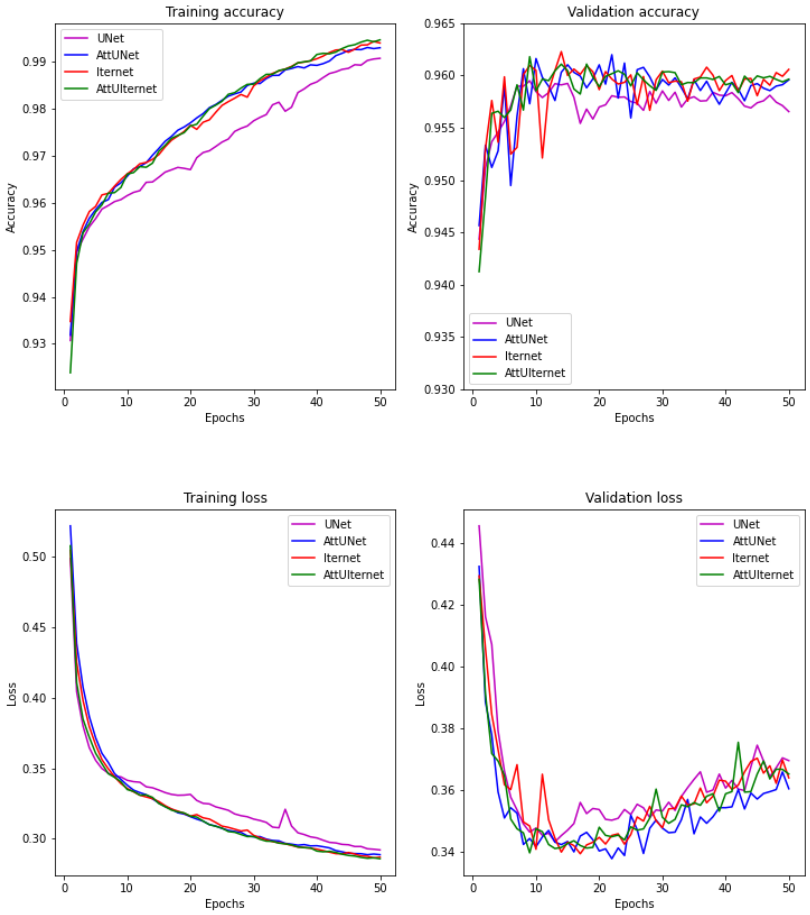


FIGURE 5.2: Training and validation accuracies and losses on the DRIVE dataset of U-Net, Attention U-Net, Iternet and Attention Iternet

Dataset	Method	Year	Acc	SE	SP	DC	JC
DRIVE	U-Net	2020	0.991890	0.932161	0.997930	0.954688	0.913328
	Att. U-Net	2020	0.993851	0.955710	0.997690	0.966063	0.934385
	Iternet	2020	0.995570	0.962162	0.998285	0.975571	**0.952560**
	Att. Iternet	2020	**0.995580**	**0.968794**	**0.998958**	**0.975694**	0.952329

TABLE 5.1: Results on the training set of the DRIVE dataset.

Dataset	Method	Year	Acc	SE	SP	DC	JC
DRIVE	U-Net Reported	2018	0.9555	0.7822	0.9808	0.8174	
	U-Net Baseline	2020	0.9567	0.4405	**0.9934**	0.5716	
	U-Net Final	2020	0.957435	0.724005	0.980274	0.750485	0.600678
	Att. U-Net	2020	0.959394	0.727828	0.982068	0.760281	0.613386
	Iternet Reported	2018	0.9573	**0.7735**	0.9838	**0.8205**	
	Iternet	2020	**0.960288**	0.731314	0.980032	0.761219	0.614543
	Att. Iternet	2020	0.959426	0.758845	0.981762	0.771606	0.628204

TABLE 5.2: Results on the test set of the DRIVE dataset.

Model	Time (s)
U-Net	4131
Attention U-Net	7169
Iternet	9133
Attention Iternet	10473

TABLE 5.3: Time taken to train the models for 50 epochs using Nvidia Tesla-P100 GPU.

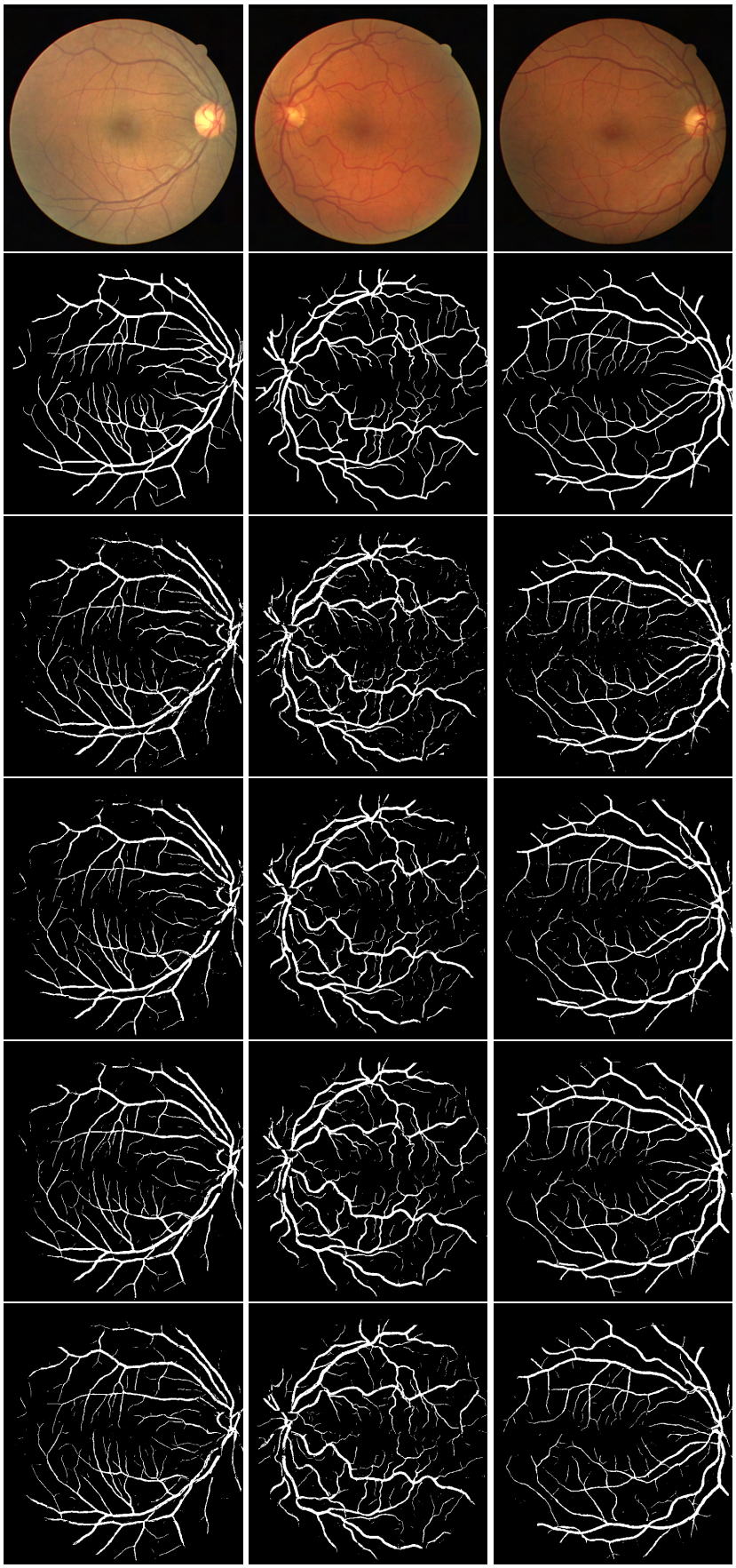


FIGURE 5.3: Result on the test images of the DRIVE dataset. The first row shows the raw images. The second row shows the expertly annotated ground truth. The third, forth, fifth and sixth rows show the result from the U-Net, Attention U-Net, Iternet and Attention Iternet models respectively.

5.1.2 STARE

Experiments were then conducted on the STARE dataset. Here, the only difference in model configuration was the choice of the loss function. The same patch extraction process was followed. It was seen that using a BCE loss function achieved superior performance in this scenario. All models perform well and extremely similar on the images presented with slight differences only seen after close inspection. The training and validation accuracies are shown in Figure 5.4 where it is clearly seen that the Iternet and U-Net seem to perform better on the training data than when these models incorporate attention. The Iternet is seen to perform best on the validation data here too; however, the U-Net is outperformed by those models including attention. Figure 5.5 shows a comparison between the different models tested.

An interesting result is seen when we take a closer look into the differences in the output of the models, as shown in Figure 5.6. Here, it is clear (even though the segmentation results may be substandard due to the large variation of vessel sizes in the image) that those models which include attention are able to segment more of the inferior vessels. The problem, however is that these inferior vessels are too pronounced in the segmented output covering more pixels than necessary (misclassifying background as vessel in pixels neighbouring the vessel). This may cancel out the increased accuracy metric of actually detecting these inferior vessels. This further suggests how the accuracy metric may be substandard in the situation of vessel segmentation. In fact, when we analyse the results in Table 5.4, it may be suggested that most of these metrics may be misleading except for sensitivity in cases where there are blood vessels of varying shapes and sizes. This may suggest that sensitivity is the most important metric when analysing blood vessel segmentation methods as it may be argued that overstating the thickness of the segmented vessel may be more beneficial than not finding the vessel at all. We see how the sensitivity increases when we add attention to the models. Furthermore, it is evident that the addition of the refinery modules in the iterative models increases performance through connecting vessel structures.

Dataset	Method	Year	Acc	SE	SP	DC	JC
STARE	U-Net Reported	2018	0.9752	**0.7840**	0.9880	0.7993	
	U-Net Re-Implementation	2020	0.955327	0.731733	0.981365	0.792193	0.659906
	Att. U-Net	2020	0.952018	0.762952	0.976774	0.781385	0.646088
	Iternet Reported	2019	**0.9782**	0.7715	**0.9919**	**0.8146**	
	Iternet Re-implementation	2020	0.957177	0.759659	0.987322	0.794565	0.662337
	Att. Iternet	2020	0.955917	0.767518	0.981658	0.794308	0.663228

TABLE 5.4: Results on the test set of the STARE database.

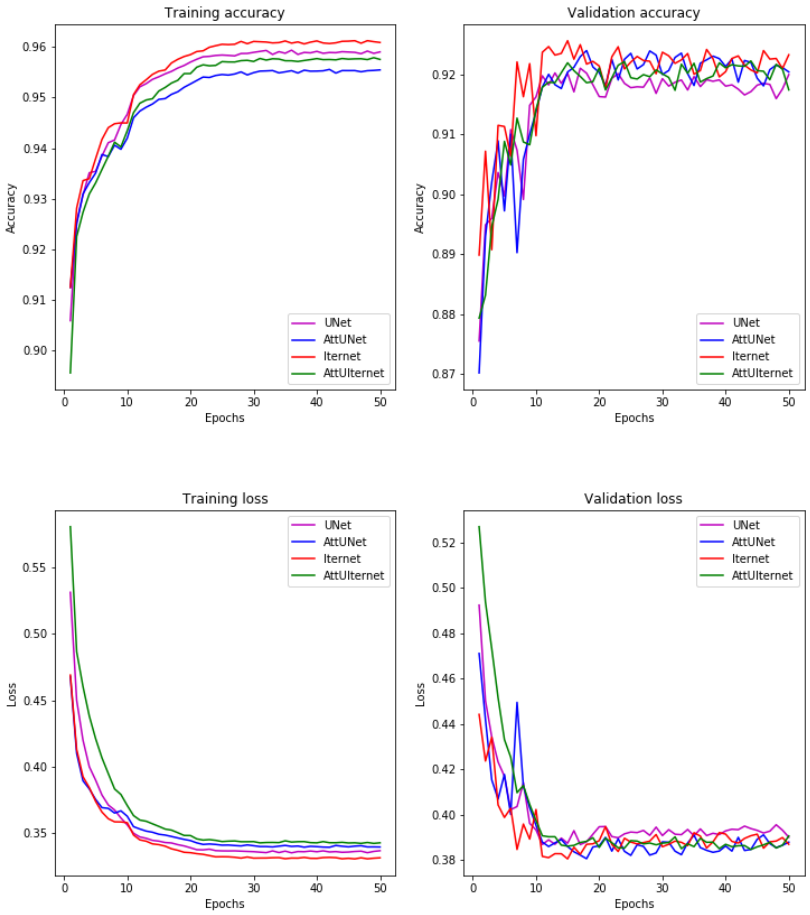


FIGURE 5.4: Training and validation accuracies and losses on the STARE dataset of U-Net, Attention U-Net, Iternet and Attention Iternet

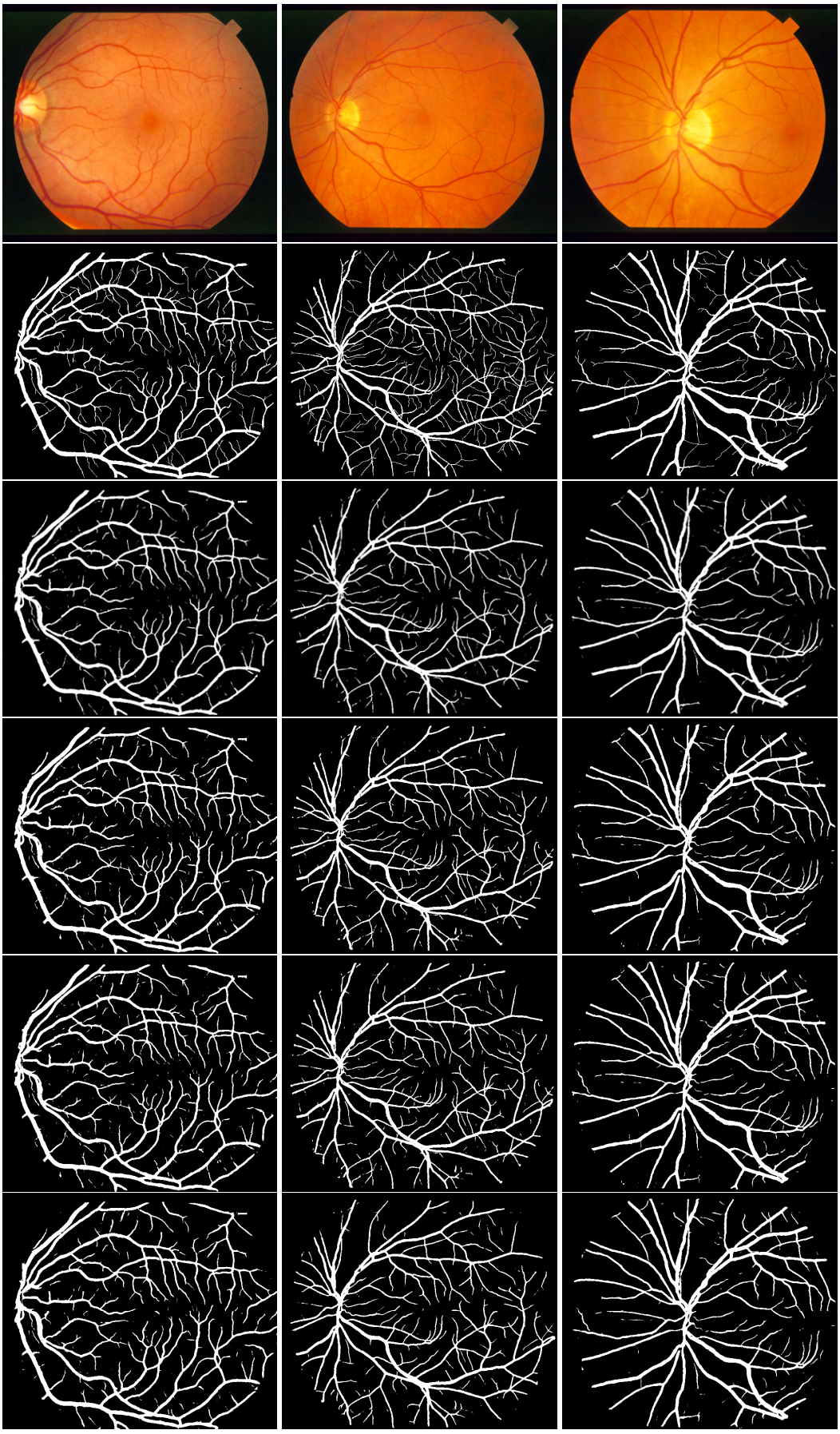


FIGURE 5.5: Result on the unseen images of the STARE dataset. The first row shows the raw images. The second row shows the expertly annotated ground truth. The third, forth, fifth and sixth rows show the result from the U-Net, Attention U-Net, Iternet and Attention Iternet models respectively.

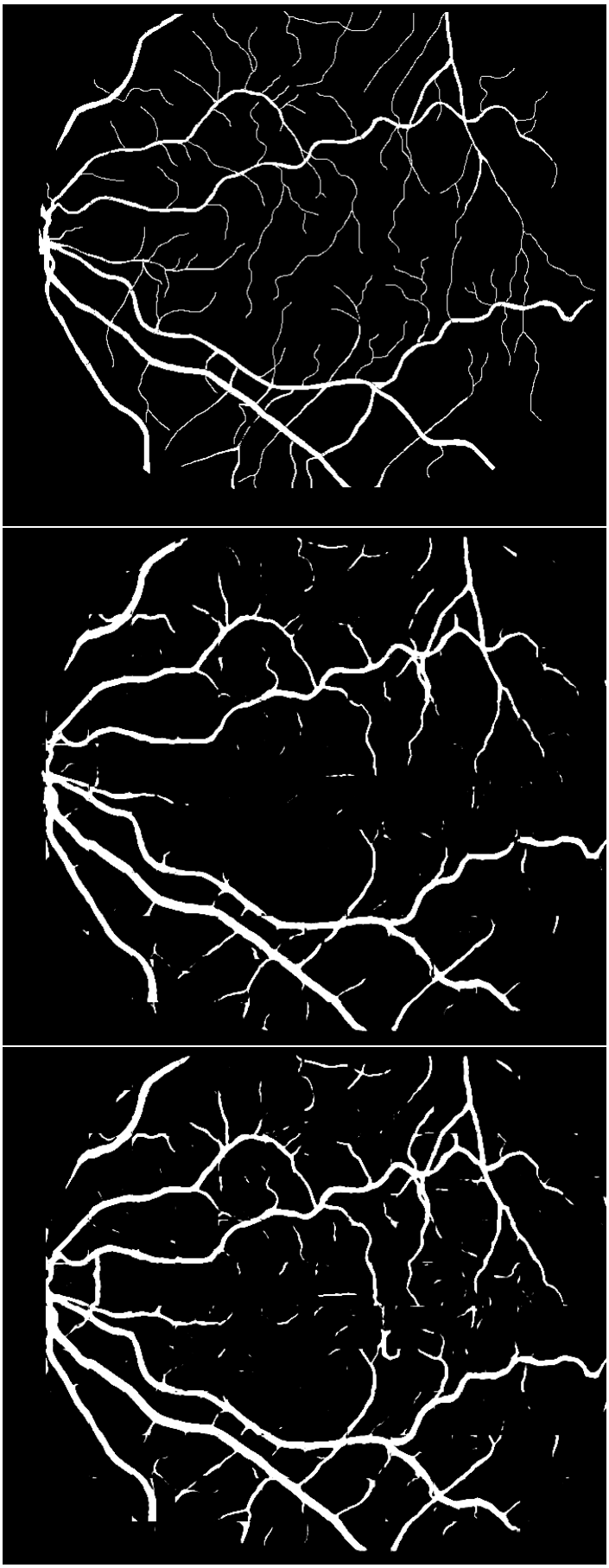


FIGURE 5.6: A closer look at the effect of using attention gates in the Iternet model on an image in the STARE database. The first row shows the ground truth. The second shows the segmentation result of the Iternet and the third shows the segmentation result of the Attention Iternet.

5.1.3 CHASE_DB1

The models were then trained and tested on the CHASE_DB1 database. A total of 2500 patches were randomly extracted (using algorithm 1) from each image, allowing the models to be trained, validated and tested on 50 000, 10 000, 10 000 patches respectively of size 48×48 pixels. There was negligent difference in performance between loss functions on this dataset, thus BCE was used in the final models due to simplicity. As seen on the STARE dataset, the iterative models outperform the others in terms of training and validation accuracy (Figure 5.7). Validation accuracy increases when incorporating attention into the U-Net model; however, this is not the case for the Iternet model. Tables 5.5 and 5.10 show how, again, those models without attention are superior in terms of accuracy, specificity, Dice coefficient, and Jaccard similarity. However, the models which incorporate attention are superior in sensitivity. Figure 5.8 shows a close comparison of the output of the different models. Again, when looking closely at those thinner vessels, it is seen that those models with attention are able to segment the smaller vessels better than those without. One downfall of those models which incorporate attention is the presence of noise in the segmentation result. This could be a major cause for the slightly inferior metrics of these models.

Although slight differences in the performance, the models produced very similar results on most images. Figures 5.9 and 5.10 show the segmentation outputs on the training and test sets respectively. This dataset consists of higher resolution images than the previous two, which, when utilising the same size image patches, seems to cause additional noise in the segmentation results. It may be beneficial to explore different sized image patches on these larger resolution images.

Dataset	Method	Year	Acc	SE	SP	DC	JC
CHASE_DB1	U-Net Reported	2018	0.9752	0.7822	0.9808	0.7993	
	U-Net Re-implementation	2020	0.980562	0.818712	**0.993557**	0.861442	0.756986
	Att. U-Net	2020	0.980388	0.819101	0.993392	0.860149	0.755065
	Iternet Reported	2019	0.9760	0.7969	0.9881	0.8072	
	Iternet Re-Implementation	2020	**0.981934**	0.840149	0.993389	**0.872992**	**0.774888**
	Att. Iternet	2020	0.981170	**0.844432**	0.992180	0.868466	0.767938

TABLE 5.5: Results on the training set of the CHASE_DB1 database.

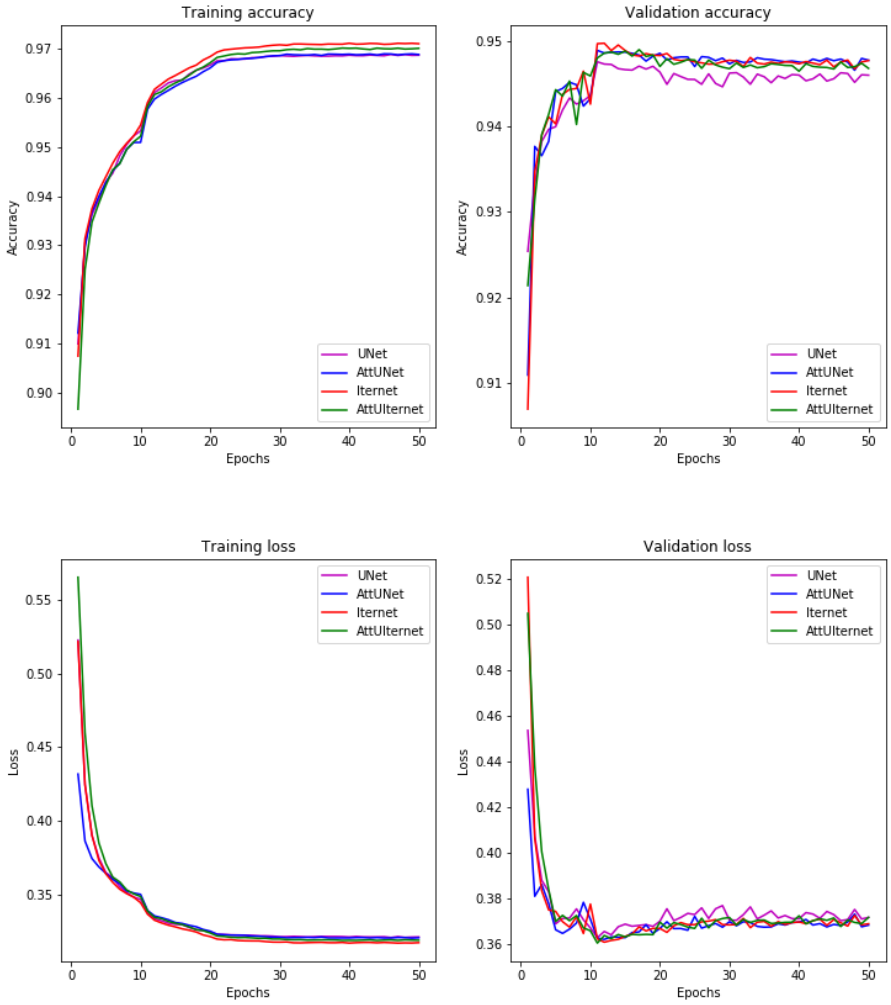
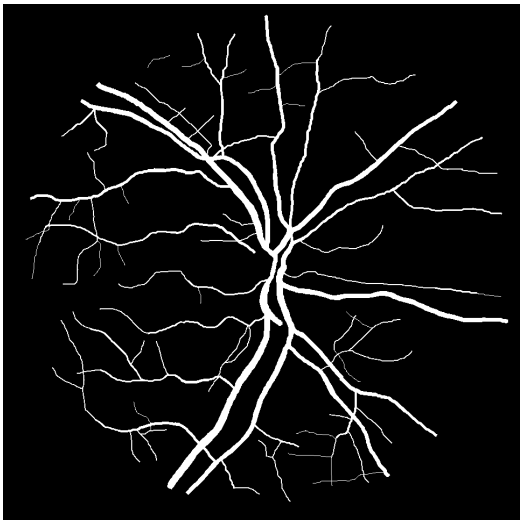


FIGURE 5.7: Training and validation accuracies and losses on the CHASE dataset of U-Net, Attention U-Net, Iternet and Attention Iternet

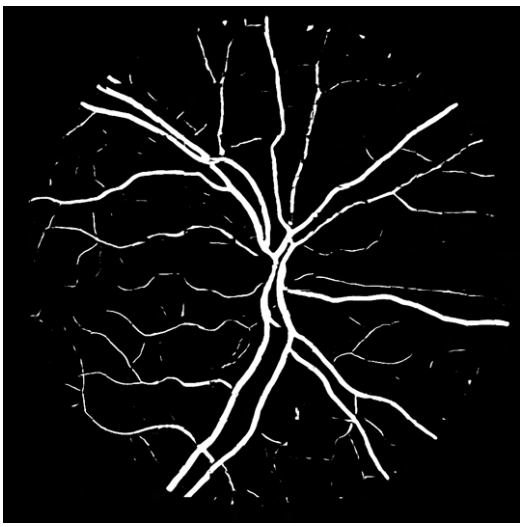
Dataset	Method	Year	Acc	SE	SP	DC	JC
CHASE_DB1	U-Net Reported	2018	0.9752	0.7822	0.9808	0.7993	
	U-Net Re-Implementation	2020	0.967826	0.690193	0.987264	0.735615	0.583260
	Att. U-Net	2020	0.966475	0.675250	0.986885	0.723770	0.568205
	Iternet Reported	2019	**0.9760**	**0.7969**	0.9881	**0.8072**	
	Iternet Re-implementation	2020	0.968264	0.641106	0.985443	0.747580	**0.597994**
	Att. Iternet	2020	0.966130	0.722915	**0.988878**	0.710534	0.552640

TABLE 5.6: Results on the test set of the CHASE_DB1 database.

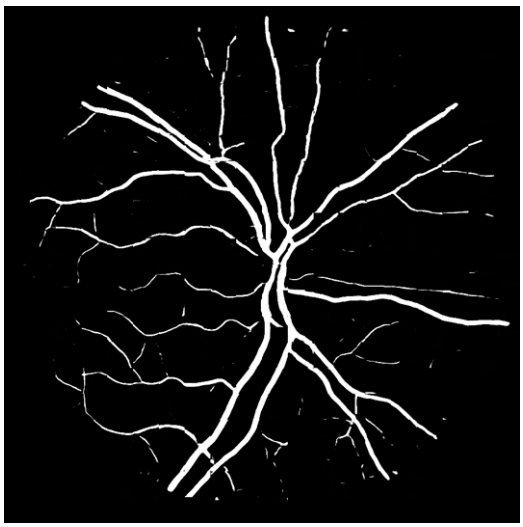


(A) Ground truth.

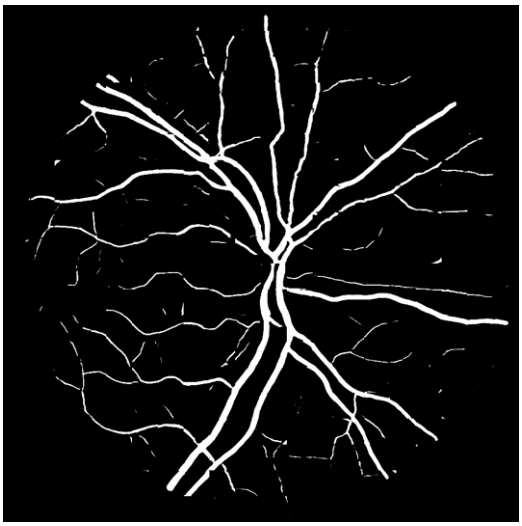
(B) U-Net.



(C) Attention U-Net.



(D) Iternet.



(E) Attention Iternet.

FIGURE 5.8: Comparison of the different models on an image in the STARE database. 5.8a shows the ground truth. 5.8b, 5.8c, 5.8d, 5.8e show the results from the U-Net, Attention U-Net, Iternet and Attention Iternet respectively.

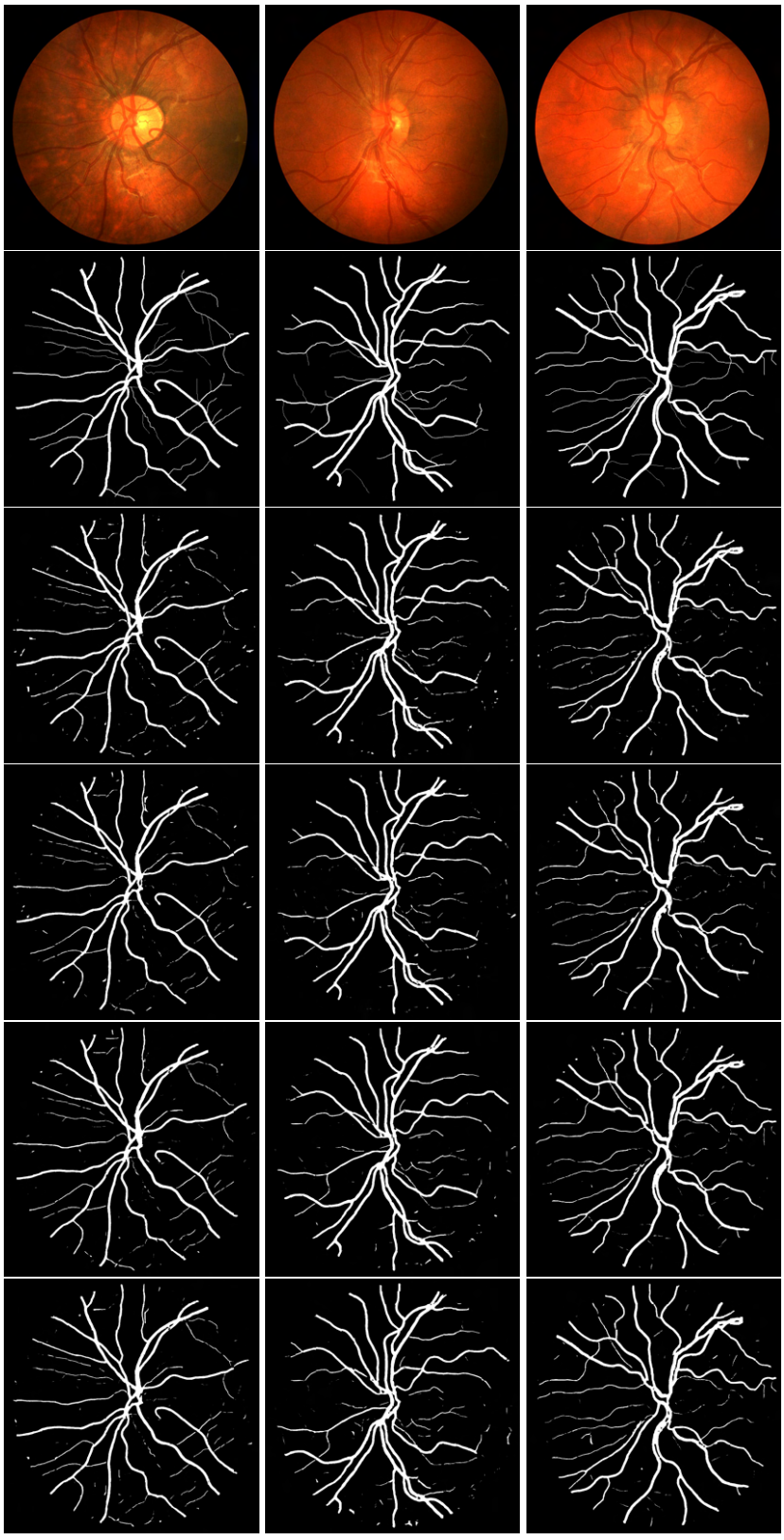


FIGURE 5.9: Comparison of the different models on the training set of the CHASE database. The first column shows the ground truth. The second, third, forth and fifth columns show the results from the U-Net, Attention U-Net, Iternet and Attention Iternet respectively.

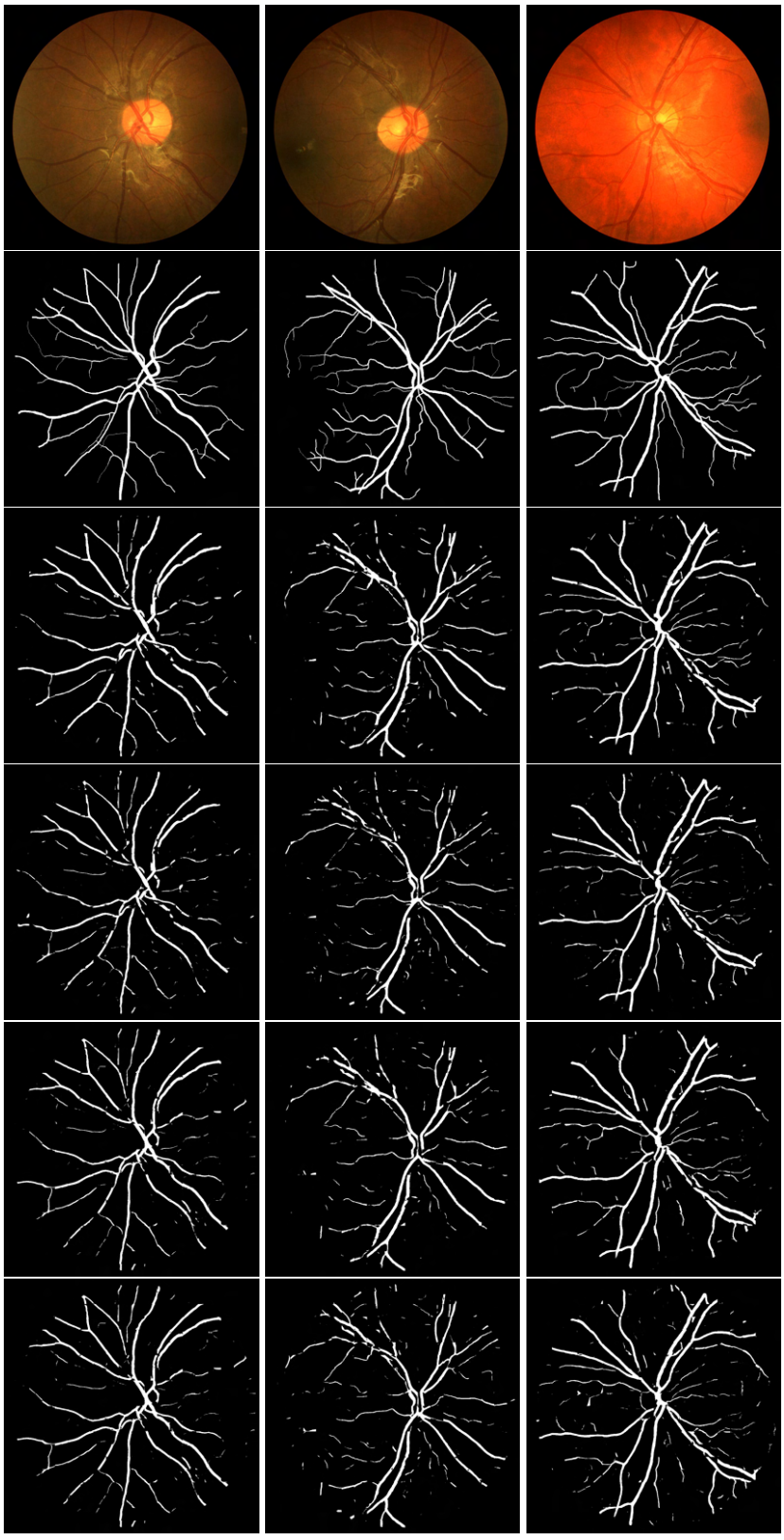


FIGURE 5.10: Comparison of the different models on the test set of the CHASE database. The first column shows the ground truth. The second, third, forth and fifth columns show the results from the U-Net, Attention U-Net, Iternet and Attention Iternet respectively.

5.1.4 HRF

This dataset is of the highest resolution amongst all those examined. For the purpose of consistency, 70 000 patches of size 48×48 were extracted from the images. 50 000 were used for training with 10 000 for validation and the remaining 10 000 for testing. Here, surprising results were seen with the the training and validation accuracies in Figure 5.11 and the other metrics in Tables 5.7 and 5.8. Firstly, it was seen that the iterative models were outperformed by the base models. Secondly, adding attention gates to the U-Net model would improve its accuracy on both the training and validation sets; however, this was not the case with the Iternet. We do, however, continue to see the trend where incorporating attention into the models will increase the sensitivity. Furthermore, when incorporating attention into the U-Net model, we notice an increase in all the reported metrics, differing from what was seen on the other datasets. This was not seen to be the case for the Iternet model, as adding attention gates to this model only proved to increase the sensitivity, Dice coefficient, and Jaccard similarity, with a decrease in accuracy and specificity.

Although apparent, these differences are negligent as seen by Figures 5.12 and 5.13, with slight differences only seen upon extremely close inspection. One of the more prevailing issues in this dataset as opposed to the previous ones, is the presence of noise in the segmentation results.

In order to gain a slightly broader understanding of the models and in an attempt to conduct further analysis, the models were then trained on patches of size 128×128. Tables 5.7 and 5.8 show the increase in performance on both the training and testing when using these larger patches. Furthermore, it is evident from Figures 5.14 and 5.15 that the issue of noise is significantly reduced.

Dataset	Method	Year	Acc	SE	SP	DC	JC
HRF 48	U-Net	2020	0.985494	0.905055	0.991276	0.903312	0.824274
	Att. U-Net	2020	0.986608	0.913055	0.993124	0.909377	0.834350
	Iternet	2020	0.985285	0.885548	0.993231	0.898778	0.816847
	Att. Iternet	2020	0.985173	0.897938	0.992138	0.900002	0.818783
HRF 128	U-Net	2020	0.990042	0.920963	**0.997730**	0.933619	0.875805
	Att. U-Net	2020	**0.991874**	**0.929672**	0.994876	**0.944927**	**0.895755**
	Iternet	2020	0.987921	0.888720	0.996060	0.917391	0.847761
	Att. Iternet	2020	0.988846	0.899929	0.996115	0.923970	0.859039

TABLE 5.7: Results on the training set of the HRF database using different patch extraction processes.

Dataset	Method	Year	Acc	SE	SP	DC	JC
HRF 48	U-Net	2020	0.957509	0.695670	0.979289	0.714657	0.557389
	Att. U-Net	2020	0.959255	0.684086	0.982153	0.719738	0.563431
	Iternet	2020	0.960547	0.693395	0.982808	0.728765	0.574571
	Att. Iternet	2020	0.957711	0.686919	0.980201	0.712808	0.555226
HRF 128	U-Net	2020	0.965725	0.726590	**0.985912**	0.766519	0.622853
	Att. U-Net	2020	**0.966051**	**0.748756**	0.984328	**0.773324**	**0.631837**
	Iternet	2020	0.964026	0.718325	0.984792	0.755938	0.608879
	Att. Iternet	2020	0.963378	0.728772	0.983151	0.755028	0.607995

TABLE 5.8: Results on the test set of the HRF database.

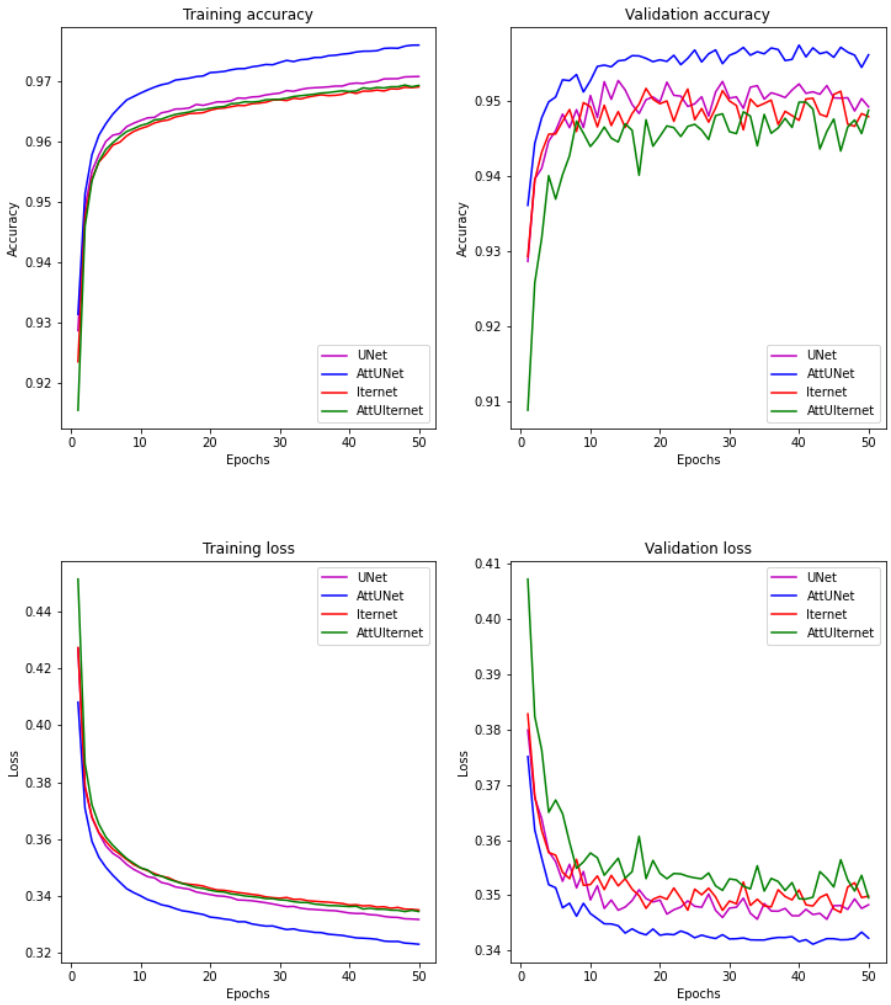


FIGURE 5.11: Training and validation accuracies and losses on the HRF dataset of U-Net, Attention U-Net, Iternet and Attention Iternet

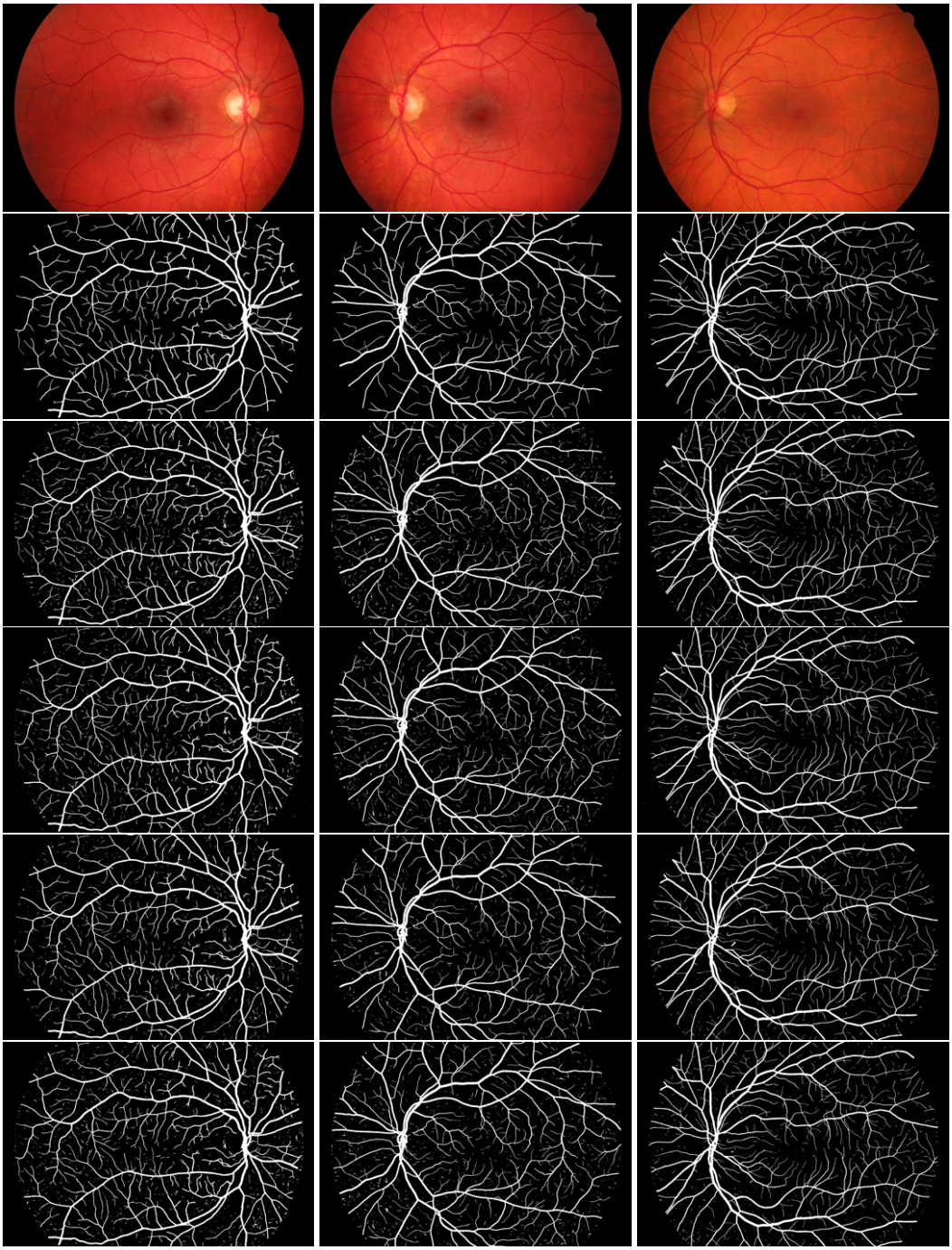


FIGURE 5.12: Comparison of the different models on an image in the training set of the HRF database when using patches of size 48×48. The first column shows the ground truth. The second, third, forth and fifth columns show the results from the U-Net, Attention U-Net, Iternet and Attention Iternet respectively.

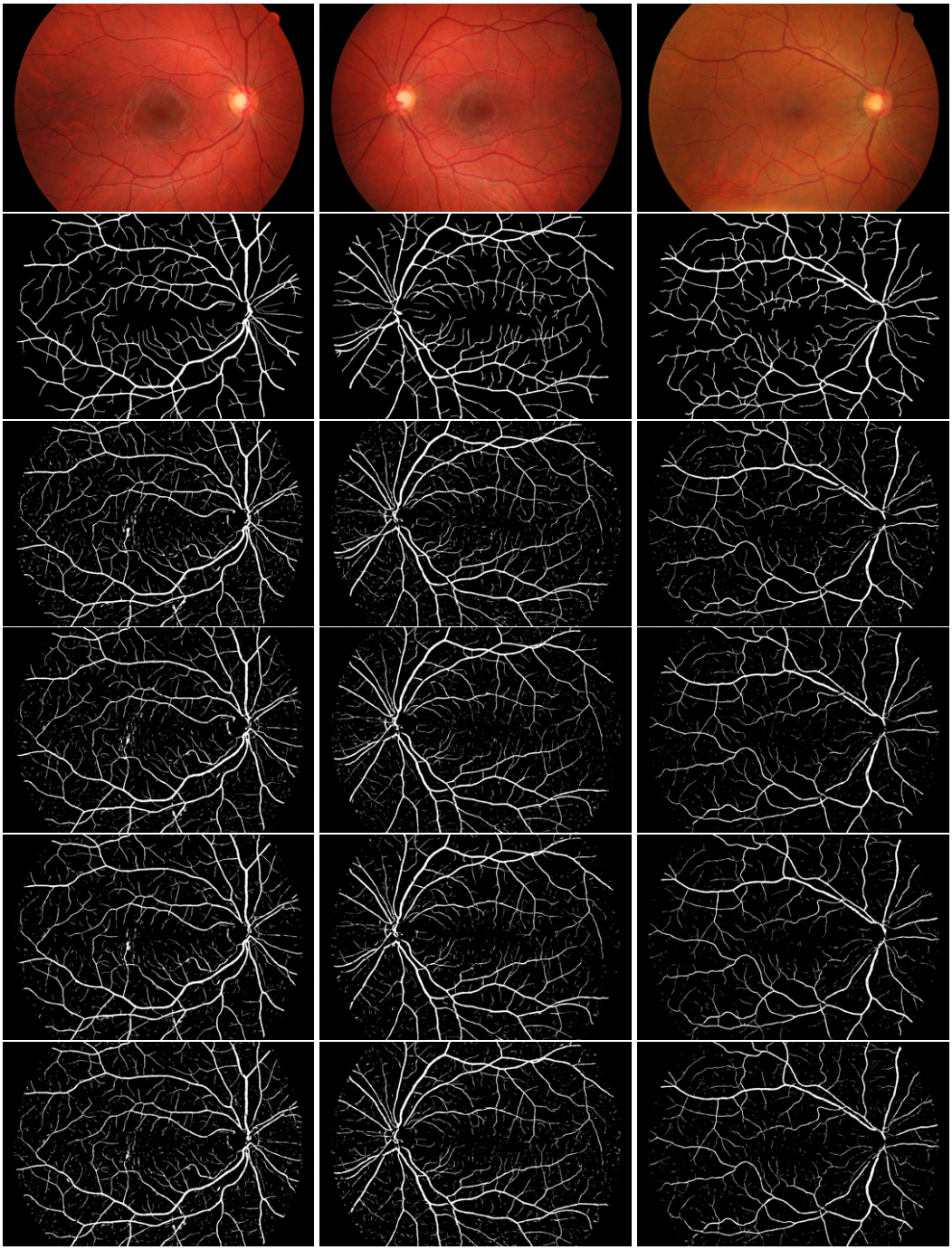


FIGURE 5.13: Comparison of the different models on an image in the test set of the HRF database using patches of size 48×48. The first column shows the ground truth. The second, third, forth and fifth columns show the results from the U-Net, Attention U-Net, Iternet and Attention Iternet respectively.

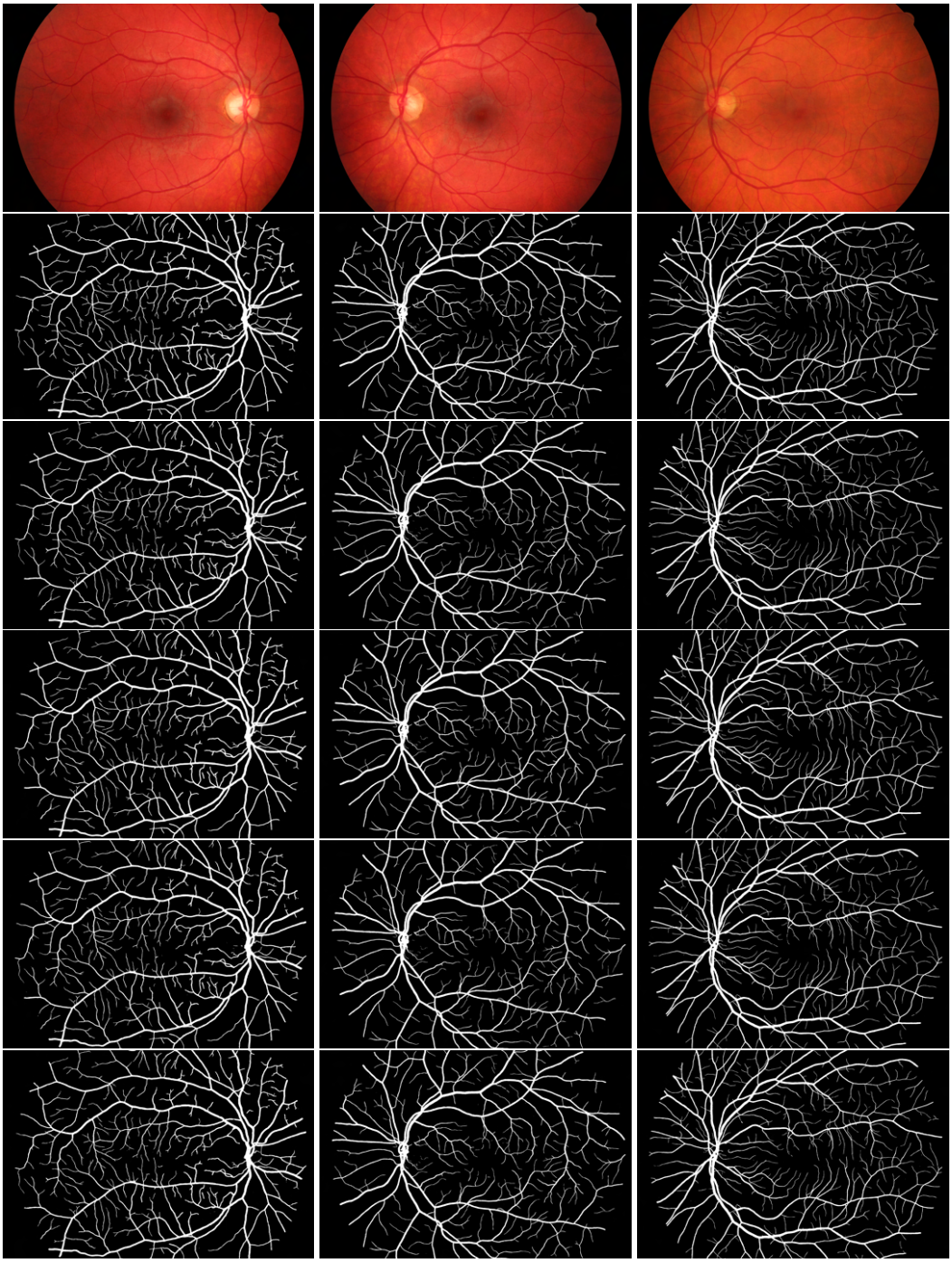


FIGURE 5.14: Comparison of the different models on an image in the training set of the HRF database when using patches of size 128×128. The first column shows the ground truth. The second, third, forth and fifth columns show the results from the U-Net, Attention U-Net, Iternet and Attention Iternet respectively.

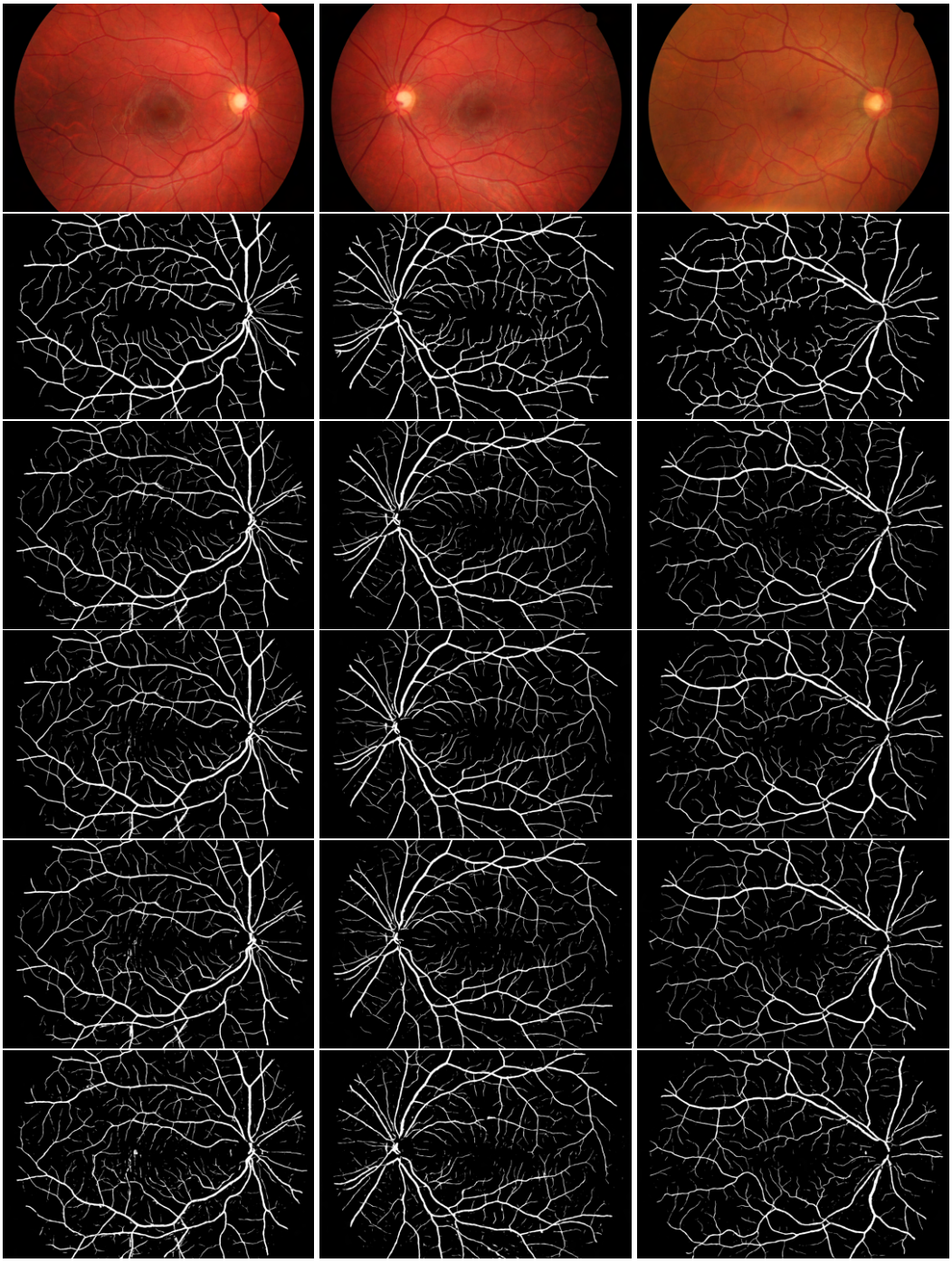


FIGURE 5.15: Comparison of the different models on an image in the test set of the HRF database using patches of size 128×128 for training. The first column shows the ground truth. The second, third, forth and fifth columns show the results from the U-Net, Attention U-Net, Iternet and Attention Iternet respectively.

Chapter 6

Summary and Conclusion

The segmentation of blood vessels from medical images is a critical issue in numerous clinical fields with research continuing throughout the 21st century and new state-of-the-art occurring year-on-year. These state-of-the-art techniques have recently been a variant of the groundbreaking U-Net model and have mostly been analysed on retinal images due to the availability of these databases. The current state-of-the-art is the Iternet model, which utilises an iterative process, mimicking that of a human expert.

The invention of AG has had a significant impact on deep learning technologies in tasks of NLP and computer vision with a particular interest in semantic segmentation (Zhang et al., 2020; Tao, Sapra, and Catanzaro, 2020). These are proposed to increase model performance without significant increase in computational overhead. Furthermore, the addition of AG enables models to learn to focus on regions of interest which may vary in shapes and sizes, which could be extremely beneficial in the scenario of vessel segmentation.

This project attempted to explore vessel segmentation techniques and loss function as well as improving on the current state-of-the-art. To this end, thorough experimentation was conducted on the U-Net and Iternet models as well as these models with incorporated attention gates to focus on salient features. A primary discovery was seen in the effect that pre-processing steps would have on the results of these models. Utilising CLAHE on raw image inputs, only taking the green channel and adopting a patch-extraction procedure proved to significantly increase baseline results.

In most of the retinal datasets, the iterative models outperformed those base models. Adding attention to the U-Net and Iternet models would then prove to increase performance in terms of sensitivity. It was also seen that while the models without attention gates might have produced some better performance metrics, that these may be misleading as the methods with attention gates over-exaggerate inferior vessels while those methods without attention would not segment these vessels at all. Interesting results were then seen on the higher resolution database where the base models outperformed the iterative models with attention making improvements on the U-Net but not the Iternet model. Furthermore, high levels of noise were presented in the segmentation results of the higher resolution dataset. Adopting a higher resolution patch-extraction procedure proved to mitigate the effect of noise.

Furthermore, we attempted to transfer the knowledge learned from these databases and apply the models on other tubular structure datasets. This task consisted of the segmentation of vessels from chloroplast in Bienertia chlorenchyma cells (data taken from Mai et al. (2019) and shown in Figure 6.1). Poor results were seen and the model was not able to segment any meaningful information. This may be due to the highly different vessel structures, shapes and sizes. Future work to improve these results could focus on training models on more similar datasets. The lack of ground truth for the chloroplast dataset makes it hard to train models on this, however, semi-supervised learning techniques could be explored using few manually segmented images as a baseline ground truth. Work will be continued on this with attempts to publish a review, current results as well as a transfer learning approach to vessel segmentation.

Ultimately, the task of vessel segmentation proved to be difficult with the need for a number of pre-processing steps. Attention was shown to increase model performance in most cases however the computational increase varied for the different models with this being much more significant for the U-Net model than the Iternet model. Attempts to transfer the knowledge learned on retinal datasets to chloroplast dataset failed, however, there may be prospects for future work on this topic and transferring knowledge to other similar datasets.

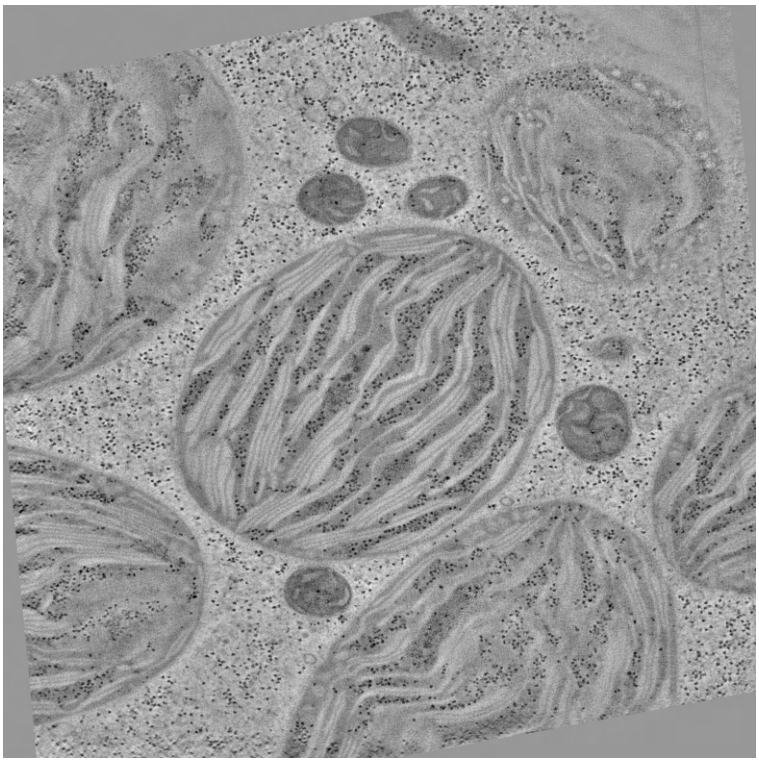


FIGURE 6.1: Electron tomography analyses of chloroplast in Bienertia chlorenchyma cells. [Source: (Mai et al., 2019)]

Deep learning is often criticised for their "black-box-ness" (Kratzert et al., 2018). This criticism is justified as the question of how and why a particular model works or not is very important. Looking within the model and behind the scenes is what makes science exciting. For this reason, it is suggested that further research could be done looking at how the models determine vessel from background. A proposed benefit of the incorporation of attention mechanisms into common architectures is its explainability. Analysing the attention coefficients could explain more about what the models are learning. Future work on deep learning-based vessel segmentation techniques could include an Iternet where the refinery modules also include attention. Furthermore, including the raw input images to the input of the refinery modules in the Iternet may be an intuitive approach which could lead to better results. Incorporating a patch-extraction procedure directly into models could be investigated as this is easily seen to improve performance drastically. A focus on a model which could be extended to multiple datasets and imaging communities would be groundbreaking and should be a topic of major interest in years to come.

Appendix A

Pre-requisites

A.1 A brief introduction to neural networks

Inspired by the biological neural networks of the brain, artificial neural networks (ANNs) are one of the most powerful tools in the field of artificial intelligence. A neural network is a system of interconnected neurons which send signals to one another (Tchircoff, 2017). The strengths of these connections between the neurons, also known as weights, determine the network's behaviour. More importantly, these weights can be systematically tuned to make the neural network behave in a highly specific, desirable way. The ANN occurs in organised layers, some of which are termed hidden layers. The number of hidden layers is controlled by the researcher and subject to the complexity required. The complexity of the model increases as the number of hidden nodes (neurons) increases. Cybenko (1989) proved that an ANN consisting of a finite-sized single hidden layer is able "to approximate any continuous function to any desired precision". Although several authors have given certain empirical equations to approximately estimate the number of neurons in the hidden layer (Tamura and Tateishi, 1997; Molga, 2003; Pendharkar and Rodger, 2003; Hunter et al., 2012), it is preferred to adopt trial and error approach for deciding the optimal number of neurons in the hidden layer.

A.1.1 Structure and notation

[1]Consider modelling a dataset consisting observations of p inputs $\boldsymbol{x}^T{}_i = [x_{i1}, x_{i2}, \ldots, x_{ip}]$ and q outputs $\boldsymbol{y}^T{}_i = [y_{i1}, y_{i2}, \ldots, y_{iq}]$ for $i = 1, 2, \ldots, N$ (a total of N training examples). Let $\boldsymbol{a}_j^l$ denote the j^{th} node on the l^{th} layer of a standard feed-forward neural network. The network structure can then be written as an updating equation:

[1]Largely taken from (Pienaar, 2018)

$$\boldsymbol{a}_j^l = \sigma_l\Big(\sum_{k=1}^{d_l-1} \boldsymbol{a}_k^{l-1}\boldsymbol{w}_{kj}^l + \boldsymbol{b}_j^l\Big) \quad l = 1,\ldots,J; \quad j = 1,\ldots,d_l, J-1. \tag{A.1}$$

Where:

- $\sigma_l(.)$ denotes an activation function on layer l. Addressed in Section A.1.2.
- d_{l-1} denotes the number of nodes in layer $l-1$.
- $\boldsymbol{w}_{kj}^l$ denotes the kj^{th} weight parameter linking the k^{th} node in layer $l-1$ and j^{th} node in layer l.
- $\boldsymbol{b}_j^l$ denotes the j^{th} bias in layer l.
- and the equation is evaluated subject to the initial conditions $\boldsymbol{a}_j^{(0)} = \boldsymbol{x}_{ij}$ for all j at the i^{th} training example.

A.1.2 Activation functions

Activation functions play an integral role in the neural network as they *decide* whether the information from a specific node is important or not. These functions are what ultimately differentiate neural networks from linear or logistic regression. A wide range of activation functions exist, each with their respective advantages and drawbacks, however, presented here are only those used in this research[2].

Sigmoid function

The sigmoid is one of the most commonly used activation functions among all applications of deep learning applications. It is a nonlinear function denoted as the following:

$$f(x) = \frac{1}{1+e^{-x}} = \frac{e^x}{e^x+1} \qquad \text{for } -\infty \leq x \leq \infty.$$

Other forms of the function include:

$$f(x) = \frac{1}{1+e^{-\alpha x}} \quad \text{and} \quad f(x) = \frac{1}{1+e^{-g(x-b)}}, \tag{A.2}$$

[2]For a full overview with implementation, please see Gupta (2020) blog.

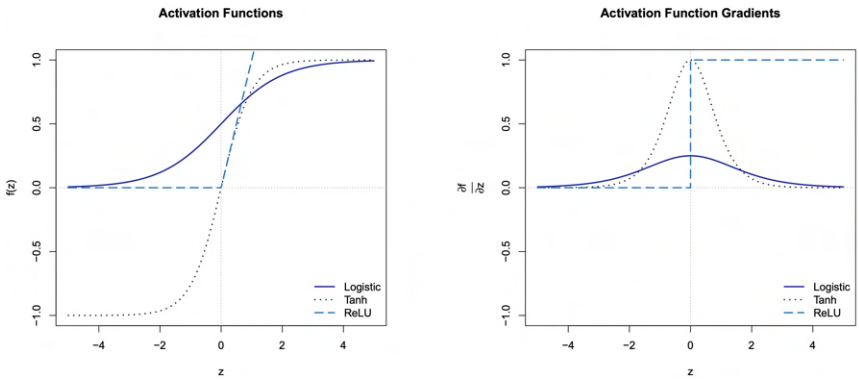


FIGURE A.1: Activation Functions. [Source: Pienaar (2018)]

where α, g, and b are the learning rate, gain and bias respectively.

Figure A.1 above graphically represents the sigmoid activation function and its derivative. One of the issues regarding the sigmoid functions is that the gradient approaches zero as X becomes more extreme, which means at this point, the network does not do much learning. The Range of the sigmoid function is between $(0, 1)$, which prevents activations from blowing up, but it returns all positive values. This issue can be addressed with the hyperbolic tangent function (**??**).

Rectified linear units

Rectified linear units (ReLU) are a special kind of maxout (Goodfellow et al., 2013) function, simply defined as the positive part of its argument:

$$f(x) = x^+ = max(0, x). \tag{A.3}$$

When looking at the gradient functions for each activation in Figure A.1, it is seen that for sufficiently large inputs, there may be very little change in the value of the activation function. The effects of this will become clear when one implements backpropagation.

Soft-max function

The output layer of a neural network represents the final evaluation of the updating equation, which defines the model. This layer is directly compared to the expected output data via an appropriate cost function and thus is usually chosen in accordance with the task at hand. For classification problems, such as pixel-wise semantic segmentation, it is necessary to choose a function which matches the encoded output

range. In an attempt to ensure the output acts as a distribution, a common approach is to use the soft-max activation function on the output layer. This is defined as:

$$a_j^L = \frac{e^{z_j}}{\sum_{k=1}^{d_L} e^{z_k}}, \tag{A.4}$$

for $j = 1, 2, \ldots, d_L$. The outputs are therefore treated as probabilities with predictions corresponding to that of the highest probability.

A.1.3 Loss functions

In the context of training machine learning models, the aim is to solve a configuration which replicates the data in some sense. We wish to find model parameters which give the most accurate approximation of the relationship between the input and the desired response. Loss functions represent the difference between the output of the model and the desired output. These can take on numerous forms and are highly dependent on the nature of the task.

Loss functions are based on statistical distributions of desired outputs. For example, Binary Cross Entropy is formed on the basis of the Bernoulli distribution with Categorical Cross Entropy on the Multinoulli distribution (Jadon, 2020). For the task of semantic segmentation, Jadon (2020) outlines numerous loss functions which have proven extremely effective in recent years. Here, those familiar and exciting (according to myself) are presented.

Semantic segmentation is evaluated in terms of pixel-wise classification with pixels belonging to either the region of interest of the background. Thus, the problem is essentially that of binary classification. One of the most popular loss functions used in scenarios such as these is that of Binary Cross Entropy. Ma Yi-de, Liu Qing, and Qian Zhi-bai (2004) define Cross Entropy as a difference measure between two distributions, given mathematically as:

$$C_{BCE}(y, \hat{y}) = -\sum_i (y_i log(\hat{y}_i) + (1 - y_i) log(1 - (\hat{y}_i))) \tag{A.5}$$

Variants of this include the Weighted Binary Cross Entropy:

$$C_{W-BCE}(y,\hat{y}) = -\sum_i (\beta y_i \log(\hat{y}_i) + (1-y_i)\log(1-(\hat{y}_i))), \tag{A.6}$$

where the β coefficient is a tuning parameters which trades-off between false negatives and false positives. $\beta > 1$ will decrease false negatives while $\beta < 1$ will decrease false positives. And Balanced Cross Entropy:

$$C_{BalCE}(y,\hat{y}) = -\sum_i (\beta y_i \log(\hat{y}) + (1-\beta)(1-y_i)\log(1-(\hat{y}_i))), \tag{A.7}$$

where $\beta = 1 - \frac{y}{Height \times Width}$.

Developed by Lin et al. (2017), Focal Loss is also a variation of Binary Cross Entropy. The aim of this loss was to steer the learning process to focus on training examples which are deemed harder to classify. Their goal was to address the issue of large data imbalance between regions of interest and background. This may prove highly beneficial in tasks of vessel segmentation where large vessels are easily segmented, with thinner vessels being completely overlooked. The large majority of vessel images are also background thus the problem is highly imbalanced. In order define the focal loss, we must rewrite binary cross entropy as a piece-wise function:

$$\mathrm{C_{BCE}}(p,y) = \begin{cases} -\log(p) & \text{if } y = 1 \\ -\log(1-p) & \text{otherwise} \end{cases}, \tag{A.8}$$

where $p \in [0,1]$ is the models output probability for the class labelled $y = 1$ and $y \in \{\pm 1\}$ is the ground truth. Thus we can write:

$$p_\mathrm{t} = \begin{cases} p & \text{if } y = 1 \\ 1-p & \text{otherwise} \end{cases}, \tag{A.9}$$

such that $\mathrm{C_{BCE}}(p,y) = \mathrm{C_{BCE}}(p_\mathrm{t}) = -\log(p_\mathrm{t})$. Now, we can define the focal loss:

$$\mathrm{C_{FL}}(p_\mathrm{t}) = -(1-p_\mathrm{t})^\gamma \log(p_\mathrm{t}), \tag{A.10}$$

for some tunable focusing parameter $\gamma \geq 0$. Notice how the focal loss is the binary cross entropy loss multiplied by $(1-p_\mathrm{t})^\gamma$, known as the "modulating factor". Lin et al. (2017) described two important aspects of focal loss:

1. Misclassified examples with small p_t values, will cause the modulating factor to be close to 1, unaffecting the loss. Where as p_t approaches 1, this modulating factor approaches 0, effectively down-weighting the loss for well classified examples.

2. The 'focusing parameter', γ, has the ability to adapt the rate at which simpler examples are down-weighted in a smooth manner. This parameter controls the modulating factors effect, where $\gamma = 0$ reduces the loss to binary cross entropy .

Lin et al. (2017) found a γ value of 2 to be superior in their experiments. In practice, it may be beneficial to use a weighted focal loss:

$$C_{FL}\left(p_t\right) = -\alpha_t \left(1 - p_t\right)^{\gamma} \log\left(p_t\right), \tag{A.11}$$

for some α.

Another popular method used in semantic segmentation is known as Dice Loss. The Dice coefficient is used in computer vision problems to determine the equivalence between images (Jadon, 2020). Sudre et al. (2017) adapted this metric in order to be incorporated into a loss function defined as:

$$C_{DL}(y, \hat{p}) = 1 - \frac{2y\hat{p} + 1}{y + \hat{p} + 1}. \tag{A.12}$$

Finally, a combination of different losses may also be used. An example of this technique is a novel loss function known as Combo Loss:

$$C_{CL}(y, \hat{y}) = \alpha C_{BalCE} - (1 - \alpha) C_{DL}(y, \hat{y}). \tag{A.13}$$

A.1.4 Backpropagation

During the CNN training process, the kernels are updated during each epoch in an attempt to minimise the given loss. The foundations of backpropagation were laid in the 1960s (Kelley, 1960) in the context of control theory, yet only coined for its use in machine learning two decades later (Rumelhart, Hinton, and Williams, 1986a). This algorithm (as well as its variants) has proven efficient in gradient-based optimisation of weights in all types of neural networks. The algorithm requires both

a continuous and differentiable loss function as its gradients are computed at each iteration. Backpropagation is not the entire learning algorithm for deep networks, but rather the commonly used method for gradient computation, which some other algorithm uses in order to undertake learning.

Mitchell (1997) defines machine learning as: "A computer program is said to learn from experience E with respect to task T and performance measure P, if its performance at task T, as measured by P, improves with experience E". This is the adaption of weights in which the difference between the network and the desired outputs are minimised (Snuverink, 2017). There are two phases through which the backpropagation algorithm must pass, the forward pass and the backward pass:

1. **Forward pass**: Input (an image in this case) is passed through the network which has been initialised with random weights giving some output.
2. **Backward pass**: Information flows back from the output and calculated cost function in order to compute its gradient in terms of the weights.

The weights are then updated in the direction of the negative gradient through some learning algorithm. Essentially, backpropagation is used to calculate the partial derivative of the loss function with respect to the network weights, $\frac{\partial L}{\partial w}$.

The following derives the backpropagation algorithm for a given cost function C, the error e_l between the true output, y_i, and the output from the network, $\hat{y}_i$. We wish to evaluate the value of the loss function with respect to the weights, w, and biases, w, working backwards from the loss through the network layers. We define the loss:

$$\begin{aligned} C &= \frac{1}{2N}\sum_{i}^{N}(e_i)^2 \\ e_l &= y_i - \hat{y}_i \end{aligned} \qquad \text{(A.14)}$$

,

and the linear component of a layer l:

$$z_j^l = \sum_{k=1}^{d_l-1} a_k^{l-1} w_{kj}^l + b_j^l, \qquad \text{(A.15)}$$

where a_j^l denotes the j^{th} node on the l^{th} layer of a neural network. The working gradient is defined as:

$$\delta_j^l = \frac{\partial C}{\partial z_j^l}. \tag{A.16}$$

We can now derive the general expressions for the required gradients, starting with the terminal condition and propagating the errors backwards:

$$\begin{aligned}
\delta_j^L &= \sum_{k=1}^{d_L=q} \frac{\partial C}{\partial a_k^L} \frac{\partial a_k^L}{\partial z_j^L} \\
&= \frac{\partial C}{\partial a_k^L} \frac{\partial a_k^L}{\partial z_j^L} \\
&= \frac{\partial C}{\partial a_j^L} \sigma\prime(z_j^L),
\end{aligned} \tag{A.17}$$

For some activation function $\sigma(.)$. Iterating backwards from the terminal condition:

$$\begin{aligned}
\delta_j^{l-1} &= \frac{\partial C}{\partial z_j^{l-1}} \\
&= \sum_{k=1}^{d_l} \frac{\partial C}{\partial z_k^l} \frac{\partial z_k^l}{z_j^{l-1}} \\
&= \sum_{k=1}^{d_l} \frac{\partial z_k^l}{z_j^{l-1}} \delta_k^l \\
&= \sum_{k=1}^{d_l} (w_{jk}^l \sigma\prime(z_j^{l-1})) \delta_k^l \\
&= \sum_{k=1}^{d_l} w_{jk}^l \delta_k^l \sigma\prime(z_j^{l-1}).
\end{aligned} \tag{A.18}$$

We can then evaluate the gradient of the cost function with respect to the model parameters. For the biases of the model, we have:

$$\begin{aligned}
\frac{\partial C}{\partial b_j^l} &= \frac{\partial C}{\partial z_k^l} \frac{\partial z_k^l}{\partial b_j^l} \\
&= \delta_j^l.
\end{aligned} \tag{A.19}$$

Next, for the weights:

$$\begin{aligned}\frac{\partial C}{\partial w_{kj}^l} &= \frac{\partial C}{\partial z_j^l}\frac{\partial z_j^l}{\partial w_{kj}^l} \\ &= a_k^{l-1}\delta_j^l.\end{aligned} \tag{A.20}$$

The updating equations for the back-propagation procedure are defined by Equation A.19 and A.20 as well as the terminal condition in Equation A.17.

A.1.5 Optimisation

In order to train neural networks, some variant of Stochastic Gradient Decent (SGD) is used. This algorithm updates weight parameters according to some rule in some way as to minimise a cost function. SGD is an efficient approximation or online version of Gradient Descent (GD) as instead of calculating the gradient through the use of the entire dataset like in GD, SGD makes use of one or a batch of training examples in order to make an update. This is known as mini-batch SGD.

Mini-batch SGD updates the parameters after viewing and calculating the gradient of the cost function for a subset of n training samples. The update rule at iteration i is defined by the previous weights θ_{i-1}, the gradient of the proposed cost function $\nabla_\theta C(\theta)$, and a learning rate η and given by:

$$\theta_i = \theta_{i-1} - \eta \nabla_{\theta_i} C(\theta_i) \tag{A.21}$$

Deciding on an appropriate learning rate is a complex issue faced by all deep learning researchers. Too high a learning rate could result in divergence from an optimum, where one too low would result in slow convergence. It is often the case to choose a dynamic learning rate in accordance with the magnitude of the slope of the cost function. Momentum is a method which aims at assisting in avoiding getting stuck in a local optimum. A proportion, γ, of the update in the preceding iteration is added to the update in the current iteration. The update rule with momentum is defined as:

$$
\begin{aligned}
v_i &= \gamma v_{i-1} + \eta \nabla_{\theta_i} L(\theta i) \\
\theta_i &= \theta_{i-1} - v_i
\end{aligned}
\tag{A.22}
$$

Introduced by Kingma and Ba (2014), the Adaptive Moment Estimation (Adam) optimisation technique is one of the most popular in recent years due to the strong performances achieved with its use and little memory requirement. Adam introduces a new variable into the SGD with momentum algorithm which represents the exponentially decaying average of past gradients. Adam combines advantages of AdaGrad (Duchi, Hazan, and Singer, 2011) and RMSProp (Tieleman and Hinton, 2012) which work well with sparse gradients and in on-line settings respectively. Bias-corrected estimates of the mean $\hat{v}_t$ and variance $\hat{m}_t$ of the gradient are used in the update rule in this technique:

$$
\theta_t = \theta_{t_1} - \frac{\eta}{\sqrt{\hat{v}_t} + \epsilon} \cdot \hat{m}_t. \tag{A.23}
$$

A.2 Convolutional Neural networks

Deep learning-based image segmentation and classification has exploded in recent years following Krizhevsky, Sutskever, and Hinton (2012) winning ImageNet. Originally proposed by LeCun and Bengio (1998), Convolutional Neural Networks (CNNs) have since dominated the field in both medical and commercial settings (Khan et al., 2020). These models are designed in such a way where the data which is processed has a grid-like topology, much like an image which can be seen as a 2D grid of pixels. Any neural network which includes a special kind of linear mathematical operation called a **convolution** is known as a CNN. "Convolutional neural networks are simply neural networks that use convolution in place of general matrix multiplication in at least one of their layers" (Goodfellow, Bengio, and Courville, 2016).

The architecture of CNNs was inspired by the structure of the visual cortex in a human brain where single neurons respond to stimuli in a restricted region only. This region is known as the receptive field. Much like general feed-forward neural networks, CNNs consist of multiple layers of neurons; however, these layers are somewhat different in CNNs. For the purpose of this study, it is important to understand and outline the motivations and operations used in the CNN architectures

such as the convolution and transpose convolution operations, pooling layers and fully-connected layers.

A.2.1 Convolution

The convolution operation can be defined as an operation on two functions of a real-valued argument (Goodfellow, Bengio, and Courville, 2016). This definition is motivated with an example.

Suppose we model a response function, $x(t)$, of some input, t, such that $x(t), t \in \mathbb{R}$. Now, it is justified to assume that instruments used to measure response may be noisy; thus one may wish to obtain a less noisy prediction of the response to an input. A common approach to this would be to take an average across multiple measurements. However, for example, with time-series data, recent measurements may be more relevant in predicting a response; thus it may be beneficial to apply a weighted average operation with more weight given to recent responses. Defining a weighting function $w(a)$, where (in the example of time-series data) a is the age of the response or the time since the response was measured. In order to obtain a de-noised or smoothed estimate, $s(t)$, of the response, a weighted average operation can be applied at every instance. The convolution operation can be defined by:

$$s(t) = \int x(a)w(t-a)da, \tag{A.24}$$

and is usually denoted with an asterisk such that:

$$s(t) = (x * w)(t). \tag{A.25}$$

In order for $s(t)$ to be a weighted average, w must be a probability density function. However, in general, convolutions are defined for any function which A.24 is defined. We term $x(t)$ the **input** to the convolution and $w(a)$ the **kernel**.

The use of the integral can be replaced with a summation in problems where t is discrete. We can then define the discrete convolution operation as:

$$s(t) = (x * w)(t) = \sum_{a=-\infty}^{\infty} x(a)w(t-a). \tag{A.26}$$

In such applications, where imaging data is used as input, I, it is useful to a two-dimensional kernel, K, during convolution:

$$S(i,j) = (I * K)(i,j) = \sum_m \sum_n I(m,n)K(i-m, j-n). \tag{A.27}$$

An important property of convolution is that, when the kernel is flipped relative to the input, it is commutative. We can then write:

$$S(i,j) = (K * I)(i,j) = \sum_m \sum_n I(i-m, j-n)K(m,n), \tag{A.28}$$

which is useful in implementation. Many implementations do however make use of **cross-correlation** instead of convolution which can be written as:

$$S(i,j) = (K * I)(i,j) = \sum_m \sum_n I(i+m, j+n)K(m,n), \tag{A.29}$$

which is the same as convolution except the kernel is not flipped.

A.2.2 Convolutional layer

Imaging data are usually represented by tensors which contain colour information in the form of red-green-blue (RGB) channels. Alternative image representations exist, such as HSL (hue, saturation, lightness) and HSV (hue, saturation, value) or HSB (hue, saturation, brightness). In these representations, the colours of each hue are arranged in a radial slice, around a central axis of neutral colours which ranges from black to white. Images are represented in the shape $h \times w \times c$, where $c = 3$ for (RGB) colour images. The purpose of a convolutional layer in a CNN is to extract high-level feature maps from images.

The kernels, K, used in convolutional layers are represented by their size $K_x \times K_y \times c$, where the receptive field on the image can be defined as $K_x \times K_y$. The kernel is convolved with the image in order to gain feature maps as output (Figure A.2). The number of channels used in the kernel is always the same as those used in the input to the convolution; thus, c changes throughout the architecture. The kernel moves along the image from left to right with a particular stride until it has travelled along the whole width of the image, where it then returns to the left and down with

the same stride. This process is continued until the kernel has parsed over the entire image or input. The stride parameter is defined in the construction of the network as its value affects the output of the convolutional layer such that the row output size is defined:

$$O_x = \frac{I_x - K_x}{s} + 1, \tag{A.30}$$

and the column output size is given by:

$$O_y = \frac{I_y - K_y}{s} + 1. \tag{A.31}$$

Several convolutions are usually performed in parallel to produce a set of linear activation's which are then processed through a non-linear activation function such as the Rectified Linear Units (ReLU) activation function.

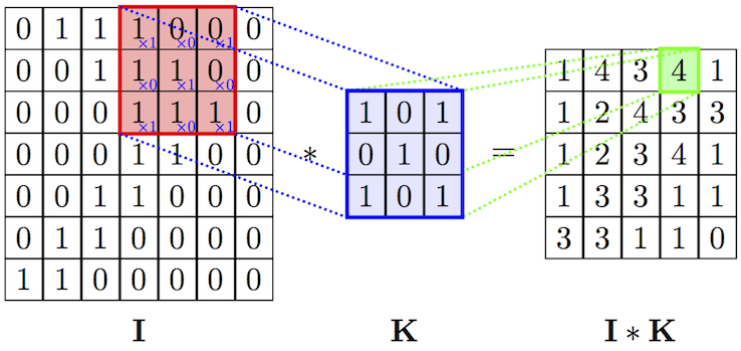


FIGURE A.2: Convolution in CNNs. [Source: (S. Mohamed, 2017)]

Convolution leverages significant properties which increase the performance of deep learning machines: sparse interaction, parameter sharing, and equivariant representations (Goodfellow, Bengio, and Courville, 2016). In traditional feed-forward neural networks, layers utilise matrix multiplication with individual parameters for each input and output unit. Convolutional layers, however, have sparse connectivity which may be achieved through the kernel size is smaller than that of the input. This allows much fewer parameters to be stored in turn, improving statistical efficiency while decreasing memory requirements as well as time complexity. Figure A.3 presents a graphical representation of this concept.

Goodfellow, Bengio, and Courville (2016) defines parameter sharing as "using

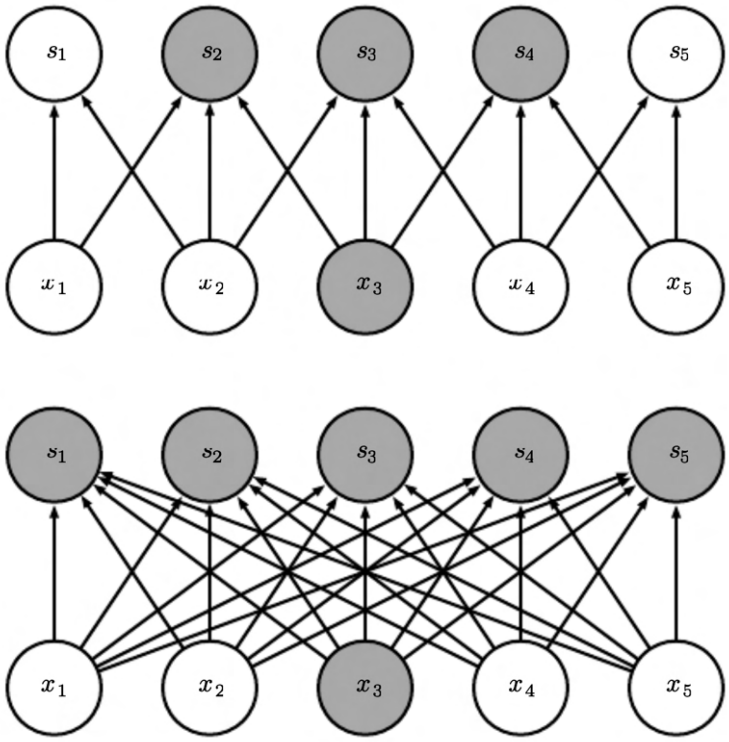


FIGURE A.3: Sparse connectivity, highlighting one output unit, s_3, and the input units in x that affect this unit. These units are known as the receptive field of s_3. (Top) When s is formed by convolution with a kernel of width 3, only three inputs affect s_3. (Bottom) When s is formed by matrix multiplication, connectivity is no longer sparse, so all the inputs affect s_3. [Source: (Goodfellow, Bengio, and Courville, 2016)]

the same parameter for more than one function in a model". In computing the output of traditional layers, each individual entry to the matrix of weights is utilised only once and never reused. In convolutional layer, however, every element in the kernel is utilised at each position of the input. This allows the machine to learn only one set of parameters which further decreases the storage requirements of the machine.

The property possessed by convolution to be equivariant to translation follows directly from this parameter sharing. A function is equivariant when a change to its inputs causes an identical change to its outputs. Formally, a function f is equivariant to another function g if $f(g(x)) = g(f(x))$, for some input x. Thus, the order of convolution and transformation on an input has no effect on the output produced. This equivariance property is useful in extracting different features in different layers of a CNN.

A.2.3 Pooling layer

Also known as downsampling layers, pooling layers are usually between most partially connected convolutional layers. Dissimilar to convolutional layers, pooling layers do not contain learnable parameters. Their function is to instead replace the output of a convolutional layer at a particular location with a summary statistic of the nearby outputs. This is a form of dimension reduction where the information outputted from the convolutional layer is compressed.

The most commonly used pooling operation in the literature, max-pooling (Zhou and Chellappa, 1988), returns the maximum output within a pre-defined region (Figure A.4). Other favoured pooling operations include average pooling, weighted average pooling with weights determined by the distance from the centre pixel, and L^2 norm pooling. The pooling operation acts independently on each channel or depth of the input in order to spatially reshape it. Pooling has an important effect which assists in allowing the output of the layer to be invariant to input translation. This is a useful property if the problem of object presence in an image is more important to localisation.

The use of pooling adds a significantly strong prior that the model learnt is invariant to small translations. If this prior assumption is valid, pooling can appreciably ameliorate the statistical efficiency of a learning machine. This may pose a problem in semantic segmentation where localisation is critical. Non-overlapping pooling grids (where the size of the pooling filter is equal to the stride, i.e. 2x2 region with a stride of 2), however, an interesting observation made by Krizhevsky, Sutskever, and Hinton (2012) showed that overlapping pooling with equivalent output dimensions (e.g. a 33 window with a stride of 2) reduces over-fitting. Pooling significantly reduces the number of learnable parameters in a model. For example, a pooling window of size 2x2 with a stride of 2 discards 75% of the activations. The depth dimension from the input, however, remains unchanged. This extreme scrapping of information has caused the use of pooling to come under scrutiny in recent years (Ruderman et al., 2018). Instead, techniques which utilise multiple

convolutional layers before implementing any pooling layer (Simonyan and Zisserman, 2014), or completely disregarding pooling layers in the architecture have been adopted (Springenberg et al., 2014). Instead, to reduce the dimensions of feature maps outputted by convolutional layers, the use of larger strides in these layers may be adopted. It is unclear whether the deep learning community will unanimously decide on their use or misuse in future applications. It is worth noting that in the training of generative models, such as Goodfellow et al.'s (2014) generative adversarial network, it has been found paramount to discard pooling.

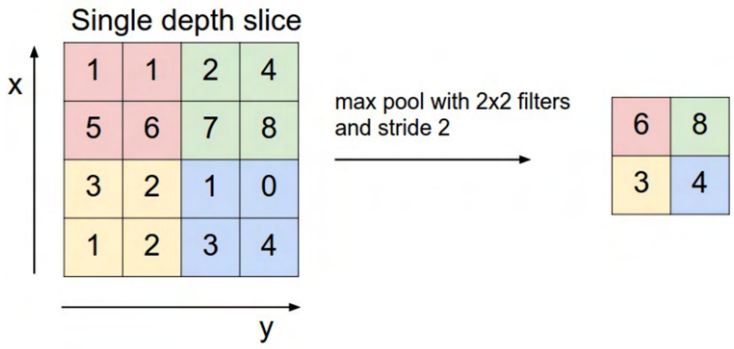


FIGURE A.4: A graphical representation of max-pooling. [Source: (Li, 2019)]

A.2.4 Data Augmentation

Although first introduced in 1988 (Lecun et al., 1998), CNNs took time to discover their true potential due to inadequate computing hardware. However, with any model fitting technique comes the threat of over-fitting. Over-fitting can be defined as a modelling error which occurs when a function is too closely fit to a finite set of data points. This will generally take the form of using an overly complex model in order to explain idiosyncrasies in the data used to train the machine. A consequence of over-fitting is the ability for a model to perform flawlessly on the training data but fails to generalise on unseen data. A machine might over-fit if the training data contains accidental or uncommon regularities and if an overly flexible machine is used. Infinitely expressive machines have an infinity of factitious rules which they may learn.

A successful machine should be able to capture some structure in the data. When there are few training examples used to train a complex model, the model is able to memorise the training set, i.e. learn the correct answer for each training example with no ability to classify novel examples. This is due to complex models having too large a capacity, or too many trainable parameters. The idea is that these trainable parameters are able to store information so that models with more trainable parameters are able to store more information. When the number of training examples increases, it becomes increasingly difficult for the model to learn a spurious rule and furthermore, for any test point, it becomes more likely that there will be a training example which is closely related. With a larger training set, there are relatively fewer accidental regularities, and thus the model will be forced to learn the true underlying structure in the data. This is the basis for adding more training examples to a complex model in order to improve the generalisation error, with possibly increasing the training error. This is known as decreasing the variance of a machine learning model.

With most applications, it is not possible to increase the number of data points, especially in medical imaging where data sets may be relatively small. However, we may *create* more data through augmentation. It may be reasonably straightforward, in classification tasks, to augment datasets. In these applications, we generate new $(\boldsymbol{x}, \boldsymbol{y})$ pairs through the transformation of the $\boldsymbol{x}$ inputs in the training set. Imaging data is considered to be high dimensional with multiple elements of disparity, most of which may be simply replicated. Wang et al. (2019a) claimed that "data augmentation is more important than model architectures for retinal vessel segmentation." Their experiments showed how simple U-Net models might outperform state-of-the-art complex architectures through effective augmentation regimes. A major finding was the extreme effect that image patch sampling at numerous orientation angles would have on the generalisation error. Common data augmentation operations include shifts, rotation and scaling images. With segmentation algorithms, it is essential that if these transformations are undertaken on the inputs, they must be done on the outputs as the gold standard segmentation will change with shift and rotation of the input image. The injection of noise into training images has also

proven to increase the performance of vessel segmentation (Shorten and Khoshgoftaar, 2019). When comparing varying learning algorithms or deep learning architectures, it is thus essential to take into account data augmentation. Often this is not the case as state-of-the-art results shown in papers may be achieved through thorough data pre-processing and augmentation rather than the carefully curated models. To accurately compare models, controlled experiments must be performed.

A.2.5 Regularisation

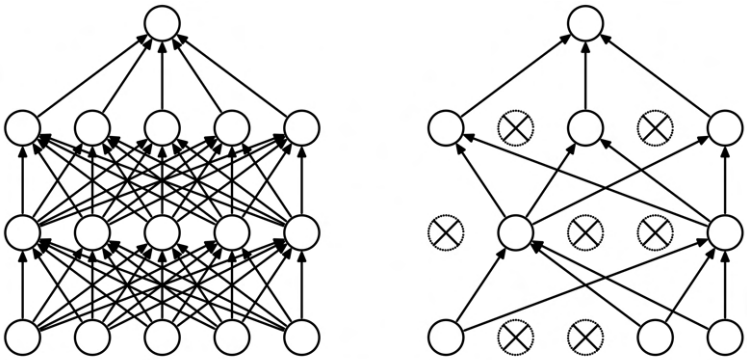
Along with techniques to increase the dataset size, researchers have explored methods which attempt to decrease the capacity of their models. Regularisation has been defined as "any modification we make to a learning algorithm that is intended to reduce its generalisation error but not its training error" (Goodfellow, Bengio, and Courville, 2016). Countless forms of regularisation techniques exist in the handbook of a deep learning practitioner. Furthermore, a major effort has been placed on the development of novel regularisation strategies in the literature. Regularisation strategies in deep learning work by trading an increase in bias for a decrease in variance. The most effective regularisers aim to profit from the bias-variance trad-off, by diminishing the variance substantially without much, if any, increase in bias. Controlling the capacity of a model does not merely entail building a model with the perfect number of learnable parameters, rather best-performing learning machines are large, yet suitably regularised.

Developed by Caruana (1997), multitask learning has played an important role in numerous deep learning tasks. The intuition is that when portions of learning models are distributed amongst various tasks, those portions are forced towards appropriate values, allowing better generalisation on unseen data. In this problem class, the learnt model is split into portions with their accompanying parameters. The first of which is task-specific, while the others are generic. The improved generalisation error will only materialise in problems where assumptions about the statistical relationships between tasks hold. Multitask learning has recently transpired in vessel segmentation tasks (Ma et al., 2019) with propositions to segment thin and thick vessels as separate tasks or vessels, arteries, and veins (Ma et al., 2019).

Learning machines which contain ample capacity will continuously decrease the training error towards zero over time. However, after a certain point, the validation error will not continue to decrease accordingly. Rather the validation error may begin to rise. This is a common consequence of over-fitting. Thus, by returning to this point after the training has completed and saving the model weights where the validation error was the lowest, will produce a better machine. This is one of the most commonly used forms of regularisation in the deep learning literature, known as early stopping. Early stopping, however, does come with costs of evaluating the validation error continuously after every epoch. Preferably, this is done separately from the main GPU used in training. The advantages of early stopping lie in its simplicity with no change needed to the cost function or constraints on the parameters. It is common practice to utilise early stopping in conjunction with other regularisation techniques. It is easily seen from analysing training curves and with a little intuition, how early stopping will lead to better generalisation; however, (Bishop, 1995) shows how early stopping is analogous to L_2 regularisation.

First introduced in 2012 (Hinton et al., 2012) and formally defined in 2014 (Srivastava et al., 2014), dropout has been critical in preventing state-of-the-art methods from over-fitting (Krizhevsky, Sutskever, and Hinton, 2012). Dropout is established as a powerful means to achieving model aggregation unaccompanied by the significant price of creating numerous models. Specifically, this technique trains this ensemble of sub-models which are created through the removal of neurons with a certain probability whenever the model is presented with a new batch of training data. These neurons are effectively discarded from the network by multiplying their value by zero.

Dropout aims to approximate the operation of bootstrap aggregation (bagging) (Breiman, 1996), through the use of multitudes of neural sub-networks. It is common practice to include an input unit with 0.8 probability, and a hidden unit with 0.5 probability. Dropout *approximates* bagging as it is still dissimilar. In bagging, it is required that all the models be independent of each other while in dropout, the models have parameters in common. Srivastava et al. (2014) described the purpose of dropout to eliminate or significantly reduce *complex co-adaptions* between neurons.



(a) Standard Neural Net (b) After applying dropout.

FIGURE A.5: A graphical representation of dropout. [Source: (Srivastava et al., 2014)]

This occurrence emerges when different neurons in the same layer of a network depend on each other's results. In this scenario, few neurons recognise meaningful information while the others "piggy-back" and would not contribute if their reliable counterparts were removed from the network. Dropout thus enforces each neuron to be robust on its own. Interestingly, Srivastava et al. (2014) cited sexual reproduction and survival of the fittest as an inspiration to dropout. In reproduction, roughly half of the genes from both the male and the female are not present in the offspring. This forces robust genes to be selected as if those which are present are not robust and depend on those which are not, the offspring will not survive. This thought process can be extended to models which include dropout: the final neurons are robust, no matter what other neurons are present in the current model. Figure A.5 presents the difference between a fully-connected network and one which applies dropout.

Further interpretations which account for the model improvement when using dropout are terms of and model averaging ensemble methods. In one layer of a standard feed-forward network which incorporates dropout with 0.5 probability, there are 2^N different sub-networks known as "thinned networks" (Srivastava et al., 2014). When a batch is presented to the model, a single thinned network is trained. In the testing phase, all the neurons are present in the network as dropout is abandoned. The weights of these are, however, scaled by a factor of 0.5. Thus, the resulting model is an approximation to the average of the 2^N thinned networks. Srivastava et al. (2014) revealed how dropout might be more successful than other regularisation techniques, such as sparse activity regularisation and weight decay. A major advantage of dropout is its computational efficiency. Its computational cost is linear

in the number of training examples ($O(n)$).

Appendix B

Code

B.1 Pre-processing

B.1.1 Patch extraction

```
"""
Random patch extraction for training
"""

import csv
import os
import urllib.request
import zipfile
from random import randint
from shutil import copyfile

from PIL import Image
from tqdm import tqdm
import numpy as np

from util import create_dir_if_not_exist


def create_patch(image_path, gt_path, patch_dir, patch_size, patch_per_image, inside=True):
    # Create dirs
    gt_dir = patch_dir + "_GT"
    image_dir = patch_dir
    create_dir_if_not_exist(gt_dir)
    create_dir_if_not_exist(image_dir)
    create_dir_if_not_exist("random")

    # Iterate through files to split and group them
    image_files = os.listdir(image_path)
    print(len(image_files), "slide images found")
    iter_tot = 0
    for image_file in tqdm(image_files, desc="Splitting images"):
        if "DS_Store" not in image_file:
            image = Image.open(image_path + "/" + image_file)
            image_np = np.asarray(image)
            # print(image_np.shape)

            gt = Image.open(gt_path + "/" + image_file[:-3] + 'png')
            gt_np = np.asarray(gt)
            # print(gt_np.shape)
            gt_np = np.reshape(gt_np, (gt_np.shape[0], gt_np.shape[1], 1))
            # print(gt_np.shape)
```

```
            width, height = image.size

            save_dir_image = image_dir
            save_dir_gt = gt_dir


            k = 0
            patches = []
            patches_gt = []
            while k < patch_per_image:
                x_center = randint(0 + int(patch_size / 2), width - int(patch_size / 2))
                # print "x_center " +str(x_center)
                y_center = randint(0 + int(patch_size / 2), height - int(patch_size / 2))
                # print "y_center " +str(y_center)
                # check whether the patch is fully contained in the FOV
                if inside == True:
                    if is_patch_inside_FOV(x_center, y_center, img_w, img_h, patch_h) == False:
                        continue
                patch = image_np[x_center - int(patch_size / 2):x_center + int(patch_size / 2),
                        y_center - int(patch_size / 2):y_center + int(patch_size / 2),
                        :]
                patch_mask = gt_np[x_center - int(patch_size / 2):x_center + int(patch_size / 2),
                             y_center - int(patch_size / 2):y_center + int(patch_size / 2),
                             :]
                patches.append(patch)
                patches_gt.append(patch_mask)

                box = (x_center - int(patch_size / 2), y_center - int(patch_size / 2),
                       x_center + int(patch_size / 2), y_center + int(patch_size / 2))
                cropped_data = image.crop(box)
                cropped_data_gt = gt.crop(box)

                cropped_image = Image.new('RGB', (patch_size, patch_size), 255)
                cropped_image.paste(cropped_data)

                cropped_image_gt = Image.new('RGB', (patch_size, patch_size), 255)
                cropped_image_gt.paste(cropped_data_gt)


                iter_tot += 1  # total
                k += 1  # per full_img

                cropped_image.save(save_dir_image + "/" + str(iter_tot).zfill(5) + ".png")

                cropped_image_gt.save(save_dir_gt + "/" + str(iter_tot).zfill(5) + ".png")

    print('Created', iter_tot, 'split images')


# check if the patch is fully contained in the FOV
def is_patch_inside_FOV(x, y, img_w, img_h, patch_h):
    x_ = x - int(img_w / 2)  # origin (0,0) shifted to image center
    y_ = y - int(img_h / 2)  # origin (0,0) shifted to image center
    R_inside = 270 - int(patch_h * np.sqrt(2.0) / 2.0)
    # radius is 270 (from DRIVE db docs), minus the patch diagonal
    # (assumed it is a square # this is the limit to contain the full
    # patch in the FOV
    radius = np.sqrt((x_ * x_) + (y_ * y_))
    if radius < R_inside:
        return True
    else:
        return False
```

B.1.2 CLAHE

```
# for using Clahe
import cv2
        image = cv2.imread(image_path)
        GT = cv2.imread(GT_path)

        # image
        lab = cv2.cvtColor(image, cv2.COLOR_BGR2LAB)
        lab_planes = cv2.split(lab)
        clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(10, 10))
        lab_planes[0] = clahe.apply(lab_planes[0])
        lab = cv2.merge(lab_planes)
        image = cv2.cvtColor(lab, cv2.COLOR_LAB2BGR)
        # GT
        lab = cv2.cvtColor(GT, cv2.COLOR_BGR2LAB)
        lab_planes = cv2.split(lab)
        clahe = cv2.createCLAHE(clipLimit=2.0, tileGridSize=(10, 10))
        lab_planes[0] = clahe.apply(lab_planes[0])
        lab = cv2.merge(lab_planes)
        GT = cv2.cvtColor(lab, cv2.COLOR_LAB2BGR)
```

B.2 Networks

B.2.1 U-Net parts

```
class DoubleConv(nn.Module):
    """(convolution => [BN] => ReLU) * 2"""

    def __init__(self, in_channels, out_channels, mid_channels=None):
        super().__init__()
        if not mid_channels:
            mid_channels = out_channels
        self.double_conv = nn.Sequential(
            nn.Conv2d(in_channels, mid_channels, kernel_size=3, padding=1),
            nn.BatchNorm2d(mid_channels),
            # nn.Dropout2d(0.5, inplace=True),
            nn.ReLU(inplace=True),
            nn.Conv2d(mid_channels, out_channels, kernel_size=3, padding=1),
            nn.BatchNorm2d(out_channels),
            # nn.Dropout2d(0.5,inplace=True),
            nn.ReLU(inplace=True)
        )

    def forward(self, x):
        return self.double_conv(x)


class Down(nn.Module):
    """Downscaling with maxpool then double conv"""

    def __init__(self, in_channels, out_channels):
        super().__init__()
        self.maxpool_conv = nn.Sequential(
            nn.MaxPool2d(2),
            DoubleConv(in_channels, out_channels)
        )

    def forward(self, x):
        return self.maxpool_conv(x)
```

```
class Up(nn.Module):
    """Upscaling then double conv"""

    def __init__(self, in_channels, out_channels, bilinear=True):
        super().__init__()

        # if bilinear, use the normal convolutions to reduce the number of channels
        if bilinear:
            self.up = nn.Upsample(scale_factor=2, mode='bilinear', align_corners=True)
            self.conv = DoubleConv(in_channels, out_channels, in_channels // 2)
        else:
            self.up = nn.ConvTranspose2d(in_channels, in_channels // 2, kernel_size=2, stride=2)
            self.conv = DoubleConv(in_channels, out_channels)

    def forward(self, x1, x2):
        x1 = self.up(x1)
        # input is CHW
        diffY = x2.size()[2] - x1.size()[2]
        diffX = x2.size()[3] - x1.size()[3]

        x1 = F.pad(x1, [diffX // 2, diffX - diffX // 2,
                        diffY // 2, diffY - diffY // 2])
        # if you have padding issues, see
        # https://github.com/HaiyongJiang/U-Net-Pytorch-Unstructured-Buggy/commit/0e854509c2cea854e247a9c615f175f76fbb2e3a
        # https://github.com/xiaopeng-liao/Pytorch-UNet/commit/8ebac70e633bac59fc22bb5195e513d5832fb3bd
        x = torch.cat([x2, x1], dim=1)
        return self.conv(x)


class OutConv(nn.Module):
    def __init__(self, in_channels, out_channels):
        super(OutConv, self).__init__()
        self.conv = nn.Conv2d(in_channels, out_channels, kernel_size=1)

    def forward(self, x):
        return self.conv(x)

class conv_block(nn.Module):
    def __init__(self, ch_in, ch_out):
        super(conv_block, self).__init__()
        self.conv = nn.Sequential(
            nn.Conv2d(ch_in, ch_out, kernel_size=3, stride=1, padding=1, bias=True),
            nn.BatchNorm2d(ch_out),
            nn.ReLU(inplace=True),
            nn.Conv2d(ch_out, ch_out, kernel_size=3, stride=1, padding=1, bias=True),
            nn.BatchNorm2d(ch_out),
            # nn.Dropout2d(0.5, inplace=True),
            nn.ReLU(inplace=True)
        )

    def forward(self, x):
        x = self.conv(x)
        return x


class up_conv(nn.Module):
    def __init__(self, ch_in, ch_out):
        super(up_conv, self).__init__()
        self.up = nn.Sequential(
            nn.Upsample(scale_factor=2),
            nn.Conv2d(ch_in, ch_out, kernel_size=3, stride=1, padding=1, bias=True),
            nn.BatchNorm2d(ch_out),
            nn.ReLU(inplace=True)
        )
```

```
    def forward(self, x):
        x = self.up(x)
        return x


class single_conv(nn.Module):
    def __init__(self, ch_in, ch_out):
        super(single_conv, self).__init__()
        self.conv = nn.Sequential(
            nn.Conv2d(ch_in, ch_out, kernel_size=3, stride=1, padding=1, bias=True),
            nn.BatchNorm2d(ch_out),
            # nn.Dropout2d(0.5),
            nn.ReLU(inplace=True)
        )

    def forward(self, x):
        x = self.conv(x)
        return x
```

B.2.2 U-Net

```
class UNet(nn.Module):
    def __init__(self, n_channels, n_classes, out_channels=32):
        super(UNet, self).__init__()
        self.n_channels = n_channels
        self.n_classes = n_classes
        bilinear = False

        self.inc = DoubleConv(n_channels, out_channels)
        self.down1 = Down(out_channels, out_channels * 2)
        self.down2 = Down(out_channels * 2, out_channels * 4)
        self.down3 = Down(out_channels * 4, out_channels * 8)
        factor = 2 if bilinear else 1
        self.down4 = Down(out_channels * 8, out_channels * 16 // factor)
        self.up1 = Up(out_channels * 16, out_channels * 8 // factor, bilinear)
        self.up2 = Up(out_channels * 8, out_channels * 4 // factor, bilinear)
        self.up3 = Up(out_channels * 4, out_channels * 2 // factor, bilinear)
        self.up4 = Up(out_channels * 2, out_channels, bilinear)
        self.outc = OutConv(out_channels, n_classes)

    def forward(self, x):
        x1 = self.inc(x)
        x2 = self.down1(x1)
        x3 = self.down2(x2)
        x4 = self.down3(x3)
        x5 = self.down4(x4)
        x = self.up1(x5, x4)
        x = self.up2(x, x3)
        x = self.up3(x, x2)
        x = self.up4(x, x1)
        logits = self.outc(x)
        return logits
```

B.2.3 Attention parts

```
class Attention_block(nn.Module):
    def __init__(self, F_g, F_l, F_int):
        super(Attention_block, self).__init__()
        self.W_g = nn.Sequential(
```

```
            nn.Conv2d(F_g, F_int, kernel_size=1, stride=1, padding=0, bias=True),
            nn.BatchNorm2d(F_int)
        )

        self.W_x = nn.Sequential(
            nn.Conv2d(F_l, F_int, kernel_size=1, stride=1, padding=0, bias=True),
            nn.BatchNorm2d(F_int)
        )

        self.psi = nn.Sequential(
            nn.Conv2d(F_int, 1, kernel_size=1, stride=1, padding=0, bias=True),
            nn.BatchNorm2d(1),
            nn.Sigmoid()
        )

        self.relu = nn.ReLU(inplace=True)

    def forward(self, g, x):
        g1 = self.W_g(g)
        x1 = self.W_x(x)
        psi = self.relu(g1 + x1)
        psi = self.psi(psi)

        return x * psi
```

B.2.4 Attention U-Net

```
class AttU_Net(nn.Module):
    def __init__(self, img_ch=3, output_ch=1):
        super(AttU_Net, self).__init__()

        self.Maxpool = nn.MaxPool2d(kernel_size=2, stride=2)

        self.Conv1 = conv_block(ch_in=img_ch, ch_out=64)
        self.Conv2 = conv_block(ch_in=64, ch_out=128)
        self.Conv3 = conv_block(ch_in=128, ch_out=256)
        self.Conv4 = conv_block(ch_in=256, ch_out=512)
        self.Conv5 = conv_block(ch_in=512, ch_out=1024)

        self.Up5 = up_conv(ch_in=1024, ch_out=512)
        self.Att5 = Attention_block(F_g=512, F_l=512, F_int=256)
        self.Up_conv5 = conv_block(ch_in=1024, ch_out=512)

        self.Up4 = up_conv(ch_in=512, ch_out=256)
        self.Att4 = Attention_block(F_g=256, F_l=256, F_int=128)
        self.Up_conv4 = conv_block(ch_in=512, ch_out=256)

        self.Up3 = up_conv(ch_in=256, ch_out=128)
        self.Att3 = Attention_block(F_g=128, F_l=128, F_int=64)
        self.Up_conv3 = conv_block(ch_in=256, ch_out=128)

        self.Up2 = up_conv(ch_in=128, ch_out=64)
        self.Att2 = Attention_block(F_g=64, F_l=64, F_int=32)
        self.Up_conv2 = conv_block(ch_in=128, ch_out=64)

        self.Conv_1x1 = nn.Conv2d(64, output_ch, kernel_size=1, stride=1, padding=0)

    def forward(self, x):
        # encoding path
        x1 = self.Conv1(x)

        x2 = self.Maxpool(x1)
        x2 = self.Conv2(x2)
```

```
        x3 = self.Maxpool(x2)
        x3 = self.Conv3(x3)

        x4 = self.Maxpool(x3)
        x4 = self.Conv4(x4)

        x5 = self.Maxpool(x4)
        x5 = self.Conv5(x5)

        # decoding + concat path
        d5 = self.Up5(x5)
        x4 = self.Att5(g=d5, x=x4)
        d5 = torch.cat((x4, d5), dim=1)
        d5 = self.Up_conv5(d5)

        d4 = self.Up4(d5)
        x3 = self.Att4(g=d4, x=x3)
        d4 = torch.cat((x3, d4), dim=1)
        d4 = self.Up_conv4(d4)

        d3 = self.Up3(d4)
        x2 = self.Att3(g=d3, x=x2)
        d3 = torch.cat((x2, d3), dim=1)
        d3 = self.Up_conv3(d3)

        d2 = self.Up2(d3)
        x1 = self.Att2(g=d2, x=x1)
        d2 = torch.cat((x1, d2), dim=1)
        d2 = self.Up_conv2(d2)

        d1 = self.Conv_1x1(d2)

        return d1
```

B.2.5 Iternet

```
class UNetForIter(nn.Module):
    def __init__(self, n_channels, n_classes, out_channels=32):
        super(UNetForIter, self).__init__()
        self.n_channels = n_channels
        self.n_classes = n_classes
        bilinear = False

        self.inc = DoubleConv(n_channels, out_channels)
        self.down1 = Down(out_channels, out_channels * 2)
        self.down2 = Down(out_channels * 2, out_channels * 4)
        self.down3 = Down(out_channels * 4, out_channels * 8)
        factor = 2 if bilinear else 1
        self.down4 = Down(out_channels * 8, out_channels * 16 // factor)
        self.up1 = Up(out_channels * 16, out_channels * 8 // factor, bilinear)
        self.up2 = Up(out_channels * 8, out_channels * 4 // factor, bilinear)
        self.up3 = Up(out_channels * 4, out_channels * 2 // factor, bilinear)
        self.up4 = Up(out_channels * 2, out_channels, bilinear)
        self.outc = OutConv(out_channels, n_classes)

    def forward(self, x):
        x1 = self.inc(x)
        x2 = self.down1(x1)
        x3 = self.down2(x2)
        x4 = self.down3(x3)
        x5 = self.down4(x4)
        x = self.up1(x5, x4)
```

```
        x = self.up2(x, x3)
        x = self.up3(x, x2)
        x = self.up4(x, x1)
        logits = self.outc(x)
        return x1, x, logits


class MiniUNet(nn.Module):
    def __init__(self, n_channels, n_classes, out_channels=32):
        super(MiniUNet, self).__init__()
        self.n_channels = n_channels
        self.n_classes = n_classes
        bilinear = False

        self.inc = DoubleConv(n_channels, out_channels)
        self.down1 = Down(out_channels, out_channels * 2)
        self.down2 = Down(out_channels * 2, out_channels * 4)
        self.down3 = Down(out_channels * 4, out_channels * 8)
        self.up1 = Up(out_channels * 8, out_channels * 4, bilinear)
        self.up2 = Up(out_channels * 4, out_channels * 2, bilinear)
        self.up3 = Up(out_channels * 2, out_channels, bilinear)
        self.outc = OutConv(out_channels, n_classes)

    def forward(self, x):
        x1 = self.inc(x)
        x2 = self.down1(x1)
        x3 = self.down2(x2)
        x4 = self.down3(x3)
        x = self.up1(x4, x3)
        x = self.up2(x, x2)
        x = self.up3(x, x1)
        logits = self.outc(x)
        return x1, x, logits


class Iternet(nn.Module):
    def __init__(self, n_channels, n_classes, out_channels=32, iterations=3):
        super(Iternet, self).__init__()
        self.n_channels = n_channels
        self.n_classes = n_classes
        self.iterations = iterations

        # define the network UNet layer
        self.model_unet = UNetForIter(n_channels=n_channels,
                                      n_classes=n_classes, out_channels=out_channels)

        # define the network MiniUNet layers
        self.model_miniunet = ModuleList(MiniUNet(
            n_channels=out_channels * 2, n_classes=n_classes, out_channels=out_channels) for i in range(iterations))

    def forward(self, x):
        x1, x2, logits = self.model_unet(x)
        for i in range(self.iterations):
            x = torch.cat([x1, x2], dim=1)
            _, x2, logits = self.model_miniunet[i](x)

        return logits
```

B.2.6 Attention Iternet

```
class AttUNetForIter(nn.Module):
    def __init__(self, img_ch=3, output_ch=1):
        super(AttUNetForIter, self).__init__()
```

```
        self.Maxpool = nn.MaxPool2d(kernel_size=2, stride=2)

        self.Conv1 = conv_block(ch_in=img_ch, ch_out=32)
        self.Conv2 = conv_block(ch_in=32, ch_out=64)
        self.Conv3 = conv_block(ch_in=64, ch_out=128)
        self.Conv4 = conv_block(ch_in=128, ch_out=256)
        self.Conv5 = conv_block(ch_in=256, ch_out=512)

        self.Up5 = up_conv(ch_in=512, ch_out=256)
        self.Att5 = Attention_block(F_g=256, F_l=256, F_int=128)
        self.Up_conv5 = conv_block(ch_in=512, ch_out=256)

        self.Up4 = up_conv(ch_in=256, ch_out=128)
        self.Att4 = Attention_block(F_g=128, F_l=128, F_int=64)
        self.Up_conv4 = conv_block(ch_in=256, ch_out=128)

        self.Up3 = up_conv(ch_in=128, ch_out=64)
        self.Att3 = Attention_block(F_g=64, F_l=64, F_int=32)
        self.Up_conv3 = conv_block(ch_in=128, ch_out=64)

        self.Up2 = up_conv(ch_in=64, ch_out=32)
        self.Att2 = Attention_block(F_g=32, F_l=32, F_int=16)
        self.Up_conv2 = conv_block(ch_in=64, ch_out=32)
        # self.Up_conv1 = conv_block(ch_in=64, ch_out=32)

        self.Conv_1x1 = nn.Conv2d(32, output_ch, kernel_size=1, stride=1, padding=0)

    def forward(self, x):
        # encoding path
        x1 = self.Conv1(x)

        x2 = self.Maxpool(x1)
        x2 = self.Conv2(x2)

        x3 = self.Maxpool(x2)
        x3 = self.Conv3(x3)

        x4 = self.Maxpool(x3)
        x4 = self.Conv4(x4)

        x5 = self.Maxpool(x4)
        x5 = self.Conv5(x5)

        # decoding + concat path
        d5 = self.Up5(x5)
        x4 = self.Att5(g=d5, x=x4)
        d5 = torch.cat((x4, d5), dim=1)
        d5 = self.Up_conv5(d5)

        d4 = self.Up4(d5)
        x3 = self.Att4(g=d4, x=x3)
        d4 = torch.cat((x3, d4), dim=1)
        d4 = self.Up_conv4(d4)

        d3 = self.Up3(d4)
        x2 = self.Att3(g=d3, x=x2)
        d3 = torch.cat((x2, d3), dim=1)
        d3 = self.Up_conv3(d3)

        d2 = self.Up2(d3)
        x1 = self.Att2(g=d2, x=x1)
        d2 = torch.cat((x1, d2), dim=1)
        d2 = self.Up_conv2(d2)
        # d2 = self.Up_conv1(d2)
```

```
        d1 = self.Conv_1x1(d2)

        return x1, d2, d1


class AttUIternet(nn.Module):
    def __init__(self, n_channels, n_classes, out_channels=32, iterations=3):
        super(AttUIternet, self).__init__()
        self.n_channels = n_channels
        self.n_classes = n_classes
        self.iterations = iterations

        # define the network UNet layer
        self.model_unet = AttUNetForIter(n_channels, 1)

        # define the network MiniUNet layers
        self.model_miniunet = ModuleList(MiniUNet(
            n_channels=out_channels * 2, n_classes=n_classes, out_channels=out_channels) for i in range(iterations))

    def forward(self, x):
        x1, x2, logits = self.model_unet(x)
        for i in range(self.iterations):
            x = torch.cat([x1, x2], dim=1)
            _, x2, logits = self.model_miniunet[i](x)

        return logits
```

Bibliography

Abbas, Waseem et al. (2019). "Patch-Based Generative Adversarial Network Towards Retinal Vessel Segmentation". In: *Neural Information Processing*. Ed. by Tom Gedeon, Kok Wai Wong, and Minho Lee. Cham: Springer International Publishing, pp. 49–56. ISBN: 978-3-030-36808-1.

Afifi, Ashraf et al. (June 2015). "New Region Growing based on Thresholding Technique Applied to MRI Data". In: *International Journal of Computer Network and Information Security* 7, pp. 61–67. DOI: 10.5815/ijcnis.2015.07.08.

Ahlem, Melouah and Soumaya Layachi (Nov. 2015). "A Novel Automatic Seed Placement Approach for Region Growing segmentation in Mammograms". In: pp. 1–5. DOI: 10.1145/2816839.2816892.

Al-Rawi, Mohammed and Huda Karajeh (2007). "Genetic algorithm matched filter optimization for automated detection of blood vessels from digital retinal images". In: *Computer Methods and Programs in Biomedicine* 87.3, pp. 248 –253. ISSN: 0169-2607. DOI: https://doi.org/10.1016/j.cmpb.2007.05.012. URL: http://www.sciencedirect.com/science/article/pii/S0169260707001332.

Ali, Aziah, Wan Mimi Diyana Wan Zaki, and Aini Hussain (2018). "Blood Vessel Segmentation from Color Retinal Images Using K-Means Clustering and 2D Gabor Wavelet". In: *Applied Physics, System Science and Computers*. Ed. by Klimis Ntalianis and Anca Croitoru. Cham: Springer International Publishing, pp. 221–227. ISBN: 978-3-319-53934-8.

Alom, Md Zahangir et al. (2018). *Recurrent Residual Convolutional Neural Network based on U-Net (R2U-Net) for Medical Image Segmentation*. arXiv: 1802.06955 [cs.CV].

Andermatt, Simon, Simon Pezold, and Philippe Cattin (Oct. 2016). "Multi-dimensional Gated Recurrent Units for the Segmentation of Biomedical 3D-Data". In: pp. 142–151. DOI: 10.1007/978-3-319-46976-8_15.

Annunziata, R. et al. (2016). "Leveraging Multiscale Hessian-Based Enhancement With a Novel Exudate Inpainting Technique for Retinal Vessel Segmentation". In: *IEEE Journal of Biomedical and Health Informatics* 20.4, pp. 1129–1138.

Asad, Ahmed, Ahmad Azar, and Aboul Ella Hassanien (Oct. 2014). "A New Heuristic Function of Ant Colony System for Retinal Vessel Segmentation". In: *International Journal of Rough Sets and Data Analysis (IJRSDA)* 1, pp. 15–30. DOI: 10.4018/ijrsda.2014070102.

Aslani, Shahab and Haldun Sarnel (2016). "A new supervised retinal vessel segmentation method based on robust hybrid features". In: *Biomedical Signal Processing and Control* 30, pp. 1 –12. ISSN: 1746-8094. DOI: https://doi.org/10.1016/j.bspc.2016.05.006. URL: http://www.sciencedirect.com/science/article/pii/S1746809416300489.

Azzopardi, George and Nicolai Petkov (2013). "A Shape Descriptor Based on Trainable COSFIRE Filters for the Recognition of Handwritten Digits". In: *Computer Analysis of Images and Patterns*. Ed. by Richard Wilson et al. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 9–16. ISBN: 978-3-642-40246-3.

Bahdanau, Dzmitry, Kyunghyun Cho, and Yoshua Bengio (2014). *Neural Machine Translation by Jointly Learning to Align and Translate*. arXiv: 1409.0473 [cs.CL].

Bai, W. et al. (2013). "A Probabilistic Patch-Based Label Fusion Model for Multi-Atlas Segmentation With Registration Refinement: Application to Cardiac MR Images". In: *IEEE Transactions on Medical Imaging* 32.7, pp. 1302–1315. ISSN: 1558-254X. DOI: 10.1109/TMI.2013.2256922.

Bai, Wenjia et al. (2019). "Self-Supervised Learning for Cardiac MR Image Segmentation by Anatomical Position Prediction". In: *Medical Image Computing and Computer Assisted Intervention – MICCAI 2019*. Ed. by Dinggang Shen et al. Cham: Springer International Publishing, pp. 541–549. ISBN: 978-3-030-32245-8.

Bhuiyan, A. et al. (2007). "Blood Vessel Segmentation from Color Retinal Images using Unsupervised Texture Classification". In: *2007 IEEE International Conference on Image Processing*. Vol. 5, pp. V –521–V –524.

Bilal Khomri Argyrios Christodoulidis, Leila Djerou Mohamed Chaouki Babahenini Farida Cheriet (2018). "Retinal blood vessel segmentation using the elite-guided multi-objective artificial bee colony algorithm". English. In: *IET Image Processing* 12 (12), 2163–2171(8). ISSN: 1751-9659. URL: https://digital-library.theiet.org/content/journals/10.1049/iet-ipr.2018.5425.

Bishop, Christopher M. (1995). "Regularization and complexity control in feed-forward networks". International Conference on Artificial Neural Networks ICANN'95. URL: http://publications.aston.ac.uk/id/eprint/524/.

Bisong, Ekaba (2019). "Google Colaboratory". In: *Building Machine Learning and Deep Learning Models on Google Cloud Platform: A Comprehensive Guide for Beginners*. Berkeley, CA: Apress, pp. 59–64. ISBN: 978-1-4842-4470-8. DOI: 10.1007/978-1-4842-4470-8_7. URL: https://doi.org/10.1007/978-1-4842-4470-8_7.

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

Breiman, Leo (1996). "Bagging Predictors". In: *Machine Learning* 24.2, pp. 123–140.

Bruyninckx, Pieter et al. (2009). "Segmentation of lung vessel trees by global optimization". In: *Medical Imaging 2009: Image Processing*. Ed. by Josien P. W. Pluim and Benoit M. Dawant. Vol. 7259. International Society for Optics and Photonics. SPIE, pp. 367 –378. DOI: 10.1117/12.811570. URL: https://doi.org/10.1117/12.811570.

— (Mar. 2010). "Segmentation of liver portal veins by global optimization". In: *Proc SPIE*. DOI: 10.1117/12.843995.

Caruana, Rich (1997). "Multitask Learning". In: *Machine Learning* 28.1, pp. 41–75.

Chalakkal, R. J. and W. H. Abdulla (2018). "Improved Vessel Segmentation Using Curvelet Transform and Line Operators". In: *2018 Asia-Pacific Signal and Information Processing Association Annual Summit and Conference (APSIPA ASC)*, pp. 2041–2046.

Chan, T. F. and L. A. Vese (2001). "Active contours without edges". In: *IEEE Transactions on Image Processing* 10.2, pp. 266–277.

Charbonnier, Jean-Paul et al. (Nov. 2016). "Improving Airway Segmentation in Computed Tomography using Leak Detection with Convolutional Networks". In: *Medical Image Analysis* 36. DOI: 10.1016/j.media.2016.11.001.

Charbonnier, Jean-Paul et al. (2017). "Improving airway segmentation in computed tomography using leak detection with convolutional networks". In: *Medical image analysis* 36, 52—60. ISSN: 1361-8415. DOI: 10.1016/j.media.2016.11.001. URL: https://doi.org/10.1016/j.media.2016.11.001.

Chen, Jianxu et al. (2016). *Combining Fully Convolutional and Recurrent Neural Networks for 3D Biomedical Image Segmentation*. arXiv: 1609.01006 [cs.CV].

Cheung, N., T.Y. Wong, and L. Hodgson (Jan. 2009). "Retinal vascular changes as biomarkers of systemic cardiovascular diseases". In: pp. 185–220.

Cireşan, Dan C. et al. (2012). "Deep Neural Networks Segment Neuronal Membranes in Electron Microscopy Images". In: *Proceedings of the 25th International Conference*

on Neural Information Processing Systems - Volume 2. NIPS'12. Lake Tahoe, Nevada: Curran Associates Inc., 2843–2851.

Clark, Alex (2015). *Pillow (PIL Fork) Documentation*. URL: https://buildmedia.readthedocs.org/media/pdf/pillow/latest/pillow.pdf.

Cordella, L. P. et al. (1999). "Reliability Parameters to Improve Combination Strategies in Multi-Expert Systems". In: *Pattern Analysis & Applications* 2.3, pp. 205–214. DOI: 10.1007/s100440050029. URL: https://doi.org/10.1007/s100440050029.

Cybenko, G. (1989). "Approximation by superpositions of a sigmoidal function". In: *Mathematics of Control, Signals and Systems* 2.4, pp. 303–314.

Dai, Chengliang et al. (2019). "Transfer Learning from Partial Annotations for Whole Brain Segmentation". In: *Domain Adaptation and Representation Transfer and Medical Image Learning with Less Labels and Imperfect Data*. Ed. by Qian Wang et al. Cham: Springer International Publishing, pp. 199–206. ISBN: 978-3-030-33391-1.

Dai, Jifeng et al. (2017). *Deformable Convolutional Networks*. arXiv: 1703.06211 [cs.CV].

Dash, J. and N. Bhoi (2018). "Retinal blood vessel segmentation using Otsu thresholding with principal component analysis". In: *2018 2nd International Conference on Inventive Systems and Control (ICISC)*, pp. 933–937.

De Momi, Elena et al. (Apr. 2014). "Multi-trajectories automatic planner for StereoElectroEncephaloGraphy (SEEG)". In: *International journal of computer assisted radiology and surgery* 9. DOI: 10.1007/s11548-014-1004-1.

Dempster, A. P., N. M. Laird, and D. B. Rubin (1977). "Maximum Likelihood from Incomplete Data via the EM Algorithm". In: *Journal of the Royal Statistical Society. Series B (Methodological)* 39.1, pp. 1–38. ISSN: 00359246. URL: http://www.jstor.org/stable/2984875.

Deng, Jia et al. (2009). "Imagenet: A large-scale hierarchical image database". In: *2009 IEEE conference on computer vision and pattern recognition*. Ieee, pp. 248–255.

Descombes, X. et al. (2012). "Vascular network segmentation: An unsupervised approach". In: *2012 9th IEEE International Symposium on Biomedical Imaging (ISBI)*, pp. 1248–1251.

Dice, Lee R. (1945). "Measures of the Amount of Ecologic Association Between Species". In: *Ecology* 26.3, pp. 297–302. DOI: 10.2307/1932409. eprint: https://esajournals.onlinelibrary.wiley.com/doi/pdf/10.2307/1932409. URL: https://esajournals.onlinelibrary.wiley.com/doi/abs/10.2307/1932409.

Dougherty, Geoff (2009). *Digital Image Processing for Medical Applications*. Cambridge University Press. DOI: 10.1017/CBO9780511609657.

Duchi, John, Elad Hazan, and Yoram Singer (July 2011). "Adaptive Subgradient Methods for Online Learning and Stochastic Optimization". In: *Journal of Machine Learning Research* 12, pp. 2121–2159.

Ebner, Michael et al. (2020). "An automated framework for localization, segmentation and super-resolution reconstruction of fetal brain MRI". In: *NeuroImage* 206, p. 116324. ISSN: 1053-8119. DOI: https://doi.org/10.1016/j.neuroimage.2019.116324. URL: http://www.sciencedirect.com/science/article/pii/S1053811919309152.

Esteva, Andre et al. (Jan. 2017). "Dermatologist-level classification of skin cancer with deep neural networks". In: *Nature* 542. DOI: 10.1038/nature21056.

Fan, Shengyu et al. (2020). "Unsupervised Cerebrovascular Segmentation of TOF-MRA Images Based on Deep Neural Network and Hidden Markov Random Field Model". In: *Frontiers in Neuroinformatics* 13, p. 77. ISSN: 1662-5196. DOI: 10.3389/fninf.2019.00077. URL: https://www.frontiersin.org/article/10.3389/fninf.2019.00077.

Fan, Zhun et al. (2019). *Accurate Retinal Vessel Segmentation via Octave Convolution Neural Network*. arXiv: 1906.12193 [eess.IV].

Folkman, Judah (1995). "Angiogenesis in cancer, vascular, rheumatoid and other disease". In: *Nature Medicine* 1.1, pp. 27–30. DOI: 10.1038/nm0195-27. URL: https://doi.org/10.1038/nm0195-27.

From not working to neural networking (2016). https://www.economist.com/special-report/2016/06/23/from-not-working-to-neural-networking.

Fu, H. et al. (2016). "Retinal vessel segmentation via deep learning network and fully-connected conditional random fields". In: *2016 IEEE 13th International Symposium on Biomedical Imaging (ISBI)*, pp. 698–701.

Fu, Huazhu et al. (2016). "DeepVessel: Retinal Vessel Segmentation via Deep Learning and Conditional Random Field". In: *Medical Image Computing and Computer-Assisted Intervention – MICCAI 2016*. Ed. by Sebastien Ourselin et al. Cham: Springer International Publishing, pp. 132–139. ISBN: 978-3-319-46723-8.

Fu, Jun et al. (2018). *Dual Attention Network for Scene Segmentation*. arXiv: 1809.02983 [cs.CV].

Ganin, Yaroslav et al. (Jan. 2016). "Domain-Adversarial Training of Neural Networks". In: *J. Mach. Learn. Res.* 17.1, 2096–2030. ISSN: 1532-4435.

Garg, S., J. Sivaswamy, and S. Chandra (2007). "Unsupervised curvature-based retinal vessel segmentation". In: *2007 4th IEEE International Symposium on Biomedical Imaging: From Nano to Macro*, pp. 344–347.

GeethaRamani, R. and Lakshmi Balasubramanian (2016). "Retinal blood vessel segmentation employing image processing and data mining techniques for computerized retinal image analysis". In: *Biocybernetics and Biomedical Engineering* 36.1, pp. 102 –118. ISSN: 0208-5216. DOI: https://doi.org/10.1016/j.bbe.2015.06.004. URL: http://www.sciencedirect.com/science/article/pii/S020852161500042X.

Gegundez-Arias, M. E. et al. (2012). "A Function for Quality Evaluation of Retinal Vessel Segmentations". In: *IEEE Transactions on Medical Imaging* 31.2, pp. 231–239. ISSN: 1558-254X. DOI: 10.1109/TMI.2011.2167982.

Ghafoorian, Mohsen et al. (Apr. 2016). "Non-uniform patch sampling with deep convolutional neural networks for white matter hyperintensity segmentation". In: pp. 1414–1417. DOI: 10.1109/ISBI.2016.7493532.

Ghazal, M., Y. Al Khalil, and A. El-Baz (2018). "An Unsupervised Parametric Mixture Model for Automatic Cerebrovascular Segmentation". In: *Cardiovascular Imaging and Image Analysis*. CRC Press. Chap. 5.

Goceri, Evgin, Zarine Shah, and Metin Gurcan (June 2016). "Vessel Segmentation From Abdominal MR Images: Adaptive and Reconstructive Approach". In: *International Journal for Numerical Methods in Biomedical Engineering* 33, e02811. DOI: 10.1002/cnm.2811.

Goldbaum, Michael (1975). *The STARE Project- STructured Analysis of the REtina*. https://cecas.clemson.edu/~ahoover/stare/.

Goodfellow, Ian, Yoshua Bengio, and Aaron Courville (2016). *Deep Learning*. http://www.deeplearningbook.org. MIT Press.

Goodfellow, Ian et al. (2014). "Generative Adversarial Nets". In: *Advances in Neural Information Processing Systems 27*. Ed. by Z. Ghahramani et al. Curran Associates, Inc., pp. 2672–2680. URL: http://papers.nips.cc/paper/5423-generative-adversarial-nets.pdf.

Goodfellow, Ian J. et al. (2013). "Maxout Networks". In: *Proceedings of the 30th International Conference on International Conference on Machine Learning - Volume 28*.

ICML'13. Atlanta, GA, USA: JMLR.org, pp. III–1319–III–1327. URL: http://dl.acm.org/citation.cfm?id=3042817.3043084.

Grinsven, Mark J. J. P. van et al. (2016). "Fast Convolutional Neural Network Training Using Selective Data Sampling: Application to Hemorrhage Detection in Color Fundus Images." In: *IEEE Trans. Med. Imaging* 35.5, pp. 1273–1284. URL: http://dblp.uni-trier.de/db/journals/tmi/tmi35.html#GrinsvenGHTS16.

Guo, Changlu et al. (2020). "SA-UNet: Spatial Attention U-Net for Retinal Vessel Segmentation". In: *ArXiv* abs/2004.03696.

Gupta, Dishashree (2020). *Fundamentals of Deep Learning – Activation Functions and When to Use Them?* https://www.analyticsvidhya.com/blog/2020/01/fundamentals-deep-learning-activation-functions-when-to-use-them/.

Gur, Shir et al. (2019). "Unsupervised Microvascular Image Segmentation Using an Active Contours Mimicking Neural Network". In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*.

Hassouna, M. Sabry et al. (2006). "Cerebrovascular segmentation from TOF using stochastic models". In: *Medical Image Analysis* 10.1, pp. 2 –18. ISSN: 1361-8415. URL: http://www.sciencedirect.com/science/article/pii/S1361841505000186.

He, Kaiming et al. (2015). *Deep Residual Learning for Image Recognition*. arXiv: 1512.03385 [cs.CV].

Hesamian, Mohammad Hesam et al. (2019). "Deep Learning Techniques for Medical Image Segmentation: Achievements and Challenges". In: *Journal of Digital Imaging* 32.4, pp. 582–596. DOI: 10.1007/s10278-019-00227-x. URL: https://doi.org/10.1007/s10278-019-00227-x.

Hinton, Geoffrey E. et al. (2012). *Improving neural networks by preventing co-adaptation of feature detectors*. arXiv: 1207.0580 [cs.NE].

Hore, Prodip and Sayan Chatterjee (2019). *A Comprehensive Guide to Attention Mechanism in Deep Learning for Everyone*.

Hu, J. et al. (2019). "S-UNet: A Bridge-Style U-Net Framework With a Saliency Mechanism for Retinal Vessel Segmentation". In: *IEEE Access* 7, pp. 174167–174177.

Hunter, D. et al. (2012). "Selection of proper Neural Network sizes and architectures:A comparative study". In: *IEEE Transaction on Industrial Informatics* 8.2, 228–240.

Jadon, Shruti (2020). *A survey of loss functions for semantic segmentation*. arXiv: 2006.14822 [eess.IV].

Jia, Yangqing et al. (2014). *Caffe: Convolutional Architecture for Fast Feature Embedding*. arXiv: 1408.5093 [cs.CV].

Jiang, Yun et al. (2019). "Automatic Retinal Blood Vessel Segmentation Based on Fully Convolutional Neural Networks". In: *Symmetry* 11.9. ISSN: 2073-8994. DOI: 10.3390/sym11091112. URL: https://www.mdpi.com/2073-8994/11/9/1112.

Jiang, Zhexin et al. (2018a). "Retinal blood vessel segmentation using fully convolutional network with transfer learning". In: *Computerized Medical Imaging and Graphics* 68, pp. 1 –15. ISSN: 0895-6111. DOI: https://doi.org/10.1016/j.compmedimag.2018.04.005. URL: http://www.sciencedirect.com/science/article/pii/S0895611118302313.

— (2018b). "Retinal blood vessel segmentation using fully convolutional network with transfer learning". In: *Computerized Medical Imaging and Graphics* 68, pp. 1 –15. ISSN: 0895-6111. DOI: https://doi.org/10.1016/j.compmedimag.2018.04.005. URL: http://www.sciencedirect.com/science/article/pii/S0895611118302313.

— (Apr. 2018c). "Retinal blood vessel segmentation using fully convolutional network with transfer learning". In: *Computerized Medical Imaging and Graphics* 68. DOI: 10.1016/j.compmedimag.2018.04.005.

Jin, Qiangguo et al. (2019). "DUNet: A deformable network for retinal vessel segmentation". In: *Knowledge-Based Systems* 178, 149–162. ISSN: 0950-7051. DOI: 10.1016/j.knosys.2019.04.025. URL: http://dx.doi.org/10.1016/j.knosys.2019.04.025.

Kamnitsas, Konstantinos et al. (2017). "Efficient multi-scale 3D CNN with fully connected CRF for accurate brain lesion segmentation". In: *Medical Image Analysis* 36, pp. 61 –78. ISSN: 1361-8415. DOI: https://doi.org/10.1016/j.media.2016.10.004. URL: http://www.sciencedirect.com/science/article/pii/S1361841516301839.

Kande, Giri Babu, P. Venkata Subbaiah, and T. Satya Savithri (2010). "Unsupervised Fuzzy Based Vessel Segmentation In Pathological Digital Fundus Images". In: *Journal of Medical Systems* 34.5, pp. 849–858. DOI: 10.1007/s10916-009-9299-0. URL: https://doi.org/10.1007/s10916-009-9299-0.

Kelley, Henry J (1960). "Gradient theory of optimal flight paths". In: *Ars Journal* 30.10, pp. 947–954.

Khan, Asifullah et al. (2020). "A survey of the recent architectures of deep convolutional neural networks". In: *Artificial Intelligence Review*.

Kingma, Diederik and Jimmy Ba (Dec. 2014). "Adam: A Method for Stochastic Optimization". In: *International Conference on Learning Representations*.

Kooi, Thijs et al. (Aug. 2016). "Large Scale Deep Learning for Computer Aided Detection of Mammographic Lesions". In: *Medical Image Analysis* 35. DOI: 10.1016/j.media.2016.07.007.

Kratzert, F. et al. (2018). "Rainfall-Runoff modelling using Long-Short-Term-Memory (LSTM) networks". In: *Hydrology and Earth System Sciences Discussions* 2018, pp. 1–26. DOI: 10.5194/hess-2018-247. URL: https://www.hydrol-earth-syst-sci-discuss.net/hess-2018-247/.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). "ImageNet Classification with Deep Convolutional Neural Networks". In: *Advances in Neural Information Processing Systems 25*. Ed. by F. Pereira et al. Curran Associates, Inc., pp. 1097–1105. URL: http://papers.nips.cc/paper/4824-imagenet-classification-with-deep-convolutional-neural-networks.pdf.

Köhler, T. et al. (2013). "Automatic no-reference quality assessment for retinal fundus images using vessel segmentation". In: *Proceedings of the 26th IEEE International Symposium on Computer-Based Medical Systems*, pp. 95–100.

Lahiri, A. et al. (2016). "Deep neural ensemble for retinal vessel segmentation in fundus images towards achieving label-free angiography". In: *2016 38th Annual International Conference of the IEEE Engineering in Medicine and Biology Society (EMBC)*, pp. 1340–1343.

Lahiri, A. et al. (2017). "Generative Adversarial Learning for Reducing Manual Annotation in Semantic Segmentation on Large Scale Miscroscopy Images: Automated Vessel Segmentation in Retinal Fundus Image as Test Case". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pp. 794–800.

Lahiri, Avisek et al. (2018). *Retinal Vessel Segmentation under Extreme Low Annotation: A Generative Adversarial Network Approach*. arXiv: 1809.01348 [cs.CV].

Lam, B. S. Y. and H. Yan (2008). "A Novel Vessel Segmentation Algorithm for Pathological Retina Images Based on the Divergence of Vector Fields". In: *IEEE Transactions on Medical Imaging* 27.2, pp. 237–246.

LeCun, Yann and Yoshua Bengio (1998). "Convolutional Networks for Images, Speech, and Time Series". In: *The Handbook of Brain Theory and Neural Networks*. Cambridge, MA, USA: MIT Press, 255–258. ISBN: 0262511029.

Lecun, Yann et al. (1998). "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE*, pp. 2278–2324.

Lee, Lay, Siau-Chuin Liew, and Weng Jie Thong (Nov. 2015). "A Review of Image Segmentation Methodologies in Medical Image". In: *Lecture Notes in Electrical Engineering* 315, pp. 1069–1080. DOI: 10.1007/978-3-319-07674-4_99.

Lenchik, Leon et al. (2019). "Automated Segmentation of Tissues Using CT and MRI: A Systematic Review". In: *Academic Radiology* 26.12, pp. 1695 –1706. ISSN: 1076-6332. DOI: https://doi.org/10.1016/j.acra.2019.07.006. URL: http://www.sciencedirect.com/science/article/pii/S1076633219303538.

Li, Fei-Fei (2019). *Stanford CS class CS231n: Convolutional Neural Networks for Visual Recognition notes*. https://cs231n.github.io/convolutional-networks/#add.

Li, Hanchao et al. (2018). *Pyramid Attention Network for Semantic Segmentation*. arXiv: 1805.10180 [cs.CV].

Li, Liangzhi et al. (2019). *IterNet: Retinal Image Segmentation Utilizing Structural Redundancy in Vessel Networks*. arXiv: 1912.05763 [eess.IV].

Li, Q. et al. (2016). "A Cross-Modality Learning Approach for Vessel Segmentation in Retinal Images". In: *IEEE Transactions on Medical Imaging* 35.1, pp. 109–118.

Li, T., M. Comer, and J. Zerubia (2018). "A Connected-Tube MPP Model for Object Detection with Application to Materials and Remotely-Sensed Images". In: *2018 25th IEEE International Conference on Image Processing (ICIP)*, pp. 1323–1327.

— (2020). "An Unsupervised Retinal Vessel Extraction and Segmentation Method Based On a Tube Marked Point Process Model". In: *ICASSP 2020 - 2020 IEEE International Conference on Acoustics, Speech and Signal Processing (ICASSP)*, pp. 1394–1398.

Lin, Tsung-Yi et al. (2017). *Focal Loss for Dense Object Detection*. arXiv: 1708.02002 [cs.CV].

Litjens, Geert et al. (2017a). "A survey on deep learning in medical image analysis". In: *Medical Image Analysis* 42, pp. 60 –88. ISSN: 1361-8415. DOI: https://doi.org/10.1016/j.media.2017.07.005. URL: http://www.sciencedirect.com/science/article/pii/S1361841517301135.

Litjens, Geert et al. (Feb. 2017b). "A Survey on Deep Learning in Medical Image Analysis". In: *Medical Image Analysis* 42. DOI: 10.1016/j.media.2017.07.005.

Liu, Bo, Lin Gu, and Feng Lu (2019). "Unsupervised Ensemble Strategy for Retinal Vessel Segmentation". In: *Medical Image Computing and Computer Assisted Intervention – MICCAI 2019*. Ed. by Dinggang Shen et al. Cham: Springer International Publishing, pp. 111–119. ISBN: 978-3-030-32239-7.

Liu, Yihao et al. (2020). "Variational intensity cross channel encoder for unsupervised vessel segmentation on OCT angiography". In: *Medical Imaging 2020: Image Processing*. Ed. by Ivana Išgum and Bennett A. Landman. Vol. 11313. International Society for Optics and Photonics. SPIE, pp. 206 –212. DOI: 10.1117/12.2549967. URL: https://doi.org/10.1117/12.2549967.

Liu, Ze-Fan et al. (2018). "Retinal Vessel Segmentation Using Densely Connected Convolution Neural Network with Colorful Fundus Images". In: *Journal of Medical Imaging and Health Informatics* 8.6, pp. 1300–1307. ISSN: 2156-7018. DOI: doi:10.1166/jmihi.2018.2429. URL: https://www.ingentaconnect.com/content/asp/jmihi/2018/00000008/00000006/art00028.

Livne, Michelle et al. (2019). "A U-Net Deep Learning Framework for High Performance Vessel Segmentation in Patients With Cerebrovascular Disease". In: *Frontiers in Neuroscience* 13, p. 97. ISSN: 1662-453X. DOI: 10.3389/fnins.2019.00097. URL: https://www.frontiersin.org/article/10.3389/fnins.2019.00097.

Long, Jonathan, Evan Shelhamer, and Trevor Darrell (2014). *Fully Convolutional Networks for Semantic Segmentation*. arXiv: 1411.4038 [cs.CV].

Luo, Z. et al. (2019). "Micro-Vessel Image Segmentation Based on the AD-UNet Model". In: *IEEE Access* 7, pp. 143402–143411.

Lupaşcu, Carmen Alina and Domenico Tegolo (2011). "Automatic Unsupervised Segmentation of Retinal Vessels Using Self-Organizing Maps and K-Means Clustering". In: *Computational Intelligence Methods for Bioinformatics and Biostatistics*. Ed. by Riccardo Rizzo and Paulo J. G. Lisboa. Berlin, Heidelberg: Springer Berlin Heidelberg, pp. 263–274. ISBN: 978-3-642-21946-7.

Lupascu, Carmen Alina and Domenico Tegolo (2016). "A Multiscale Approach to Automatic and Unsupervised Retinal Vessel Segmentation Using Self-Organizing Maps". In: *Proceedings of the 17th International Conference on Computer Systems and Technologies 2016*. CompSysTech '16. Palermo, Italy: Association for Computing Machinery, 182–189. ISBN: 9781450341820. DOI: 10.1145/2983468.2983478. URL: https://doi.org/10.1145/2983468.2983478.

Ma, Wenao et al. (Sept. 2019). "Multi-Task Neural Networks with Spatial Activation for Retinal Vessel Segmentation and Artery/Vein Classification". In:

Ma Yi-de, Liu Qing, and Qian Zhi-bai (2004). "Automated image segmentation using improved PCNN model based on cross-entropy". In: *Proceedings of 2004 International Symposium on Intelligent Multimedia, Video and Speech Processing, 2004.* Pp. 743–746.

Mai, Keith Ka Ki et al. (2019). "Electron Tomography Analysis of Thylakoid Assembly and Fission in Chloroplasts of a Single-Cell C4 plant, Bienertia sinuspersici". In: *Scientific Reports* 9.1, p. 19640.

Mapayi, Temitope, Jules-Raymond Tapamo, and Serestina Viriri (Sept. 2015). "Retinal Vessel Segmentation: A Comparative Study of Fuzzy C-Means and Sum Entropy Information on Phase Congruency". In: *International Journal of Advanced Robotic Systems* 12. DOI: 10.5772/60581.

Maulik, Ujjwal and Sanghamitra Bandyopadhyay (2000). "Genetic algorithm-based clustering technique". In: *Pattern Recognition* 33.9, pp. 1455 –1465. ISSN: 0031-3203. DOI: https://doi.org/10.1016/S0031-3203(99)00137-5. URL: http://www.sciencedirect.com/science/article/pii/S0031320399001375.

McDonald, Donald M. and Peter Baluk (2002). "Significance of Blood Vessel Leakiness in Cancer". In: *Cancer Research* 62.18, pp. 5381–5385. ISSN: 0008-5472. eprint: https://cancerres.aacrjournals.org/content/62/18/5381.full.pdf. URL: https://cancerres.aacrjournals.org/content/62/18/5381.

Memari, Nogol et al. (2019). "Retinal Blood Vessel Segmentation by Using Matched Filtering and Fuzzy C-means Clustering with Integrated Level Set Method for Diabetic Retinopathy Assessment". In: *Journal of Medical and Biological Engineering* 39.5, pp. 713–731.

Menze, B. H. et al. (2015). "The Multimodal Brain Tumor Image Segmentation Benchmark (BRATS)". In: *IEEE Transactions on Medical Imaging* 34.10, pp. 1993–2024. ISSN: 1558-254X. DOI: 10.1109/TMI.2014.2377694.

Meyer-Baese, Anke and Volker Schmid (2014). "Chapter 13 - Computer-Aided Diagnosis for Diagnostically Challenging Breast Lesions in DCE-MRI". In: *Pattern Recognition and Signal Analysis in Medical Imaging (Second Edition)*. Ed. by Anke Meyer-Baese and Volker Schmid. Second Edition. Oxford: Academic Press, pp. 391 –420. ISBN: 978-0-12-409545-8. DOI: https://doi.org/10.1016/B978-0-12-

409545-8.00013-3. URL: http://www.sciencedirect.com/science/article/pii/B9780124095458000133.

Milletari, F., N. Navab, and S. Ahmadi (2016). "V-Net: Fully Convolutional Neural Networks for Volumetric Medical Image Segmentation". In: *2016 Fourth International Conference on 3D Vision (3DV)*, pp. 565–571. DOI: 10.1109/3DV.2016.79.

Miri, Maliheh et al. (2017). "A Comprehensive Study of Retinal Vessel Classification Methods in Fundus Images". In: *Journal of medical signals and sensors* 7.2, pp. 59–70. URL: https://pubmed.ncbi.nlm.nih.gov/28553578.

Mitchell, Thomas M. (1997). *Machine Learning*. 1st ed. USA: McGraw-Hill, Inc. ISBN: 0070428077.

Moccia, Sara et al. (2018a). "Blood vessel segmentation algorithms — Review of methods, datasets and evaluation metrics". In: *Computer Methods and Programs in Biomedicine* 158, pp. 71 –91. ISSN: 0169-2607. DOI: https://doi.org/10.1016/j.cmpb.2018.02.001. URL: http://www.sciencedirect.com/science/article/pii/S0169260717313421.

— (2018b). "Blood vessel segmentation algorithms — Review of methods, datasets and evaluation metrics". In: *Computer Methods and Programs in Biomedicine* 158, pp. 71 –91. ISSN: 0169-2607. DOI: https://doi.org/10.1016/j.cmpb.2018.02.001. URL: http://www.sciencedirect.com/science/article/pii/S0169260717313421.

Molga, E.J. (2003). "Neural network approach to support modeling of chemical reactors: problems, resolutions, criteria of application". In: *Chemical Engineering and Processing: Process Intensification* 42.8–9, 675–695.

Nekovei, R. and Ying Sun (1995). "Back-propagation network and its configuration for blood vessel detection in angiograms". In: *IEEE Transactions on Neural Networks* 6.1, pp. 64–72. ISSN: 1941-0093. DOI: 10.1109/72.363449.

Neto], Luiz [Câmara et al. (2017). "An unsupervised coarse-to-fine algorithm for blood vessel segmentation in fundus images". In: *Expert Systems with Applications* 78, pp. 182 –192. ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2017.02.015. URL: http://www.sciencedirect.com/science/article/pii/S0957417417300970.

Niu, Ruigang et al. (2020). *HMANet: Hybrid Multiple Attention Network for Semantic Segmentation in Aerial Images*. arXiv: 2001.02870 [cs.CV].

Noh, Hyeonwoo, Seunghoon Hong, and Bohyung Han (2015). *Learning Deconvolution Network for Semantic Segmentation*. arXiv: 1505.04366 [cs.CV].

Nowińska, Anna, Zafer Yavuz, and Cemal Köse (2017). "Blood Vessel Extraction in Color Retinal Fundus Images with Enhancement Filtering and Unsupervised Classification". In: *Journal of Healthcare Engineering* 2017, p. 4897258. DOI: 10.1155/2017/4897258. URL: https://doi.org/10.1155/2017/4897258.

Oktay, O. et al. (2018). "Anatomically Constrained Neural Networks (ACNNs): Application to Cardiac Image Enhancement and Segmentation". In: *IEEE Transactions on Medical Imaging* 37.2, pp. 384–395. ISSN: 1558-254X. DOI: 10.1109/TMI.2017.2743464.

Oktay, Ozan et al. (2018). *Attention U-Net: Learning Where to Look for the Pancreas*. arXiv: 1804.03999 [cs.CV].

Oliveira, Dário AB, Raul Q. Feitosa, and Mauro M. Correia (2011). "Segmentation of liver, its vessels and lesions from CT images for surgical planning". In: *BioMedical Engineering OnLine* 10.1, p. 30. DOI: 10.1186/1475-925X-10-30. URL: https://doi.org/10.1186/1475-925X-10-30.

Oliveira, W. S., T. I. Ren, and G. D. C. Cavalcanti (2012a). "An Unsupervised Segmentation Method for Retinal Vessel Using Combined Filters". In: *2012 IEEE 24th International Conference on Tools with Artificial Intelligence*. Vol. 1, pp. 750–756.

— (2012b). "An Unsupervised Segmentation Method for Retinal Vessel Using Combined Filters". In: *2012 IEEE 24th International Conference on Tools with Artificial Intelligence*. Vol. 1, pp. 750–756.

Orlando, J. I., E. Prokofyeva, and M. B. Blaschko (2017). "A Discriminatively Trained Fully Connected Conditional Random Field Model for Blood Vessel Segmentation in Fundus Images". In: *IEEE Transactions on Biomedical Engineering* 64.1, pp. 16–27.

Owen, Christopher G. et al. (2009). "Measuring retinal vessel tortuosity in 10-year-old children: validation of the Computer-Assisted Image Analysis of the Retina (CAIAR) program." In: *Investigative ophthalmology visual science* 50 5, pp. 2004–10.

Paszke, Adam et al. (2019). "PyTorch: An Imperative Style, High-Performance Deep Learning Library". In: *Advances in Neural Information Processing Systems 32*. Ed. by H. Wallach et al. Curran Associates, Inc., pp. 8024–8035. URL: http://papers.neurips.cc/paper/9015-pytorch-an-imperative-style-high-performance-deep-learning-library.pdf.

Pendharkar, P.C. and J.A. Rodger (2003). "Technical efficiency based selection of learning cases to improve the forecasting efficiency of neural networks under monotonicity assumption". In: *Decision Support Systems* 36.1, 117–136.

Pienaar, Etienne A.D. (2018). *Introduction to Neural Networks*.

Pizer, Stephen M. et al. (1987). "Adaptive histogram equalization and its variations". In: *Computer Vision, Graphics, and Image Processing* 39.3, pp. 355 –368. ISSN: 0734-189X. DOI: https://doi.org/10.1016/S0734-189X(87)80186-X. URL: http://www.sciencedirect.com/science/article/pii/S0734189X8780186X.

Poudel, Rudra P K, Pablo Lamata, and Giovanni Montana (2016). *Recurrent Fully Convolutional Neural Networks for Multi-slice MRI Cardiac Segmentation*. arXiv: 1608.03974 [stat.ML].

Powers, David (Jan. 2008). "Evaluation: From Precision, Recall and F-Factor to ROC, Informedness, Markedness Correlation". In: *Mach. Learn. Technol.* 2.

Rajan, S. Palanivel and V. Kavitha (2017). "Diagnosis of Cardiovascular Diseases Using Retinal Images Through Vessel Segmentation Graph". In: *Current Medical Imaging Reviews* 13.4, pp. 454–459. ISSN: 1573-4056. DOI: doi:10.2174/1573405613666170111153207. URL: https://www.ingentaconnect.com/content/ben/cmir/2017/00000013/00000004/art00013.

Ramakonar, Hari et al. (Dec. 2018). "Intraoperative detection of blood vessels with an imaging needle during neurosurgery in humans". In: *Science advances* 4.12, eaav4992–eaav4992. DOI: 10.1126/sciadv.aav4992. URL: https://pubmed.ncbi.nlm.nih.gov/30585293.

Rocha, Douglas Abreu da et al. (2020). "An unsupervised approach to improve contrast and segmentation of blood vessels in retinal images using CLAHE, 2D Gabor wavelet, and morphological operations". In: *Research on Biomedical Engineering* 36.1, pp. 67–75.

Rodrigues, Pedro et al. (Dec. 2013). "Two-dimensional segmentation of the retinal vascular network from optical coherence tomography". In: *Journal of biomedical optics* 18, p. 126011. DOI: 10.1117/1.JBO.18.12.126011.

Ronneberger, Olaf, Philipp Fischer, and Thomas Brox (2015). *U-Net: Convolutional Networks for Biomedical Image Segmentation*. arXiv: 1505.04597 [cs.CV].

Roychowdhury, S., D. D. Koozekanani, and K. K. Parhi (2015). "Blood Vessel Segmentation of Fundus Images by Major Vessel Extraction and Subimage Classification". In: *IEEE Journal of Biomedical and Health Informatics* 19.3, pp. 1118–1128.

Ruderman, Avraham et al. (2018). *Pooling is neither necessary nor sufficient for appropriate deformation stability in CNNs*. arXiv: 1804.04438 [cs.CV].

Rumelhart, David E., Geoffrey E. Hinton, and Ronald J. Williams (1986a). "Learning representations by back-propagating errors". In: *Nature* 323.6088, pp. 533–536.

— (1986b). "Learning representations by back-propagating errors". In: *Nature* 323.6088, pp. 533–536.

Rundo, Leonardo et al. (2020). "Tissue-specific and interpretable sub-segmentation of whole tumour burden on CT images by unsupervised fuzzy clustering". In: *Computers in Biology and Medicine* 120, p. 103751. ISSN: 0010-4825. DOI: https://doi.org/10.1016/j.compbiomed.2020.103751. URL: http://www.sciencedirect.com/science/article/pii/S0010482520301293.

S. Mohamed, Ihab (Sept. 2017). "Detection and Tracking of Pallets using a Laser Rangefinder and Machine Learning Techniques". PhD thesis. DOI: 10.13140/RG.2.2.30795.69926.

Santhosh Krishna, B. V., T. Gnanasekaran, and S. Aswini (2018). "Unsupervised Morphological Approach for Retinal Vessel Segmentation". In: *Progress in Computing, Analytics and Networking*. Ed. by Prasant Kumar Pattnaik et al. Singapore: Springer Singapore, pp. 743–752. ISBN: 978-981-10-7871-2.

Sateesh, Srinivas (2018). *Have you asked why F1-Score is a Harmonic Mean(HM) of Precision and Recall?* https://medium.com/@srinivas.sateesh/have-you-asked-why-f1-score-is-a-harmonic-mean-hm-of-precision-and-recall-febc233ce247.

Savelli, B. et al. (2017). "Illumination Correction by Dehazing for Retinal Vessel Segmentation". In: *2017 IEEE 30th International Symposium on Computer-Based Medical Systems (CBMS)*, pp. 219–224.

Schlemper, Jo et al. (2018). *Attention Gated Networks: Learning to Leverage Salient Regions in Medical Images*. arXiv: 1808.08114 [cs.CV].

— (2019). "Attention gated networks: Learning to leverage salient regions in medical images". In: *Medical Image Analysis* 53, pp. 197 –207. ISSN: 1361-8415. DOI: https://doi.org/10.1016/j.media.2019.01.012. URL: http://www.sciencedirect.com/science/article/pii/S1361841518306133.

Shah, S. A. A. et al. (2019). "Unsupervised Method for Retinal Vessel Segmentation Based on Gabor Wavelet and Multiscale Line Detector". In: *IEEE Access* 7, pp. 167221–167228.

Shalaby, A. et al. (2018). "Accurate Unsupervised 3D Segmentation of Blood Vessels Using Magnetic Resonance Angiography". In: *Cardiovascular Imaging and Image Analysis*. CRC Press. Chap. 4.

Shapey, Jonathan et al. (2019). "An artificial intelligence framework for automatic segmentation and volumetry of vestibular schwannomas from contrast-enhanced T1-weighted and high-resolution T2-weighted MRI". In: *Journal of Neurosurgery JNS*, pp. 1 –9. URL: https://thejns.org/view/journals/j-neurosurg/aop/article-10.3171-2019.9.JNS191949/article-10.3171-2019.9.JNS191949.xml.

Sharma, Neeraj and Lalit M Aggarwal (Jan. 2010). "Automated medical image segmentation techniques". In: *Journal of medical physics* 35.1, pp. 3–14. DOI: 10.4103/0971-6203.58777. URL: https://pubmed.ncbi.nlm.nih.gov/20177565.

Shin, Seung Yeon et al. (2019). "Deep vessel segmentation by learning graphical connectivity". In: *Medical Image Analysis* 58, p. 101556. ISSN: 1361-8415. DOI: https://doi.org/10.1016/j.media.2019.101556. URL: http://www.sciencedirect.com/science/article/pii/S1361841519300982.

Shorten, Connor and Taghi M. Khoshgoftaar (2019). "A survey on Image Data Augmentation for Deep Learning". In: *Journal of Big Data* 6.1, p. 60.

Shrivastava, Neeraj and Jyoti Bharti (2019). "A Comparative Analysis of Medical Image Segmentation". In: *International Conference on Advanced Computing Networking and Informatics*. Ed. by Raj Kamal, Michael Henshaw, and Pramod S. Nair. Singapore: Springer Singapore, pp. 459–467. ISBN: 978-981-13-2673-8.

Simonyan, Karen and Andrew Zisserman (2014). *Very Deep Convolutional Networks for Large-Scale Image Recognition*. arXiv: 1409.1556 [cs.CV].

Sinha, Ashish and Jose Dolz (2019). *Multi-scale self-guided attention for medical image segmentation*. arXiv: 1906.02849 [cs.CV].

Smistad, Erik and Lasse Lovstakken (Oct. 2016). "Vessel Detection in Ultrasound Images Using Deep Convolutional Neural Networks". In: DOI: 10.1007/978-3-319-46976-8_4.

Snuverink, I.A.F. (2017). "Deep Learning for Pixelwise Classification of Hyperspectral Images". PhD thesis.

Soares, J. V. B. et al. (2006). "Retinal vessel segmentation using the 2-D Gabor wavelet and supervised classification". In: *IEEE Transactions on Medical Imaging* 25.9, pp. 1214–1222. ISSN: 1558-254X. DOI: 10.1109/TMI.2006.879967.

Son, Jaemin, Sang Jun Park, and Kyu-Hwan Jung (2017). *Retinal Vessel Segmentation in Fundoscopic Images with Generative Adversarial Networks*. arXiv: 1706.09318 [cs.CV].

Springenberg, Jost Tobias et al. (2014). *Striving for Simplicity: The All Convolutional Net*. arXiv: 1412.6806 [cs.LG].

Sreejini, K.S. and V.K. Govindan (2015). "Improved multiscale matched filter for retina vessel segmentation using PSO algorithm". In: *Egyptian Informatics Journal* 16.3, pp. 253 –260. ISSN: 1110-8665. DOI: https://doi.org/10.1016/j.eij.2015.06.004. URL: http://www.sciencedirect.com/science/article/pii/S111086651500033X.

Srivastava, Nitish et al. (2014). "Dropout: A Simple Way to Prevent Neural Networks from Overfitting". In: *Journal of Machine Learning Research* 15.56, pp. 1929–1958. URL: http://jmlr.org/papers/v15/srivastava14a.html.

Staal, J. et al. (2004a). "Ridge-based vessel segmentation in color images of the retina". In: *IEEE Transactions on Medical Imaging* 23.4, pp. 501–509. ISSN: 1558-254X. DOI: 10.1109/TMI.2004.825627.

— (2004b). "Ridge-based vessel segmentation in color images of the retina". In: *IEEE Transactions on Medical Imaging* 23.4, pp. 501–509.

— (2004c). "Ridge-based vessel segmentation in color images of the retina". In: *IEEE Transactions on Medical Imaging* 23.4, pp. 501–509.

Stockman, George and Linda G. Shapiro (2001). *Computer Vision*. 1st. USA: Prentice Hall PTR. ISBN: 0130307963.

Stollenga, Marijn et al. (June 2015). "Parallel Multi-Dimensional LSTM, With Application to Fast Biomedical Volumetric Image Segmentation". In:

Strisciuglio, Nicola et al. (Oct. 2015). "Unsupervised delineation of the vessel tree in retinal fundus images". In: DOI: 10.1201/b19241-26.

Sudre, Carole H. et al. (2017). "Generalised Dice Overlap as a Deep Learning Loss Function for Highly Unbalanced Segmentations". In: *Lecture Notes in Computer Science*, 240–248. ISSN: 1611-3349. DOI: 10.1007/978-3-319-67558-9_28. URL: http://dx.doi.org/10.1007/978-3-319-67558-9_28.

Sulaiman, Hamzah Asyrani et al. (2016). *Advanced Computer and Communication Engineering Technology: Proceedings of the 1st International Conference on Communication and Computer Engineering*. 1st. Springer Publishing Company, Incorporated. ISBN: 3319384163.

Szegedy, Christian et al. (2014). *Going Deeper with Convolutions*. arXiv: 1409.4842 [cs.CV].

Szpak, Zygmunt and Jules-Raymond Tapamo (Jan. 2008). "Automatic and Interactive Retinal Vessel Segmentation". In: *South African Computer Journal* 40, pp. 23 –30.

Tajbakhsh, Nima et al. (2019). *Embracing Imperfect Datasets: A Review of Deep Learning Solutions for Medical Image Segmentation*. arXiv: 1908.10454 [eess.IV].

Tamura, S. and M. Tateishi (1997). "Capabilities of a four-layered feedforward neural network: four layers versus three". In: *IEEE Transactions on Neural Networks* 8.2, 251–255.

Tao, Andrew, Karan Sapra, and Bryan Catanzaro (2020). *Hierarchical Multi-Scale Attention for Semantic Segmentation*. arXiv: 2005.10821 [cs.CV].

Tavakoli, M. et al. (2017). "Automated Fovea Detection Based on Unsupervised Retinal Vessel Segmentation Method". In: *2017 IEEE Nuclear Science Symposium and Medical Imaging Conference (NSS/MIC)*, pp. 1–7.

Tavares, Joao and Renato Natal Jorge (Jan. 2009). "A Review on the Current Segmentation Algorithms for Medical Images." In: pp. 135–140.

Tchircoff, Andrew (2017). *The mostly complete chart of Neural Networks, explained*. https://towardsdatascience.com/the-mostly-complete-chart-of-neural-networks-explained-3fb6f2367464. Accessed: 2018-09-01.

Tetteh, Giles et al. (2017). "Deep-FExt: Deep Feature Extraction for Vessel Segmentation and Centerline Prediction". In: *Machine Learning in Medical Imaging*. Ed. by Qian Wang et al. Cham: Springer International Publishing, pp. 344–352. ISBN: 978-3-319-67389-9.

Tetteh, Giles et al. (2018). *DeepVesselNet: Vessel Segmentation, Centerline Prediction, and Bifurcation Detection in 3-D Angiographic Volumes*. arXiv: 1803.09340 [cs.CV].

Tieleman, T. and G. Hinton (2012). *Lecture 6.5 - RMSProp*.

Tiu, Ekin (2019). *Metrics to Evaluate your Semantic Segmentation Model*. https://towardsdatascience.com/metrics-to-evaluate-your-semantic-segmentation-model.

Tsang, Sik-Ho (2018). *Review: FCN — Fully Convolutional Network (Semantic Segmentation)*.

van Opbroek, Annegreet, Fedde van der Lijn, and Marleen de Bruijne (2013). "Automated brain-tissue segmentation by multi-feature SVM classification". English.

In: *The MICCAI Grand Challenge on MR Brain Image Segmentation (MRBrainS13)*. The MIDAS Journal. OA; The MICCAI Grand Challenge on MR Brain Image Segmentation, MRBrainS13 ; Conference date: 26-09-2013 Through 26-09-2013.

Vlachos, Marios and Evangelos Dermatas (2010). "Multi-scale retinal vessel segmentation using line tracking". In: *Computerized Medical Imaging and Graphics* 34.3, pp. 213 –227. ISSN: 0895-6111. DOI: https://doi.org/10.1016/j.compmedimag.2009.09.006. URL: http://www.sciencedirect.com/science/article/pii/S0895611109001177.

Vswitchs (2020). *Adaptive histogram equalization*. https://commons.wikimedia.org/w/index.php?curid=17500777.

Wang, Yong et al. (2020). "Augmenting Vascular Disease Diagnosis by Vasculature-aware Unsupervised Learning". In: *bioRxiv*. DOI: 10.1101/2020.02.07.938282. eprint: https://www.biorxiv.org/content/early/2020/02/07/2020.02.07.938282.full.pdf. URL: https://www.biorxiv.org/content/early/2020/02/07/2020.02.07.938282.

Wang, Zhaolei et al. (2019a). "Data Augmentation is More Important Than Model Architectures for Retinal Vessel Segmentation". In: *Proceedings of the 2019 International Conference on Intelligent Medicine and Health*. ICIMH 2019. Ningbo, China: Association for Computing Machinery, 48–52. ISBN: 9781450372862. DOI: 10.1145/3348416.3348425. URL: https://doi.org/10.1145/3348416.3348425.

— (2019b). "Data Augmentation is More Important Than Model Architectures for Retinal Vessel Segmentation". In: *Proceedings of the 2019 International Conference on Intelligent Medicine and Health*. ICIMH 2019. Ningbo, China: Association for Computing Machinery, 48–52. ISBN: 9781450372862. DOI: 10.1145/3348416.3348425. URL: https://doi.org/10.1145/3348416.3348425.

Warfield, Simon, Kelly Zou, and William Wells (Aug. 2004). "Simultaneous Truth and Performance Level Estimation (STAPLE): An Algorithm for the Validation of Image Segmentation". In: *IEEE transactions on medical imaging* 23, pp. 903–21. DOI: 10.1109/TMI.2004.828354.

Williams, H. et al. (2019). "3D Convolutional Neural Network for Segmentation of the Urethra in Volumetric Ultrasound of the Pelvic Floor". In: *2019 IEEE International Ultrasonics Symposium (IUS)*, pp. 1473–1476. DOI: 10.1109/ULTSYM.2019.8925792.

Xiang, Yi, Yuren Zhou, and Hailin Liu (2015). "An elitism based multi-objective artificial bee colony algorithm". In: *European Journal of Operational Research* 245.1, pp. 168 –193. ISSN: 0377-2217. DOI: https://doi.org/10.1016/j.ejor.2015.03.005. URL: http://www.sciencedirect.com/science/article/pii/S0377221715001988.

Xiao, X. et al. (2018). "Weighted Res-UNet for High-Quality Retina Vessel Segmentation". In: *2018 9th International Conference on Information Technology in Medicine and Education (ITME)*, pp. 327–331.

Xie, Yuanpu et al. (Oct. 2016). "Spatial Clockwork Recurrent Neural Network for Muscle Perimysium Segmentation". In: *Medical image computing and computer-assisted intervention : MICCAI ... International Conference on Medical Image Computing and Computer-Assisted Intervention* 9901, pp. 185–193. DOI: 10.1007/978-3-319-46723-8{_}22. URL: https://pubmed.ncbi.nlm.nih.gov/28090603.

Xu, F. et al. (2010). "Segmentation algorithm of brain vessel image based on SEM statistical mixture model". In: *2010 Seventh International Conference on Fuzzy Systems and Knowledge Discovery*. Vol. 4, pp. 1830–1833.

Xu, Kelvin et al. (2015). *Show, Attend and Tell: Neural Image Caption Generation with Visual Attention*. arXiv: 1502.03044 [cs.LG].

Yan, Z., X. Yang, and K. Cheng (2018). "Joint Segment-Level and Pixel-Wise Losses for Deep Learning Based Retinal Vessel Segmentation". In: *IEEE Transactions on Biomedical Engineering* 65.9, pp. 1912–1923.

— (2019). "A Three-Stage Deep Learning Model for Accurate Retinal Vessel Segmentation". In: *IEEE Journal of Biomedical and Health Informatics* 23.4, pp. 1427–1436.

Yang, Tiejun et al. (2020). "SUD-GAN: Deep Convolution Generative Adversarial Network Combined with Short Connection and Dense Block for Retinal Vessel Segmentation". In: *Journal of Digital Imaging*.

Yang, Wei et al. (Jan. 2017). "Cascade of multi-scale convolutional neural networks for bone suppression of chest radiographs in gradient domain". In: *Medical Image Analysis* 35, pp. 421–433. DOI: 10.1016/j.media.2016.08.004.

Yu, Honggang et al. (Feb. 2012). "Fast Vessel Segmentation in Retinal Images Using Multiscale Enhancement and Second-order Local Entropy". In: *Proceedings of SPIE - The International Society for Optical Engineering*, pp. 45–. DOI: 10.1117/12.911547.

Zaidi, Habib and William Erwin (Aug. 2007). "Quantitative Analysis in Nuclear Medicine Imaging". In: *Journal of Nuclear Medicine* 48, pp. 1401–1401. DOI: 10.2967/jnumed.107.042598.

Zeng, Ye Zhan et al. (2016). "Liver vessel segmentation based on extreme learning machine". In: *Physica Medica* 32.5, pp. 709 –716. ISSN: 1120-1797. DOI: https://doi.org/10.1016/j.ejmp.2016.04.003. URL: http://www.sciencedirect.com/science/article/pii/S1120179716300187.

Zhang, Hang et al. (2020). *ResNeSt: Split-Attention Networks*. arXiv: 2004.08955 [cs.CV].

Zhang, J. et al. (2016). "Robust Retinal Vessel Segmentation via Locally Adaptive Derivative Frames in Orientation Scores". In: *IEEE Transactions on Medical Imaging* 35.12, pp. 2631–2644.

Zhang, Jingdan et al. (2015). "Blood Vessel Segmentation of Retinal Images Based on Neural Network". In: *Image and Graphics*. Ed. by Yu-Jin Zhang. Cham: Springer International Publishing, pp. 11–17. ISBN: 978-3-319-21963-9.

Zhao, Hanli et al. (2020a). "High-quality retinal vessel segmentation using generative adversarial network with a large receptive field". In: *International Journal of Imaging Systems and Technology*. DOI: 10.1002/ima.22428. eprint: https://onlinelibrary.wiley.com/doi/pdf/10.1002/ima.22428. URL: https://onlinelibrary.wiley.com/doi/abs/10.1002/ima.22428.

Zhao, Peng et al. (2020b). "SCAU-Net: Spatial-Channel Attention U-Net for Gland Segmentation". In: *Frontiers in Bioengineering and Biotechnology* 8, p. 670. ISSN: 2296-4185. DOI: 10.3389/fbioe.2020.00670. URL: https://www.frontiersin.org/article/10.3389/fbioe.2020.00670.

Zhou, Yi Tao and Rama Chellappa (1988). "Computation of optical flow using a neural network". In: *IEEE 1988 International Conference on Neural Networks*, 71–78 vol.2.

Zhou, Zongwei et al. (2018a). "UNet++: A Nested U-Net Architecture for Medical Image Segmentation". In: *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*. Ed. by Danail Stoyanov et al. Cham: Springer International Publishing, pp. 3–11. ISBN: 978-3-030-00889-5.

Zhou, Zongwei et al. (2018b). "Unet++: A Nested U-Net Architecture for Medical Image Segmentation". In: *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*. Springer, pp. 3–11.

Zhou, Zongwei et al. (2019). "UNet++: Redesigning Skip Connections to Exploit Multiscale Features in Image Segmentation". In: *IEEE Transactions on Medical Imaging*.

Zhu, Chengzhang et al. (2017). "Retinal vessel segmentation in colour fundus images using Extreme Learning Machine". In: *Computerized Medical Imaging and Graphics* 55. Special Issue on Ophthalmic Medical Image Analysis, pp. 68 –77. ISSN: 0895-6111. DOI: https://doi.org/10.1016/j.compmedimag.2016.05.004. URL: http://www.sciencedirect.com/science/article/pii/S0895611116300416.

Zhuang, Juntang (2018). *LadderNet: Multi-path networks based on U-Net for medical image segmentation*. arXiv: 1810.07810 [cs.CV].

Zhun Fan, Jiewei Lu, and Yibiao Rong (2016). "Automated blood vessel segmentation of fundus images using region features of vessels". In: *2016 IEEE Symposium Series on Computational Intelligence (SSCI)*, pp. 1–6.

Çiçek, Özgün et al. (2016). *3D U-Net: Learning Dense Volumetric Segmentation from Sparse Annotation*. arXiv: 1606.06650 [cs.CV].