University of Cape Town

Automated quantification of plant water transport network failure using deep learning

Student:
Tristan Naidoo
NDXTRI015

Supervisor:
Mr Stefan Britz
Co-supervisor:
Dr Glenn Moncrieff

Minor dissertation for the degree

M.Sc. Advanced Analytics

Department of Statistical Sciences

September 17, 2021

Abstract

Droughts, exacerbated by anthropogenic climate change, threaten plants through hydraulic failure. This hydraulic failure is caused by the formation of embolisms which block water flow in a plant's xylem conduits. By tracking these failures over time, vulnerability curves (VCs) can be created. The creation of these curves is laborious and time consuming. This study seeks to automate the creation of these curves. In particular, it seeks to automate the optical vulnerability (OV) method of determining hydraulic failure. To do this, embolisms need to be segmented across a sequence of images. Three fully convolutional models were considered for this task, namely U-Net, U-Net (ResNet34), and W-Net. The sample consisted of four unique leaves, each with its own sequence of images. Using these leaves, three experiments were conducted. They considered whether a leaf could generalise across samples from the same leaf, across different leaves of the same species, and across different species.

The results were assessed on two levels; the first considered the results of the segmentation, and the second considered how well VCs could be constructed. Across the three experiments, the highest test precision-recall AUCs achieved were 81%, 45%, and 40%. W-Net performed the worst across the models, while U-Net and U-Net (ResNet-34) performed similarly to one another. VC reconstruction was assessed using two metrics. The first is Normalised Root Mean Square Error. The second is the difference in Ψ_{50} values between the true VC and the predicted VC, where Ψ_{50} is a physiological value of interest. This study found that the shape of the VCs could be reconstructed well if the model was able to recall a portion of embolisms across all images which had embolisms. Moreover, it found that some images may be more important than others due to a non-linear mapping between time and water potential. VC reconstruction was satisfactory, except for the third experiment. This study demonstrates that, in certain scenarios, automation of the OV method is attainable.

To support the ubiquitous use and development of the work done in this study, a website was created to document the code base. In addition, this website contains instructions on how to interact with the code base. For more information please visit: https://plant-network-segmentation.readthedocs.io/.

Acknowledgements

I would firstly like to thank Stefan Britz for being an exemplary supervisor. He taught me the skill of building a coherent narrative in research. He also kept me out of many rabbit holes; without him this thesis would have been 100 pages longer. I would equally like to thank Glenn Moncrieff, my second supervisor, for sharing his technical expertise and contextual knowledge. His constant support, willingness to help me think through problems, and overall assistance were invaluable. I would also like to thank Rob Skelton, who provided the data used in this thesis and assisted with the construction of Vulnerability Curves.

Running my code would not have been possible without access to additional computational power. In this regard, I would like acknowledge SAEON for providing me with access to their Graphical Processing Unit server. I would similarly like to acknowledge Glenn Moncrieff for providing me with funds on Google Cloud Platform.

Finally, I would like to thank: my parents, Patrick and Mavis Naidoo for their unconditional support, and my brother Jarryd Naidoo, for his motivation and an endless supply of braai meat.

Contents

List of Figures

List of Tables

List of Abbreviations

AE	**A**coustic **E**mission
AUC	**A**rea **U**nder a **C**urve
BHO	**B**ayesian **H**yperparameter **O**ptimisation
CNN	**C**onvolutional **N**ueral **N**etwork
CT	**C**ohesion-**T**ension
FCN	**F**ully **C**onvolutional **N**etwork
FN	**F**alse**N**egative
FP	**F**alse **P**ositive
GPU	**G**raphics **P**rocessing **U**nit
OV	**O**ptical **V**ulnerability
NRMSE	**N**ormalised **R**oot **M**ean **S**quare **E**rror
PR	**P**recision-**R**ecall
RMSE	**R**oot **M**ean **S**quare **E**rror
SGD	**S**tochastic **G**radient **D**escent
TN	**T**rue **N**egative
TP	**T**rue **P**ositive
Qg	**Q**uercus **g**ambelii
Qk	**Q**uercus **k**elloggii
Qp	**Q**uercus **p**almeri
Qp1	**Q**uercus **p**almeri$_1$
Qp2	**Q**uercus **p**almeri$_2$
VC	**V**ulnerability **C**urve

To my loving parents, Patrick and Mavis Naidoo

Chapter 1

Introduction

Plants are paramount to the functioning of ecosystems and consequently to the functioning of the planet; indeed, the study of plant life, Botany, is centuries old - often thought to date back to the Greek philosopher Theophrastus in the latter half of the 4th century BC (Morton, 1981). This long-standing pursuit of understanding plant life has become more relevant due to global warming.

Global warming, a multifarious threat, once considered minor, has begun to manifest in many ecosystems (Parmesan and Yohe, 2003). Projections by Trenberth et al., 2013 suggest that land surface warming, as a consequence of global warming, may lead to longer and more severe droughts. The effect of this phenomenon on vascular plants is increased risk of hydraulic failure, where hydraulic failure is the inability of a plant to move water through its vascular system (Petruzzellis et al., 2020; Powell et al., 2017; Nardini and Luglio, 2014)

Initial methods of quantifying hydraulic failure were both invasive to plants and difficult to apply. Brodribb et al., 2016 propose a non-invasive method, called the optical vulnerability (OV) method, which uses the change in light to quantify the extent of hydraulic failure. While this method is both easier and less invasive compared to other methods, it is laborious to apply, requiring manual labelling of images. The broad goal of this study is to automate the application of the OV method by leveraging fully convolutional deep learning models, which have shown success in similar tasks. Successful automation will allow for the increased application of this method, which may prove important in our fight against global warming.

1.1 Hydraulic vulnerability

Hydraulic vulnerability is important to plant physiologists as it is closely correlated to a plant's drought tolerance. This tolerance is becoming ever more relevant due to the effect of anthropogenic climate change on droughts. The effects of both increased and more severe droughts are already evident in many biomes. Moreover, many species operate closely to their threshold of hydraulic failure, and hence are at heightened risk. There is extensive evidence in the literature pertaining to the devastating effects of major loss of plant life (Allen et al., 2010; Anderegg, Kane, and Anderegg, 2013). To understand this close correlation, one needs to understand the mechanism of water transportation in a plant.

1.1.1 The transport of water in plants

A plant's vascular, or water transport, system consists of conduits called xylem. The transport of water in plants occurs primarily through these conduits. Xylem comprise two types of cells, which have the peculiar property of being dead at maturity, the result of which is that these cells are hollowed out with thick, lignified walls - providing a path of low resistance for water to the plant's leaves (Taiz and Zeiger, 2010). However, despite this pathway, the mechanism of transporting water through xylem needs to overcome both gravity and the lesser frictional resistance within the cell (Venturas, Sperry, and Hacke, 2017).

Dixon and Joly (1895) provided a widely accepted explanation for how water is transported through the xylem, called Cohesion-Tension (CT) theory. CT theory asserts that water travels in a metastable state through the xylem as a result of evaporating water at leaf surfaces. The danger of water travelling in a metastable state is that the xylem are susceptible to cavitation, which means that an air bubble forms within a liquid. When cavitation occurs, the xylem is said to embolise, and the air bubble is known as an embolism. The formation of an embolism can have dire consequences for a plant as it blocks water flow in the conduit.

In general, embolisms form within a plant when it is under extreme stress, such as during a drought (Brodribb et al., 2016). Moreover, different plant species embolise at different rates, with some species being more resistant to stress than others. Embolisms are rare events from which plants do not often recover. Once a plant has fully embolised, the plant dies. Hence, the ability to quantify the characteristics of this mechanistic trait is important both to understand hydraulic failure in plants and to ascertain which plants are at risk due to anthropogenic climate change.

1.1.2 Optical Vulnerability Method

Many methods have been proposed to quantify hydraulic failure, the OV method being one of the most recent. The OV method entails removing the leaves from the main plant, exposing them to increasing hydraulic deprivation, and taking sequential images of the leaves. This process of data gathering is less invasive in comparison to other methods of quantification. The image sequence obtained in this procedure is then analysed to create a profile of hydraulic failure over time. These profiles, known as vulnerability curves (VCs), display species-specific xylem vulnerability to embolisms as a function of water potential (Skelton et al., 2021). These curves are of inferential value to plant physiologists, and, in addition to the inferential value of VCs, quantification of xylem vulnerability will allow for this mechanistic trait to be included in ecological studies (Petruzzellis et al., 2020).

The ease of application of OV methods has been improved by Petruzzellis et al. (2020), who have developed an easy way to generate the data necessary for this method, using only a smartphone (with a camera), an LED light source, and a notebook. However, while gathering data is easier, there is no attempt in the literature to make the method of analysis easier. Specifically, the image output from the OV method needs to be segmented, where image segmentation, in the context of this task, requires labelling each pixel as either an embolism or not. This will be the focus of this study.

Machine learning, and its subfield deep learning, has permeated many fields of research (Litjens et al., 2017; Kamilaris and Prenafeta-Boldú, 2018; Hatcher and Yu, 2018). This study will aim to continue the trend, leveraging fully convolutional models in this segmentation task. Fully convolutional models have shown promise in many

similar tasks, most notable of which are road extraction and retinal vessel segmentation. It is hypothesised that this class of models, with appropriate pre-processing and post-processing, will be able to segment input images accurately enough to produce VCs. The specific aims and objectives of this study will be discussed in the next section.

1.2 Research Aims and Objectives

This study aims to segment an image sequence, produced by applying the OV method, well enough to recreate a VC. The accuracy of the segmentation models will be assessed at a pixel level. However, adequacy of segmentation performance will be determined by how well the recreated VC corresponds to the true VC, which is obtained from human annotated OV images. In addition, this study aims to create reproducible analysis which can be used by plant physiologists without the requirement of in-depth knowledge of the analysis process. The research objectives necessary to achieve these aims are listed below:

Build an analysis pipeline to segment OV images, where the pipeline will include the following:

1. Data preparation
2. Pre-processing
3. Segmentation using deep learning
4. Post-processing
5. VC generation

Achieving these aims will allow for the rapid implementation of the OV method, resulting in an improved understanding of hydraulic failure in plants. Furthermore, it will allow for a better understanding of how xylem vulnerability varies within species and between different species, and which species are at most risk due to climate change. Moreover, successful application of fully convolutional models will contribute to the plant physiology knowledge base, in turn contributing to an increase in the adoption of deep learning in the field.

1.3 Conclusion

This chapter has introduced hydraulic vulnerability, a mechanistic plant trait, and the OV method, which can be used to quantify this vulnerability. It has motivated the relevance of this study by explaining why quantifying hydraulic vulnerability is important, especially in the face of anthropogenic climate change, and how automating the OV method can increase the extent of hydraulic vulnerability quantification. The chapter ends by outlining the aim of this study, and the objectives that need to be met to achieve it.

Chapter two will review the literature relevant to this segmentation task. The review starts with a brief history of plant hydraulics, and methods which are currently used for quantifying xylem vulnerability. It will then provide an overview of image segmentation and of convolutional models. This will provide a foundation to review fully convolutional models, which have performed well in segmentation tasks. The

chapter will end with by reviewing applications of fully convolutional models to tasks similar to the one addressed in this study.

Chapter three will both present and interrogate the data used in this chapter. It will present in more detail how VCs are constructed. It will also present class imbalance, explain why it is an issue, and suggest ways to address it. The chapter will conclude by presenting an interesting property inherent in the data used by the OV method, and explain how this property can be used both in pre- and post-processing techniques.

Chapter four will present the methodology – inspired by the literature reviewed in Chapter two – that will be used to produce predicted masks and predicted VCs. It will start by explaining the three different experiments that will be conducted, using different combinations of the leaves presented in Chapter three. It will then look at the building blocks of convolutional models, and explain how these blocks are combined to form the three models used in this study. The chapter will conclude by explaining how these models will be trained and assessed.

Chapter five will present the results of applying the methodology presented in Chapter four. It will assess each of the three experiments conducted, both in terms of segmentation performance and in terms of VC reconstruction.

Chapter six will discuss the key findings from Chapter five. This discussion will be focused both on the performance of the chosen methodology and on the possible real world applications. It will present the limitations of the study and provide recommendations for future work. It will end by concluding this study.

Chapter 2

Literature review

This literature review is divided into two broad themes: the first is plant hydraulics and the second is image segmentation. The theme of plant hydraulics will review the literature on the history of plant hydraulics, plant cavitation, and methods to measure such cavitation. The theme of image segmentation will look at convolutional neural networks (CNNs), fully convolutional networks (FCNs), and the application of FCNs to road extraction and retinal segmentation.

2.1 Plant hydraulics

Plant hydraulics – the science of how water moves through a plant vascular system – was founded on the revolutionary work of Dixon and Joly (1895). Since then, the field has advanced steadily; the last 50 years, however, have witnessed accelerated advancement (McDowell, Brodribb, and Nardini, 2019). Included in this advancement is the foundational work by early researchers, who investigated plant embolisms and their associated factors, which paved the way for research into vulnerability of damage in plant vascular systems, or "hydraulic vulnerability". (Tyree and Sperry, 1989; Brodribb et al., 2016).

2.1.1 History of plant hydraulics

Although Dixon and Joly (1895) are credited with proposing Cohesion-Tension (CT) theory, it is important to highlight earlier work which contributed to their findings. Notably, there are two contributions, in the same decade, which were important (Brown, 2013). The first came in 1891, in an encyclopaedic compilation by Eduard Strasburger, which contained work done on pathways and mechanisms of water transport in the plant body (Richter and Cruiziat, 2002). Included in this compilation were experiments which showed that living cells were not responsible for the ascent of water inside a plant (Strasburger, 1891; Richter and Cruiziat, 2002; Brown, 2013). After conducting experiments which supported Strasburger's initial findings in 1892, Joseph Bohm made the second notable contribution one year later. He made two important assertions that are foundational to CT theory. Firstly, he argued that transpiration played an important role in starting the process of water ascent. Secondly, he argued that the cohesive property of water allowed water columns to withstand negative tension involved during the ascent, hence maintaining their integrity.

Two years after Joseph Bohm's work, Dixon and Joly (1895) published their seminal paper; while revolutionary, this work was not without controversy. A similar paper, developed independently, was published by Askenasy (1895), prompting a debate of

priority, which Askenasy conceded (Pickard, 1981). Moreover, CT theory has proved divisive in the plant sciences community, with many initially refusing to accept the theory; the notion that water is able to travel in a metastable state, as intact water columns, is a particular point of contention (Tyree and Sperry, 1989). The first research in support of this notion was conducted by Renner (1911), who showed through experiments based on leafy twigs that the integrity of water persists under negative tension (Bentrup, 2017).

In the following years, further research was conducted in quantifying tension in xylem and investigating whether water in plants cavitates(Dixon, 1914; Stocking, 1945; Temperley, 1947; Preston, 1959; Milburn and Johnson, 1966). However, a drawback of this research was that the techniques used could not measure xylem tensions directly, with indications in support of CT theory depending on indirect measures (Stocking, 1945; Milburn, 1966). This was until Scholander et al. (1965) proposed the pressure bomb technique[1]. This innovative research allowed for the direct measurement of water pressure and tension in plant xylem. Moreover, Scholander et al. (1965) confirmed that water was transported by negative tension and observed that these tensions varied between different species across different environments, after which widespread acceptance of CT Theory ensued (Tyree and Sperry, 1989).

Accompanying this widespread acceptance of CT theory was the conclusion that since negative tension transports water, cavitation must be rare (Tyree and Sperry, 1989). This prompted a shift in focus from whether plants cavitate (due to negative pressure), to how often they cavitate. A breakthrough on this front was made by John Milburn in 1966. Milburn (1966) began by showing that water cavitation reduces xylem conductivity; he also suggested that this reduction may be temporary. In a sequential paper, Milburn and Johnson (1966) recorded an acoustic emission (AE), described as a "click", that leaves make when subject to water stress. He argued this cavitation was likely to be the cause of these emissions, as these emissions occurred at times when cavitation is expected. This expectation was based on research done in his paper published earlier that year (Milburn, 1966). Moreover, he argued that a plant's anatomical features were correlated with the number of emissions. Milburn's work had two important implications: firstly, cavitation is progressive. Secondly, cavitation varies across species. This non-invasive technique reignited research in CT theory, with a slew of research supporting the correlation of AEs with cavitation across a wide variety of species (Milburn, 1973a; Milburn, 1973b; Milburn and McLaughlin, 1974; West and Gaff, 1976; Crombie, Milburn, and Hipkins, 1985; Crombie, Hipkins, and Milburn, 1985; Tyree and Sperry, 1989).

The discovery that plants do cavitate frequently raised questions around the rate of cavitation and which vulnerabilities were associated with causing cavitation. This prompted research efforts into creating new methods to measure cavitation. Tyree and Dixon (1983) proposed a new and improved AE method, by restricting measurement to ultrasonic ranges. Working in the ultrasonic frequency range instead of the audible frequency range had two clear advantages: firstly, ambient and background noise could be filtered out, and secondly, the method could be automated, with the ability to count AE events at rates up to 1000 sec^{-1} (Tyree and Dixon, 1983; Tyree and Sperry, 1989). Using this technique, researchers were able to provide further support that AEs were related to cavitation events, reinforcing the notion that plants do cavitate (Tyree and Sperry, 1989).

[1]Spanner (1951) proposed the Thermocouple psychrometer, which made direct measurements, but his results were variable

Shortly after the inception of the ultrasonic AE method, Sperry (1985) proposed the "bench-top hydraulic method", which is able both to assess the impact of cavitation directly and to measure cavitation caused by a variety of factors (Tyree and Sperry, 1989). The method uses the extent of reduction in hydraulic conductivity (through flow resistance) to quantify the extent to which a plant has embolised. However, direct measurement comes at a cost, as this method is more destructive than AE methods, requiring stems and petioles to be excised. Using this method, Sperry and Tyree (1988) formalised xylem vulnerability curves (VCs). Owing to direct measurement, this method was able to produce accurate curves, and for the first time plant physiologists were able to quantify the loss of xylem conductance as a function of time. This is a plausible reason for its widespread adoption as the most common method to measure embolisms (Cochard et al., 2013; Fontes and Cavender-Bares, 2020).

The work of these pioneering researchers laid down the foundations to create a new area of research called "plant hydraulics", which is the study of vulnerability to damage of the plant water transport system (Brodribb et al., 2016; Tyree and Sperry, 1989). Research efforts in plant hydraulics have since continued, with a focus on understanding xylem vulnerability. This has amounted to efforts in proposing new techniques that are both more accurate and less invasive. The forerunners of these efforts are visual techniques.

2.1.2 Visual methods for quantifying xylem vulnerability

Visual techniques measure xylem vulnerability through direct observation. Initial efforts to quantify embolisms visually were made either by eye or under a light microscope, but their use-cases were limited due both to difficulty in preparation and to their inability to quantify total xylem failure (Dixon, 1914; Haines, 1935; Cochard et al., 2013). As technology has advanced, so too have the methods of direct observation (Canny, 1997; Holbrook et al., 2001; Cochard et al., 2013; Fontes and Cavender-Bares, 2020).

One such advancement was the development of micro-computed tomography (micro-CT), which uses x-rays to create cross-sections of an object – effectively allowing a view inside an object. Micro-CT was initially used in both the health sciences and the material sciences, until Fromm et al. (2001) applied the method to plants by using micro-CT to measure the xylem water content of whole spruce and oak trees. This method provides a non-invasive observation of xylem contents at a high spatial resolution. However, the application of micro-CT has remained niched due to the technology (synchrotron facilities) required (Brodersen et al., 2010; Cochard et al., 2013; Cochard, Delzon, and Basel, 2015). In recent years, synchrotron facilities have become cheaper, and consequently more accessible (Nolf et al., 2017). Cheaper and more powerful computer desktop machines have allowed for a more accessible desktop-based micro-CT; however, there are disadvantages to this compared to synchrotron approaches, particularly relating both to the specimen size and to the long term health of the specimen. In addition, synchrotron approaches have better beam intensity and better temporal resolution (Suuronen et al., 2013; Torres-Ruiz et al., 2014; Cochard, Delzon, and Basel, 2015). There is an active effort to improve the quality of these desktop-based approaches, with many of the disadvantages being addressed (Nolf et al., 2017).

Micro-CT has grown in popularity, establishing itself as an accurate and reliable way to quantify xylem vulnerability *in vivo* (Losso et al., 2019). Indeed, the literature

is rife with examples of its application (Brodersen et al., 2013; Knipfer et al., 2015; Bouche et al., 2016; Longuetaud et al., 2016; Choat et al., 2016; Losso et al., 2019; Pratt et al., 2020). In addition, to support the accuracy of the micro-CT approach, comparative studies have recently been undertaken in an attempt to show alignment between micro-CT and the commonly used bench-top dehydration technique (Savi et al., 2017; Nolf et al., 2017; Nardini et al., 2017; Pratt et al., 2020). However, despite this success, micro-CT is not without disadvantages (Skelton, Brodribb, and Choat, 2017; Brodribb et al., 2017; Petruzzellis et al., 2020). In particular, micro-CT techniques are currently limited in their applicability to large sample sizes and to large species' assemblages due to the specialised equipment and facilities required (Hochberg et al., 2017; Brodribb et al., 2017; Petruzzellis et al., 2020). Moreover, this technique cannot be applied over the period of time required to dehydrate plants, due to the harmful effects of radiation caused by repeated scans, yielding limited temporal information (McElrone et al., 2013; Hochberg et al., 2017; Savi et al., 2017; Brodribb et al., 2017; Skelton, Brodribb, and Choat, 2017).

Recently, Brodribb et al. (2016), have developed a new, non-invasive visual method, called the optical vulnerability (OV) method, which addresses some of the disadvantages of previous methods. The technique leverages work by Haines (1935), who showed that embolised vessels are more reflective (white) when compared to water filled conduits (typically dark) (Brodribb et al., 2016; Brodribb et al., 2017). The OV method requires sequential images of a dehydrating leaf; an embolism would be indicated by a change in light transmission between sequential images. Brodribb et al. (2017) and Zhang and Brodribb (2017) have extended this method both to stems and to flowers respectively. The advantages of the OV method can be summarised as follows: this is a low-cost, simple, and portable method that provides a wide spatial and temporal view of the process of cavitation until desiccation (Brodribb et al., 2017; Skelton, Brodribb, and Choat, 2017).

Remarkably, Petruzzellis et al. (2020) were able to improve on the initial low-resource setup required to apply this method. They proposed a new set-up based on a smartphone (with a camera), an LED light source, and a notebook. These items are easily available and according to the authors "would potentially allow any laboratory to apply the OV method."[2] Although this method has been recently proposed, there are already many examples of its successful use in the literature, alluding both to its promise and to the importance of measuring xylem vulnerability (Brodribb, Bienaimé, and Marmottant, 2016; Skelton, Brodribb, and Choat, 2017; Skelton et al., 2018; Cardoso et al., 2018; Skelton et al., 2019; Creek et al., 2019; Smith-Martin et al., 2020; Johnson et al., 2020).

The literature reviewed above presented many methods which have been used in pursuit of better understanding xylem vulnerability. The numerous attempts at an improved methodology to measure this mechanistic trait alludes to its importance.

2.1.3 The importance of xylem vulnerability

Concerns have recently been raised about both the reliability and the validity of indirect, destructive hydraulic methods due to potential artefacts that skew the results; the literature on this issue is rich with debate, reports of methodological bias, and concern (Knipfer et al., 2015; Cochard, Delzon, and Basel, 2015; Nardini et al., 2017; Savi et al., 2017). In his review of current methods, Cochard et al. (2013) states that

[2] A more detailed explanation of the required resources can be found here.

it is "crucial to consolidate the methods for measuring cavitation in order to guide research on plant hydraulics in the most relevant and fruitful directions.", a sentiment which Knipfer et al. (2015) echoed. The current proposed visual methods perhaps have the greatest potential for doing so. The OV method in particular is a standout candidate, characterised by its ability to visualise the complete process of embolism formation (Skelton, Brodribb, and Choat, 2017). A consolidated technique, which has been validated, will allow for unified efforts and results, lending evidence to many long-standing controversies and debates to be settled (Tyree, 1997; Zimmermann et al., 2002; Brown, 2013; Brodribb et al., 2016). But perhaps the strongest motivation for a consolidated approach is the threat of climate change, and the consequent importance of understanding xylem vulnerability to drought.

Severe droughts have devastating effects on plant communities, as the resultant increased dehydration increases cavitation and drought-induced mortality (Brodribb and Cochard, 2009; Urli et al., 2013; Adams et al., 2010; Choat et al., 2016; Skelton et al., 2018; Skelton et al., 2019; Creek et al., 2019). In fact, cavitation and consequent embolisms are considered to be a principal cause of drought-induced plant mortality (Anderegg et al., 2016; Nolf et al., 2017). It is worrying then that anthropogenic climate change, and the associated global warming, have increased both the severity and the frequency of droughts. Consequently, the risk of widespread death in many plants, and more particularly across all major forest biomes, has increased (Choat et al., 2012; Engelbrecht, 2012; Trenberth et al., 2013; Urli et al., 2013; Choat et al., 2016; Nardini et al., 2017; Choat et al., 2018; Johnson et al., 2020; Smith-Martin et al., 2020). Against this backdrop, being able to accurately predict and mitigate plant mortality has elevated importance (Choat et al., 2018).

The ability of xylem to resist embolisms is a key physiological trait relating to a plant's drought tolerance, and it can hence be used to predict plant mortality (Nardini et al., 2017; Skelton, Brodribb, and Choat, 2017; Skelton et al., 2018; Skelton et al., 2019; Choat et al., 2018; Smith-Martin et al., 2020). The inclusion of this property is usually made through a proxy measure of the threshold at which water potential corresponds to a particular loss of hydraulic conductance. Usually the water potential at which 50% of hydraulic conductance is lost – denoted as Ψ_{50} – is used[3] (Skelton et al., 2018; Creek et al., 2019; Skelton et al., 2019). These values can be obtained using the VCs, which can of course be constructed using the methods described above (Skelton, Brodribb, and Choat, 2017; Savi et al., 2017).

The OV method has the potential for rapid determination of xylem VCs. This rapid determination presents the opportunity for the widespread inclusion of Ψ_{50} in ecological studies, and is hence an additional advantage of the method. Recently, Brodribb, Cochard, and Dominguez (2019) used this property in such a predictive model, where Ψ_{50} was obtained from VCs generated using the OV method. Moreover, it is important to highlight that vulnerability may differ between species within genera (Johnson et al., 2020; Skelton et al., 2019). This variability emphasises the importance of fast resolution of VC curves in an ideal method, as many species and/or samples will need to be measured for accurate values.

To automate the OV method, the OV images will need to be segmented. Deep learning approaches, and particularly fully convolutional models, have shown success in such

[3]This threshold can differ across different plant species; for example, in their study, Skelton et al. (2018) use 50% in conifers and 88% in angiosperms.

tasks. Image segmentation will be discussed next, with a focus on convolutional models.

2.2 Image segmentation

Image segmentation can be traced back approximately 60 years, to Roberts (1963) , who proposed in his PhD thesis the "Roberts edge detector" to extract edge information from a digital photograph (Davis, 1975; Zhang, 2006). Remarkably, Roberts (1963)'s work came only 6 years after the world's first digital photograph was scanned by Russel Kirsch (Kirsch et al., 1957). Following Roberts (1963)'s initial work, image segmentation has been an active area of research. In his review, Zhang (2006) summarises the activity in this area over the preceding 40 years. At a high level, the groups of algorithms proposed are: clustering (thresholding if binary target classes), edge detection, region extraction, fuzzy set theory approaches, and iterative pixel classification – in which both Markov random field (MRF) based approaches and neural network based approaches are included (Fu and Mui, 1981; Pal and Pal, 1993; Zhang, 2006).

Recent years have seen rapid advancement in the approaches to image segmentation. In particular, this advancement has been through the use of fully convolutional networks (FCNs). These models are adapted from Convolutional Neural Networks (CNNs), which are discussed next.

2.2.1 Convolutional Neural Networks

The biological inspiration of CNNs was provided by Hubel and Wiesel (1962), who studied the primary visual cortex in cats and showed that it consists of two types of cells. Using these findings, Fukushima and Miyake (1982) proposed the neocognitron. Le Cun et al. (1989) simplified this model to propose the first CNN trainable by backpropagation, which was later called LeNet[4] (LeCun, Kavukcuoglu, and Farabet, 2010). LeCun et al. (1998) performed a comparative study between variants of LeNet and other methods as applied to both handwritten character and handwritten digit recognition; in both cases they found that CNNs outperformed all other models. Importantly, this study compared test error rates, meaning that these models were able to generalise. This success, and the generalisability of the CNN approach, at the time, was emphasised by their commercial deployment. By the end of the 1990s, 10% of all cheques were being read by CNNs, and in the early 2000s several optical character recognition (OCR) CNNs were being deployed by Microsoft (Goodfellow et al., 2016).

However, deep learning had lost popularity in the early 2000s. Perhaps this is why, despite early success, CNNs did not garner widespread interest. This changed in 2012. First, Ciresan, Meier, and Schmidhuber (2012) successfully improved the state-of-the-art performance on 6 well known computer vision benchmark datasets using a CNN termed Multi-column Deep Neural Networks (MCDNN). The authors concluded their paper by stating: "This is the first time human-competitive results are reported on widely used computer vision benchmarks." Following this, a significant breakthrough was made when Krizhevsky, Sutskever, and Hinton (2012) won the ImageNet Large Scale Visual Recognition Challenge (ILSVRC), a benchmark dataset renowned for

[4]This model was trained with a reduced version of the now popular MNIST dataset.

its scale.[5](Russakovsky et al., 2015; Goodfellow et al., 2016). Their success was a statement victory, beating the second-best entry by more than 10%, and also the previous state-of-the-art. The network they used – called AlexNet – had 5 convolutional layers, 3 fully connected layers, 650 000 neurons and 60 million parameters (see section 4.3.1 for an explanation of CNN architecture). At the time, it was one of the largest convolutional networks ever trained (Krizhevsky, Sutskever, and Hinton, 2012). To train this large network, Krizhevsky, Sutskever, and Hinton (2012) spread training across two Graphics Processing Units (GPUs); this application of GPUs was novel. Moreover, two additional novel applications were the use of rectified linear unit (ReLU) activation functions and dropout for regularisation.

The success of AlexNet sparked a renewed interest in using CNNs in large-scale computer vision problems. Benchmark datasets (of which there are many) promote development by giving researchers a state-of-the-art score to beat. In particular, the ILSVRC has promoted architectural development in CNNs (Simonyan and Zisserman, 2014). The winner of the 2013 ILSVRC was also a CNN, called ZFNet (Zeiler and Fergus, 2014), although this was architecturally very similar to AlexNet. 2014, however, saw two novel networks proposed: Inception[6] and VGG, which claimed first and second place respectively in the 2014 ILSRVC.

The focus of Simonyan and Zisserman (2014), when developing VGG, was depth. They varied the convolutional layers between 11 and 19, which were followed by 3 fully connected layers. What was notable about their approach, in addition to depth, was the use of smaller 3×3 receptive fields throughout the network. This was in contrast to both AlexNet and ZFNet, which used larger convolutions in their first layers. In addition, the use of stacked layers was notable. The reason for stacking layers is that a wider effective receptive field could be achieved using smaller receptive field filters, and consequently with much fewer weights[7].

When Szegedy et al. (2015) proposed the Inception network, depth was also in mind. However, they approached depth from two directions. The first was through organisation, achieved by what they called the "Inception module", and the second was through usual network depth. The inception module was at the heart of this approach. The inception module entails feeding input from the previous layer to multiple filters in parallel and concatenating the result. To contain computational cost introduced by this concatenation, the authors used Lin, Chen, and Yan (2013)'s Network-in-Network approach – which can be thought of as a 1×1 convolution in this context – to reduce channel dimensions. In addition to the Inception module, an important component of the network was the set of auxiliary classifiers which addressed issues associated with propagating errors through such a large network. Interestingly, this approach used significantly fewer parameters than AlexNet.

In 2015, He et al. (2016) proposed the revolutionary Residual Neural Network (ResNet). An ensemble of these networks won the ILSVRC, achieving an error score close to half of what was achieved by GoogleLeNet, and better than that of a human expert, Andrej Karpathy. Inspired by the benefit that increased depth had yielded, the authors were driven to the following question: "Is learning better networks as easy as stacking more layers?". The authors began by showing that at a certain point adding more layers increases error (training included), where this error is not due to overfitting.

[5]The ImageNet dataset contains more than 10 million labelled images depicting 10 000+ object categories; the challenge uses a subset of 1000 categories and 1.2 million images.

[6]A particular version of Inception, called GoogleLeNet was submittied to ILSRVC.

[7]See Araujo, Norris, and Sim (2019) for more details on receptive field arithmetic

To address this issue, the authors introduced the residual block. Each residual block stacks weight layers – as with VGG. The addition to the residual block is the identity mapping, which is achieved through a shortcut connection. This mapping simply adds the input into a residual block to the output of the same block. The idea behind this is that if depth has degraded the output, then it is easier to set the output of a block to zero and use the block's input, than it is to learn a non-linear identity mapping function. The authors tested multiple configurations of ResNet depth, ranging from 18 layers all the way to an extraordinary 152 layers.

Since AlexNet and, at the time of writing, CNNs have both exclusively won the ILSVRC[8] and set new benchmarks (Stojnic et al., 2021). This has continued to drive innovation in proposed CNN architectures beyond the foundational networks discussed above. However, rather than discussing these newer architectures, attention will be shifted to fully convolutional models as they are often used for image segmentation — the focus of this study.

2.2.2 Fully convolutional networks

Image segmentation is characterised by the way pixels are classified. Currently there are three types, namely semantic segmentation, instance segmentation, and panoptic segmentation. Semantic segmentation entails labelling every pixel in an image; historically this was the general definition of segmentation (Fu and Mui, 1981). In contrast, instance segmentation requires a particular type of object to be detected and for those pixels to be classified. Hence, instance segmentation considers objects of the same type as unique, each with a distinct label. Panoptic segmentation, which Kirillov et al. (2019) recently proposed, is a unification of the instance and semantic segmentation. This type of segmentation both labels every pixel in an image and treats objects as unique. This study will focus on semantic segmentation, and hence so too will the literature reviewed below.

Given the success of CNNs in image classification tasks, researchers turned their attention to classification tasks on a finer spatial scale, in which object detection[9] and segmentation are included. In 2013, Girshick et al. (2014) proposed R-CNN for object detection, where its name was derived from the fact that it combined region proposals with a CNN feature extractor. Specifically they used AlexNet as the feature extractor, but it could be built on any of the aforementioned architectures (Guo et al., 2018). The use of a CNN as a feature extractor was noteworthy, as previously handcrafted machine learning features had been integral to success (Sultana, Sufian, and Dutta, 2020). While this model was designed for object detection, the authors realised that since region classification is a standard technique for semantic segmentation, their model could be extended to the task with minor modification. Using R-CNN, the authors were able to achieve state-of-the-art results on both the PASCAL VOC 2011 segmentation task[10] and the PASCAL VOC 2010-2012 object detection. This proved that the recent success of CNNs in image classification tasks could extend to additional computer vision tasks.

In 2015, Long, Shelhamer, and Darrell (2015) proposed Fully Convolutional Networks (FCNs) as an alternative to region-based segmentation, which addressed the latter's shortcomings. These models are trainable end-to-end. The authors converted three

[8] This challenge was moved to Kaggle in 2018.

[9] Object detection entails classifying and localising multiple instances of an object, by drawing a bounding box around each occurrence of the object (Girshick et al., 2014).

[10] Using the VOC 2011 test set.

well known architectures to FCNs, namely AlexNet, VGG-19, and GoogleLeNet. To convert a model to a FCN, all fully connected layers are converted to convolutional layers – which allows for arbitrarily sized inputs – followed by a 1×1 convolution. This 1×1 convolution is coupled with a deconvolution layer, which bi-linearly up-samples the output of the 1×1 layer back to original input image size. The authors found that the loss of spatial resolution which occurs in the final layer (prior to up-sampling) causes the up-sampled output to be coarse. This was successfully remedied by the introduction of skip connections that combine lower layers – which have greater spatial resolution – with the final prediction layer. Moreover, the authors made use of transfer learning[11] by fine-tuning pre-trained versions of models that won their respective ILSVRC[12], rather than training from scratch. Their model achieved state-of-the-art performance on both PASCAL VOC 2011 and 2012 semantic segmentation tasks.

In the same year, Ronneberger, Fischer, and Brox (2015), built on Long's FCN to propose U-Net. There are a few key differences between the two models. Firstly, U-Net has an expansive path symmetric to its contracting path, resulting in a U-shaped architecture. Hence, U-Net has multiple up-sampling steps. This allows context information to be better preserved at a higher resolution at later layers. Secondly, rather than the addition operation Long's FCN uses in its skip connections, U-Net concatenates the feature map from the expansive path with the corresponding feature map from the contracting path. Finally, U-Net uses learnable filters in its up-sampling step, as opposed to the bilinear interpolation Long's FCN uses. This network won biomedical image segmentation challenges, setting new records in the process. Key to this success was the use of both excessive augmentation and a custom weighted loss.

FCNs have since become a popular approach to semantic segmentation and are considered state-of-the-art in various applications (Li et al., 2020), with many variations to the above two approaches suggested (Noh, Hong, and Han, 2015; Badrinarayanan, Kendall, and Cipolla, 2017; Chen et al., 2017; Jégou et al., 2017; Zhao et al., 2017). Much of the innovation in both image classification and semantic segmentation has been driven by testing models against benchmark datasets. Once new models are established, they are applied to specific use cases, with further domain specific adaptation.

2.2.3 Domain-specific applications of FCNs

To the best of the author's knowledge, no literature exists on the application of FCNs, or similar deep learning based architectures, to segmentation of OV images. Hence, models applied to similar tasks will be reviewed. Two such tasks are retinal vessel segmentation and road extraction from aerial images. It is widely accepted that deep learning approaches are popular for these tasks (Alom et al., 2018). In particular, U-Net and its variants have been used in both tasks. This section will focus on both the architecture and the training procedure.

Alom et al. (2018) updated the original U-Net and proposed the Recurrent Residual Convolutional Neural Network (R2U-Net), which they applied to retinal vessel

[11] Transfer learning is the use of model weights trained on one task as initialisation weights for a different task, with the aim of utilising the "knowledge" acquired in the previous task.

[12] Where possible the authors used the publicly available versions from the Caffee model zoo. In the case of GoogleLeNet, which was not publicly available, they had to train their own version

segmentation. Two additions were made with regards to the original U-Net convolutional block. The first addition implements a residual unit, as is used in ResNet. This addition allows for a deeper model to be trained. The second addition is the use of Recurrent Convolutional Layers (Ming Liang and Xiaolin Hu, 2015), which allows for efficient feature accumulation. Feature accumulation in this context allows for stronger feature representation, which translates to the extraction of more granular features. An additional change made was the replacement of the copy and crop skip connections between encoder-decoder units with concatenation operations. Despite these additions, R2U-Net has the same number of parameters as U-Net. The authors compared the original U-Net to R2U-Net over 5 different benchmark datasets – 3 of which were retinal vessel focussed – and in all cases R2U-Net was found to perform better.

The use of residual blocks and recurrent layers has also been successfully applied to road extraction. Zhang, Liu, and Wang (2018) successfully replaced the original U-Net convolutional blocks with residual blocks to proposed ResUnet; their reasons for doing so were similar to above. Their model outperformed a "vanilla" U-Net as well as other state-of-the-art approaches to road segmentation using the Massachusetts roads dataset. Yang et al. (2019) proposed RCNN-UNet by replacing the convolutional block in U-Net with a recurrent convolutional block. The recurrent block better utilises both spatial context and low-level features to create more detailed features.

Zhuang (2018) proposed LadderNet for retinal vessel segmentation, which can be viewed as a chain of U-Nets. Due to its architecture, this network allows multiple paths of information flow, where each path can be thought of as a FCN. Hence, this network can equivalently be thought of as an ensemble of multiple FCNs. The advantage of this interconnectivity is that the network has the potential to learn more complicated features. In addition, the authors, inspired by the success of R2U-Net, proposed a modified residual block. This block is influential in the multi-path information flow. Moreover, convolutional layers in the block share the same weights, which greatly reduces the number of parameters in the network. This sharing of weights addresses the increase in parameters that is introduced by stacking U-Nets.

Li et al. (2020) proposed IterNet, which adds multiple refinery modules – called mini U-Nets – after an initial U-Net. The output of the initial U-Net is used as the initial input to the chain of mini U-Nets, whilst the input for the remaining mini U-Nets is a feature map from the penultimate layer of a preceding mini U-Net. Interestingly, this strategy means that each refinery model will get a different input, despite a fixed input sample size. Analogous to excessive augmentation, the refinery modules will be exposed to a large number of incorrect vessel patterns, which helps with the training process. Through iterative application of the refinery module, IterNet is able to refine the initial output both by removing errors and by adding predictions. Furthermore, through iterative application, IterNet, without explicitly modelling it, takes into account structural redundancy, which according to the authors is exploited by human annotators. Rather than making an architectural change, the authors aim to make use of the extracted features of U-Net. The mini U-Nets are smaller since the inputs to the refinery modules are much simpler than the original images. In addition, to assist with training, additional skip connections are added, and the mini U-Net models share the same weights and biases to prevent overfitting.

The notion of exploiting structural dependency also extends to road extraction. Although not applied to a convolutional approach, Mnih and Hinton (2010) applied a post-processing step, in a road extraction task, which uses the output for a patch

from one neural network as input to another. The authors note the following: "The process clearly removes disconnected blotches, fills in the gaps in the roads, and generally improves the quality of the prediction". In addition, Conditional Random Fields (CRFs) have been applied as a post-processing step to utilise structural dependency (Rao et al., 2018). The use of a CRF as a post processing step has been made easier due to Zheng et al. (2015)'s work, which formulated a CRF as an RNN; this made it possible to integrate the CRF (as an RNN) with a FCN architecture, allowing for end-to-end training (Jégou et al., 2017).

Galdran et al. (2020), similar to IterNet, also proposed a repeated U-Net variant, with a focus on model complexity, and whether it is necessary. According to the authors, "[o]ne of the main goals of this work is to explore the lower limits in model complexity for the task of retinal vessel segmentation." They firstly showed that a U-Net with a small number of parameters performs similarly to the current state-of-the-art techniques. The authors then combined two mini U-Nets to propose W-Net. The output of the first U-Net is concatenated to the original input image, which is then used as input to the second U-Net, which helps focus on regions the first U-Net marked as being important. This is in contrast to IterNet, which only passes feature maps to its refinery modules. The W-Net used in the paper is the simplest architecture (in terms of parameters), at the time of writing, ever proposed for retinal vessel segmentation Galdran et al. (2020).

Recently, Guo et al. (2020) achieved state-of-the-art performance on two publicly available retinal vessel segmentation datasets, namely DRIVE and CHASE_DB1. They achieved their result by modifying the U-Net architecture rather than iterating models. They use a modified version of U-Net, called SA-UNet, which has two notable changes. The first is to integrate DropBlock[13] and batch normalisation to address overfitting. The second is the introduction of the spatial attention module at the bottleneck of the encoder-decoder structure. The spatial attention module indicates where the network should place its attention by highlighting which features should be emphasised and which should be suppressed (Woo et al., 2018). The authors' results confirmed this function as they found that the addition of the module highlighted blood vessels and reduced the influence of the background.

Training deep learning models, and consequently fully convolutional models, is a complicated task. There are many choices which can be made during the training process, more specifically the choice of: initialisation, loss, regularisation, optimiser, batch size, epoch, and learning rate. A summary of these choices – where available – for the literature reviewed in this section is given in Table 2.1.

2.3 Conclusion

The current wave of deep learning, preceded by a long and rich history, has sparked innovation and development in the field of computer vision. In semantic segmentation tasks, fully convolutional models have become popular. In particular, U-Net and its variants have been successful in many segmentation tasks (Zhuang, 2018), success which is illustrated when looking at the best-in-class models for road extraction and retinal vessel segmentation – tasks that are intrinsically similar to identifying embolisms in OV images. Training of these models is intricate and task-specific, as illustrated by the varying combinations of training schemes summarised.

[13] A structured form of dropout intended for convolutional layers, which drops contiguous regions from a feature map rather than random units (Ghiasi, Lin, and Le, 2018)

This chapter provided a review of the history of plant hydraulics. Moreover, it reviewed literature that highlights the need to automate the OV method and supports the exploration of U-Net-based architectures to achieve the goal of automation. The following chapter explores the data which will be used in this study.

Model	Epochs	Activation	Loss	Optimiser	Learning Rate	Regulariser	Metrics
R2U-Net	150	ReLU	Binary Cross-Entropy	Adam	2×10^{-4}	-	SE SP JSC F1-score AC AUC DC
ResUNet	50	ReLU	MSE	SGD	lr= 0.001 and reduced by a factor of 0.1 in every 20 epochs	BN	relaxed precision and recall scores ($\phi = 3$)
Laddernet	250	ReLU	BCE	Adam	learning rate as 0.01, 0.001, 0.0001 on epochs 0, 20 and 150 respectively	dropout, BN	F1 ACC AUC (ROC) SE SP
Iternet	200	ReLU	weighted sum of BCE for each out	Adam	$1e-3$	Dropout	AUC Connectivity, F1, SE, SP, ACC, AUC (ROC)
W-Net	iteration, rather than epoch based; set to 4000 iterations	ReLU	Binary Cross-Entropy	Adam	$\lambda = 10^{-2}$ anealed using a cosine law until it $\lambda = 10^{-8}$	Batch-Norm	MCC, DICE, AUC
SA-Unet	150	ReLU	BCE	Adam	first 100 epochs: 0.001, then 0.0001	DropBlock, BN	Matthews Correlation Coefficient (MCC), AUC (ROC), F1, SE AND SP

TABLE 2.1: Training configurations of different models used for road network and retinal vessel segmentation

Chapter 3

Data

The data used for a problem necessarily defines the task, and hence is an important component of a modelling pipeline. This chapter discusses the data used for this study.

3.1 Data Source

The data used for this study was provided by Dr. Robert Skelton, who originally used the data to construct vulnerability curves (VCs) using the OV method. The data consists of 4 unique leaves, each with their own sample set of images. All leaves are from the same genus – *Quercus* – commonly known as oaks. There are three unique species, with two leaves belonging to the same species. Figure 3.1 shows these leaves and the natural ranges where they occur. *Quercus gambelii*, more commonly known as a Gambel oak, is a small tree or shrub, commonly found in the western regions of North America. *Quercus kelloggii*, also known as the California black oak, is a larger oak variant. As its name suggests, this tree is commonly found in California. The last leaf, *Quercus palmeri*, is the smallest tree in the sample. This tree is commonly found in dry habitats, and is commonly known as a Palmer oak.

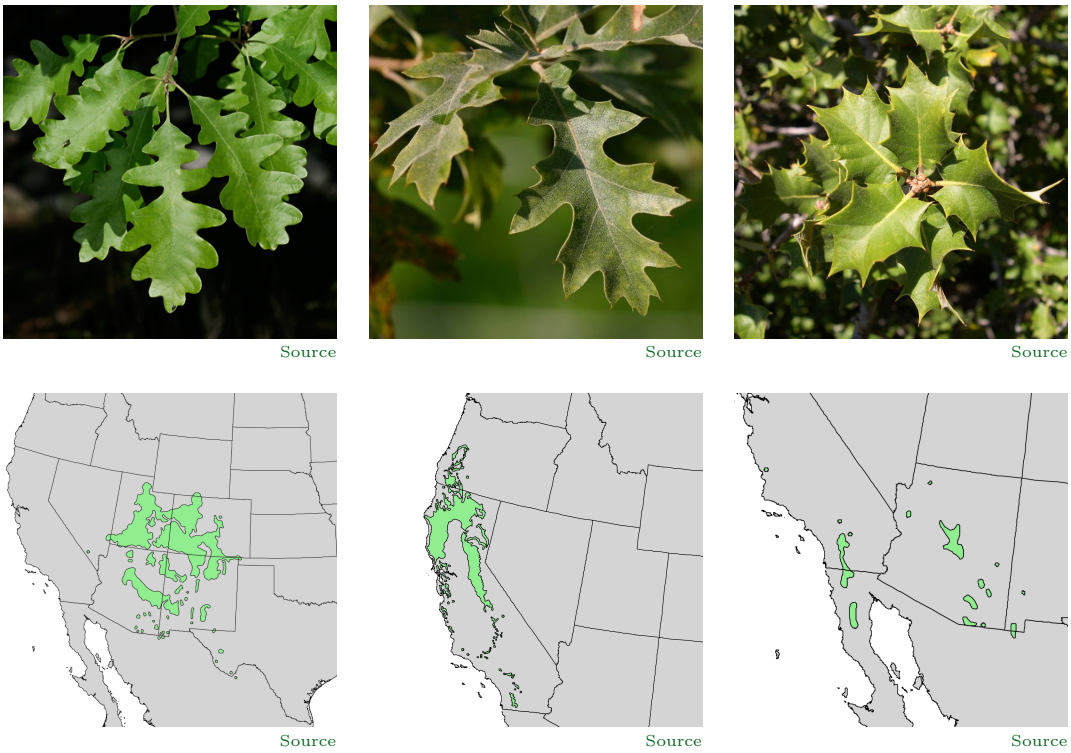
Each sample set was constructed by sequentially photographing an excised leaf until complete desiccation. Some leaves will have more samples than others, as the speed of desiccation is species specific. The photographs were taken at a DPI (dots per inch) of 72, and saved as grayscale in a Tagged Image File Format (.tif). A grayscale image is a single channel image with a pixel intensity between 0 and 255 indicating brightness. Details about each leaf are shown in Table 3.1.

Leaf name	Sample size	Image dimensions	Storage size[1]
Quercus kelloggii	742	2616 × 1950	3.3MB
Quercus gambelii	642	2304 × 2570	4.4MB
Quercus palmeri$_1$	395	1400 × 1934	1.9MB
Quercus palmeri$_2$	206	1104 × 1030	0.8MB

Table 3.1: Technical specifications of each leaf used in the study

The OV method uses changes in light between sequential images to show that a xylem conduit has embolised, as explained in Section 2.1.2. This change is easily seen if the image at step $t + 1$ is subtracted from the image at step t; an image of this form

[1] average storage of leaves and masks combined



(D) *Quercus gambelii* (Gambel oak)

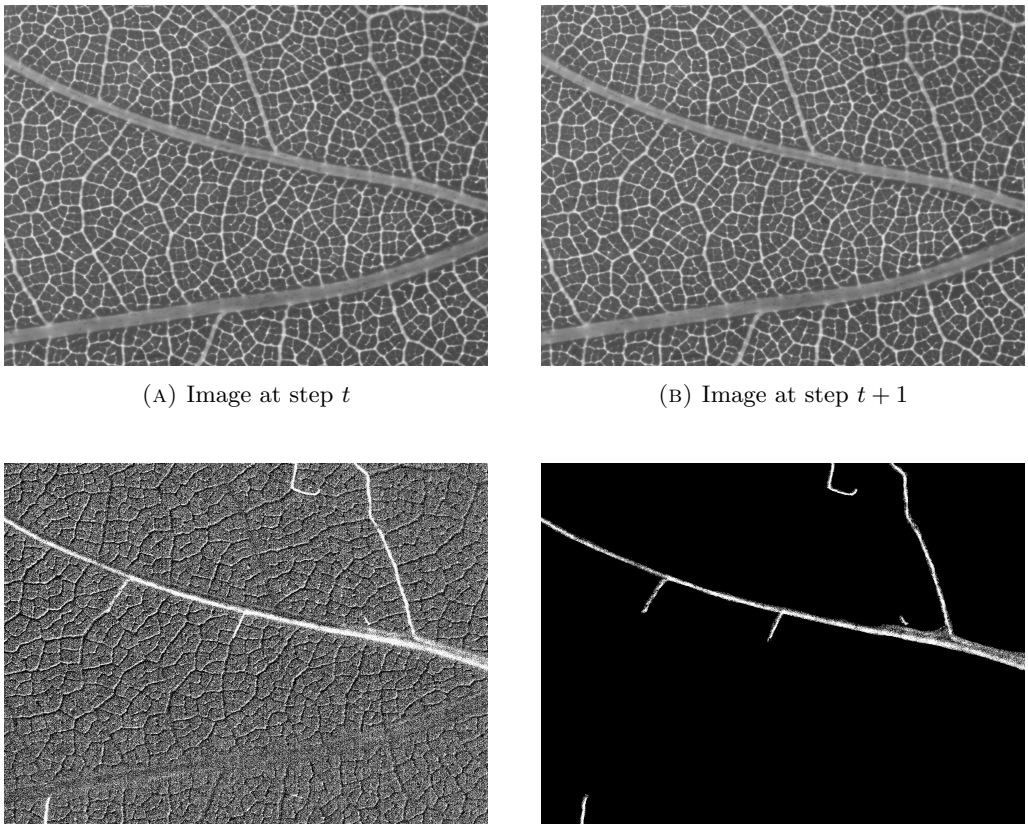
(E) *Quercus kelloggii* (California black oak)

(F) *Quercus palmeri* (Palmer oak)

FIGURE 3.1: **Top:** Photographs of each plant species in its natural habitat
Bottom: The natural range where each plant species occurs

will henceforth be referred to as a *differenced image*. The results of this process is illustrated in Figure 3.2 below. Upon first inspection of Figures 3.2a and 3.2b, it is not possible to see a difference between the two images. However, once a differenced image has been created, shown in Figure 3.2c, the locations of embolism are clearly indicated by the structured sequences of white pixels. Each unique embolism will only be displayed in one time step, since once an embolism has occurred, subsequent images will have the same pixel intensity at that location. Hence, a differenced image will show only those embolisms that occurred at a given time step. However, it is possible for embolisms to appear in the same location more than once, since the xylem in a leaf are layered. Stated differently, the 2-dimensional images are capturing a 3-dimensional process.

To learn the process that generates an input, one requires labelled data; in the context of semantic segmentation this comes in the form of an annotated image, which assigns a label to every pixel in an image. The annotations are referred to as *masks*. In this problem each differenced image has a corresponding mask, which labels a pixel as 1 if it indicates an embolism and 0 if not. The mask that corresponds to the example differenced image discussed in the preceding paragraph is shown in Figure 3.2d. Comparing the mask to the differenced image, it is clear that the differenced image contains many structures not useful in diagnosing embolisms (noise) – herein lies the difficulty of the segmentation task. To create a sample suitable for training, differenced images need to be extracted from the raw images series. Each differenced image is paired with a mask, and together they comprise a single sample.



(A) Image at step t

(B) Image at step $t+1$

(C) Differenced image (Image$_{t+1}$ - Image$_t$)

(D) Mask

FIGURE 3.2: **Top:** Raw images of *Quercus kelloggii*, taken at sequential time steps
Bottom: A single sample, obtained from the images in the top row, which consists of a differenced image and a corresponding mask. The structured white pixels in both the differenced image and the mask indicate embolisms.

In addition to providing annotations for differenced images, masks can also be used to construct the target VC for a leaf.

3.2 Constructing Vulnerability Curves

The samples in the dataset can be grouped by leaf. In this view there are two tiers to the dataset – the lowest tier considers samples within a leaf, and the second tier considers samples across different leaves. Constructing a VC considers samples within a leaf independently.

By the design of the data gathering process, which photographs a leaf until complete desiccation, each leaf sample captures the full process of hydraulic failure for that leaf. Therefore, images later in the series correspond to increased leaf desiccation. Hence by using this sequence a temporal profile of failure can be constructed. The extent of embolism at a time step is quantified as the total number of pixels with embolism divided by the total number of pixels which had embolisms across all images in the series. Using this, a temporal profile shows cumulative embolism % (y-axis) against time (x-axis). A temporal profile of each leaf used in this study is shown in Figure

3.3. The comparison of these curves aligns with expectation; *Quercus Palmeri* has a smaller gradient (slower cavitation) since it is found in drier habitats and is thus more resistant to drought.

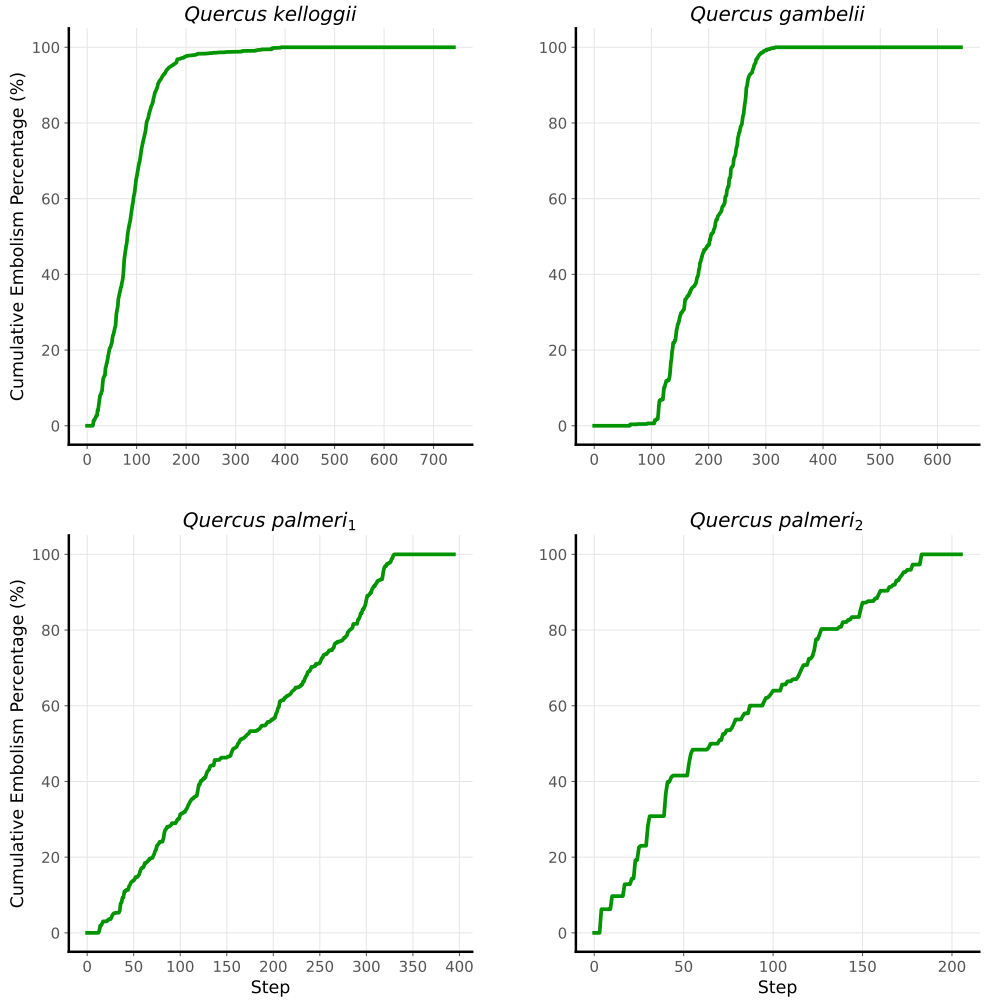


FIGURE 3.3: Temporal profiles of failure of each of the four leaves used in this study

A VC shows cumulative embolism % (y-axis) against water potential which is measured in Megapascals (Ψ). Hence, to construct a VC, the time on the x-axis of the temporal profile needs to be replaced by the water potential at that step. Xylem water potential was measured – either every fifteen or every ten minutes – on the same branches as were being dried down on the scanners using stem psychrometers (ICT International, Australia). These potentials were extrapolated between time points to form a continuous series of water potentials over the entire period of dessication. Using this extrapolated series, water potentials are mapped to the time that a leaf image was taken; in the case of differenced images the timestamp of the second raw image was used. The VC for each leaf used in this study is shown in Figure 3.4. This figure has two x-axes. The top axis, which shows time, is included to illustrate that the mapping between water potential and time is non-linear for the leaves used.

This dynamic process of hydraulic failure, over time, is captured using an animation

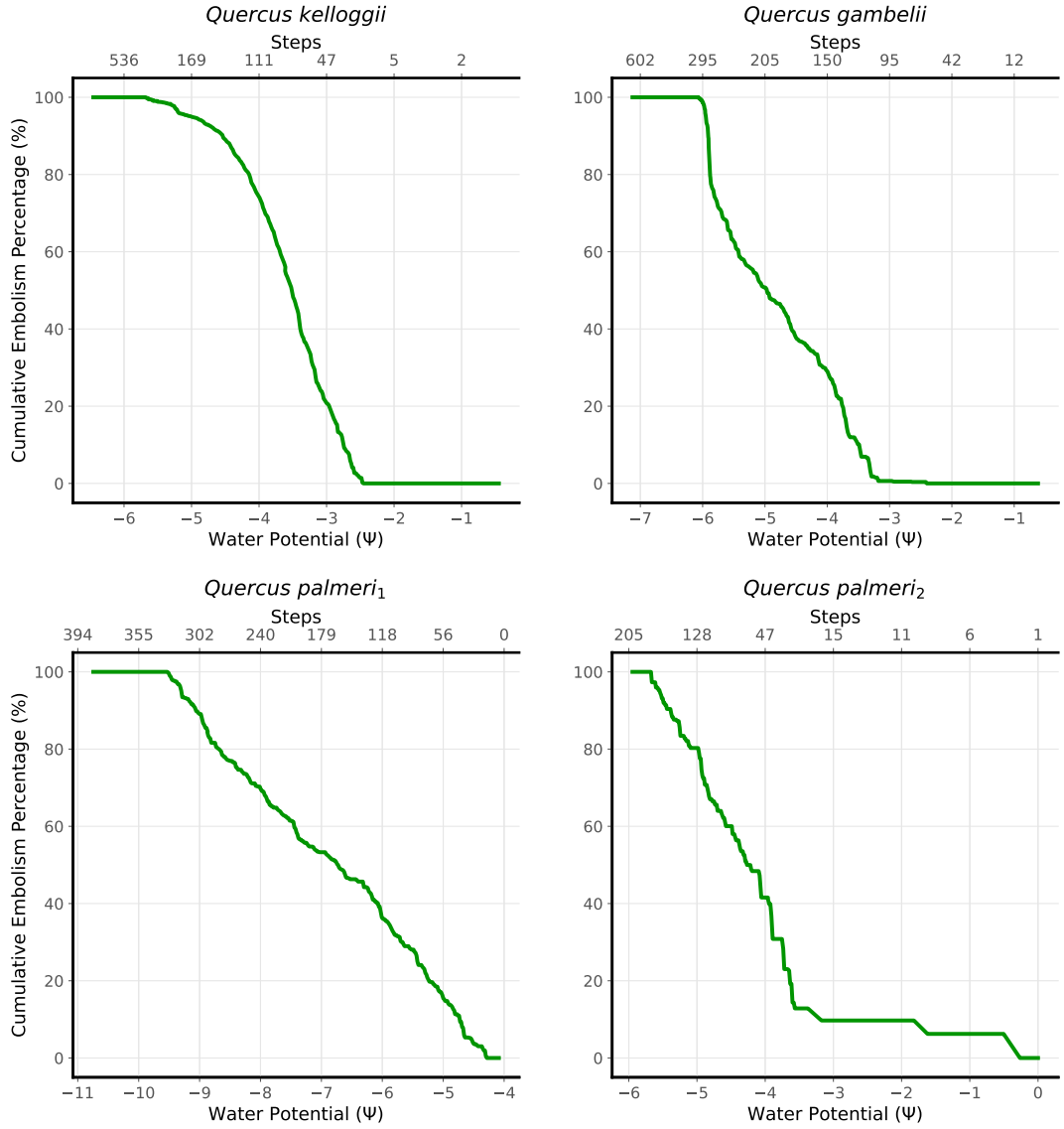


FIGURE 3.4: Vulnerability Curves of each of the four leaves used in this study

shown in Figure 3.5. The figure on the left shows the sequential occurrence of embolisms. Rather than showing the embolism for a single time step, masks are stacked on top of each other, with the colour indicating when the embolism occurred; a darker colour indicates an earlier time step, while a lighter colour indicates a later time step. The final image in the animation displays all embolisms which occurred. The figure on the right shows the corresponding construction of a VC. The final plot in this animation displays a complete VC. The faster speed of the animation before a water potential of 5 further illustrates the non-linear mapping between time and the water potential. The speed of the animation corresponds to the time axis shown in Figure 3.4; most of the embolisms occur before time step 170. These animations help clarify the goal of the study – while the primary goal is to segment images using masks as the target, the secondary goal is to segment masks well enough to reconstruct the VC, hence capturing this temporal process. However, these animations also reveal that there are many more pixels without embolisms than there are pixels with embolisms. This is expected as the total area of xylem may be less than the total area of a leaf,

and every xylem in a leaf does not need to cavitate for it to fail. When the ratio of one class is much higher than the other – non-embolisms to embolisms in our task – the problem is said to be class imbalanced.

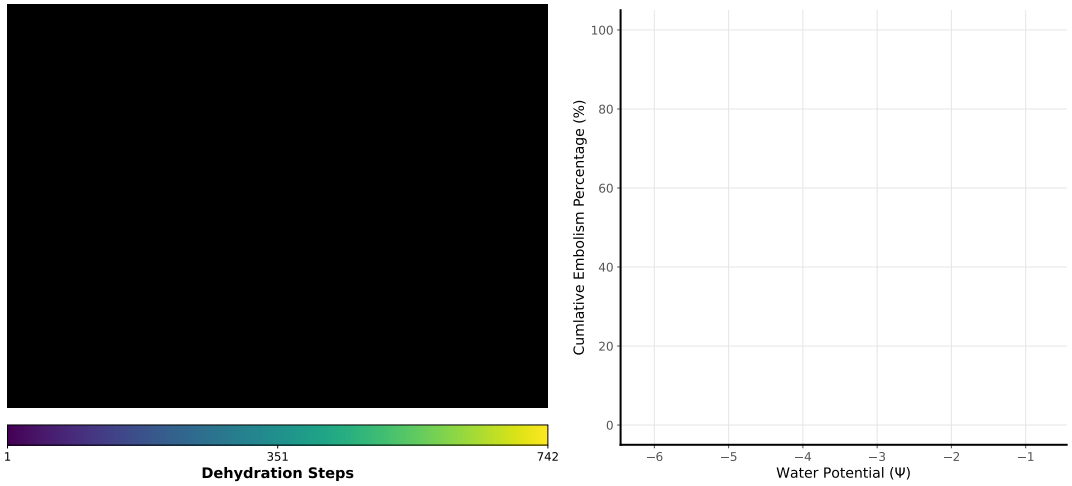


FIGURE 3.5: **Left:** Masks sequentially stacked and coloured by the step at which they occur
Right: Vulnerability Curve which increments proportionally to the stacked mask

3.3 Class imbalance

Class imbalance poses a serious challenge to many tasks; it can be difficult to learn a process when there aren't many examples of output generated by the process. Given that this is a segmentation task, there exists two sources of imbalance – both within an image and across images. The imbalance in this dataset will be examined using both full images, and tiles. The motivation for examining tiles is discussed below.

3.3.1 Full images

The distribution of embolisms over images depends on the rate at which a leaf desiccates. The extent of the across-image class imbalance is illustrated in Figure 3.6. The clustered bar chart shows both the percentage of images with and without embolisms per leaf. Inspecting the figure, we see that *Quercus gambelii* is the most imbalanced with only 29% of its samples having images with embolisms. In contrast, *Quercus palmeri*$_1$ is the most balanced with 52% of its sample containing embolisms. Considering the entire dataset, there are 699 images with embolisms and 1629 images without. This is a ratio of 1:2.3, which is not a severe imbalance.

While across-image imbalance does not seem too severe, the within-image imbalance is stark in comparison. As explained in Section 3.1, only new embolisms are shown in each time step. This exacerbates the issue of within-image imbalance, as the individual embolisms at a time step are usually a small fraction of the entire image. For each leaf, less than one percent of the total number of pixels had embolisms. This is illustrated in Table 3.2, which shows the average percentage of embolisms in each image. Futhermore, Table 3.2 contains both the ratio of images with at least one embolism to those with none, and the percentage of pixels with embolisms for

each leaf. These two metrics further illustrate that while there are many images with embolisms, embolism pixels are far outnumbered by non-embolism pixels.

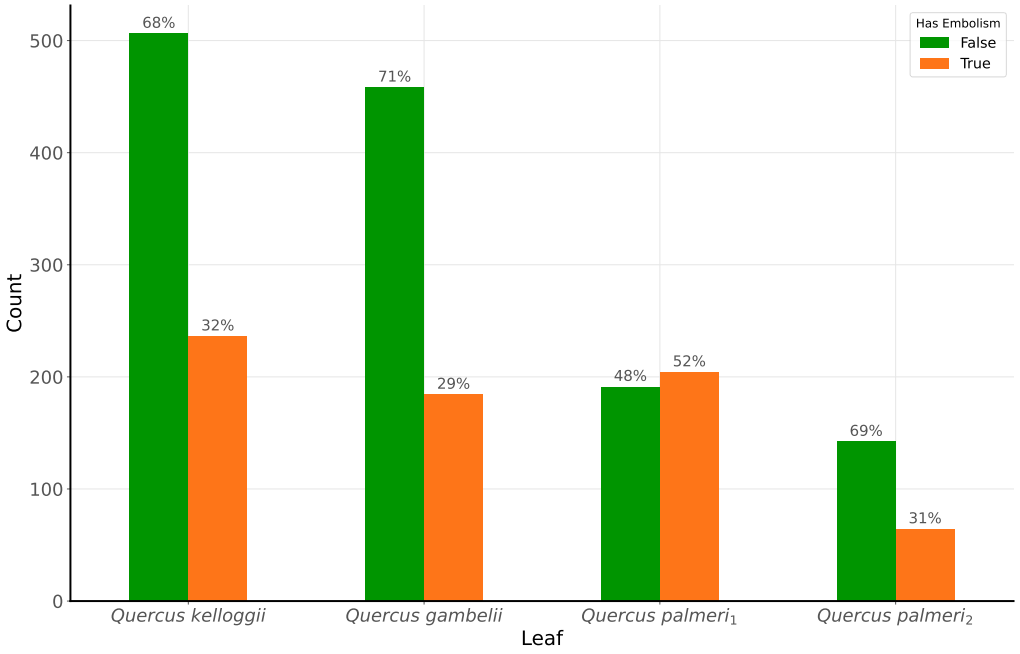


FIGURE 3.6: Clustered bar chart indicating the number of samples in a leaf with and without embolisms, annotated by percentage of sample size

Measure	*kelloggii*	*gambelii*	$palmeri_1$	$palmeri_2$
% Embolism Pixels	0.16	0.11	0.04	0.04
Embolism Ratio	1:2.1	1:2.5	1:0.9	1:2.2
Avg % Embolism per Image	0.51	0.38	0.07	0.12

TABLE 3.2: Details of the distribution of embolisms across full size images in each leaf

3.3.2 Tiles

The models that will be used in this study will be fully convolutional neural networks. These models were introduced in Section 2 and will be explained in further detail in Section 4. Although these models are flexible in their input size, a limitation is the memory available on the hardware a model is run on. The image dimensions and storage size, which are given in Table 3.1, are prohibitively large to run. For this reason it will be necessary to tile the images used in this study. The first consideration was tile size. A larger tile size would allow the network to learn on a bigger area, while a smaller tile size would reduce the chances of overfitting. A tile size of 512×512 will be used in this study. This was determined to be large enough to maintain spatial structure, while being small enough to fit in memory. The second consideration was whether tiles should overlap, for which the decision was made not to overlap the tiles. Overlapping tiles would result in larger datasets, which would increase training time.

Moreover, deciding the extent of overlap would add additional complexity to this study.

The results of tiling each dataset are shown in Table 3.3. Tiling has an interesting effect on class imbalance, resulting in a higher average percentage of embolism pixels per image. This reduces the within-image class imbalance. However, given that the majority of the image does not have embolisms, there will be many more tiles without embolisms than with embolisms. Thus, tiling increases the embolism ratio manifold. Consequently, tiling exacerbates across-image class imbalance. A strategy to deal with this particular imbalance is addressed in Section 3.4.2.

Measure	*kelloggii*	*gambelii*	*palmeri*$_1$	*palmeri*$_2$
Embolism Ratio	1:6	1:13	1:8	1:8
Avg % Embolisms per Image	1.17	1.52	0.35	0.33

TABLE 3.3: Details of the distribution of embolisms across 512×512 tiles created using the full size images in each leaf

3.4 Addressing class imbalance

Imbalanced datasets require adjustments both to find a good solution and for the solution to generalise. This study addresses class imbalance in three ways: augmentation, downsampling, and through class weighted loss functions. Downsampling and augmentation will be discussed next, while the relevant loss functions will be discussed in Section 4.5.1.

3.4.1 Augmentation

Data augmentation refers to augmenting the original dataset, using perturbations of original images. It is commonly applied in computer vision tasks and has a variety of benefits, although we will only discuss two of these advantages. Firstly, it adds additional training data, which is especially useful in scenarios where the data are imbalanced. Secondly, by providing perturbations of the input data, it makes the learning more robust to similar perturbations. This both helps in preventing overfitting and helps with model generalisation. Both model generalisation and overfitting are explained in Section 4.2.1.

Operation	Range
Reflection	Pr(x-axis) = 0.5
Translation	-25% – 25% (for x-axis and y-axis each)
Rotation	-90° – 90°
Sheering	-30° – 30°
Cropping	5% – 30% on each axis
Zooming	50% – 150% on each axis

TABLE 3.4: Augmentation operations and their ranges, which will be applied to embolism samples

When augmenting a dataset, it is important to augment images in ways that could reasonably exist in the population from which the dataset was drawn. Due to the

class imbalance present in this dataset, the decision was made to only augment images which had embolisms in them. In addition, augmentation was done at a tile level to create increased variability in the augmented dataset. Based on this decision to only augment tiles with embolisms, augmentation was done prior to training; often augmentation is done at training time by including the perturbations as a pre-processing layer. The details of the augmentations used are provided in Table 3.4. Each augmentation was applied independently with a probability of 0.5. Three examples of augmented samples are shown in Figure 3.7. Full size images are used as they better display the effect of the augmentation, however the augmentation will be applied to tiles.

3.4.2 Downsampling

Augmenting the dataset addresses class imbalance by providing more images with embolisms, helping to address the issue of across-image imbalance. However, there are still a large number of images without embolisms. These images exacerbate the imbalance at a pixel level. By having too many images without embolisms, a model may be biased towards always predicting pixels as non-embolisms. Moreover, motivated by computational constraints, removing images speeds up training. Therefore, downsampling was adopted to address imbalance and improve computational performance. Datasets were downsampled by removing 80% of non-embolism tiles.

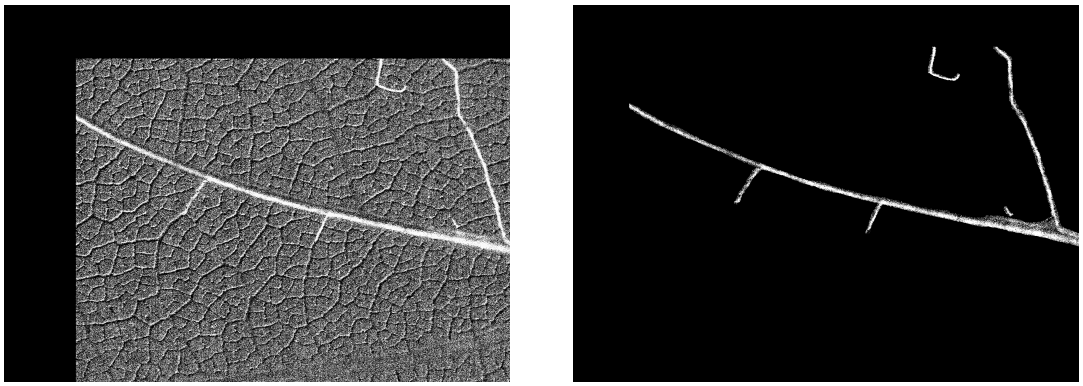
Once the final dataset size has been fixed, the tiles can be further processed in order to facilitate the learning process.

3.5 $\mathbf{Im}_{t+1} < \mathbf{Im}_t$ Property

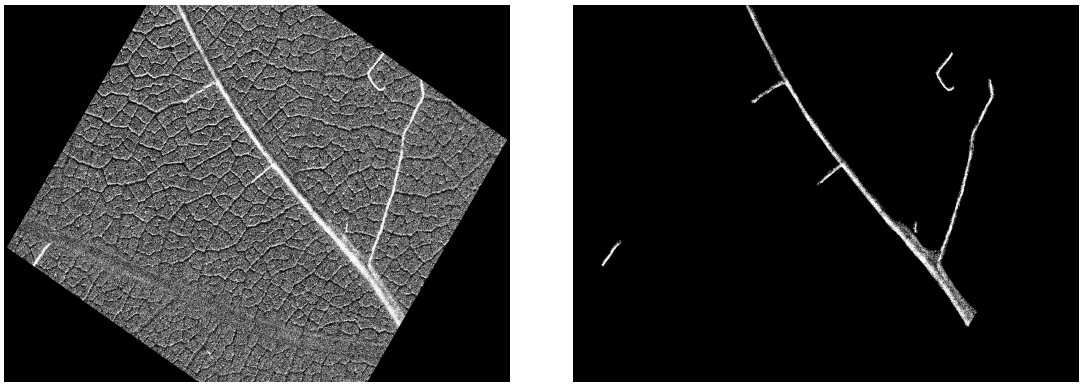
A hypothesised property that could be useful for further processing tiles, is that the difference in pixel intensities where embolisms occur should always be negative. This property will henceforth be referred to as the *image difference property*. The motivation behind this hypothesis is that the OV method detects an embolism through a change of light between sequential raw images. This is in line with the theoretical motivation provided by (Haines, 1935), as explained in Section 2.1.2. More specifically, pixels in image_{t+1} will be **darker** than the corresponding pixels in image_t if an embolism has occurred. Hence, embolisms should only occur where the difference in pixel values is negative.

Before this property can be investigated, an identifiability issue needs to be addressed. The images presented thus far have been in a uint8 format – the standard format for saving images. This means that pixels can only take on an unsigned intensity between 0 and 255 ($2^8 - 1$). When two uint8 images are subtracted, a difference which is less than 0 wraps around; for example, on a uint8 scale, since 250 - 254 is negative we would need to add 255 to the answer, that is, 250 - 254 = 251. Herein lies the identifiability issue: using a uint8 format, there are two sources for pixels with high intensities in an image, either a lighter pixel minus a darker pixel, for example, 255 - 4 = 251, or the wrap-around example provided. Thus, before this hypothesis can be tested, the data first needs to be converted to a float format, which allows for negative values.

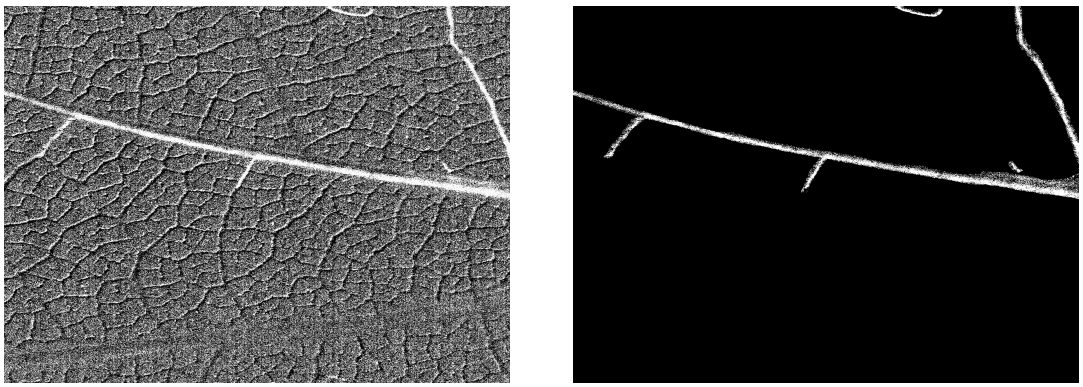
Using the locations of embolisms in the corresponding mask as indices, the regions where an embolism occurred were extracted from the differenced image. This was



(A) Translate, x: 15%, y: 10%



(B) Shear, 30°



(C) Zoom, x: 150%, y: 115%

FIGURE 3.7: Three examples of augmentations applied to the sample shown in Figure 3.2

done for each leaf. The distribution of intensities is shown in Figure 3.8. This figure shows that every pixel which was identified as an embolism – across all differenced images – has an intensity less than zero. This provides empirical support for the hypothesised property that $\text{image}_{t+1} < \text{image}_t$.

This property is helpful as it allows the candidate solution space for an image to be restricted to only those locations where there are negative pixel intensities. The candidate solution space refers to locations which can be classified as embolisms. The effect of this property can be quantified as follows: 44% of all pixels in the dataset

are negative, hence there is a massive reduction in the candidate solution space, since now only 44% of the pixels need to be considered. In this setting, the percentage of embolisms pixels in candidate solution space increases by an approximate factor of two, from 0.12% to 0.27%. It is important to remember that non-embolism pixels can also have negative intensities. Hence, this property can only be used to inform which pixels are definitely non-embolisms. This serves as a reminder to understand the extent of noise. This also helps remind us of the necessity of a learning system – if only embolisms pixels induced this property, one could simply select these pixels from the original image to segment it.

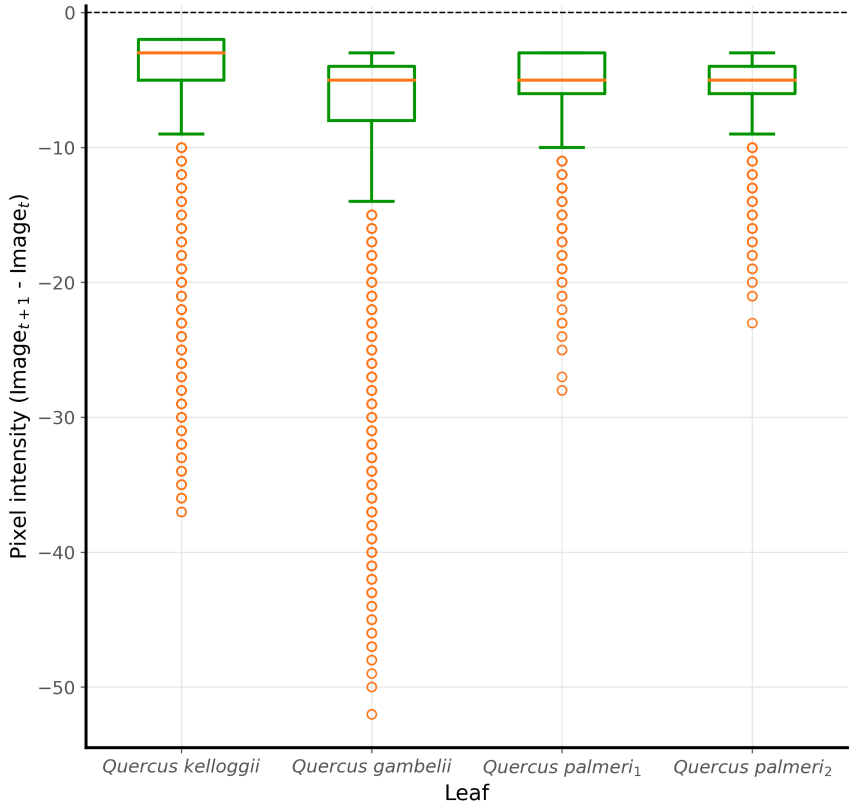


FIGURE 3.8: Box plot of embolism pixel intensities from the differenced image. Note that all intensities are less than zero

3.6 Image processing

Image processing can be divided into two groups: pre-processing and post-processing. As their names suggest, they are applied before and after predictions are made, respectively.

3.6.1 Pre-processing

To utilise the image difference property, pixel intensities in the range of $[-255, 255]$ are required. Common image formats are saved as unsigned integers as explained above. An alternative would be to save an image as something other than an image, but this introduces unnecessary complexity. Rather, the solution was to add 255 to pixel intensities. In memory, the image has a range of $[-255, 255]$, hence adding 255 changes this range to $[0, 510]$, ensuring that all intensities are positive. To save an

image with this range, a uint16 format is used. A uint16 pixel has a range of 2^{16} possible values, which includes $[0, 510]$. Once the images are read into memory, 255 is subtracted again, which converts the range back to $[-255, 255]$.

The first preprocessing step was to set all pixels with positive intensities to 0. The reasoning behind this is to restrict the candidate solution space – all possible pixels that a model may predict as being an embolism. This can be thought of as denoising the image, if we consider pixels which cannot be embolisms as noise.

The models used, which will be introduced in Section 4.4, train better when pixel values are confined to a smaller range. A smaller range reduces the influence of larger pixel values on the model weights. Pixels are normalised by dividing all pixel intensities by 255. This maps the range of pixel intensities from $[-255, 255]$ to $[-1, 1]$.

3.6.2 Post-processing

Only a single post-processing step, which is based on the image difference property, will be applied. To use this property as a post-processing step, all predicted pixels where the original input is greater than or equal to zero are set to zero. This is aimed at increasing precision without affecting recall, as only predictions which are incorrectly classified as embolisms (false positives) will be corrected. These metrics are introduced and discussed in Section 4.6.1. Many additional post-processing steps have been recommended for future work in Section 6.2.

3.7 Conclusion

This chapter started by presenting the four leaves that will be used in this thesis. It then explained how the raw images of these leaves are differenced to expose which xylem have embolised. In addition, it explained how these samples are used to construct VCs. The focus of the chapter then shifted to the severe class imbalance present in this dataset. It explained why this is an issue and ways to address this imbalance were suggested. The chapter concluded by looking at the image difference property, and how this property could be used in pre-processing and post-processing techniques. The methodology that will be used to produce both predicted masks and VCs is presented next.

Chapter 4

Methodology

4.1 The Learning Problem

The problem of *learning* from data was first introduced in the Chapter 2. Learning is usually approached from two similar schools of thought: Machine Learning (ML) and Statistical Learning Theory (SLT). This section will expand on the idea of learning, using a blend of both approaches, and also introduce the idea of a learner.

A learner – also known as a model or predictor – is a parametrised function that determines how input variables should be combined to produce an output. Moreover, the learner defines the functional space that can be explored; a functional space is the space of all possible functions that can be expressed using a learner[1].

To obtain the optimal function for a given problem, a learner is updated and fine-tuned using data. The data used is an input, target pair – (X, Y), which has been gathered by observing (or simulating) a process that relates X to Y. In this application, both X and Y are real-valued $s \times s$ random matrices, where Y has joint distribution $P(X, Y)$ (Hastie, Tibshirani, and Friedman, 2009). Both these components have many different names, usually discipline dependent; in this study, we will use predictor to refer to X, and response to refer to Y. Assuming there exists a relationship relating X to Y, the purpose of the learning system is to learn a function that captures this relationship as well as possible. A learning problem that uses data in this format is called supervised learning since the response is used to *supervise* the learning of the function.

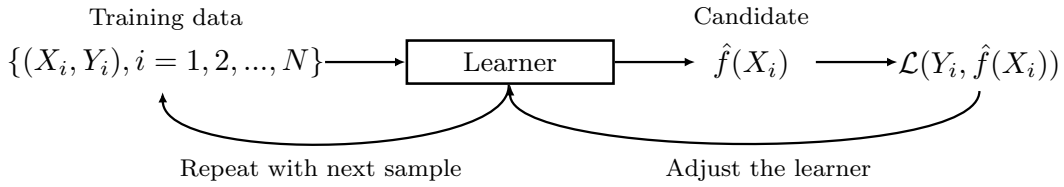


Figure 4.1: A simplified diagram of the process of *learning*

A diagram of the learning process is shown in Figure 4.1. For a given predictor, a learning system provides a candidate output according to the current state of the system – $\hat{f}(X)$ – which is equivalent to $\hat{\Pr}(Y = 1|X)$ in our task. This candidate output is then compared to the response which corresponds to the input. This comparison is

[1] A learner whose functional space spans all possible functions is called a universal approximator.

made using some evaluation criterion, $\mathcal{L}\left(Y_i, \hat{f}(X_i)\right)$. Depending on the result of the evaluation, an algorithm adjusts the learning system. The formal term for the process where a learning system learns the function which maps X to Y is called *training*. Training will be discussed in more detail in Section 4.5. To train a learning system, a suitable dataset, with N samples, is required. The term "learner" was used in this section as it makes the explanation of the learning problem more intuitive. However, this study will use the term model as it is more common in statistical literature.

4.2 Experimental design

To conduct experiments, we require a dataset that is representative of some population. In our case, each experiment we conduct will require its own dataset. Each dataset will need to be prepared as described in Chapter 3. In addition, each dataset will also need to be split in a format that is both suitable for training a model and testing the trained model.

4.2.1 Splitting the dataset

Learning from data is done using a sample of N points. The assumption is that this sample is representative of a larger population of similar points. We usually try to learn a population-level relationship using sample data, such that the analysis is generalisable. Generalisation refers to how well a model would perform on an unseen input. A big challenge in this regard is that the sample data in this study is noisy. Noise is interspersed with the true signal that relates the input to the underlying function we want to learn. Moreover, noise varies between samples. This makes learning more difficult; if noise has influenced the model too much, then the learnt function may not generalise.

One approach to addressing how well a model generalises is to hold out a fraction of the data for testing, referred to as the test set. The remaining fraction is used to train the model, and is called the training set. The motivation for doing this is that the optimal function will not have been determined using the test set, thus any performance evaluation done using the test set will indicate how well the model will generalise. If the model performs well on the training set, but poorly on the test set, then the parameters may have been influenced too strongly by the noise in the training set and the model is said to be overfitted. When a model has both low training error and low testing error, it is called underfitting since the model has not learnt well from the training set.

If a model either overfits or underfits, adjustments may be made to the learner. Often learners will have parameters, in addition to weights, that are user-specified; these parameters are known as hyperparameters. If a learner performs poorly on a held-out set then a possible adjustment would be to change the model's hyperparamters[2]. However, an issue with using the test set to adjust the learner is that the model will be biased to perform well on it. In this case, the test set would no longer be an unbiased measure of model generalisation. Hence, in addition to the test set, an additional set, called the validation set, is held out from the training set. This set serves as a proxy for the test set. Choosing hyperparameters is called tuning and will be discussed in more detail in Section 4.5.3.

[2]In such cases, we could also adjust the specification of the learning problem by, for example, adjusting the loss functions or optimiser.

4.2.2 Experiments

This study conducts three experiments to investigate the following questions:

1. Can a model generalise to within a single leaf?
2. Can a model generalise to leaves in the same species?
3. Can a model generalise to leaves across different species?

The first experiment will be tested by training a model using each of the four species individually. To get the training, validation and test sets, the complete dataset will first be split using a ratio of 80:20, where the 20% split will be used as the test set. The remaining 80% will then be split again, again using a ratio of 80:20, where 20% will be used as the validation set. The remainder constitutes the training set. The second experiment will be conducted using the *Quercus palmeri* (Qp) species, which is the only species for which we have data on more than one leaf. One complete leaf, Qp1, will be used as the training set, while the second leaf, Qp2, will be used to construct both the validation and test sets. The final experiment will be conducted using all four species. Models will train using *Quercus kelloggii* (Qk) and *Quercus gambelii* (Qg) and use Qp1 and Qp2 for validation and testing respectively.

To conduct these experiments we require a model that accurately represents the relationship between the input and the output. This will be discussed next.

4.3 Learning Architecture

As detailed in Section 2.2.1, convolutional neural network (CNN) architectures are ubiquitous in image recognition applications. An understanding of CNNs is necessary to understand the architectures that will be used in this study. Hence, CNNs will be the starting point. Following this introduction, the specific fully convolutional architectures used for this study will be discussed.

4.3.1 Convolutional Neural Networks

A CNN, used in the context of image recognition, takes as input an image and returns a prediction of which class the image may belong to, where these classes are defined by the problem. To achieve this, a CNN has different types of layers, characterised by both their structure and their operations. These different layers will be discussed in the subsequent sections, followed by a final section which will explain how the layers combine to form a complete model.

The Convolutional Layer

A convolutional layer is the first layer and the main feature extractor in a CNN, and there must be at least one of these layers in every CNN. According to Goodfellow et al. (2016), "[c]onvolutional networks are simply neural networks that use convolution in place of general matrix multiplication in at least one of their layers."

The components of a convolutional layer are shown in Figure 4.2. The filter, or kernel, is smaller than the input, and hence it will need to be applied multiple times, each time to a different region of the input. Hence, in general, the filter will only be applied to a particular region of an input; this region is known as a *receptive field*. The output of the repeated application of a filter to an input is called a *feature map*, where each element in the feature map is a linear combination of the weights in the

filter and the input. This repeated application is shown in Figure 4.3 and is called a convolution.

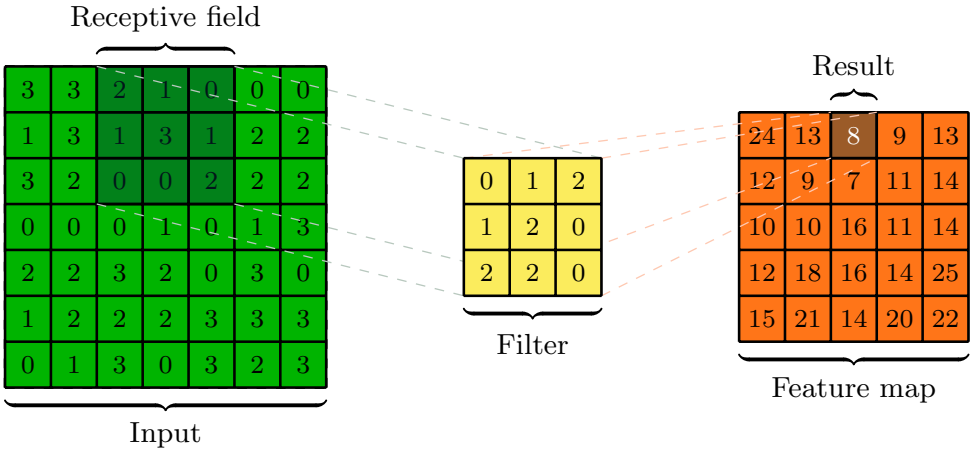


FIGURE 4.2: The components used when applying the convolution operation in a CNN; the figure shows the result of convolving the highlighted receptive field with a filter

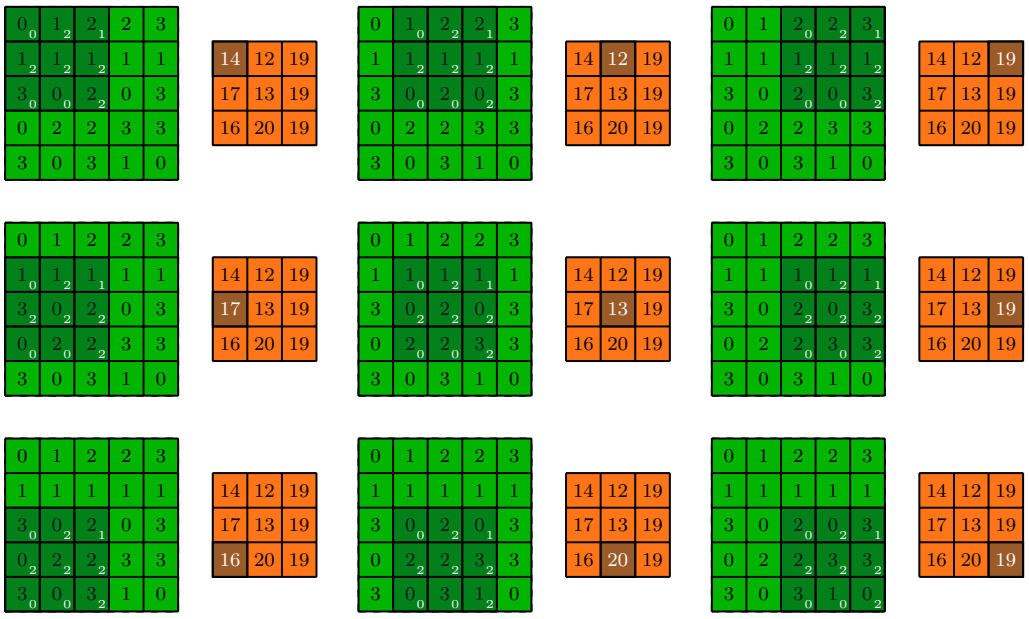


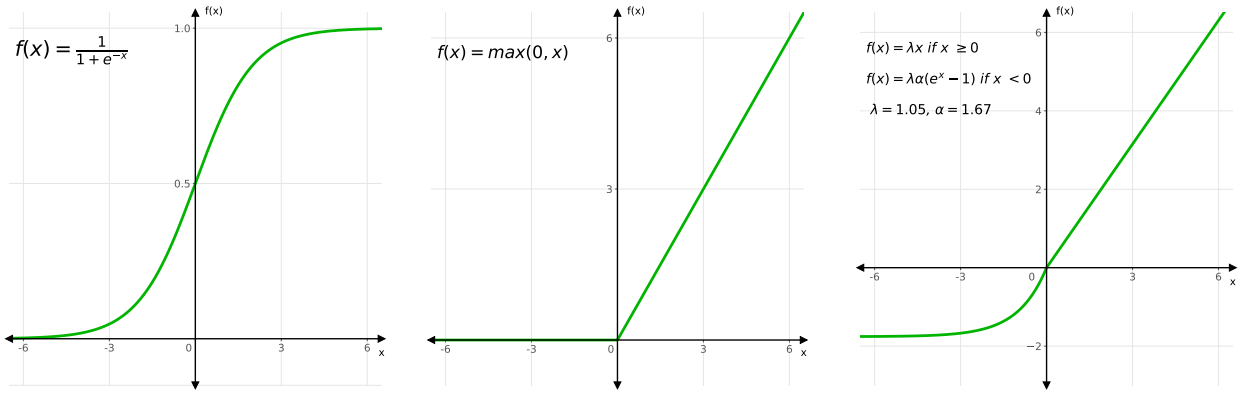
FIGURE 4.3: Unrolling a discrete convolution, with the following configuration: filter = 3×3, stride = 1, and padding=0

There are two aspects of filters which are important to understand. Firstly, the filters shown in Figures 4.2 and 4.3 have depths of one. A filter has the same depth as the input. In general, the input to a convolutional layer is an N-dimensional

tensor[3]. In the case of an image, these dimensions represent the height, width, and depth, (H, W, D), respectively. Usually, the initial input is either an RGB image (a colour image with channels for red, green, and blue), which has a depth of three, or a grayscale image, which has a depth of one. In the case of an RGB image, each respective filter will be applied to a corresponding channel of the image. The result of this process will be a feature map with a depth of one. For example, a $3 \times 3 \times 3$ filter convolved with an input will be a weighted sum of $3^3 = 27$ numbers, yielding a single value on the resulting feature map.

The second important aspect of filters is that they are constructed to detect specific features in an input. When applying a convolution, the presence of a feature, which a filter was designed to find, will result in a stronger correlation between that region of the input and the filter. This correlation will consequently be reflected in the feature map. Hence, a feature map simply shows whether or not a feature has been detected. Prior to CNNs, these filters would be hand-crafted to detect specific features, such as vertical or horizontal lines. However, in a CNN, and specifically in a convolutional layer, these features are learnt from data. A caveat is that since the filters are learnt, it is difficult to know which feature a filter is designed to detect.

The final step of creating a feature map is to introduce non-linearity to the extracted features. To do this, an activation function is applied to each value of a feature map, across all feature maps. The introduction of non-linearity increases the model's flexibility, which allows more complex features to be learnt. Often, the relationship between the input and output in classification tasks will be non-linear. Some examples of commonly used activation functions are given in Figure 4.4. Klambauer et al. (2017) recently presented the SELU activation function, which has been used successfully in other segmentation tasks (Maier et al., 2020). The values of λ and α used in the SELU function in Figure 4.4 are the same values used by Klambauer et al. (2017).



(A) Sigmoid (B) Rectified Linear Unit (ReLU) (C) Scaled Exponential Linear Unit (SELU)

FIGURE 4.4: Different activation functions which can be used to introduce non-linearity to a linear combination

Now that a comprehensive understanding of convolutions has been established, two additional considerations when using a convolution need to be discussed, namely padding and stride length. Padding refers to padding an image – usually with zeroes

[3]In layman's terms, a tensor is a generalised matrix, which can house an N-dimensional array of data

– at its borders. In the convolutions considered above no padding was used; this is also known as *valid padding*. Another type of padding is *same* padding; in this case, we pad an image such that the input is the same size as the output. Same padding, illustrated in Figure 4.5, allows for greater interaction between the pixels at the border and a filter, hence allowing for more emphasis on detecting features at the image borders. Only regions where the filter and image perfectly overlap are considered, therefore, padding can be thought of as extending an image such that there are more overlapping regions.

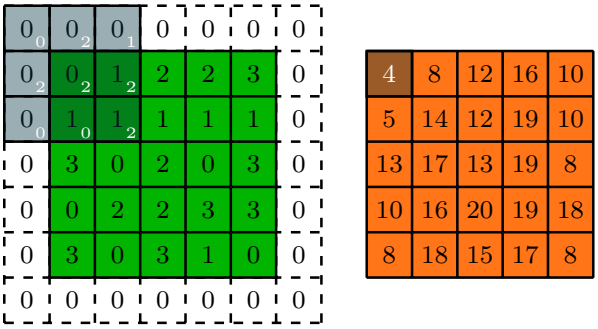


FIGURE 4.5: An illustration of same padding. Note that the output size is the same as the input size

The second consideration is stride length. The stride length controls how many steps a filter will move as it passes over an input; in Figure 4.3 a stride of 1 is used. It is possible for the horizontal and vertical stride size to be different, but in this study it will be assumed that they are the same. Setting a stride length greater than 1 can be used to downsample an image; this can also be thought of as a form of regularisation, since features are not constructed using the entire image and hence the model does not fit the data as closely. With this final component of a convolution, we can obtain a formula for the output size of a feature map, which is given below. This formula combines all the elements discussed in this section and gives an idea of how they interact to yield a feature map.

$$\text{output size} : \left\lfloor \frac{I_H + 2p_H - F_H}{s_H} - 1 \right\rfloor \times \left\lfloor \frac{I_W + 2p_W - F_W}{s_W} - 1 \right\rfloor$$

Where:
$F_{H\times W}$, is a Filter with height H and width W
$I_{H\times W}$, is an Input with height H and width W
S_H and s_W are the horizontal and vertical strides respectively
p_H and p_W are the horizontal and vertical padding respectively
we assume that the horizontal paddings and vertical paddings are the same

The Pooling Layer

The next layer in a CNN is the pooling layer. A potential flaw of feature maps is that they are sensitive to the location of features, hence even small translations of the input would result in a different feature map. To address this issue, a summary of the

feature map is rather used. This reduced version ideally captures only the important information of the feature map. A common approach to achieve this is by using a pooling operation.

Pooling creates a reduced output by returning a summary statistic for regions of the feature map, hence retaining the most important signal in the image. The application of the pooling operation results in the output being invariant to local translation. This follows since small changes in the feature map would not result in as many (if any) changes to the pooled output. This is a desirable property if the focus is whether a feature exists, rather than where the feature is (Goodfellow et al., 2016). Similar to convolutions, pooling also requires specifying a filter size and stride length. These components are applied as above, however, in this case, the filter specifies the region that a pooling operation is applied to. Moreover, the operation is applied to each feature map separately rather than across the depth of a feature map, as would be the case with a convolution. An additional difference is that since pooling summarises all points in the filter region, it can be applied to contiguous rather than overlapping regions. Padding is not usually applied during the pooling operation, but it may be used when the filter does not overlap with a feature map at its boundaries. Furthermore, pooling introduces a computational gain as it reduces the size of the input to subsequent layers, hence speeding up calculations at these layers.

Max pooling, a common type of pooling that returns the maximum value in the region considered, is used in this study. This is illustrated in Figure 4.6 where the value 3 is returned since it is the maximum value in the region considered. Note that minor changes in the feature map would not cause a notable change in the output, hence the property of local translation invariance.

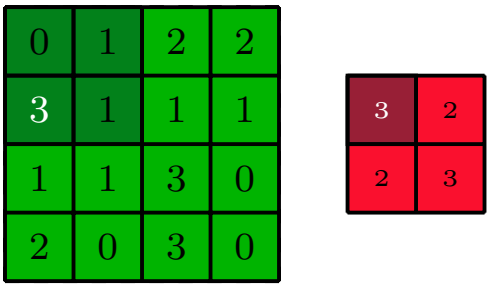


FIGURE 4.6: An illustration of max pooling

The Fully connected layer

The final layer in a CNN is the fully connected layer, which is in fact a feed-forward neural network. In the case of a CNN, the input to this model is a feature map tensor that has been flattened into a vector. Additional transformations are made to the input using hidden layers. These layers connect to the output layer, which provides an output as its name suggests. Figure 4.7 shows a simple feedforward layer, used for a binary classification task. In this case, the output layer produces probabilistic outputs, which are bound between 0 and 1, by using a sigmoid activation function.

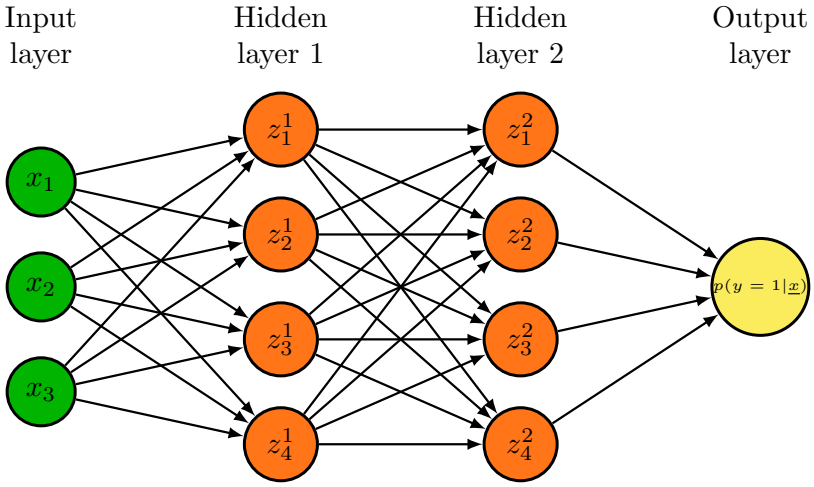


FIGURE 4.7: A fully connected layer with two hidden layers and a single output

The complete model

A complete CNN model, which combines all the layers discussed above, is shown in Figure 4.8. This is a CNN applied to a binary classification task that detects whether a grayscale image is a leaf. Using this figure, the complete process of classification will be explained.

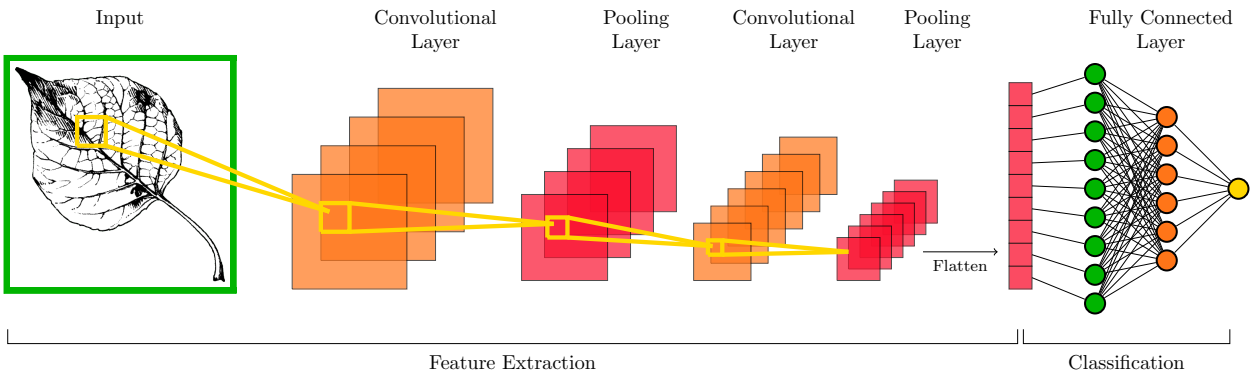


FIGURE 4.8: A complete Convolutional Neural Network, which combines convolutional layers, pooling layers, and a fully connected layer

The feature extraction process of a CNN begins by applying a filter to an image. Multiple filters can be applied at a convolutional layer, where each filter would detect a different feature, the presence of which would be represented in the resulting feature map. The depth of the feature map would be equal to the number of filters, and the filters in the convolutional next layer would need to have the same depth. In the CNN shown in Figure 4.8, four filters are applied to the input image, which is determined by the number of feature maps shown in the first convolutional layer. Following this, a pooling layer is applied to summarise the feature maps. In this case, the number of feature maps is equal to the number of feature maps in the previous convolutional layer.

The process of applying convolutional layers followed by a pooling layer is repeated

many times. This repetition allows for simpler features created earlier in the network to be combined later in the network to create more complex features. These later features will have much larger effective receptive fields compared to earlier features. Therefore, they will implicitly see more of the original image than earlier features. Considering the yellow filters in the example image, each filter consists of the values obtained by applying multiple filters at the previous levels; if this process is traced back to the input, it is clear that later levels will be indirectly exposed to more of the input image. Furthermore, pooling layers reduces the influence of noise in later, possibly more complex, features, by only allowing the important signal of earlier feature maps to be propagated forward.

Once enough pooling and convolutional layers have been applied, the resulting features will need to be classified. To do so, the final pooling layer will be flattened into a vector. If there are multiple feature maps, each map will be flattened and the vectors will be stacked to form one long vector. This vector will then be used as input to the fully connected layer. This process of classification is exactly the same as initial attempts to classify images using feed-forward neural networks. The main difference of a CNN is the initial stage of feature extraction, which can be optimised for the specific task at hand.

CNNs have shown state-of-the-art performance in image classification tasks. Previous models which performed well on image classification tasks used extensive feature engineering followed by a classification model like, for example, a support vector machine (Lowe, 2004; Dalal and Triggs, 2005). In contrast, a CNN combines both feature extraction and classification in a single model, allowing for end-to-end training. This allows a CNN to "learn" what the best features are for the classification task at hand. The fully connected models, which will be discussed next, leverage the feature extraction component of the CNNs.

4.4 Fully Convolutional Models

Image segmentation was introduced in Section 2.2. In this task the input is an image and the response is a single channel matrix, representing which class a pixel belongs to. Therefore, the goal of this task is to take in an input and return a segmentation map. In this study specifically, the goal is to take in an image of a differenced leaf and return a mask indicating where the input has embolised. To achieve this, fully convolutional models can be used. Specifically, three fully convolutional models will be used in this study: U-Net, U-Net with a ResNet34 backbone, and W-Net. In order to describe these architectures, we first need to introduce the concept of a transposed convolutional layer, which is core to these models.

4.4.1 Transposed Convolutions

A transposed convolution is used to project the input to a higher dimensional[4] space; in this context it can be thought of as the opposite to the convolutions discussed in Section 4.3.1. While a convolution aims to compress an input to a lower dimension, aiming to retain only the important signal, a transposed convolution aims to distribute the signal in the input to a higher dimension. This operation is vital in computer vision scenarios where an entire matrix is required as output, rather than a single classification.

[4]The area of the filter is its dimension, where the values in the filter represent coordinates

Although transposed convolutions can be thought of as opposite to a convolution, they can always be expressed equivalently as a convolution. To achieve this equivalent expression, rows and columns of zeros are added to the input. Exactly how these zeroes are added depends on what output size is required. In our task we want to reverse the size reduction introduced by a convolution. Consider an input that goes through a normal convolution such that the resulting output is smaller than the input. If we want to then return this output to the same size as the initial input we can use a transposed convolution. The configuration used in the transposed convolution will need to take into account the configuration used in the convolution. Specifically, we will look at three configurations applied in the fully convolutional networks used in this study. These cases are: valid padding with unit strides, same padding with unit strides, and valid padding with non-unit strides.

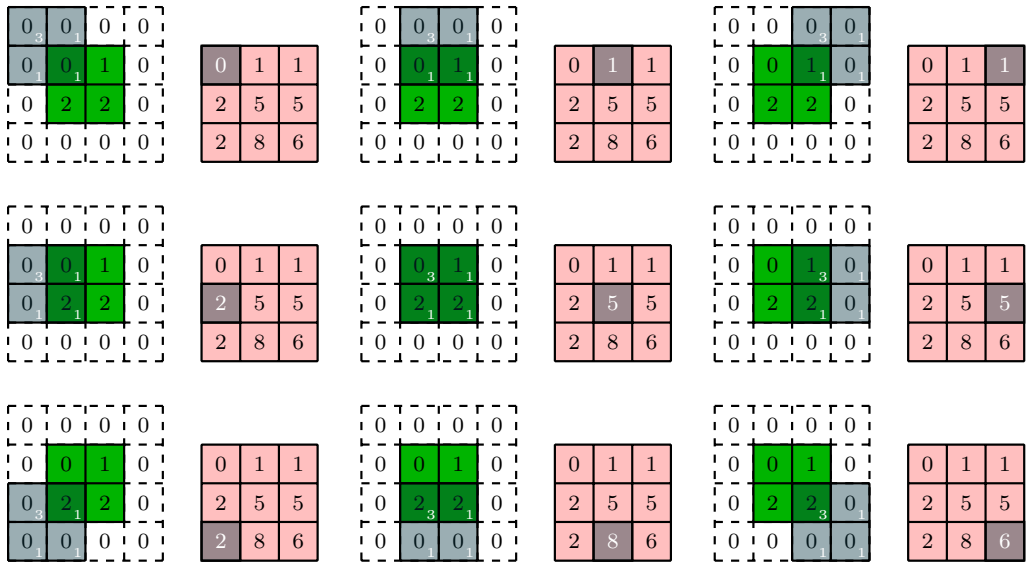


FIGURE 4.9: Unrolling a transposed convolution with the following configuration: filter=2×2, stride=1, and padding=1

An unrolled transposed convolution, for the case where the size reduction of convolution using a valid padding with unit strides is reversed, is shown in 4.9. As can be seen when inspecting the image, the input, which is a 2×2 matrix, is fully padded. This holds in general - that is, using a transposed convolution to reverse the size reduction of a convolution applied with valid padding and unit stride, is equivalent to using a fully padded convolution with unit strides (Dumoulin and Visin, 2018). As padding in the normal convolution increases, the amount of padding used in the transposed convolution would decrease, since the size reduction of the convolution would be less due to the padding. A special case is the case where same padding is used, in this case the transposed convolution would also use same padding since the input size is the same as the desired output size.

Another name for a transposed convolution is a fractionally-strided convolution. This name arises from the case where we want to reverse the size reduction of a convolution where the stride used was greater than one. Figure 4.10 shows a transposed convolution where the convolution used a stride length of two and same padding; this approach to downsampling a feature map is used in a ResNet, which is discussed in

4.4.3. Using a stride length greater than one means that the filter would move over an image faster than for a unit stride. In the case of a transposed convolution the opposite is done, that is the filter needs to move over the input slower than for a unit stride. To achieve this, zeroes are inserted between the values of the input, as shown in Figure 4.10. The number of zeros added between the input values increases as the stride length increases.

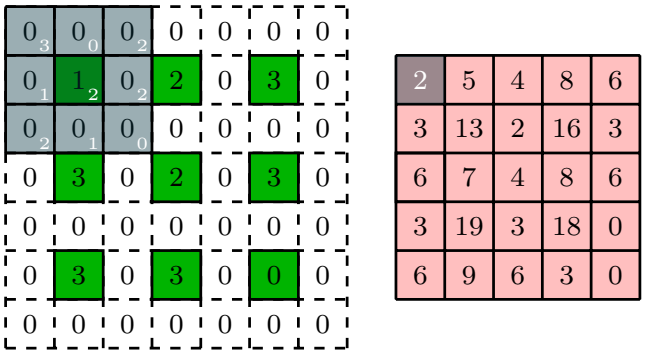


FIGURE 4.10: Transposed convolution with the following configuration: filter = 3×3, stride = 1, and padding = 1. The input (green) was obtained by applying a convolution to the output (pink) with the following configuration: filter = 3×3, stride = 2, and padding = 1

It is worth noting that there are many operations which can be used to upsample the size of an input in addition to a transposed convolution. Some examples include nearest neighbour interpolation, bi-linear interpolation, and max unpooling. The difference between transposed convolutions and these operations is that it has a learnable filter which allows it to determine the optimal way to distribute the input signal to a higher dimension. Although, if one of the other methods are used, they can be followed by a normal convolution to achieve a similar mechanism. Transposed convolutions are preferred for their simplicity of doing both operations at once. With this basic understanding of how inputs can be upsampled, fully convolutional models can be discussed.

4.4.2 U-Net

U-Net, shown in Figure 4.11, is a popular choice for image segmentation tasks and is the initial model that will be used in this study. The network's U-shape architecture is the source of its name. This network consists of a contracting path and an expansive path. In this network, the fully connected component of the CNN is replaced by the expansive path. This allows the network to return a segmentation map rather than an image.

The motivation for using the contracting path is the same as for a CNN: the contracting path is used as a feature extractor. The contracting part consists of all layers prior to the bottleneck, which is at the middle of the "U" of the architecture. In other words, it consists of all layers on the left of the figure, above the last red arrow. At each step there are 2 convolutional layers, and each layer is followed by a ReLU activation. The stacking of convolutional layers was inspired by VGG, as it allows

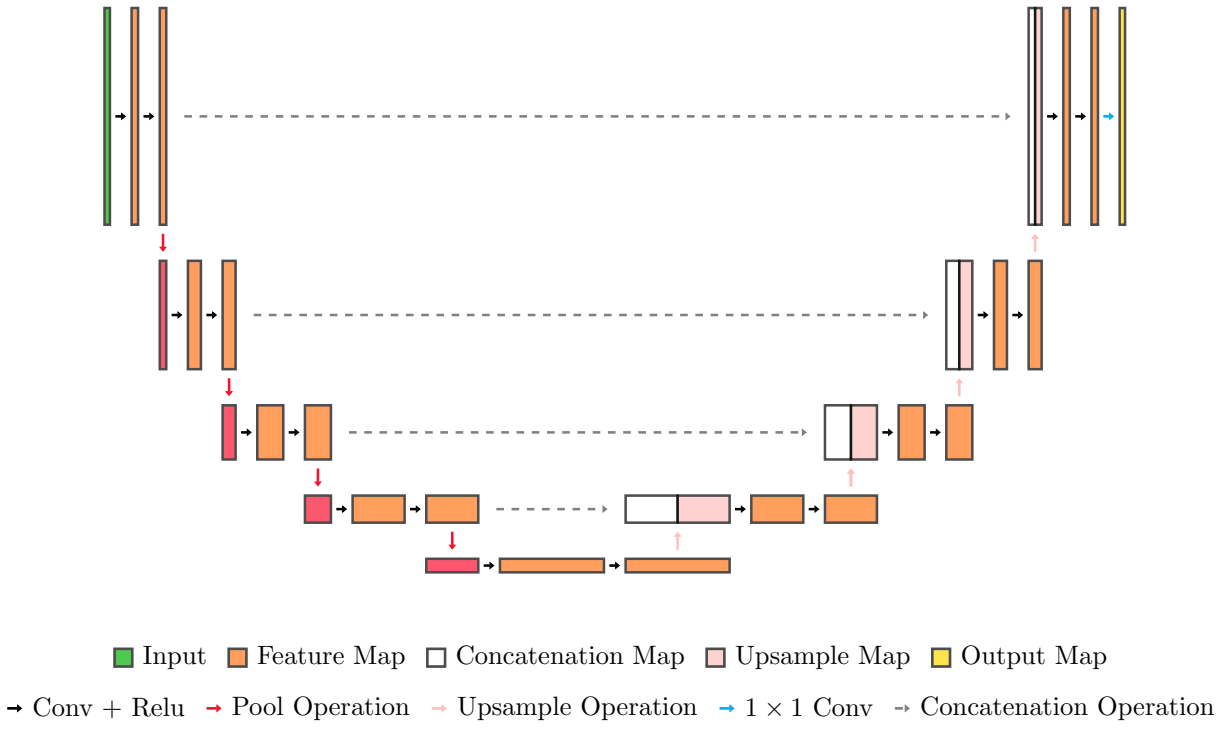


FIGURE 4.11: An illustration of a U-Net that takes in an image (green) and produces a segmentation map (yellow)

for larger receptive fields while using fewer parameters. As an example, consider a stack of 2 layers using 3×3 filters. This stack will have a receptive field size of 5×5. Assuming the input has C channels, the number of weights for the stacked layer is $2(3^2C^2) = 18C^2$. In contrast, the number of weights for a single convolutional layer with a 5×5 filter, will be $5^2C^2 = 25C^2$, resulting in 39% more parameters. Owing to this stacking achieving a larger receptive field, the network exclusively uses 3×3 filters in its convolutional layers.

The contracting path is used to compress the input image. The idea is that only the signal in an image important for extracting a mask should be retained. Hence, we squeeze an image into a bottleneck where features are used to retain the signal important to the task, while concurrently a reduction in image size is used to remove noise and focus the features. This is achieved through using two processes. The first process is pooling, which was explained in Section 4.3.1. At each step, excluding the first, a pooling operation with a 2×2 filter is used to halve the input from the previous step. In this study, max pooling is used. The second process is the doubling of the number of filters at each step. This compensates for the loss of information caused by pooling by creating both more and increasingly complex feature maps. The first step starts with 64 features which increases to 1024 by the bottleneck.

The bottleneck of the network is between the contracting and expansive path. It contains the most summarised version of the input with the most features. Using the final set of feature maps from the bottleneck, the expansive path aims to recreate a segmentation map.

The expansive path, in contrast with the contracting path, aims to decompress the signal in the bottleneck, distributing it across feature maps that double in size at

each step. To create these larger feature maps, a transposed convolution is used, which reverses the size reduction of the corresponding convolution in the contracting path, as explained in Section 4.4.1. However, an issue with this decompression is that we are trying to recreate an output from a bottleneck that is $\frac{1}{16}^{th}$ of the target size; compressing the initial input to this extent results in a loss of spatial resolution. In other words, when upsampling, spatial correlation from the original input may be lost. To address this, the last feature map (prior to pooling) at the corresponding step in the contracting path is appended to the upsampled feature map. By appending this feature map, we are able to add some of the missing spatial correlation. We can think of this concatenation as providing additional information as to how the signal should be distributed in the newly upsampled feature map[5]. Therefore, the depth of the first feature map at each step of the expansive path will be double the size of the other feature maps in the same step. The other feature maps in the step are convolutional layers that match the contracting path in configuration. The notion here is that by using these layers we can effectively combine the concatenated feature map and further refine the result.

The final step in the expansive path is to return a segmentation map. In general, the depth of this map will be equal to the number of classes in the task. However, in the case of binary segmentation, the output map will be a single channel where each pixel will represent $\Pr(Y = 1|X)$. In this application, each output pixel will represent the probability that an embolism has occurred at that pixel location in the input. One problem, however, is that at the final layer of the expansive path the final convolutional layer has a depth of 64, rather than the single channel we require. To achieve a single channel, a 1×1 convolution is used. This type of convolution can be used to change the depth of a feature map by expanding or combining the existing features element-by-element along the depth axis. This is then followed by a sigmoid activation function which is used to squeeze the resulting logits between 0 and 1, hence creating classification probabilities.

While U-Net is an efficient architecture, many variations have been suggested, as discussed in Section 2.2.3. One such variation, which will be used in this study, is the U-Net architecture that uses ResNet34 as a backbone.

4.4.3 U-Net ResNet-34

One class of U-Net variations maintains the "U" structure of the network, but rather than using the same number of steps and layer configuration, a more popular CNN configuration is used. To do this, the fully connected component of the popular CNN configuration is replaced by an expansive path symmetric to the contracting path. When this is done, the underlying network is called the *backbone* of the U-Net. Examples of backbones used are AlexNet, VGG, and ResNet variants. This section will focus on explaining both ResNet34 and how it can be used as a backbone for the U-Net. A U-Net with a ResNet34 backbone is shown in Figure 4.12.

ResNet34 obtains its suffix due to having 34 layers – if we count the fully connected layer as one layer [6]. The first layer in ResNet34 is a convolutional layer with 64 7×7 filters. This layer is then followed by a max pooling operation, which is the only pooling layer used in this network. All remaining layers have 3×3 filters which are

[5] In the original U-Net architecture this operation is a copy-and-crop operation. In this application, tile sizes are chosen such that it is not necessary to crop the contracting path feature map.

[6] In this application the fully connected layer and preceding average pooling layer will not be considered.

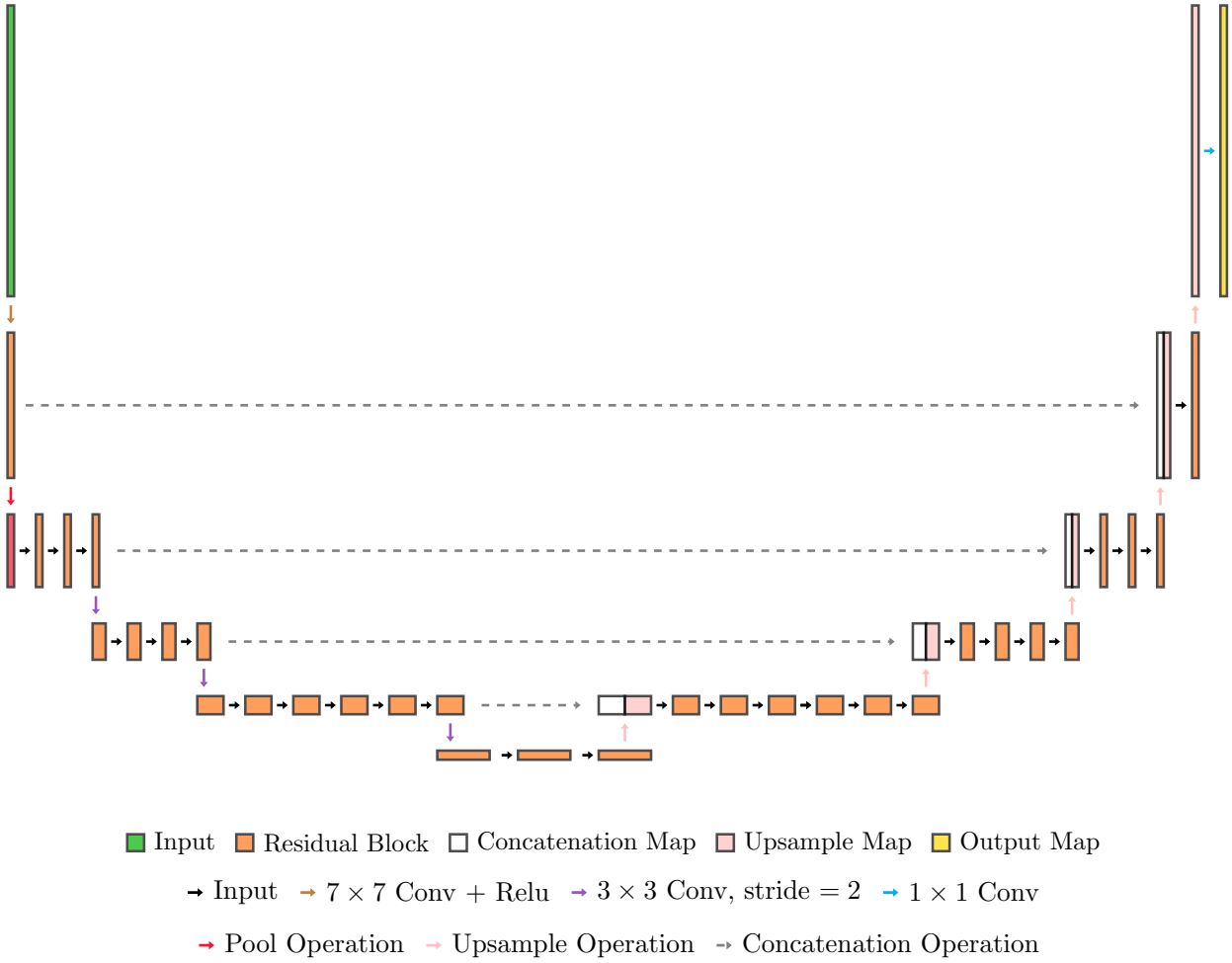


FIGURE 4.12: An illustration of a U-Net with a ResNet-34 backbone which takes in an image (green) and produces a segmentation map (yellow). The contracting path of the model is a ResNet-34 architecture, sans fully connected layer.

stacked together, also inspired by VGG. However, in contrast to U-Net, this network groups a stack of two convolutional layers together into a residual block.

Residuals blocks are core to a ResNet. As explained in Section 2.2.1, this block allows a network to use an identity mapping which prevents the network from needing to learn this mapping as a non-linear function. A residual block is illustrated in Figure 4.13. In the remainder of this paragraph, the idea of a residual block will be examined more formally. The input to the stack of convolutional layers is x. Let the application of all operations in this stack be denoted by $F_{\underline{w}}$, where this function is parametrised by a weight vector $\underline{\mathrm{w}}$. The output of this stack will be given by $F_{\underline{w}}(x)$. The residual block adds a shortcut connection between this output and the original input, to return $F_{\underline{w}}(x) + x$. When a network is sufficiently deep, it is possible that further operations – $F_{\underline{w}}$ – can degrade the input x; this includes the degradation of training loss. It is important to note that when a network is sufficiently deep, the input is a feature map which is a cumulation of many operations similar to $F_{\underline{w}}$. In cases where this feature map cannot be improved upon by the network, we would want to keep the input

to the residual block unchanged; that is, the output of this layer should be x. One possible way to do this would be for the network to learn $F_{\underline{w}}$ as an identity mapping – $F_{\underline{w}}(x) = x$. However, learning such a mapping is quite difficult as it requires a non-linear function that maps x onto itself. A simpler approach taken by the author is to use a shortcut connection. In this approach the network only needs to learn to set $\underline{w}$ to, or close to, zero, such that $F_{\underline{w}}(x) = 0$. Hence, $F_{\underline{w}}(x) + x = 0 + x = x$, which is an identity mapping. This allows for a deeper network to train effectively.

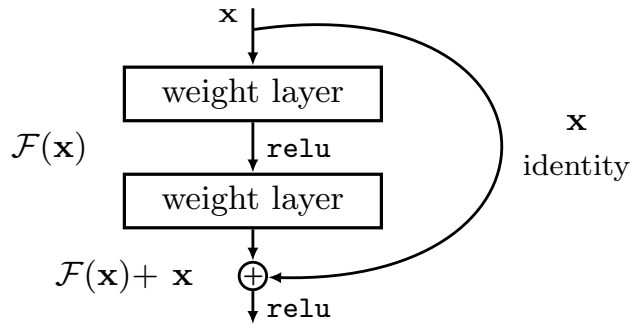


FIGURE 4.13: An illustration of a residual block

ResNet34 stacks multiple residual blocks before downsampling the feature map size. Similar to U-Net, the number of filters in a residual block remains unchanged unless the height and width of the feature maps in the stack have halved. However, in contrast to U-Net, rather than using a pooling operation, a convolutional layer with a stride of two is used to down sample the image.An issue with downsampling in a ResNet is that a dimension mismatch between the block's output and input is introduced. To remedy this, a 1×1 filter with a stride of two is used, which increases the number of channels while halving the height and width of the input.

An additional operation used in ResNet is batch normalisation (Ioffe and Szegedy, 2015). This is used after each convolutional layer prior to applying the ReLU activation. When training neural networks, the distribution of the input to a layer changes as the parameters of the previous layer are updated. Ioffe and Szegedy termed this phenomenon *internal covariate shift*. To address this, the inputs to layers are first standardised ($\mu = 0, \sigma = 1$) and then scaled and shifted. The mean and variance used for standardisation are estimated using the current mini-batch. Due to this, batch normalisation also has a slight regularisation effect since the estimated mean and variance add noise to an output. The size of this regularisation effect decreases as batch size increases. Batch normalisation makes latter layers more robust to changes earlier in the network since the distribution of the layer will be more stable regardless of changes in the input to the layer. This promotes independence between layers, hence promoting stability and speeding up learning.

To use ResNet34 as a U-Net backbone, and hence convert it to a FCN, an expansive path must be created. A stack of 3 residual blocks with a depth of 512 is used as the bottleneck layer. To achieve the U-shaped architecture, the expansive path is symmetric to the contracting path. As with U-Net, a transposed convolution is then used to upsample the dimensions of the feature maps. The number of filters is halved at each step, and the dimensions of the feature map match the corresponding feature map in the contracting path. The final feature map is then reduced using a 1×1 convolution followed by a sigmoid activation function.

Changing the backbone does not make structural changes to the original U-Net. This is in contrast to the model that will be discussed next, which is a repeated U-Net variant.

4.4.4 W-Net

W-Net is a relatively simple network that stacks two mini U-Nets together. The W-Net architecture is shown in Figure 4.14. The U-Net used in W-Net is prefixed by the term "mini" due to the low number of parameters it has. A single mini U-Net has a similar structure to its larger counterpart; however, rather than having four steps each in the contracting path and expansive path, it only has two. The number of filters in the mini U-Net starts at 8, increasing to 32 at the bottleneck. In addition, a mini U-Net differs in that it uses a residual block at each step; residual blocks were explained in Section 4.4.3.

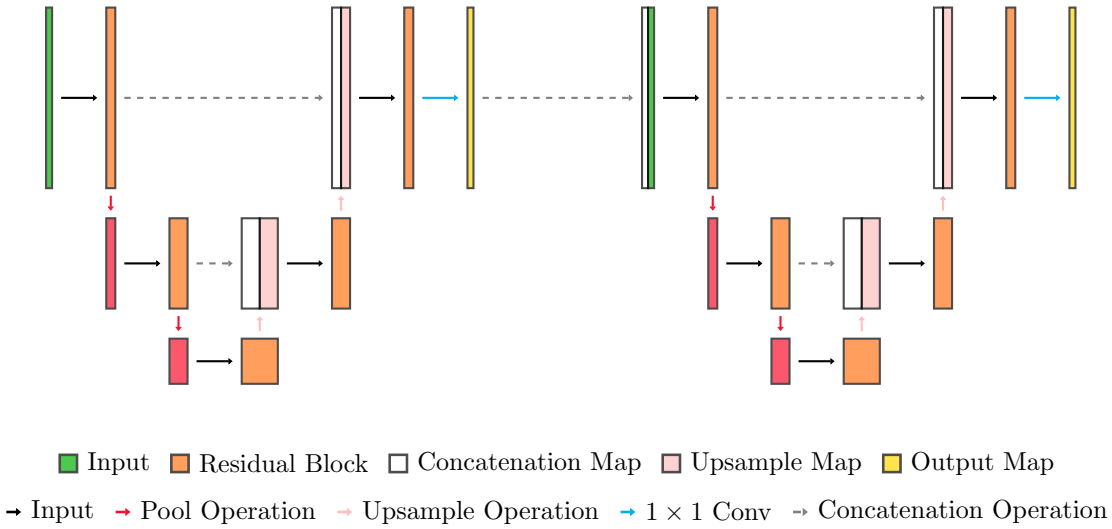


FIGURE 4.14: An illustration of a W-Net which takes in an image (green) and produces a segmentation map (yellow). It repeats two mini U-Nets and uses the output from the first to improve the output from the second.

The output from the first mini U-Net is concatenated to the original input. This stack is used as the input to the second mini U-Net. The motivation for the stack is that the segmentation map output from the first mini U-Net can be used to highlight which regions of the input the second mini U-Net should focus on. Specifically, in this study, since the segmentation map is a single channel with probabilities as elements, it weights the pixels of the input. A higher probability of embolism assigns more weight, and hence focuses the network on the corresponding input pixel. To train the network, each mini U-Net is assigned its own loss, which is summed to give the total loss of the network.

4.5 Training

The goal of the learning problem, expressed in terms of a model, is to choose a function from the model's functional space that most closely relates to the true functional relationship between X and Y. In order to choose this function we will need some

way to determine what constitutes a suitable function. This is determined by applying some evaluation criterion to the available data. The better the choice of this criterion and the more representative the available data, the closer we can get to an optimal solution. This evaluation criterion is called an objective function. When an objective function is minimised – which is typically the case when determining a model's weights – it is called a cost function or loss function. In this study, we will use the term loss function. The loss function determines how well the output produced by a candidate function matches the response.

Once a loss function has been chosen, we try to find a set of parameters that minimises the loss – in other words, we try to choose the optimal function from the functional space of the model. This search is performed using an optimiser. Hence, to train a model, the following components are required: a model, a loss function, and an optimiser. Loss functions and optimisers will be discussed next.

4.5.1 Loss functions

Binary cross entropy (BCE), also known as log-loss, is a commonly used loss function for binary classification[7]. The formula for weighted cross entropy (WCE) – an extension of binary cross entropy – is given in equation 4.1. If $\beta = 1$, then the expression is the same as for binary cross entropy. In this study, N represents the pixels across the relevant images; that is, the loss is applied at a pixel level.

$$L = -\frac{1}{N}\sum_{i=1}^{N} y_i \beta \log(\hat{p}) + (1 - y_i)(1 - \beta) \log(1 - \hat{p}) \tag{4.1}$$

Where:

- $\hat{p} = p(Y_i = 1|X_i)$
- $\beta \in [0, 1]$

The β parameter can be used to weight each term in cross entropy loss. Inspecting equation 4.1, we see that a higher value of β would assign a higher weight to the misclassification of the positive class and a lower weight to the misclassification on the negative class. This is a useful property when there is a large class imbalance, as it allows for a higher weight to be placed on misclassification of the minority class. One suggestion for choosing this weight is to use the class imbalance present in the dataset. Specifically, $\beta = 1 - \frac{\sum_{i=1}^{N} y_i}{|Y_N|}$ where Y_N is the set containing all y values (Zhou et al., 2017; Lin et al., 2017).

When this weight is used, the loss is called class-balanced cross entropy (Xie and Tu, 2015). Using this weight equates to applying a higher weight to misclassifying an embolism, where this weight is proportional to the number of non-embolism pixels. In contrast, misclassification of a non-embolism pixel would have a smaller weight since there are much fewer embolism pixels. Hence this weight balances the contribution of both embolism and non-embolism pixels to the loss function. This is used to further address the within image class imbalance raised in Section 3.3.

[7] Binary cross entropy is a special case of cross entropy which extends to multiple classes.

The final loss function considered in this study is called focal loss. It is a variant of cross entropy that was proposed for severely imbalanced datasets. The specific version of focal loss used in this study is a weighted variant, which Lin et al. (2017) found worked better in practice. The formula for this variant is given in equation 4.2.

$$L = -\frac{1}{N}\sum_{i=1}^{N} y_i\beta(1-\hat{p})^{\gamma}\log(\hat{p}) + (1-y_i)\hat{p}^{\gamma}(1-\beta)\log(1-\hat{p}) \tag{4.2}$$

Where:

- $\hat{p} = p(y_i = 1|X_i)$
- $\beta \in [0, 1]$
- $\gamma \geq 0$

This formula for weighted focal loss is almost identical to weighted cross entropy, apart from the additional $(1-\hat{p})^{\gamma}$ and $\hat{p}^{\gamma}$ terms. If we define p_t:

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

then we can rewrite the additional terms as $(1-p_t)^{\gamma}$. This term is a modulating factor, which down-weights "easy" examples, hence placing more importance on correctly classifying "harder" examples. An easy example is one that has been correctly classified with a high probability. Therefore, rather than focusing on these cases, the loss assigns more weight to examples which the network is confidently wrong about. Lin et al. (2017) found in their experiments that in a largely imbalanced dataset, easily classified negative examples overwhelm cross entropy and contribute to the majority of the loss. Hence, the modulating factor adds to weighted cross entropy which is only able to adjust the importance of negative classes compared to positive classes.

The γ term in the modulating factor is a hyperparameter which controls the effect of the modulating factor. When $\gamma = 1$, the weighted focal loss is equivalent to weighted cross entropy. The authors found $\gamma = 2$ to work best in their experiments. To avoid the complexity which would be introduced by tuning this parameter, this study uses $\gamma = 2$.

Once a loss function has been chosen, we have a way of ranking our candidate solutions, and hence have a way to find an optimal solution. This optimal solution determines the optimal parameters to use. In the case of FCNs, a numerical optimisation algorithm is usually used since a closed-form solution either does not exist or is intractable.

4.5.2 Optimisers

A common type of optimisation used in neural networks is gradient-based optimisation (Ruder, 2016). Two gradient-based optimisation algorithms were used in this study: mini-batch gradient descent and the Adam optimiser. These algorithms are explained briefly in this section. For more details on the algorithms, please see appendices A and B respectively.

Gradient-based optimisation algorithms – a popular approach in optimising neural networks – are iterative algorithms that involve updating a model's weights by taking steps in the opposite direction of the gradient of the loss function in an attempt to reach a global minimum. Each algorithm contains an update rule, which informs how to use the gradient to update the current weights at step i for the next step $i+1$. The hyperparameter that controls the size of the update to the current weight is called the learning rate, η. The loss surface is incredibly complex and will have many local minima in addition to the global minimum. An ideal learning rate will be such that it is large enough to avoid local minima while being small enough to find the global minimum.

Mini-batch gradient descent uses a subset of m data points such that the update is made on an average trend, hence reducing the variance of updates. Smaller batch sizes can be thought of as a form of regularisation since using individual examples would introduce more noise. Moreover, using smaller batches is more computationally efficient than for batch gradient descent, since it allows computation to be done in parallel[8]. Stochastic gradient descent (SGD) is often used to refer to mini-batch gradient descent, and this convention will be adopted in this study[9].

In some cases the SGD algorithm will struggle to converge, particularly when the loss surface is hard to navigate. To address this, a term called momentum (Polyak, 1964) is included. The momentum term is recursive and contains a hyperparameter, α, set between 0 and 1. This gives SGD some memory; when gradients are in the same direction for consecutive steps the algorithm speeds up since the previous gradients will reinforce subsequent ones, and when they are not in the same direction the step size will be dampened. The larger α is relative to the η, the larger the influence of previous gradients (Goodfellow et al., 2016). Although momentum improves the convergence speed of SGD, a possible caveat of SGD is that it updates all parameters equally.

A newer optimiser that addresses this issue is the Adam optimiser – derived from adaptive moment estimation (Kingma and Ba, 2014). The Adam optimiser adapts the learning rate per parameter, which allows the update size to be based on feature importance. Lower learning rates would be used for more important features since this would allow for smaller updates and hence a refined search (Ruder, 2016). The authors of the Adam optimiser aimed to combine the best properties of two other optimisers, namely RMSProp (Tieleman and Hinton, 2012) and AdaGrad (Duchi, Hazan, and Singer, 2011), but perhaps it is best seen as a combination of RMSprop and momentum, with a few distinctions (Goodfellow et al., 2016). The algorithm uses an exponentially decaying average both of the first-order moment – similar to momentum – and of the (uncentred) second-order moment – similar to AdaGrad and RMSprop – where the moments are taken with respect to the loss gradient (Ruder, 2016).

There are two additional components important in optimising the weights of a neural network – initialisation and backpropagation – which will now be introduced. To start the optimisation process, weights need to be initialised. Constant weight initialisation will perform poorly, as neurons will evolve symmetrically due to having an identical influence on the loss function (Katanforoosh and Kunin, 2019). Hence, the initialisation should be done randomly. Although many initialisation schemes exist,

[8]Mini-batch sizes are usually chosen in powers of 2 as this is more computationally efficient for vectorized processes.

[9]Stochastic gradient descent technically refers to the case when the batch size is equal to 1.

this study uses two popular schemes: the Glorot Uniform initialisation and He Normal initialisation. The Glorot Uniform initialises weights in each layer by drawing from the Uniform distribution $U\left(-\frac{\sqrt{6}}{n_l+m_l}, \frac{\sqrt{6}}{n_l+m_l}\right)$, where n_l and m_l are the number of inputs and outputs from the layer respectively. The He Normal initialisation initialises weights at each layer from a truncated normal distribution centred on zero with $\sigma = \sqrt{\frac{2}{n_l}}$. In both cases, the biases are initialised as a constant zero tensor.

Both numerical optimisation schemes introduced in this chapter require gradients to be computed. In a neural network, computing gradients can be quite complicated. Hence, when calculating the gradients in neural architecture, an algorithm called backpropagation is used. Backpropagation has two steps. The first step is a forward pass that calculates the output for the current step for a given input. The second step is a backward pass, which, by using the chain rule, calculates the partial derivatives (gradients) of each weight and bias in the network with respect to the loss function for the current step. In the case of a layer with a filter, the partial derivative is taken with respect to each weight in the filter. Using these derivatives, the chosen optimiser informs how the current weights should be updated. Hence, the backpropagation algorithm is the process of updating the weights in a network, whereas an optimiser determines what the optimal update will be. The backpropagation algorithm allows for networks to be trained end-to-end.

4.5.3 Model tuning

Tuning a model updates the functional space, making it easier to find an optimal solution for the problem at hand. Tuning is an important task, as the choice of hyperparameters influnces the final solution. There are many methods for tuning hyperparameters; common methods include random searches or a grid search, which is usually performed using K-Fold cross-validation. However, these methods can be computationally expensive if a single training pass is expensive, as is the case in this task. A more efficient alternative, which minimises the required number of evaluations, is Bayesian hyperparameter optimisation (BHO) (Snoek, Larochelle, and Adams, 2012; Brochu, Cora, and Freitas, 2010).

BHO broadly has three components: an objective function, a surrogate function, and an acquisition function. The objective function is the quantity of interest to optimise. Validation loss is a common choice for an objective function, but it is not used since the loss function is treated as a hyperparameter. Therefore, validation AUC is rather used as the objective function; AUC will be introduced in Section 4.6.1. Assessing the validation AUC for multiple hyperparameter configurations is computationally expensive. An alternative is to use a surrogate function, which serves as a proxy for the objective function. Specifically, the surrogate function in BHO is a posterior probability, where the prior is a Gaussian process (GP) – which is a function of hyperparameters to optimise – and the likelihood is a function of the results of the objective function. Using the surrogate function, an acquisition function optimally determines the next vector of hyperparameters to evaluate. This point is then evaluated using the true objective function, and this process is repeated. Hence, as more trials are performed, there are more evaluation results that can be used to update the GP prior, resulting in a more informed posterior. Through this process, BHO is able to effectively search the hyperparameter space making an informed selection of which hyperparamters to try.

Hyper Parameter	Specifications
Activation	ReLU, SELU
Initialiser	He Normal, Glorot Uniform
Filters	1, 2, 4, 8 [10]
Optimiser	Adam, SGD
Learning Rate	[0.0001, 0.01]
Momentum (SGD)	[0.5, 0.9]
Loss	BCE, WCE, Focal
Loss Weight	[0.5, 0.99]

TABLE 4.1: The hyperparameter search space

The hyperparameter search space for each model is given in Table 4.1 above. The loss function was treated as a hyperparameter due to the infeasibility of training a model for each loss function. Moreover, batch size was not optimised due to the limitation of GPU RAM. A parameter included in the search space that has not yet been introduced is the filter multiple, referred to as Filters in Table 4.1. The initial filter size was set to 8 in each model; the number of filters in each subsequent step is based on the previous as explained in the description of each model. This base is multiplied by $Filters$. At the highest filter multiple, both U-Net and U-Net (ResNet34) will be at their conventional size, while at the lowest filter multiple, W-Net will be at its conventional size. Hence, the filter multiple is a way to control model complexity. It will determine if simpler structures are better for U-Net and U-Net (ResNet34) and if a more complex structure is better for W-Net. The use of the filter multiple can be thought of as an approach to regularisation.

4.6 Assessing performance

4.6.1 Metrics

Metrics are used to evaluate how well a model has performed. They are usually more intuitive than a loss function and, in addition, multiple metrics can be applied to a single problem, each summarising a different property of the model. A loss function must meet certain properties and is designed to be optimised. However, despite these differences, a loss function should relate closely to the metrics which are important to your problem. In some cases a metric can also be used as a loss function. The metrics used in this study are precision, recall, accuracy, F1-score, and AUC. Each of these metrics will be explained briefly below.

Four of the metrics above – precision, recall, accuracy, and F1-score – can easily be explained using a confusion matrix. A confusion matrix for a binary classification problem is shown in Figure 4.15. A confusion matrix shows a comparison of predictions against the corresponding response. Often in image segmentation tasks, a metric called Intersection over Union (IoU) is used to determine if an image has been correctly classified. If IoU is above a certain threshold, then the entire image is considered to be correctly classified. In this study, this approach will not be adopted[11], rather the confusion matrix will be determined on a per-pixel basis. This means that the classification of every pixel will be looked at independently. This was done as it is

[10] W-Net did not include a filter multiple of 8.

[11] IoU was not included for brevity, given its correlation with F1.

both a fairer and stricter assessment, especially taking into account the severe class imbalance present in this task.

To determine if a prediction is positive or negative, a threshold is used. If the predicted probability is greater than the threshold, the prediction is assumed to be positive, otherwise it is negative. If a predictions is correctly classified as positive, then it is a true positive (TP), and if a prediction is correctly classified as negative, then it is a true negative (TN). If the true label is negative, but the sample is classified as positive, then it is a false positive (FP) and, similarly, if a positive sample is classified as negative, then it is a false negative (FN). Using these terms we can now provide definitions for the aforementioned metrics.

Accuracy is the total number of correctly classified pixels out of the total number of pixels and is given by $\frac{TP+TN}{TP+TN+FP+FN}$. Accuracy provides information as to how accurate a model is. However, when there is a large class imbalance accuracy is inflated. Consider the case where there only exists 10 positive samples out of 1000 samples, and consider a classifier that only predicts negative samples. This classifier would have an accuracy of 99%, despite predicting no positive classes correctly.

In such cases, precision, recall, and the F1-Score are more informative. Precision gives the proportion of positive predictions that indeed had positive labels, and is given by $\frac{TP}{TP+FP}$. Precision informs how precise a classifier is, and is a measure of prediction purity. Recall gives the proportion of positive labels that were classified as such and is given by $\frac{TP}{TP+FN}$. Recall indicates how complete our predictions are. The F1 score, which is a special case of the more general F-score, is the harmonic mean of precision and recall. It is given by

$$F1 = 2\frac{precision \times recall}{precision + recall} = \frac{TP}{TP + \frac{1}{2}(FN + FP)}.$$

An F1 score can be between 0 and 1, where 1 indicates both perfect precision and recall. In a binary classification setting, F1 is the same as the Sørensen–Dice coefficient, more commonly known as the Dice coefficient.

An important component of determining the above metrics is an appropriate choice of a threshold. As explained, a threshold determines the amount above which a predicted probability can be cast to the positive class. An incorrect threshold choice could undermine the usefulness of these metrics. Moreover, different models may have different thresholds making comparisons across models more difficult. An additional metric used to address this shortcoming is an AUC (area under a curve). There are two commonly used curves from which an AUC is calculated, namely the receiver operating characteristic curve (ROC) and the PR curve. An ROC curve measures recall on the y-axis and 1 - specificity on the x-axis, where specificity = $\frac{FP}{FP+TN}$. For imbalanced datasets, which have more negative classes, values on the x-axis may be inflated since there may be many more true negative predictions. Hence, a precision-recall (PR) curve is preferred for imbalanced datasets because, rather than 1-specificity, precision is used, which does not involve true negatives (Saito and Rehmsmeier, 2015).

To construct this curve both precision and recall are calculated using different thresholds. If we were able to calculate the precision and recall for an infinite amount of thresholds t, then we could calculate the AUC using an integral. However, this is of course not possible, and hence we calculate the precision and recall for a reasonable

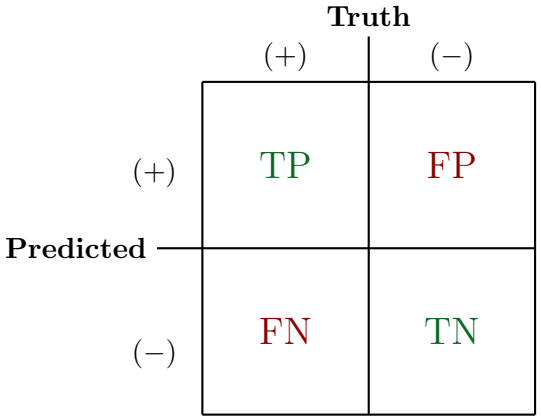


FIGURE 4.15: A confusion matrix for a binary classification problem

number of thresholds and estimate this integral using a Riemann sum (where precision is calculated using the precision at the end of the delta). The maximum AUC is 1, and occurs when the classifier is perfectly skilled; that is, the curve has a vertex at the co-ordinate (1,1). A similar metric to AUC is average precision. Due to their similarity, only AUC is used.

Using the AUC, an optimal threshold can be determined. What is considered as optimal varies depending on the task. In this study, the optimal threshold is the threshold which yields the precision and recall closest to a perfectly skilled classifier, corresponding to the point (1, 1) as mentioned above, where the notion of distance is Euclidean. Specifically the threshold used is as follows:

$$\arg\min_t d(t) = \sqrt{(1 - p(t))^2 + (1 - r(t))^2} \tag{4.3}$$

Where:
t is the threshold
$p(t)$ is the precision at threshold t
$r(t)$ is the recall at threshold t

4.6.2 Curve comparison

Root Mean Square Error

The true value of predictions in this study lies in their ability to reconstruct a Vulnerability Curve (VC). A candidate for curve comparison is Root Mean Square Error (RMSE). However, a drawback of RMSE is that comparisons across different scales is difficult, which may be the case when comparing the VCs of different leaves. Hence, RMSE is normalised (NRMSE), using the mean of the true curve. The equation of NRMSE is given below.

$$RMSE = \sqrt{\frac{\sum_{i=1}^{N} (y_i - \hat{y}_i)^2}{N}} \tag{4.4}$$

$$NRMSE = \frac{RMSE}{\sum_{i=1}^{N} \frac{1}{N} y_i} \tag{4.5}$$

NRMSE provides a metric of how well the reconstructed VC matches the true VC. A larger NRMSE indicates a larger difference between points on the curve, and is thus worse.

A disadvantage of using NRMSE in the context of this study is that it provides a summary of the average performance. In this application, certain regions of the VC may hold more physiological value than others. Hence in addition to RMSE, we will also investigate comparing a physiological value of interest between the reconstructed curve and the true curve. This is discussed next.

Comparing Ψ_{50}

The water potential at which 50% of hydraulic conductance occurs is denoted as Ψ_{50}. This value was introduced and its physiological importance explained in Section 2.1.3. To extract the Ψ_{50}, a sigmoid curve is fit to a VC. The form of the sigmoid curve is given below:

$$f(potential) = 100 - \frac{100}{1 + \mathrm{e}^{a(x-b)}} \tag{4.6}$$

The curve is fit by minimising the sum of squared errors between the sigmoid curve and the VC. The result of the process solves for the two unknowns, a and b. For an example of this process applied to the leaves used in this study, see Appendix C.

Using the resulting sigmoid curve, Ψ_{50} can be extracted. Usually, this would be done using the inverse function of the curve. However, given the definition of Ψ_{50}, it is equivalent to b, which is used instead. Sigmoid curves will be fit to both the predicted and true VCs, and the metric used will compare the Ψ_{50} of each curve. The formula of the this metric is given below:

$$\Delta(\Psi_{50}) = \Psi_{50}^{true} - \Psi_{50}^{pred} \tag{4.7}$$

4.7 Conclusion

This chapter presented the three experiments that will be conducted in this study. It presented the building blocks of convolutional models, and explained how these blocks are combined to form the models in this study, namely U-Net, U-Net (ResNet34), and W-Net. Focus was then shifted to the components required to train these models, which included loss functions, optimisers, and model tuning. The chapter concluded by explaining how the performance of the three models would be assessed, in terms of both segmentation performance and VC reconstruction.

Next, we will look at the application of the methodology which was established in this chapter, and assess the results using the metrics presented at the end of this chapter.

Chapter 5

Results

Using the methodology established in the preceding chapter, the three experiments laid out in 4.2.2 were conducted. The results of these experiments will be explored in this chapter. For each experiment, we will start by looking at the dataset created, using image tiles, followed by the model performance, and then finally we will look at VC reconstruction. We start by looking at Experiment 1, which investigates whether a model can generalise across images within the same leaf.

5.1 Experiment 1: Within leaf

5.1.1 Datasets

Four datasets were constructed for Experiment 1 – one for each leaf. The results of the data generation process, explained in sections 3.4 and 3.6, are shown in Table 5.1. The first three rows of the table indicate counts of images. The fourth row indicates the ratio of embolism images to non-embolism images; this ratio includes augmented images. The final row indicates the percentage of pixels with embolisms; the numerator is the count of pixels with embolisms and the denominator is the count of all pixels in the dataset. Each column shows a complete dataset, which were respectively split into train, validation, and test sets, as described in Section 4.2.2.

	kelloggii	*gambelii*	$palmeri_1$	$palmeri_2$
Embolism	2465	1313	518	212
Non-Embolism	4582	5367	1260	490
Augmented	3983	2082	793	313
Embolism Ratio	1:0.72	1:1.58	1:0.96	1:0.93
% Embolism Pixels	0.705	0.635	0.188	0.194

Table 5.1: Technical specification of the leaves, after downsampling and augmentation, used to investigate within leaf generalisation

5.1.2 Hyperparameter results

Experiment 1 only considered U-Net models. The reasoning for this was two-fold. Firstly, an additional eight runs each for hyperparameter tuning and consequent model fitting would have been required had all models been used. These additional runs were infeasible in terms of both time and computation. Secondly, Experiment

1 was used to determine the feasibility of the proposed methodology. Experiment 1 arguably has the largest overlap between the train, validation, and test sets, and hence should be the easiest task in terms of model generalisation. Therefore, it serves as a baseline experiment.

The Bayesian hyperparameter tuning used a configuration of 50 epochs across 30 trials per model; early stopping was triggered if the validation loss remained unchanged for 10 epochs. A batch size of four was used as this was the largest batch size that would fit into GPU memory. This batch size was also used when fitting final models. The search space used is given in Table 4.1. This configuration was used across all three experiments. Due to the time it took to tune hyperparameters, only 10% of the dataset was used for hyperparameter tuning. The intuition behind this decision is that the 10% used will be a representative sample. Hence, although the sample is limited, the choice of hyperparameters will still be based on the underlying dataset.

Hyperparameter	*kelloggii*	*gambelii*	*palmeri*$_1$	*palmeri*$_2$
Score (AUC)	0.67	0.83	0.53	0.53
Activation	ReLU	ReLU	ReLU	ReLU
Initialiser	Glorot Uniform	Glorot Uniform	He Normal	He Normal
Filters	0	3	3	3
Optimiser	Adam	Adam	Adam	Adam
Learning Rate	4×10^{-4}	2×10^{-4}	1×10^{-4}	3×10^{-4}
Loss (Weight)	Focal (0.5)	Focal (0.6)	Focal (0.75)	Focal (0.92)

TABLE 5.2: The results of hyperparameter tuning a U-Net on the dataset for each leaf

The results of the hyperparameter search of a U-Net model are shown for each leaf in Table 5.2. There are a few noteworthy points. Firstly, focal loss was the chosen loss across all leaves. The loss weight was lower for leaves that have a higher percentage of pixels with embolisms. However, this proportionality was non-linear, with *Quercus palmeri*$_2$ (Qp2) having a higher loss weight than *Quercus palmeri*$_1$ (Qp1) did, despite having a lower proportion of pixels with embolisms. Secondly, Glorot Uniform was the preferred initialisation in *Quercus kelloggii* (Qk) and *Quercus gambelii* (Qg), compared to both *Quercus palmeri* leaves. The clearest difference between these leaves is the proportion of pixels with embolisms. Thirdly, the same activation function and initialiser was used across all leaves. Finally, complex models were chosen across all leaves, except for Qk, where a filter multiple of zero was used.

In this experiment and the experiments to follow, models were fit using the optimal hyperparameters and the results of these final optimised models are presented. The results for U-Net models fit with hyperparameters presented in this section are discussed next.

5.1.3 Model Results

The model was run for 100 epochs and the point where the highest validation AUC was achieved was saved. The optimal threshold for each model was determined using equation 4.5. The optimal validation threshold for each leaf is shown in Table 5.3. In addition, Figure 5.1 shows the validation PR curves and the threshold. Post-processing had no effect on the results across all experiments, and hence only one set of results are shown. This was confirmed by comparing PR curves constructed before

Leaf	Threshold	Accuracy	Precision	Recall	AUC	F1	NRMSE	$\Delta(\Psi_{50})$
kelloggii	0.44	99.84	78.2	79.4	84.7	78.8	0.01	0.02
gambelii	0.47	99.88	80.1	78.2	84.0	79.1	0.02	-0.04
$palmeri_1$	0.36	99.94	70.4	67.5	70.9	68.9	0.06	0.18
$palmeri_2$	0.70	99.95	83.0	79.3	82.6	81.1	0.03	0.00

TABLE 5.3: Validation results of a U-Net fit using the optimal hyperparameters for each leaf

Leaf	Threshold	Accuracy	Precision	Recall	AUC	F1	NRMSE	$\Delta(\Psi_{50})$
kelloggii	0.43	99.84	81.3	80.4	86.7	80.8	0.01	0.00
gambelii	0.46	99.88	80.1	78.2	84.4	79.1	0.01	-0.02
$palmeri_1$	0.27	99.94	71.8	64.9	69.9	68.2	0.06	0.21
$palmeri_2$	0.68	99.95	75.1	69.0	73.2	71.9	0.03	-0.04

TABLE 5.4: Test results of a U-Net fit using the optimal hyperparameters for each leaf

and after post-processing. This means that in all cases the model only predicted that embolisms occur where the image difference property holds. The validation and test set results are given in Tables 5.3 and 5.4 respectively. Note that results for accuracy, precision, recall, AUC, and F1 are presented as percentages all throughout this chapter.

The same threshold – based on the validation set – was used for the test set. This is not necessarily the optimal threshold for the test set, but determining this threshold requires using the test masks, which would overstate the test set results upwards. The test PR curves, with their corresponding optimal thresholds, are shown in Appendix D.1.

Accuracy was near 100% in all cases, but this is not as meaningful due to the extreme class imbalance. In most cases, precision was higher than recall, showing that the models favoured accurate predictions, possibly incorrectly classifying some embolism pixels as non-embolism pixels. Leaves with a higher proportion of pixels with embolisms performed well. In fact, AUC, which measures overall model performance, was ordered according to the proportion of pixels with embolisms. We observe that there was large variability between the test set and validation set results for Qp2. Qp2 had the smallest dataset across 4 leaves, and hence it is expected that there is more variability between the distributions of embolisms, both within images and across images, in the validation and test sets. Additionally, this suggests that the hyperparameter tuning has been successfully influenced by the validation set – an assertion which will be supported by results in the latter two experiments. This variability was experienced across the other leaves, but to a lower extent. The leaf that performed the best across test AUC and F1 was Qk, while worst-performing leaf was Qp1.

Using the test set, VCs were constructed for each leaf. These are shown in Figure 5.2. The orange curve represents the VC recreated using predictions, while the green curve represents the true VC created using masks. These curves are created using image tiles; it is not possible to use the complete leaves since these images are also included in the training and validation sets. However, despite using tiles the true curves (shown in green in Figure 5.2) closely match the curves created using full leaf

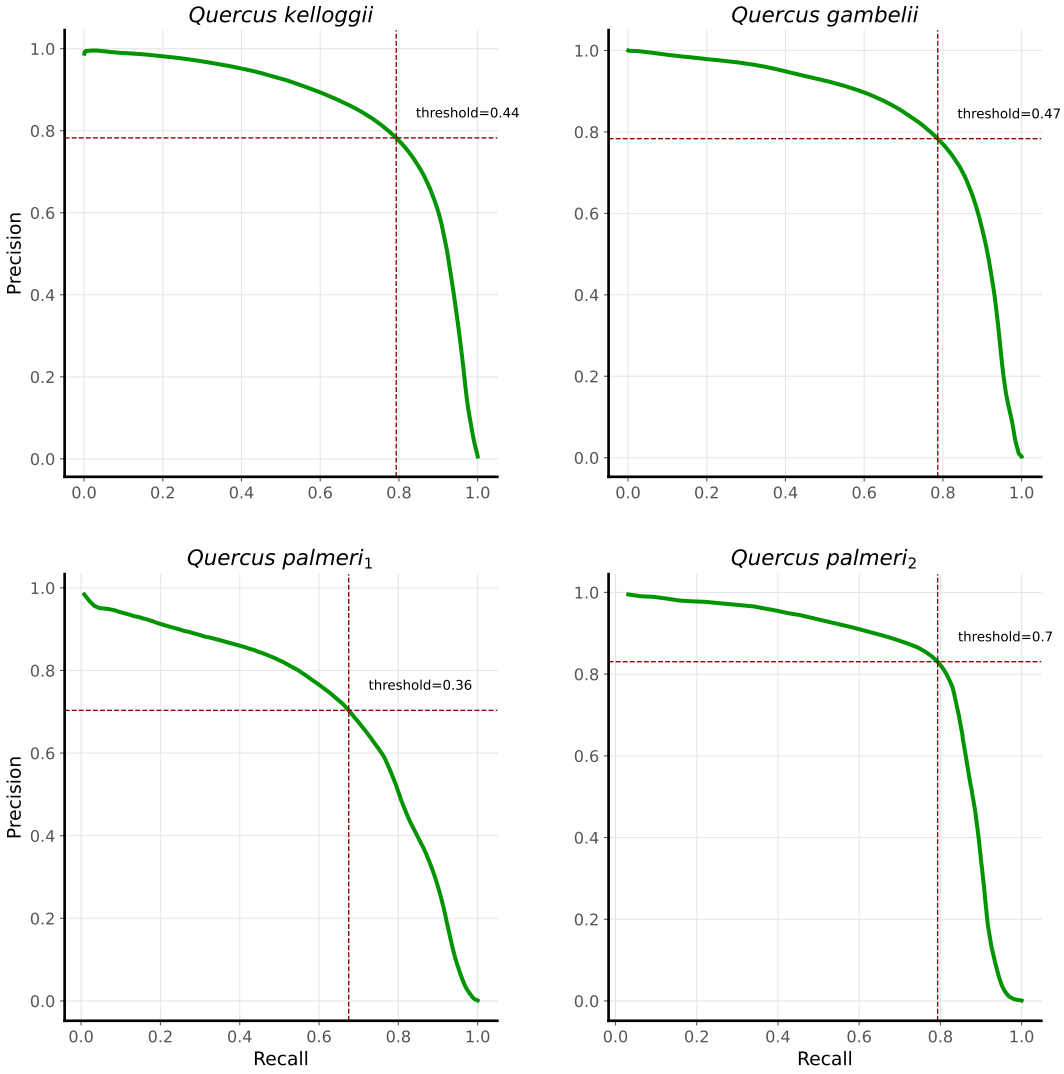


FIGURE 5.1: A Precision-Recall curve, with the optimal threshold, constructed using the within leaf validation set for each leaf

images, shown in Figure 3.4. The tail of Qp2 is an exception. At a water potential higher than -3, it is much lower than the full sample VC shown in Figure 3.4. A plausible explanation for this is again the fact that it is the leaf with the smallest sample, with the tail consisting of only 15 images. Apart from Qp2, the similarity of the test VC to the full sample VC is indicative of a successful data generation process, in which the sampled test set is representative of the full sample.

Shifting our attention to the predicted curves in Figure 5.2, we see that they closely match the true curves. Qk most closely matched the true curve, while Qp1 deviated the most. This is confirmed by the NRMSE for each curve, which was 0.01 and 0.06 respectively. The ordering of NRMSE values for each leaf matches what would be expected based on visual inspection. The extent of successful reconstruction of Qk's VC was expected due to the high precision and recall achieved for the leaf. However, perhaps unexpected was the reconstruction of Qp1, which, despite having the lowest recall and precision, still had a similar shape to the true curve. This suggests that predicting at least some of the embolisms across all the images that they occur in, is

more important than predicting all the embolisms correctly in fewer images. However, despite, having a similar shape Qp1 had a much larger $\Delta(\Psi_{50})$ compared to the other leaves.

The $\Delta(\Psi_{50})$ support the assertions made by inspecting the NRMSE. There appears to be direct proportionality between the absolute test $\Delta(\Psi_{50})$ and NRSME. However, this proportionality is not followed by the validation $\Delta(\Psi_{50})$ for Qp2. The reason for this is due to an intersection between the true and predicted sigmoid curves at 50% cumulative embolism. The sigmoid curves off which the validation and test Ψ_{50} values were read can be seen in Appendices D.3 and D.4 respectively. This highlights a potential flaw of the metric: since this metric references a single point, it will be low if the true and predicted sigmoid curves overlap, regardless of overall performance. This will be further illustrated in Section 5.3.3.

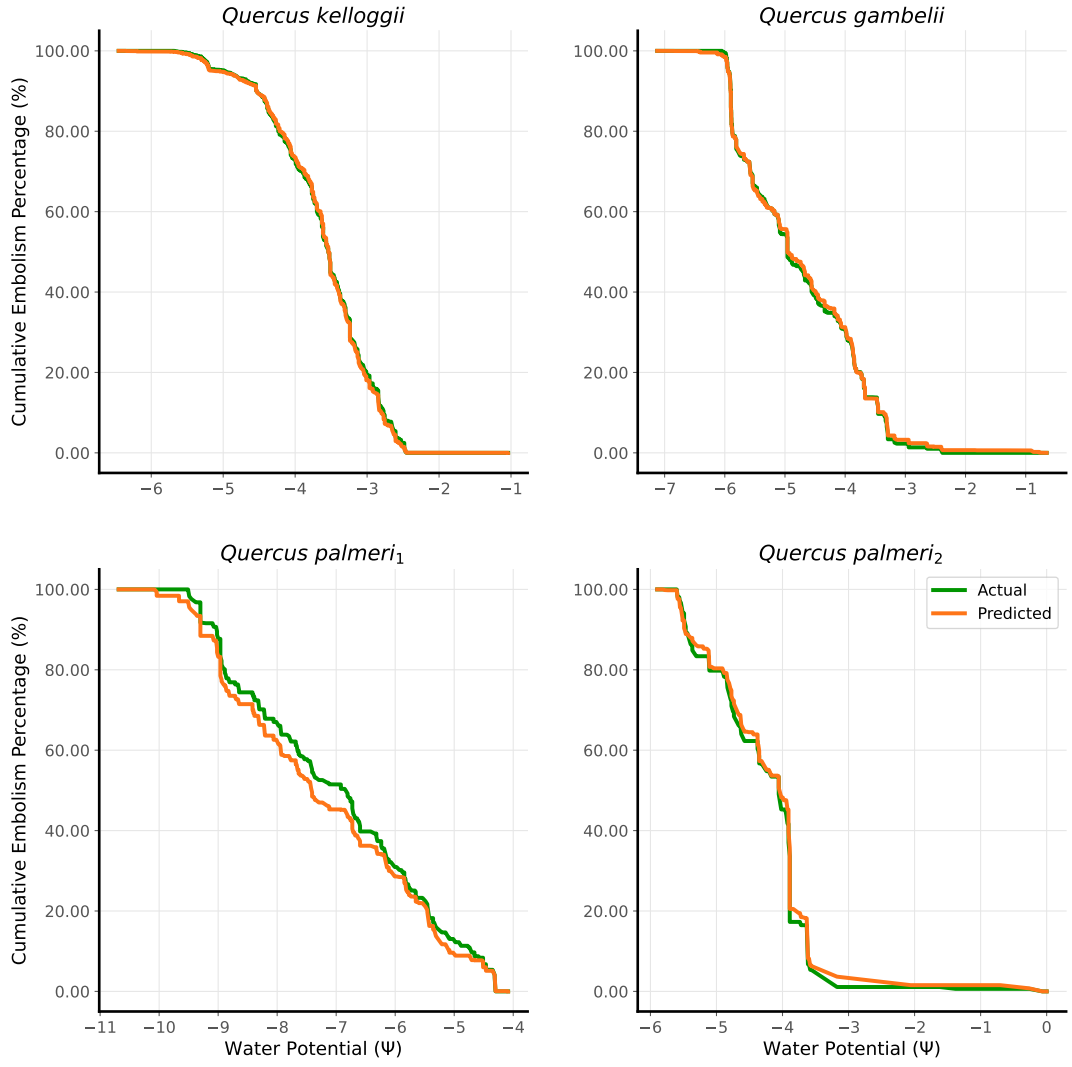


FIGURE 5.2: A comparison of the true vulnerability curve against the predicted vulnerability curve for each leaf. The curves are created using the within leaf test set.

Figure 5.3 depicts a sample of predictions from the U-Net trained on the Qk dataset. The first row shows the predictions and the second row shows inputs prior to pre-processing. The predictions are coloured according to their classification; green represents true positives, blue represents false positives, and red represents false negatives.

This representation provides unique insight into model performance. The first column shows the tile with the most true positives. In this prediction, the false negatives show that the model failed to detect the connectivity between the xylem in two regions. Moreover, looking closely at the boundaries, we see that false positives and false negatives are interspersed. The next column shows the prediction with the most false positives. Based on the input, it appears that the model has correctly identified an embolism that was not detected by the human annotator. Finally, looking at the final column, which shows the prediction with the most false negatives, we see that the network failed to detect a streak of embolisms that is, arguably, not immediately obvious. This is similar to false negatives in the first column, in which it is quite difficult to see that the xylem are connected. The Qk dataset was used since it has the most embolisms, and hence these patterns are easier to see. These assertions do extend to the other datasets; similar figures for each dataset can be found in Appendices D.5, D.6, and D.7.

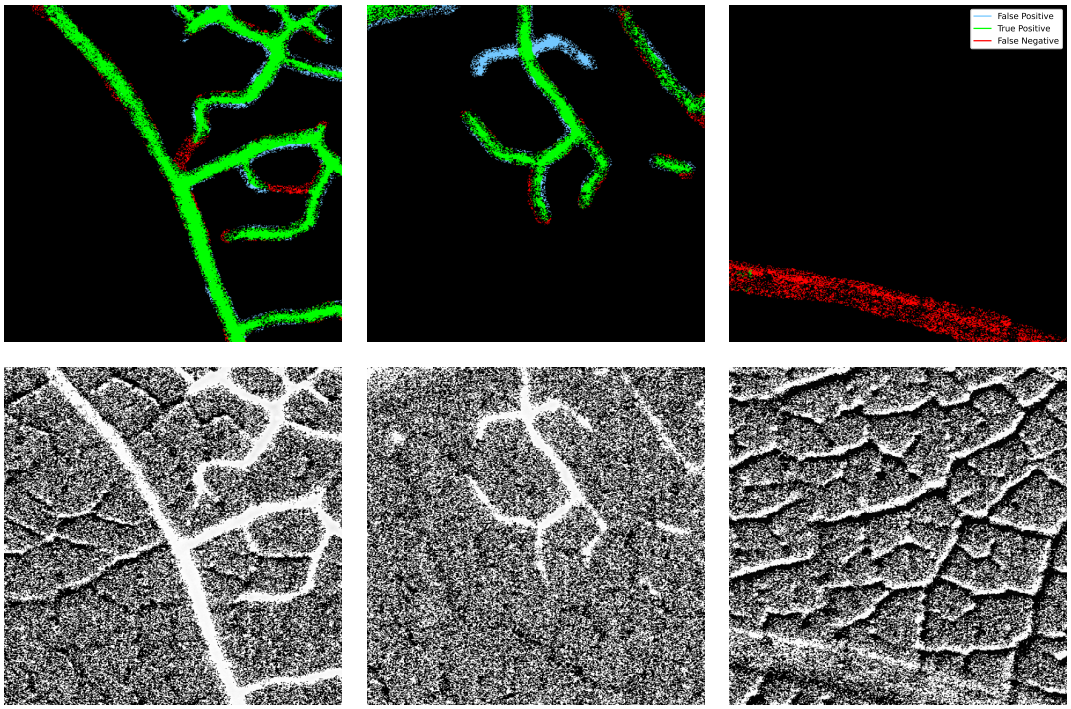


FIGURE 5.3: Predictions using samples from the Qk test set. The top row shows the prediction and the bottom row shows the unprocessed input. The columns, in order from left to right, show the prediction with the most true positives, false positives, and false negatives respectively.

The results of Experiment 1 demonstrates that it is possible for U-Net both to train and to generalise on samples of images generated by the OV method. This lays the foundation for more difficult experiments, which are considered next. In these experiments, both U-Net (ResNet-34) and W-Net are also considered. Specifically, the next experiment investigates whether a model trained on a leaf can generalise to an unseen leaf from the same species.

5.2 Experiment 2: Within Species

5.2.1 Datasets

In this experiment, the dataset was constructed using Qp1 and Qp2. Qp1 was used as the training set, as it had a larger number of tiles compared to Qp2. The respective sets from Experiment 1 for Qp2 were reused as follows: the validation and test sets in Experiment 1 were combined to make the validation set for Experiment 2 and the training set from Experiment 1 was used as the test set for Experiment 2. The results of this process are shown in Table 5.5 below.

	$palmeri_1$	$palmeri_2$	
	Train	Validation	Test
Embolism	518	77	135
Non-Embolism	1260	177	313
Augmented	1192	0	0
Embolism Ratio	1:0.74	1:2.3	1:2.32
% Embolism Pixels	0.206	0.096	0.102

TABLE 5.5: Technical specification of the dataset, after downsampling and augmentation, used to investigate within species generalisation

The difference in percentage of embolism pixels between the two sets stands out, with the percentage in the training set being approximately double that of the validation and test sets. This highlights a difference in the distributions between the two leaves. However, despite this difference, there is the potential for generalisation, if the properties that the network uses to determine embolisms are common between the two leaves.

5.2.2 Hyperparameter Results

The results of the hyperparameter search are shown in Table 5.6. The hyperparameter results were exactly the same for both U-Net and U-Net (ResNet-34), apart from their overall performance. Interestingly, very simple models were favoured, with a filter multiple of 0 being chosen. W-Net, however, whose results were similar to the preceding two models, performed much worse. An additional noteworthy result is that for all models, binary cross entropy was the chosen loss function. This is in contrast to what was chosen previously. A possible reason for this is that it is difficult to choose weights for both focal loss and weighted cross entropy. This difficulty arises since the choice of weight is usually a function of imbalance, for which the percentage of embolism pixels is a proxy measure. Therefore, the choice of loss weight which resulted in an effective loss minimisation on the training set may not have translated to the validation set. As mentioned, the percentage of embolism pixels differs between the training and validation sets because they are created from different leaves.

5.2.3 Model Results

The validation and test set results are given in Tables 5.7 and 5.8 respectively, whilst the validation and test PR curves are shown in Appendices D.8 and D.9 respectively. The focus of the comparison in these results differs from Experiment 1, in that looking

Hyper Parameter	U-Net	U-Net (ResNet34)	W-Net
Score (AUC)	0.14	0.11	0.01
Activation	ReLU	ReLU	ReLU
Initialiser	He Normal	He Normal	He Normal
Filters	0	0	2
Optimiser	Adam	Adam	Adam
Learning Rate	3×10^{-4}	3×10^{-4}	2×10^{-3}
Loss (Weight)	BCE	BCE	BCE

TABLE 5.6: The results of hyperparameter tuning U-Net, U-Net (ResNet34), and W-Net on a subset of the within species dataset

Model	Threshold	Accuracy	Precision	Recall	AUC	F1	NRMSE	$\Delta(\Psi_{50})$
U-Net	0.31	99.92	59.5	56.2	53.6	57.8	0.11	-0.21
U-Net (ResNet34)	0.23	99.88	39.4	48.8	36.3	43.6	0.14	-0.15
W-Net	0.09	99.91	52.5	42.2	46.1	46.8	0.22	-0.30

TABLE 5.7: Validation results of a U-Net, U-Net (ResNet34), and W-Net fit using the optimal hyperparameters for each model

Model	Threshold	Accuracy	Precision	Recall	AUC	F1	NRMSE	$\Delta(\Psi_{50})$
U-Net	0.27	99.91	54.0	51.5	45.3	52.7	0.15	-0.39
U-Net (ResNet34)	0.26	99.88	43.5	47.7	37.4	45.5	0.09	-0.13
W-Net	0.07	99.89	46.5	39.6	37.5	42.8	0.21	-0.47

TABLE 5.8: Test results of a U-Net, U-Net (ResNet34), and W-Net fit using the optimal hyperparameters for each model

down a column now compares different models trained on the same dataset, rather than similar models across different datasets. U-Net performed the best across all segmentation metrics, in terms of both test and validation results.

U-Net (ResNet34) and W-Net are contrasting in their performance; the former had higher recall at the cost of lower precision, while the opposite was true of W-Net. This indicates that U-Net (ResNet34) was more likely to predict that a pixel was an embolism compared to W-Net, leading to higher false positives, whereas W-Net was more likely to predict a pixel as a non-embolism if it was not confident that it was an embolism, resulting in more false negatives. Inspecting the validation set results, we see that higher precision, at the cost of recall, had a negative effect on VC reconstruction, as W-Net had poorer reconstruction in comparison with U-Net (ResNet34). Specifically W-Net had an NRMSE 1.6 times and a $\Delta(\Psi_{50})$ 2 times higher than that of U-Net (ResNet34). This comparison is reinforced by the test set result. W-Net performed notably worse in the test set compared to the validation set, with a drop in AUC of 8.6%. Moreover, this drop in performance resulted in its AUC being only 0.1% higher than the AUC of U-Net (ResNet34). However, despite similar performance between the models, the drop in performance exacerbated the poor VC reconstruction of W-Net.

Although U-Net had the best segmentation results, U-Net (ResNet34) had the superior test VC reconstruction, and a superior $\Delta(\Psi_{50})$. This is supported by Figure 5.4 which displays the predicted VC against the true curve using the test set, as was

displayed in the preceding experiment. We see that both U-Net and W-Net perform poorly on the images where water potential was less than three. It is important to remember that the denominator for the predicted compared to the true curve is different, as it is relative to the total number of embolisms present in the curve. Hence, despite having the lowest segmentation metrics on the test set, the distribution of embolisms across the U-Net (ResNet34) predictions most closely matched the distribution of embolisms across the true masks.

To further inspect this result, a comparison of the true and predicted temporal profiles for each model is shown in Figure 5.5; the time axis in Figure 5.4 is included due to this additional comparison. Rather than using the total embolism pixels for each series, the total number of pixels in the tile was used as the denominator. This allows us to compare how well predictions matched masks over time. The segmentation metrics are better reflected in this figure, where U-Net has the best temporal reconstruction. We see that the lower recall of W-Net results in an underestimation of the extent of cavitation towards steps 300 and above, whereas for U-Net (ResNet34) we see that higher recall results in a predicted curve that is higher than the true curve, showing that the model is prone to many false positives. However, despite this overestimation at later steps, we see that U-Net (ResNet34) mostly matches the true curve closely between steps 0 to 100; the images between 0 and 100 cover more than half the water potential range of the VC. This result reflects the importance of the non-linear relationship between time and the VC. Correctly predicting embolisms in earlier images, where most of the cavitation occurs, is more important than in images at later time steps, as the earlier images are more influential in determining the shape of the VC.

The result of VC reconstruction in this chapter further develops the suggestion in Section 5.1.3, namely that predicting at least some of the embolisms across all the images that they occur in is more important than predicting all the embolisms correctly in fewer images (unless these images span the majority of the water potential range). Moreover, the U-Net (ResNet34) VC reconstruction shows that despite much poorer model performance compared to Experiment 1, it is possible to recreate VCs within species.

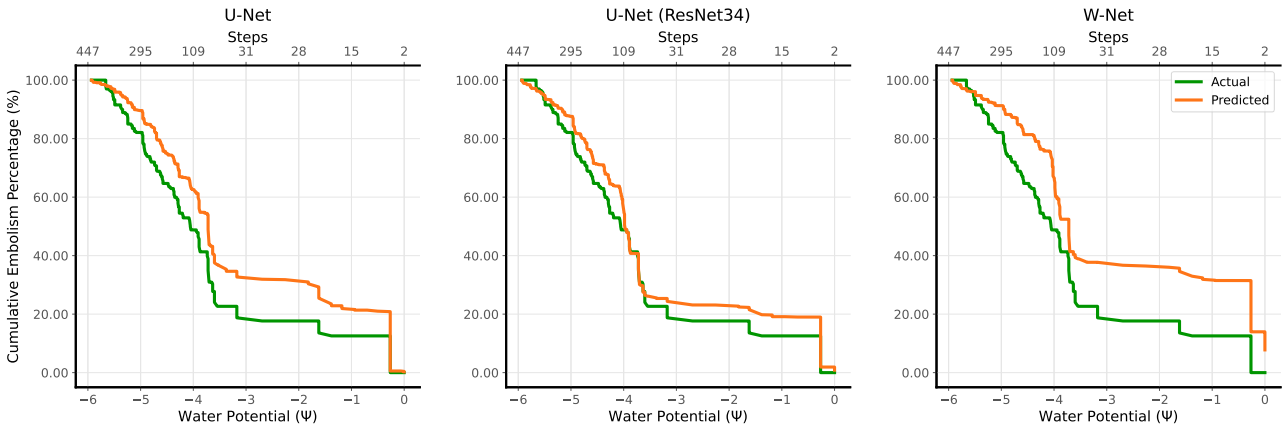


FIGURE 5.4: A comparison of the true vulnerability curve against the predicted vulnerability curve for each model. The curves are created using the within species test set.

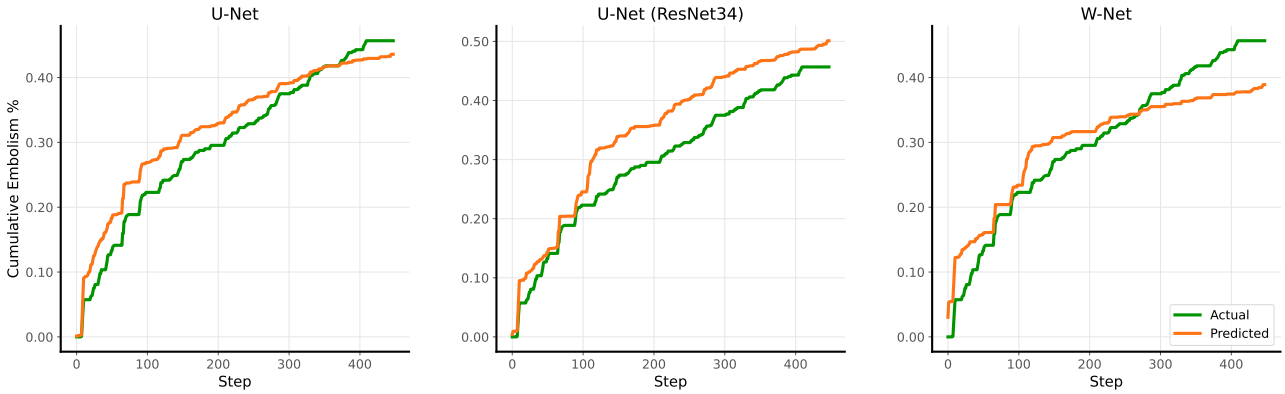


FIGURE 5.5: A comparison of the true temporal profile against the predicted temporal profile for each model. The curves are created using the within species test set.

Experiment 2 illustrates that on a more difficult task, model performance is not as good as it is in Experiment 1. However, despite this drop in performance, it is still possible to recreate a VC with a similar shape to the true VC. With this in mind, the final experiment, which considered whether a model trained on a mixture of different species could generalise to an unseen species, is discussed next.

5.3 Experiment 3: Across Species

5.3.1 Datasets

In this experiment, the training set was constructed using Qk and Qg. The validation set was created using the training set of Qp1 from Experiment 1 and the test set was created using Qp2. The results of this process are shown in Table 5.9 below. As was the case in Experiment 2, what stands out is the contrast between the percentages of embolism pixels. This highlights the difference in both the distributions and the extent of embolisms across the different sets. Additionally, no augmentation was used in this experiment. This is mainly due to the additional computational time introduced by augmentation, but also, considering that the training set has many images with embolisms, it was considered unnecessary to augment for the purpose of increased embolisms images. The assumption made is that there is enough variation in this set of images to allow the models to generalise.

	kelloggi & *gambelii* Train	*palmeri*$_1$ Validation	*palmeri*$_2$ Test
Embolism	3778	331	212
Non-Embolism	9948	806	490
Embolism Ratio	1:2.63	1:2.44	1:2.31
% Embolism Pixels	0.356	0.102	0.010

TABLE 5.9: Technical specification of the dataset, after downsampling and augmentation, used to investigate across species generalisation

5.3.2 Hyper Parameter Results

The results of the hyperparameter tuning for the three models are shown in Table 5.10. Although there was a difference in percentage of embolism pixels across all sets, similar to Experiment 2, the choice of loss function was not the same. In this experiment focal loss was again the most popular loss choice, in line with the results of Experiment 1. The activation function chosen for the U-Net (ResNet34) was SELU, the first time this function was chosen. An additional noteworthy observation is that a more complex model was chosen for U-Net in comparison to the other two models, indicated by a filter choice of 3.

Hyper Parameter	U-Net	U-Net (ResNet34)	W-Net
Score (AUC)	66.8	55.5	58.7
Activation	ReLU	SELU	ReLU
Initialiser	He Normal	He Normal	He Normal
Filters	3	1	1
Optimiser	Adam	Adam	Adam
Learning Rate	0.0001	0.0001	0.0009
Loss (Weight)	Focal(0.75)	Focal(0.73)	BCE

TABLE 5.10: The results of hyperparameter tuning U-Net, U-Net (ResNet34), and W-Net on a subset of the across species dataset

5.3.3 Model Results

As mentioned in Section 5.3.1, the dataset used in this experiment was much larger than in the preceding experiments. Therefore, models were only run for 50 epochs. This experiment had the largest difference between the validation and test set results, across all three models. These results can be seen in Tables 5.11 and 5.12. The validation results are similar to that of Experiment 2; however, the test results are much lower. A plausible reason for this is the large differences between validation and test optimal thresholds for each model, where U-Net and W-Net were particularly affected. W-Net, as in Experiment 2, had a very low threshold compared to the other models. This difference in threshold highlights the distributional differences between the validation and test sets.

To better understand the difference between validation and test performance, especially given that the validation and test sets pertain to different leaves, both validation and test PR and VCs are plotted. The contrast between validation and test performance is displayed clearly by Figures 5.6 and 5.7. U-Net and W-Net completely fail to generalise, despite doing well on the validation set. The test PR curves for both these models bow toward the point (0,0), indicating poor performance. U-Net (ResNet34) maintains similar performance, but the optimal test threshold is lower than the optimal validation threshold. The use of the validation threshold in U-Net (ResNet34) results in a much lower recall, as many of the pixels which are predicted as embolisms with probabilities in the range $[0.5, 0.12)$ are incorrectly classified as non-embolism.

Model	Threshold	Accuracy	Precision	Recall	AUC	F1	NRMSE	$\Delta(\Psi_{50})$
U-Net	0.48	99.88	44.0	56.6	49.6	49.5	0.08	-0.06
U-Net (ResNet34)	0.12	99.91	58.6	50.1	53.1	54.0	0.14	0.49
W-Net	0.006	99.89	43.9	45.4	25.7	44.6	0.10	0.32

TABLE 5.11: Validation results of a U-Net, U-Net (ResNet34), and W-Net fit using the optimal hyperparameters for each model

Model	Threshold	Accuracy	Precision	Recall	AUC	F1	NRMSE	$\Delta(\Psi_{50})$
U-Net	0.11	99.87	27.2	18.3	21.0	21.9	0.33	-0.25
U-Net (ResNet34)	0.05	99.90	52.6	38.0	40.0	44.1	0.20	-0.26
W-Net	2 $\times 10^{-}5$	99.89	30.8	0.08	0.04	13.7	0.32	-0.21

TABLE 5.12: Test results of a U-Net, U-Net (ResNet34), and W-Net fit using the optimal hyperparameters for each model

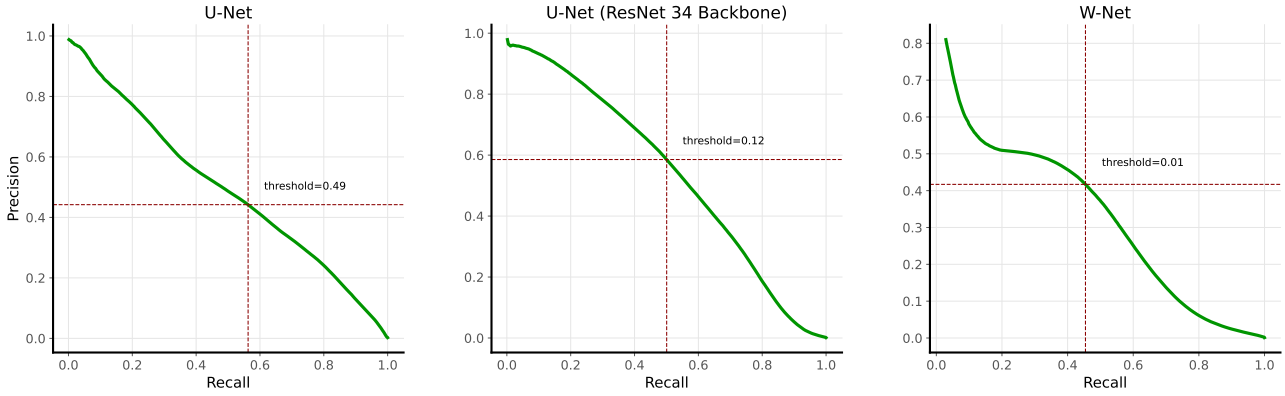


FIGURE 5.6: A Precision-Recall curve, with the optimal threshold, constructed using the across species validation set for each model

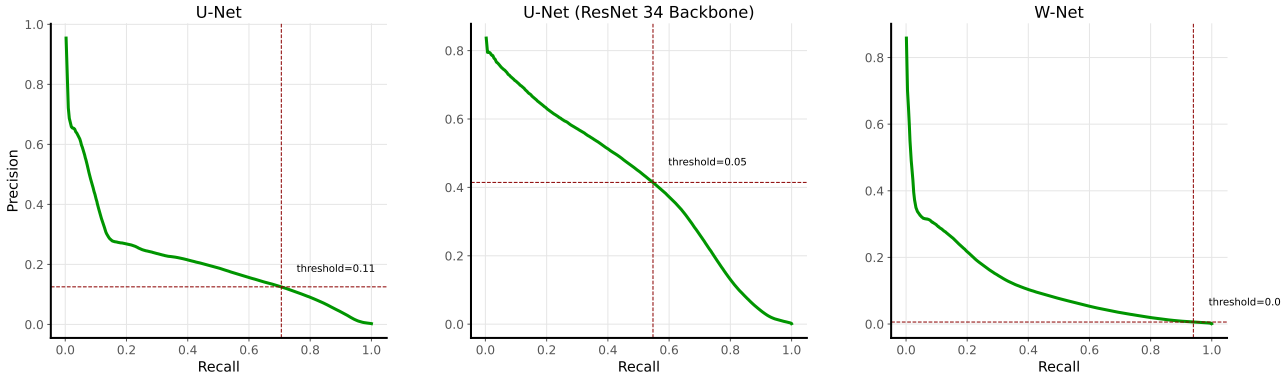


FIGURE 5.7: A Precision-Recall curve, with the optimal threshold, constructed using the across species test set for each model

Our attention now shifts to the reconstructed VCs, displayed in Figure 5.8. Inspecting the validation VCs, we see that all predicted curves track the shape of the true curves. Looking at the validation curves for W-Net and U-Net (ResNet34), we

see that W-Net performs better than U-Net (ResNet34). This is supported by the NRMSE of W-Net, which is 0.04 units lower. Despite performing the worst in terms of segmentation metrics, W-Net outperformed U-Net (ResNet34). This reinforces the findings of Experiments 1 and 2, particularly that at lower model performance, a few correct predictions over many images is more important than accurately predicting fewer images. This is supported by comparing the standard deviation of recall across individual tiles between these two models, which was 6% and 20% for W-Net and U-Net (ResNet34) respectively. Additionally, the low $\Delta(\Psi_{50})$ for U-Net compared to W-Net is interesting given their similar NRMSE performance. The VC predicted by U-Net intersects the true VC around 50% cumulative embolism %, which is the reason this metric is so low. This is supported by the sigmoid curves which are shown in Appendix D.13

The test VCs are notably worse in comparison to the validation VCs. This is supported by the higher test NRMSE. Figure 5.10 shows the test VC for each model. Clearly, U-Net and W-Net have performed poorly, which is expected based on their test segmentation metrics and NRMSE, whereas both based on metrics and visually, U-Net (ResNet34) performed considerably better. However, according to the $\Delta(\Psi_{50})$ results, U-Net (ResNet34) performed the worst. This example highlights an extreme case of the flaw of this metric that was raised both in the preceding chapter and in Experiment 1. Despite poor performance, intersection between the true VC and predicted VC around 50% of cumulative embolism percentage results in low $\Delta(\Psi_{50})$. Furthermore, the higher precision compared to the lower recall metric of U-Net (ResNet34) contrasts with what was seen in Experiment 2 and in this experiment's validation set. To better under the poor performance of the test set, the time axis was included in the figure to allow a comparison with the temporal profile of the dataset – as was done in Experiment 2.

Figure 5.10 shows a comparison of the temporal and predicted temporal profile for each model. As in Experiment 2, the denominator is the total number of pixels in the tile. This view makes clear the poor performance of both U-Net and W-Net. W-Net grossly underestimates the curves at most points. U-Net appears to have a large spike, which is due to predicting too many embolisms across a few images. U-Net (Resnet34) does well at the start but has failed past the 200^{th} step, whereby it fails to detect many of the embolisms. This failure was common across all models, with both U-Net and W-Net recalling little to no embolisms past this step 200. Based on Figure 5.10, it appears that there is an additional point of failure at step 173. This is supported by Figure 5.10 which shows that the models predict many embolisms for images between steps 173 and 200, with a large percentage of the cumulative embolisms occurring between these steps. However, as in Experiment 2, more than half the water potential range of the VC occurs before step 200, which is why U-Net (Resnet34) has a reconstructed VC with a similar shape to the true VC.

To better understand the failure at between steps 173 and 200, and past step 200, three test set input and prediction pairs are shown in Figure 5.11. The predictions were made using the U-Net (ResNet34) model. The first column shows the prediction that had the most true positives; this prediction is from a step earlier than 173. It shows an example of where the model was able to correctly detect an embolism. The middle column shows the prediction at step 173. The pronounced xylem pattern in the image is due to poor alignment between the two raw images that were differenced; it is clear that the sequences of white pixels in this image confuses the network. Many images between steps 173 and 200 had similar artefacts from differencing. The last

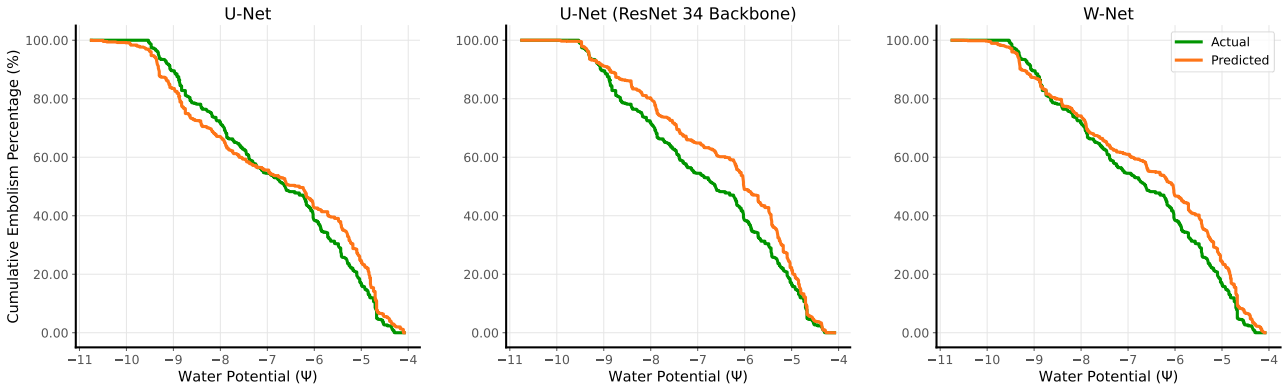


FIGURE 5.8: A comparison of the true vulnerability curve against the predicted vulnerability curve for each model. The curves are created using the across species validation set.

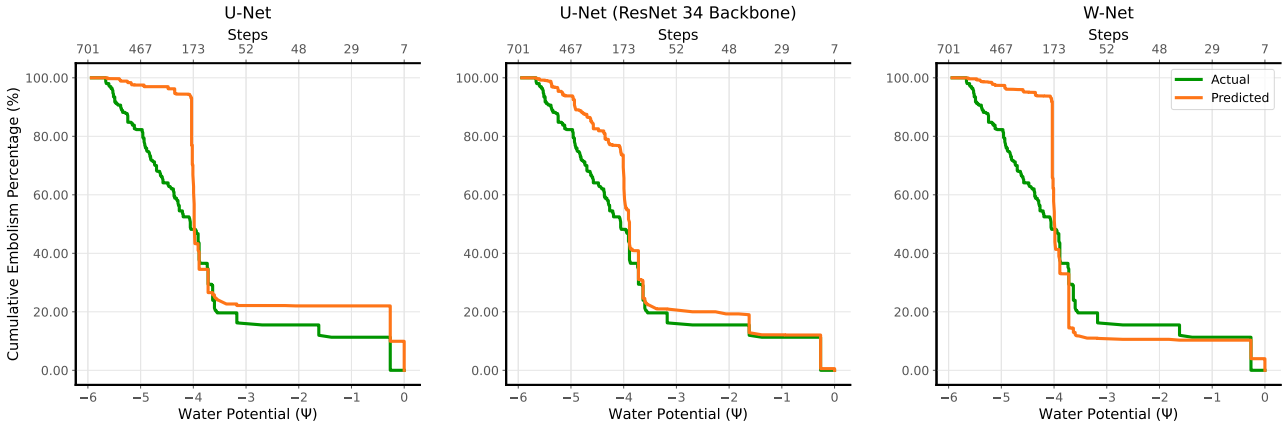


FIGURE 5.9: A comparison of the true vulnerability curve against the predicted vulnerability curve for each model. The curves are created using the across species test set.

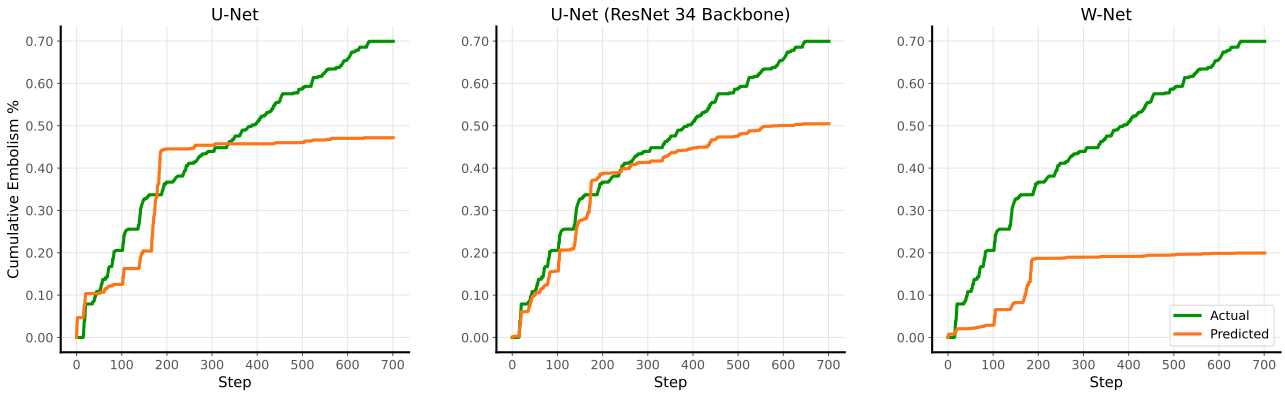


FIGURE 5.10: A comparison of the true vulnerability curve against the predicted vulnerability curve for each model. The curves are created using the across species test set.

column shows the prediction with the most true positives after step 200. The image in this column is similar to the rest of the images past this step. The inputs add some clarity as to why the predictions are so poor; it is difficult even for a human

to detect embolisms in these images. These images are considerably noisier than the images used in training. However, the Qp2 model used in Experiment 1 was able to successfully generalise to images with as much noise, which suggest it is possible. Based on these images, the test set result, and the success in Experiment 1, an additional augmentation technique which may be valuable is adding Gaussian noise to the input.

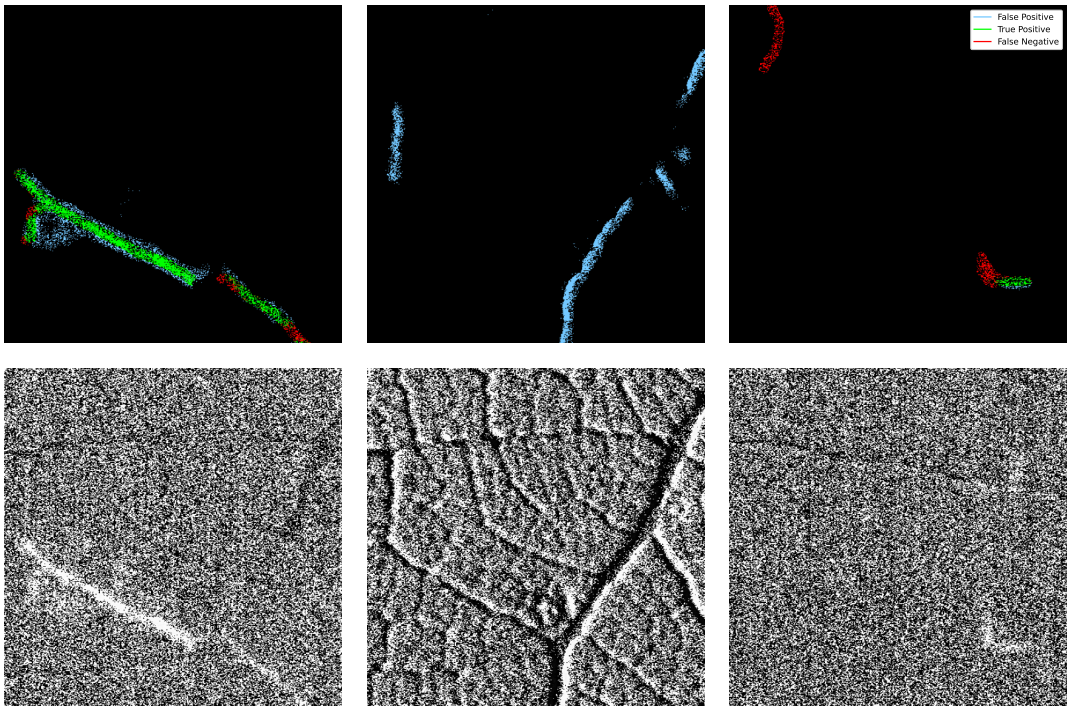


FIGURE 5.11: U-Net (ResNet34) predictions made using samples from the across species test set. The top row shows the prediction and the bottom row shows the unprocessed input. The first column shows the prediction with the most true positives, the second column shows the prediction at step 173, and the final column shows the prediction with the most true positives past step 200.

The difference between validation and test set performance is indicative of the influence of the validation set in determining hyperparameters. This supports the efficacy of the hyperparameter tuning approach – an assertion mentioned in previous experiment results – and also highlights flaws in the data construction process. This result is informative in that it sheds light on the way the network learns. There are signals specific to a species that the network uses to determine where embolisms are. The hyperparameter tuning process may bias the model to do well on the validation set, which could further affect the ability of the model to generalise. From this result, we can deduce that the network uses characteristics of embolism formations other than the fact that it is a structured collection of white pixels; possible candidate examples are size and length. In this regard, the lack of augmentation may have been detrimental to the network training, as these characteristics may not have been represented. However, based on the satisfactory validation performance, which is based on a different leaf, it is difficult to know the extent of augmentation that would be necessary for the network to detect the characteristics specific to the test set.

5.4 Conclusion

This chapter presented the results of three experiments which considered whether a model could generalise within a leaf, within a species, and across species respectively. Experiment 1 only considered U-Nets, training using datasets constructed using each of the four leaves. Experiments 2 and 3 considered U-Net, U-Net (ResNet34), and W-Net which were each trained on a single dataset per experiment, where each dataset was a combination of specific leaves. The key findings from these results will be discussed in the next chapter, followed by the conclusion of this study.

Chapter 6

Discussion and Conclusion

This study aimed to segment OV images well enough to reconstruct Vulnerability Curves (VCs), hence automating the OV method. The success of the reconstruction was based on how well Ψ_{50}, a physiological value of interest, obtained from the predicted curve matched the same value obtained from the true curve.

To identify how to approach the task relevant literature was reviewed. The first section of this review investigated plant hydraulics. It explained the recent emergence of this field, and the importance of being able to measure a plant's vulnerability to cavitation. It expanded on the visual methods of quantifying xylem vulnerability, which includes the Optical Vulnerability (OV) method. Finally, the importance of Ψ_{50}, and consequently VCs, was explained. The next section investigated how automation of the OV method could be achieved. It introduced convolutional neural networks (CNNs) and their extension, fully convolutional networks (FCNs), which are commonly used for image segmentation. The section concluded by investigating domain specific applications of FCNs. Road network and retinal vessel segmentation were investigated due to their similarity to the segmentation task of this study; to the best of this candidate's knowledge, no literature exists on the application of FCNs to this particular segmentation task. The literature review established the physiological importance of the study. Furthermore, it established a grounds for attempting to automate the OV method using deep learning.

Data generated by the OV method was examined next. This began by looking at three different leaf species, namely *Quercus kelloggii*, *Quercus gambelii*, and *Quercus palmeri* (Qp). In total there were four sets of leaf images, as there were two different Qp leaves. Using each of these leaves, VCs were constructed. The remainder of this section investigated the severe class imbalance present in the dataset. A few approaches were used to deal with this imbalance. Firstly, images were tiled; this increased the number of examples of images with embolisms, and was also necessary due to the in-memory considerations. Secondly, augmentation was applied only to those images that had embolisms. Thirdly, a useful property, called the image difference property, was established which showed that only pixels with negative intensities are candidates for embolisms. The section ends by outlining the pre-processing and post-processing steps that are applied to the dataset, which are mainly based on the image difference property.

The next chapter established the methodology. It began by outlining how models learn from data. With this understanding, the experimental design was provided, with three main experiments being conducted. These experiments investigated within leaf generalisation, within species generalisation, and across species generalisation.

With these experiments in mind, the specific models that were used were presented. This began with a presentation of CNNs, which form the foundation for the models used. Next, three models were presented. The first was a U-Shaped network, called U-Net. The next model used a common CNN architecture called ResNet34 as the backbone for U-Net, and this model was called U-Net (ResNet34). The final model was a repeated U-Net variant called W-Net. U-Net (ResNet34) and W-Net fell into two classes of U-Net variations that had done well in the literature on similar tasks. Following this, training was investigated; this included descriptions of loss functions, optimisers, and model tuning. The section concluded by explaining how model performance would be assessed, which included both segmentation performance and VC reconstruction.

The results of applying this methodology were provided in the previous chapter. A discussion of key takeaways from the results follow.

6.1 Discussion

6.1.1 Model Performance and VC reconstruction

Across all experiments, within leaf generalisation was the most successful, in terms of both segmentation performance and VC reconstruction. This result is intuitive as this experiment had the most overlap across the training, validation, and test sets. The experiments were designed to be increasingly difficult for a model to generalise and this was reflected in the results of the remaining two experiments.

Experiment 1 only considered U-Net models, whilst Experiments 2 and 3 considered all three models. In Experiment 2, U-Net showed the highest segmentation performance while U-Net (ResNet34) performed the best in Experiment 3. This performance did not necessarily translate to VC reconstruction, as will be discussed later. W-Net had notably low thresholds across both the experiments in which it was used. This is interesting, as it illustrates that W-Net is confident about which pixels are not embolisms, yet it is not confident about the pixels that are. Across all experiments, the models with the highest AUC scores had a filter parameter of either 0 or 1[1]. This supports the Galdran et al. (2020) findings that simpler models may perform adequately on difficult tasks; the predisposition to using pre-built models may bias us to believe that more complicated models are necessary.

Despite the poor segmentation performance in Experiments 2 and 3, VCs reconstruction produced predicted curves with similar shapes to the true curves. In Experiment 2, both U-Net and U-Net (ResNet34) performed well in VC reconstruction, with U-Net (ResNet34) performing slightly better. In Experiment 3, only U-Net (ResNet34) performed satisfactorily. The superior performance of U-Net (ResNet34) suggests that there is an advantage to using U-Net variants in addition to a vanilla U-Net for VC reconstruction. This is inline with what is suggested in the literature, where these variants have advantages and gains over U-Net in different tasks on benchmark datasets. Furthermore, focal loss was the most common loss function across all the experiments, supporting its usefulness in class imbalanced problems. In some cases, binary cross entropy was the chosen loss function; a hypothesised reason was given for this, although more investigation is required to further understand this.

[1] In Experiment 1, more complex models were chosen for three of the four leaves; this distinction is important as only one model was fit per leaf.

The Qp1 VC was closely reconstructed despite having the worst segmentation performance in Experiment 1. Based on this, we can assert that if VC reconstruction is the goal, it is more important to predict a few embolisms across many images, as opposed to perfectly predicting a few images. The results of Experiment 2 refined this assertion. Counter-intuitively, the model with the lowest test AUC – U-Net (ResNet34) – had the best VC reconstruction. The reason for this was that U-Net (ResNet34) performed the best on the initial images which cover more than 50% of the range of water potentials. Similarly in Experiment 3, U-Net (ResNet34) performed the best on VC reconstruction due to accurate prediction over initial images which covered 50% of the range of water potentials. However, in this experiment, this model had the best segmentation performance. The results of these two experiments indicate that images which span a larger range of water potentials are more important than later images; this importance is due to the non-linear mapping of time to water potential. The more linear this mapping is, the less importance will be placed on individual images. This helps refine our assertion that the images spanning the majority of the water potential range are the most important. This is crucial and bears repeating – it is more important to accurately predict embolisms over many **important** images compared to accurately predicting fewer images.

Assessing VCs was done through Normalised Root Mean Square Error (NRMSE) and the difference between the true and predicted Ψ_{50} values – $\Delta(\Psi_{50})$. A flaw of the $\Delta(\Psi_{50})$ metric is that it will be low if the true and predicted VC intersect close to 50% of cumulative embolisms. This flaw was exposed in Experiment 1 and especially in Experiment 3, where both U-Net and W-Net had had poor curve reconstruction, but lower $\Delta(\Psi_{50})$ values than the model with the lowest NRMSE. Hence, $\Delta(\Psi_{50})$ must be used in combination with NRMSE. Moreover, this flaw highlights that it is worth using additional metrics that are of physiological value; recommendations are given in Section 6.2.

The degradation of segmentation results across model performance over the different experiments highlights the importance of dataset construction. Moreover, the variability between validation and test performance further highlights this. The variability between the validation and test performance shows that the parameters chosen may have been too strongly influenced by the hyperparameter tuning process. This is especially highlighted by Experiment 3, where possible data mismatch was introduced between the validation and test sets by constructing each using different leaves, yielding a large difference between test and validation results. Furthermore, Experiment 3, provides insight into how the models learn. The failure of the model to generalise to the test set illustrates that the properties the network learned while training do not extend to the other leaves. This suggests that properties other than colour – which is the main characteristic used by human annotators – are important to the network. These characteristics are possibly common within species, but not across species.

The result, or lack thereof, of the post-processing step also provides insight into how the models learn. All models only predicted that embolisms occur where the image difference property holds; that is, where pixel intensities were negative. This lack of effectiveness is due to the effectiveness of the pre-processing technique, which converts all positive pixel intensities to 0. Based on these results, it is worth investigating whether both are required. This is discussed in more detail in Section 6.2.

The extent of success across the experiments varied. However, the satisfactory performance in some of the experiments supports the chosen methodology. This methodology was largely informed by applications in both road and retinal segmentation.

Therefore, this success supports the use of road and retinal segmentation to further inform the methodology of plant network segmentation, until more literature on the latter is available. Furthermore, the extent of success was good enough to warrant real world application.

6.1.2 Real World Application

A single model that could predict embolisms across all genera and species would be ideal. Investigating this was not possible due to the available sample, and hence, in the context of this study, the ideal is a single model that could predict across all species within the *Quercus* genus.

The results presented in this study illustrate that a single model cannot confidently generalise across species within the *Quercus* genus. Rather, a model can generalise to the leaf it was trained on, or within a species. The latter statement is based on the results of the Qp species, and hence may not extend to all species in the genus. However, on the premise that it does, this presents potential use cases for the models developed.

The first use case would be to let the model make predictions on unseen images, and assume this to be correct enough to construct VCs. Based on the results and reconstruction, within leaf predictions would be better suited to this approach. An issue with within leaf predictions, however, is that a new model would need to be trained for each leaf to which the developed methodology will be applied. This would be time consuming and computationally expensive, and would require annotated images. The second and more cautious use case would be to use the developed model as a decision aid tool. That is, a single model can be trained on a few leaves from the same species; each species would require a species specific model. Predictions can then be made on unseen leaves from the same species. These masks can then be confirmed by human annotators. Although not strictly automating the OV method, this would greatly speed up the labelling process.

This approach will improve understanding of within species variability, which will be beneficial for modelling that uses Ψ_{50}, such as the model created by Brodribb, Cochard, and Dominguez (2019). Rather than a single value for Ψ_{50}, researchers will now have access to a distribution of these values. This within species variability was present in this study, where Qp1 had a Ψ_{50} of -6.8, which was 2.5 units lower than the Ψ_{50} of Qp2. Moreover, this approach will allow researchers to study the influence of factors in the environment that affect species. May et al. (2009) investigated a 13000-year-old Qp which was considered a relic of a previous population that had rescinded due to historic spread of aridity in the regions of North America in which it is found. Due to this aridity, there are many isolated Qp populations (May et al., 2009). A species specific model would allow the investigation of these populations. It would allow researchers to understand whether these plants are outlying in their Ψ_{50}, and help them to understand how these plants survived while the rest of the population did not.

The OV method is less invasive and requires fewer resources than other methods to quantify xylem vulnerability. Petruzzellis et al. (2020) improved on the initial low-resource setup, allowing for more portable resources to be used in the data collection process. Currently, low cost sensors are being developed to take images of leaves in the field. The methodology and models developed in this study could be used to improve these sensors by also allowing them to make predictions. This could allow

these sensors to extend to monitoring devices. Importantly, inference time need not be instant; this makes the predictions on a low-resource device simpler and more attainable.

6.2 Limitations and Future Recommendations

The limitations of this study, listed below, were caused due to either time or computational constraints.

- The small sample of unique leaves was a major limitation in effectively investigating the experiments outlined in this study. More species, and more leaves within each species, would have allowed for a richer comparison and investigation. In addition, this study was not able to conduct experiments across genera. Additional data will allow for this.
- Due to the cost of tuning and model training, Experiment 1 only considered U-Nets. It would have been valuable to assess the performance of the other two models on this experiment given U-Net's success.
- Experimentation with batch size was difficult both due to limited GPU memory and large image sizes.
- Training time ranged from a few hours to a few days across the experiments. Training time was an important factor in deciding not to augment in Experiment 3, which may have influenced its results. Moreover, the lengthy training time made model experimentation difficult.
- Much of the noise in the input images is due to sensor noise and camera shake. This noise varied across images and may cause inputs to be quite different. Consequently, this makes generalisation across different leaves more difficult. This contributed to the poor test performance in Experiment 3.

The recommendations for future work are given below:

- In many of the leaf images shown, the annotated masks appear as spotty. Perfect overlap with these masks is difficult, as was shown in the examples of predictions. Moreover, in reality these masks should be solid, given that an embolism is a continuous air bubble. Hence, a suggestion is to blur masks as a post-processing step.
- When used in isolation, $\Delta(\Psi_{50})$ is flawed if the curves intersect at 50% of cumulative embolism. Therefore, the difference between two additional physiological values of interest, which are also read from a sigmoid curve fit to a VC, can be used. These values are Ψ_{12} and Ψ_{88}. These two points will refer to points at either tail of the sigmoid curve. These three points used in combination will provide more insight to how closely VCs match across the entire range of water potentials.
- The models often failed to detect connectivity between xylem. As mentioned in the literature review, Mnih and Hinton (2010) solved a similar problem using a CRF. With this as motivation, a CRF, formulated as an RNN, could be used as a post-processing technique, once the network has been trained.
- To restrict the candidate solution space, only pixels which could be embolisms were presented to the network. This limits the information that the network

is given. A future experiment will be to remove this step and only utilise the image difference property as a post-processing technique.

- Based on the images, and particularly on the Experiment 3 images on which the models failed, the addition of Gaussian noise as an augmentation strategy may be beneficial.
- Different tiling strategies, and particularly different tile sizes could be investigated in future experiments.
- Many of the applications presented in the literature review used learning rate cycles. An approach that has shown success on image tasks is the Learning Rate Range Finder technique (LRRF), and the subsequent One-Cycle policy that is based on the LRRF (Smith, 2017).
- IterNet, presented in Section 2.2.3, is a repeated U-Net similar to W-Net. Inspired by this architecture, the number of mini U-Nets in W-Net could be treated as a hyperparameter. This would create an iterated W-Net variant. The benefit of the increased repetition in W-Net is both more refined and more granular attention using each mini U-Net's probability map.

6.3 Conclusion

This section will conclude this study by assessing how well the aims outlined in Section 1.2 were met. Moreover, it will highlight the contributions of this study.

This study aimed to develop an analysis pipeline that would be able to create VCs. Moreover, it aimed to segment the input images well enough to recreate true VCs. Across the three experiments, which varied in how difficult it would be for a model to generalise, two were considered successful. Specifically, this study found that fully convolutional models were able to generalise within leaves, and within species. However, they were unable to generalise across species.

Three fully convolutional models were used: U-Net, U-Net (ResNet34), and W-Net. These choices were based on the successful performance of these models when applied to the similar segmentation tasks of retinal vessel and road network segmentation in the literature. This study found that all models performed equally well. Consequently, this finding establishes a link between models used for retinal vessel and road network segmentation.

This study outlines real world use cases, based on the study results. The most important of these is that within species models can be used as a decision aid tool. This can improve the understanding of Ψ_{50} within a species, and allow for more accurate inclusion when modelling the effects of climate change. More powerful use cases exist should a model be able to generalise across species and across genera. This should be the focus of future work. The many recommendations provided will be a starting point of attempting these tasks, for which generalisation is expected to be more difficult.

To support the ubiquitous use and development of the work done in this study, a website containing both instructions and documentation of the code base was created. In addition, a public GitHub repository is available. Furthermore, to support reproducibility and future development, all results and saved models can be made available on request.

To the best of this candidate's knowledge, no literature exists on segmentation of OV images. Although the findings of this study are based on a limited sample of unique leaves, it provides a new foundation on which future work can be developed. It forms a new interdisciplinary link between deep learning and plant hydraulics, with the potential for real world impact.

Appendix A

Gradient Descent

The update rule for gradient descent is given by:

$$\underline{w}_{t+1} = \underline{w}_t - \eta \nabla_{\underline{w}} [\frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(f(x^{(i)}, \underline{w}_t), y^{(i)})] \tag{A.1}$$

Where: $\nabla_{\underline{w}} [\frac{1}{N} \sum_{i=1}^{N} \mathcal{L}(f(x^{(i)}, \underline{w}_t), y^{(i)})]$ is the derivative of the loss function with respect to the entire dataset.

Gradient descent can be improved by adding a term called momentum. The momentum term is given by :

$$\nu_{t+1} = \alpha \nu_t + \eta \nabla_{\underline{w}} [\frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(f(x^{(i)}, \underline{w}_t), y^{(i)})] \tag{A.2}$$

and the updated update rule is given by:

$$\underline{w}_{t+1} = \underline{w}_t - \nu_{t+1} \tag{A.3}$$

$$= \underline{w}_t - \eta \nabla_{\underline{w}} [\frac{1}{n} \sum_{i=1}^{n} \mathcal{L}(f(x^{(i)}, \underline{w}_t), y^{(i)})] - \alpha \nu_t \tag{A.4}$$

The parameter n determines the number of data points used. Batch gradient descent uses the entire dataset, $n = N$. Stochastic gradient descent uses a single a data point, $n = 1$. Mini-batch gradient descent uses a subset of the sample $n = m$, where $1 < m < N$.

Appendix B

Adam Optimiser

As explained in section 4.5.2, the Adam optimiser uses an exponentially decaying average both of the first-order moment - similar to momentum - and of the (uncentered) second-order moment - similar to AdaGrad and RMSprop, where the moments are taken with respect to the loss gradient (Ruder, 2016). The first-order moment, m_t^1 and (uncentered) second-order moment m_t^2 are given by

$$m_t^1 = \beta_1 m_{t-1}^1 + (1 - \beta_1) g_t \tag{B.1}$$
$$m_t^2 = \beta_2 m_{t-1}^2 + (1 - \beta_2) g_t \odot g_t \tag{B.2}$$

Where:

$$g_t = \nabla_{\underline{w}} [\frac{1}{m} \sum_{i=1}^{m} \mathcal{L}(f(x^{(i)}, \underline{w}_t), y^{(i)})] \tag{B.3}$$

In the above, both the first and second order moments are initialised at the origin, which biases them toward the origin. To account for this the following correction is made.

$$\hat{m}_t^1 = \frac{m_t^1}{(1 - \beta_1^t)} \tag{B.4}$$
$$\hat{m}_t^2 = \frac{m_t^2}{(1 - \beta_2^t)} \tag{B.5}$$

Using these bias corrected estimates the weight update can be made; the update rule for the Adam optimiser is given by

$$w_{t+1} = w_t - \eta \frac{\hat{m}_t^1}{\sqrt{\hat{m}_t^2} + \epsilon} \tag{B.6}$$

Appendix C

Vulnerability Curves with Ψ_{50}

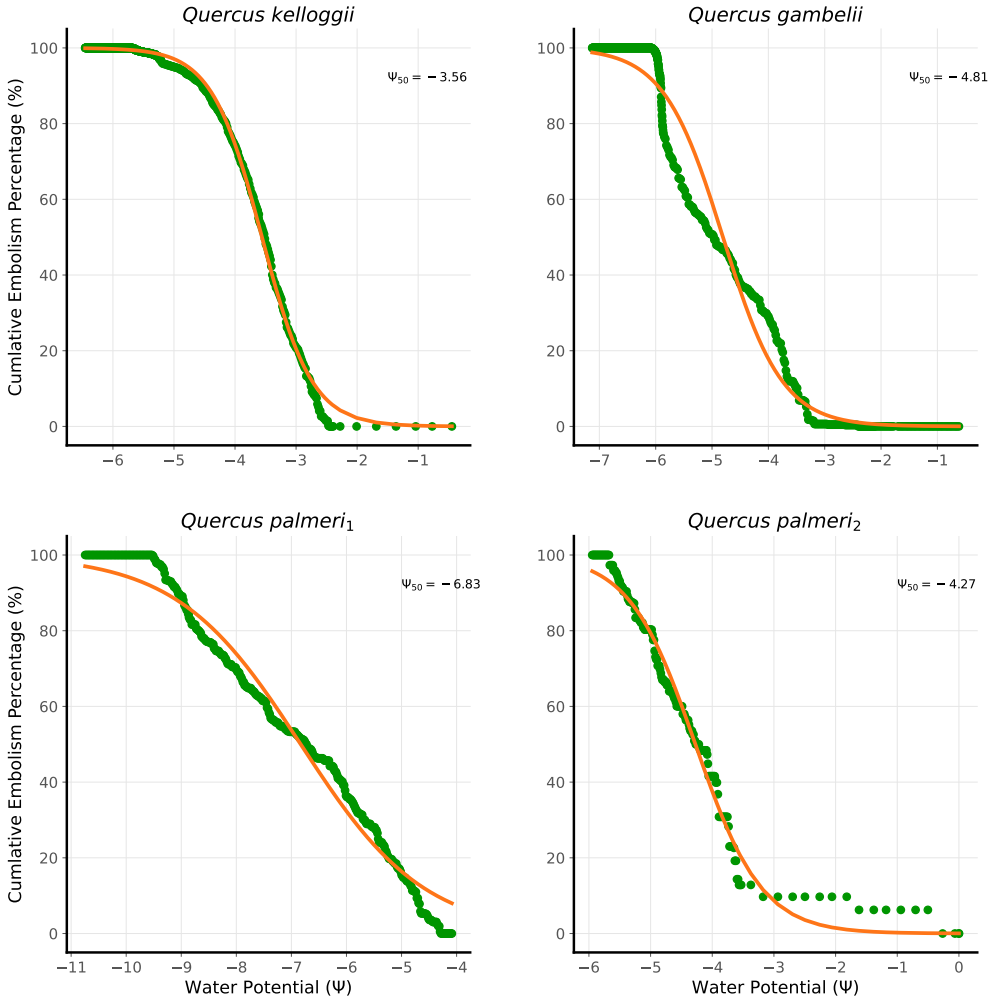


Figure C.1: A sigmoid curve fit to a VC for each leaf used in this study. The Ψ_{50} obtained from the fitted sigmoid curve is annotated.

Appendix D

Additional Results

D.1 Experiment 1

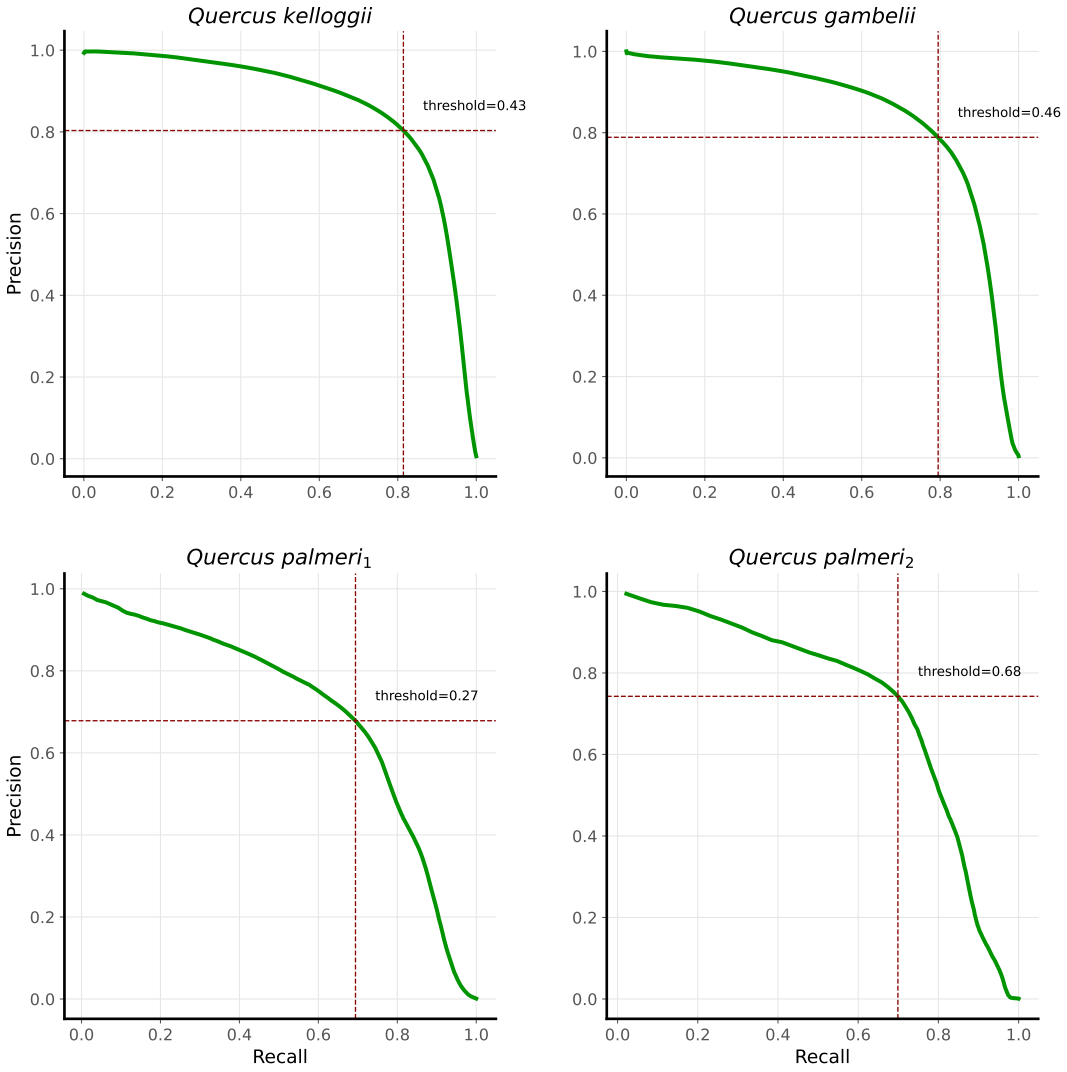


Figure D.1: A Precision-Recall curve, with the optimal threshold, constructed using the within leaf test set for each leaf

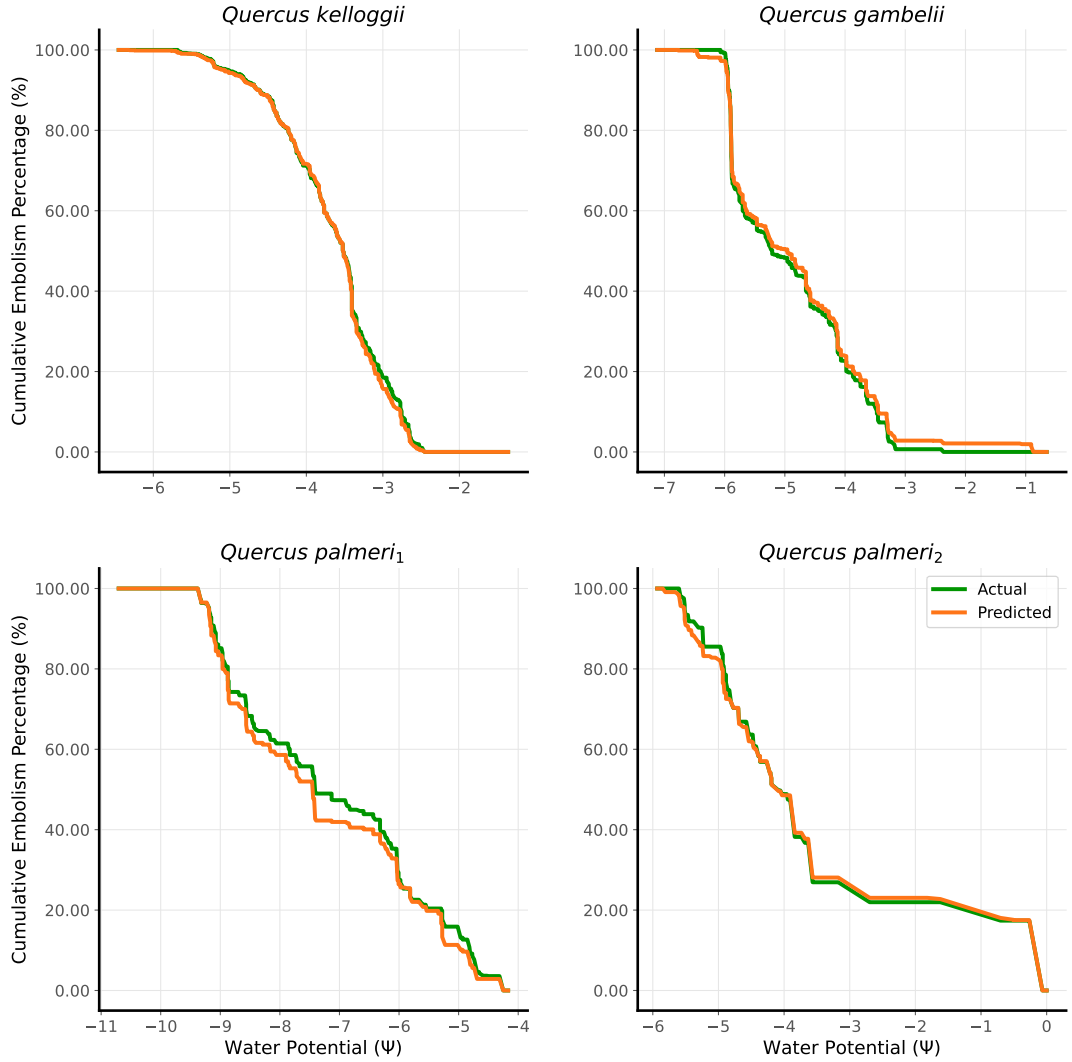


FIGURE D.2: A comparison of the true vulnerability curve against the predicted vulnerability curve for each leaf. The curves are created using the within leaf validation set.

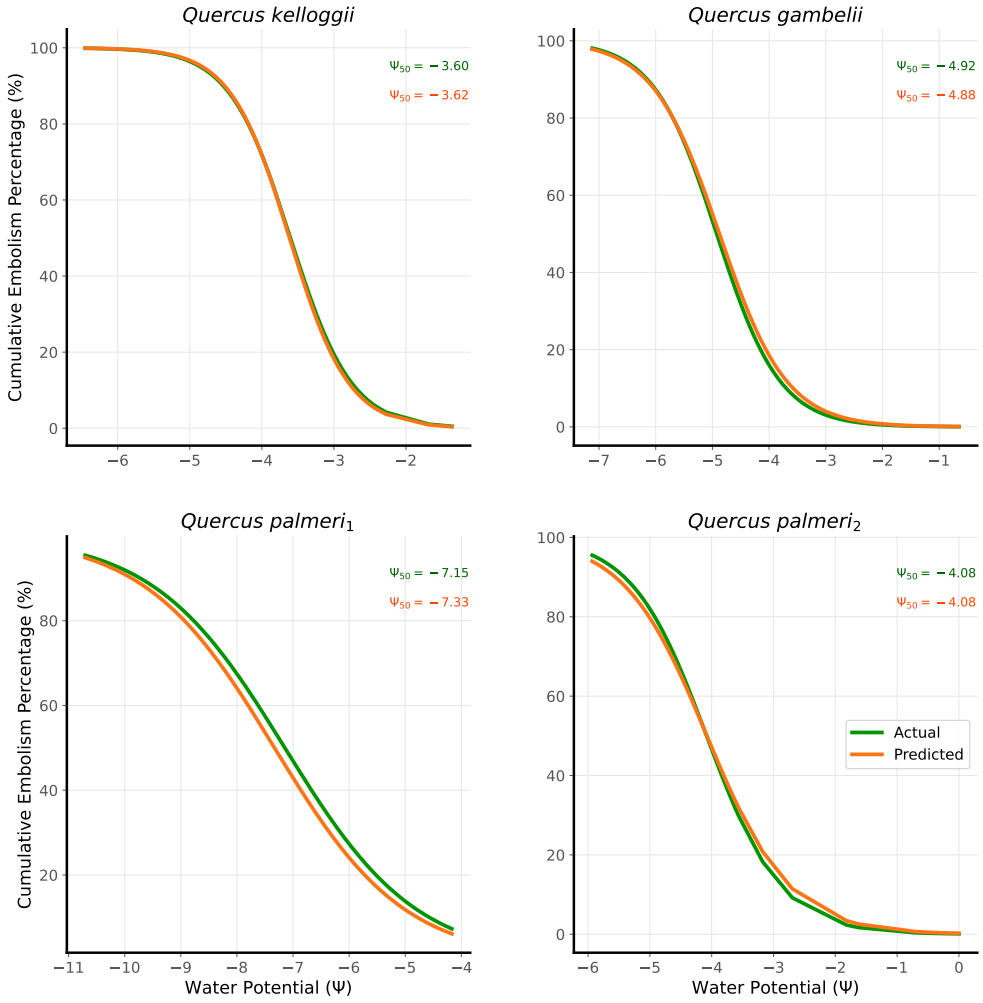


FIGURE D.3: A comparison of the true sigmoid vulnerability curve against the predicted sigmoid vulnerability curve for each leaf. The curves are created using the within leaf validation set.

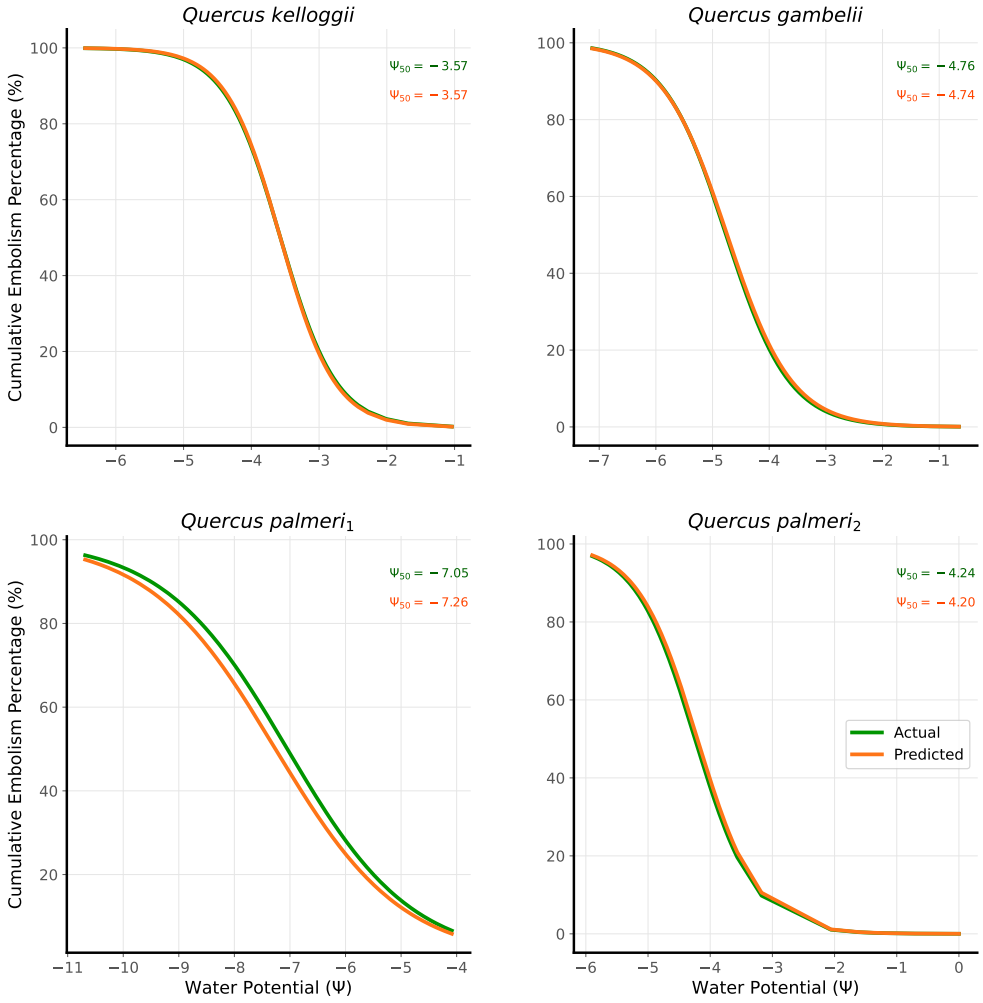


FIGURE D.4: A comparison of the true sigmoid vulnerability curve against the predicted sigmoid vulnerability curve for each leaf. The curves are created using the within leaf test set.

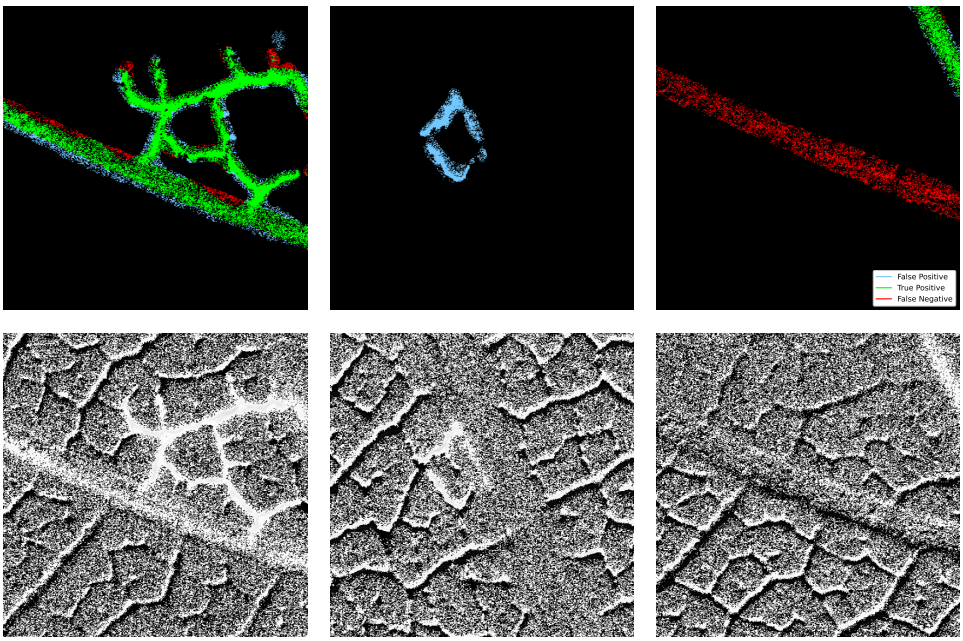


FIGURE D.5: Predictions using samples from the *Quercus gambelii* test set. The top row shows the prediction and the bottom row shows the unprocessed input. The columns, in order from left to right, show the prediction with the most true positives, false positives, and false negatives respectively.

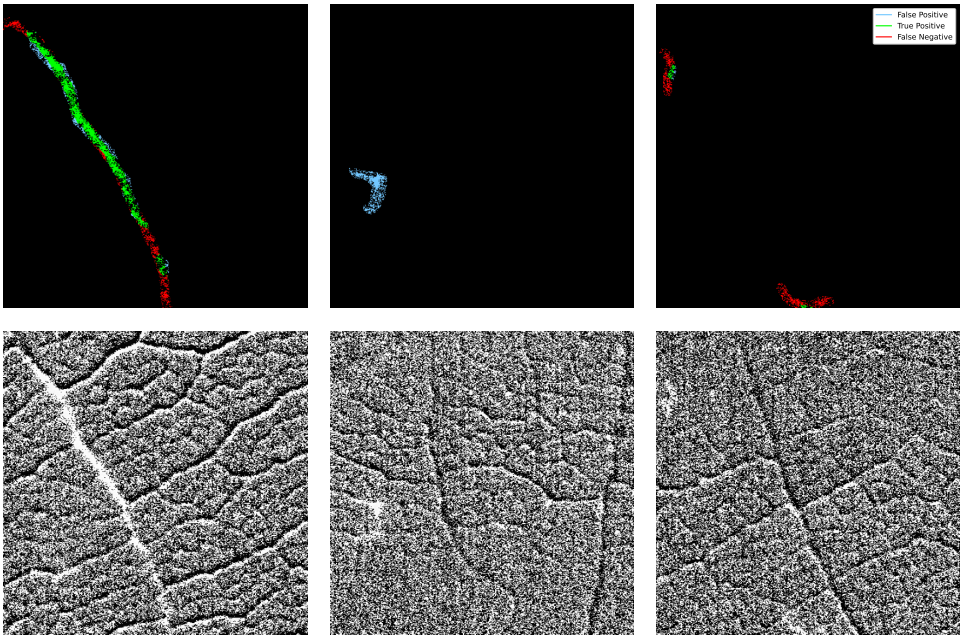


FIGURE D.6: Predictions using samples from the *Quercus palmeri*$_1$ test set. The top row shows the prediction and the bottom row shows the unprocessed input. The columns, in order from left to right, show the prediction with the most true positives, false positives, and false negatives respectively.

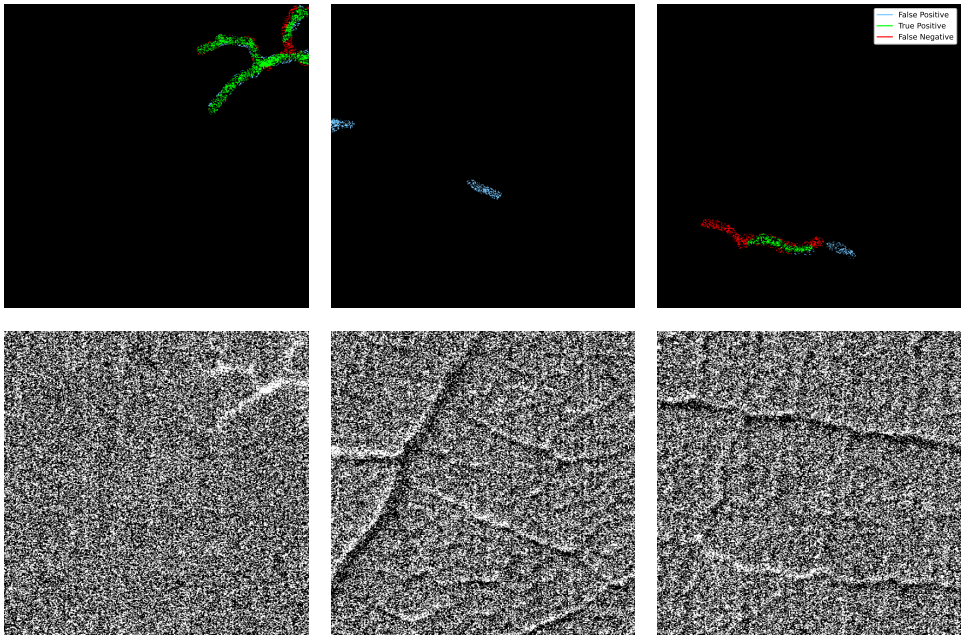


FIGURE D.7: Predictions using samples from the *Quercus palmeri*$_2$ test set. The top row shows the prediction and the bottom row shows the unprocessed input. The columns, in order from left to right, show the prediction with the most true positives, false positives, and false negatives respectively.

D.2 Experiment 2

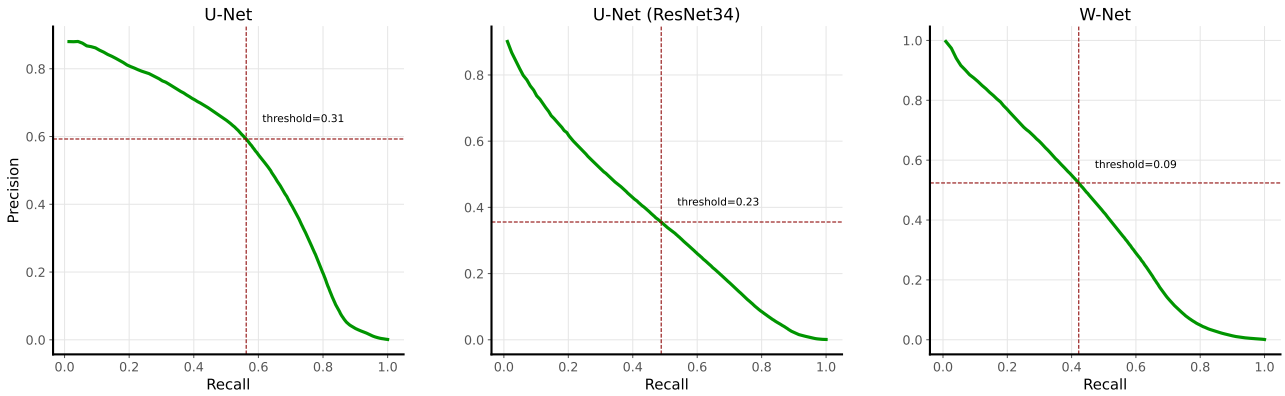


FIGURE D.8: A Precision-Recall curve, with the optimal threshold, constructed using the within species validation set for each model

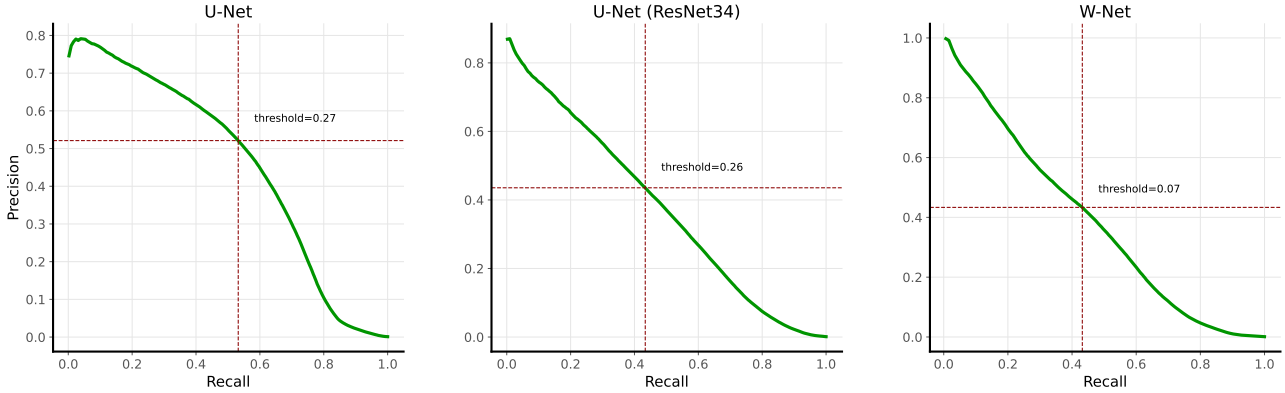


FIGURE D.9: A Precision-Recall curve, with the optimal threshold, constructed using the within species test set for each model

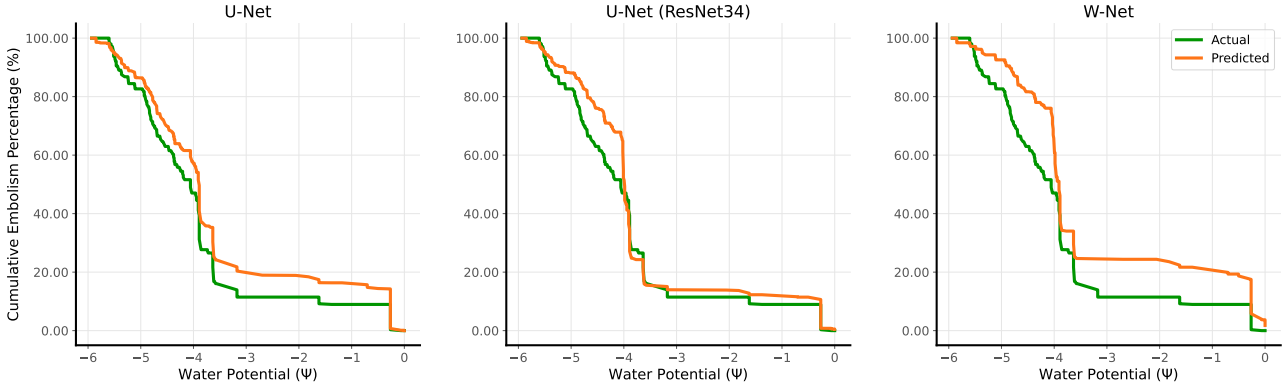


FIGURE D.10: A comparison of the true vulnerability curve against the predicted vulnerability curve for each model. The curves are created using the within species validation set.

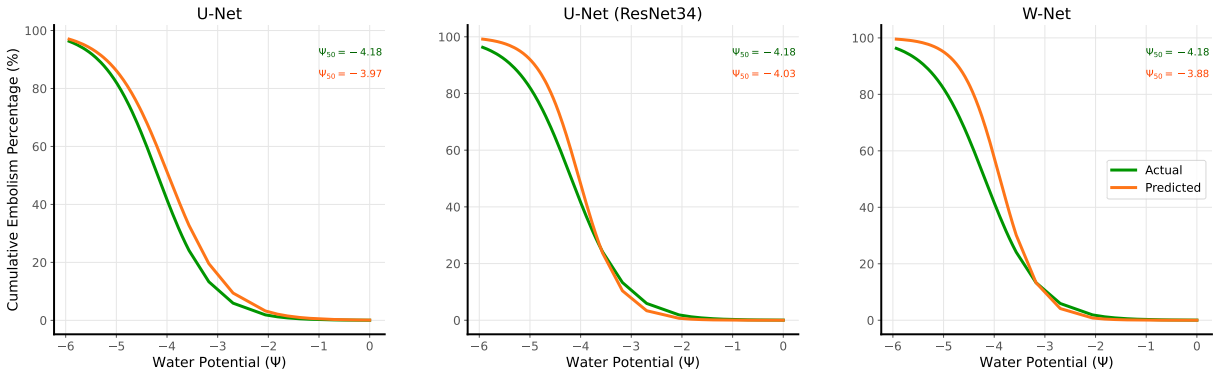


FIGURE D.11: A comparison of the true sigmoid vulnerability curve against the predicted sigmoid vulnerability curve for each model. The curves are created using the within species validation set.

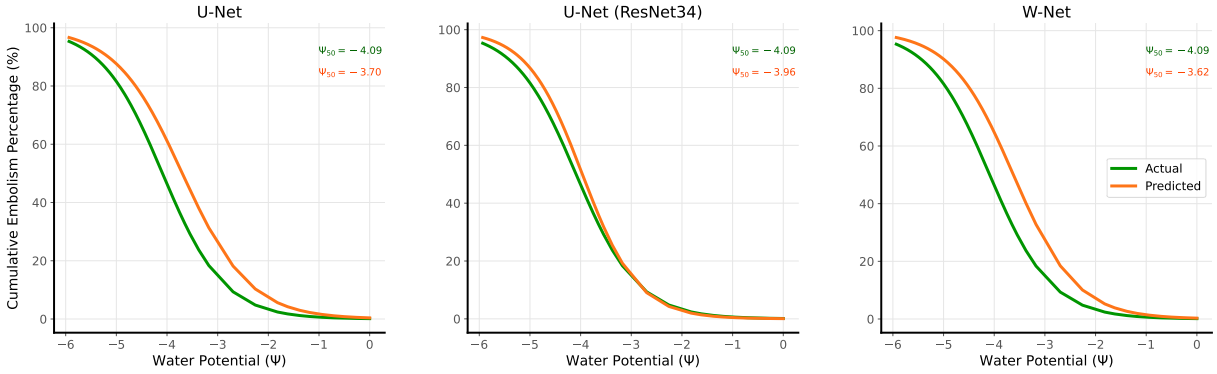


FIGURE D.12: A comparison of the true sigmoid vulnerability curve against the predicted sigmoid vulnerability curve for each model. The curves are created using the within species test set.

D.3 Experiment 3

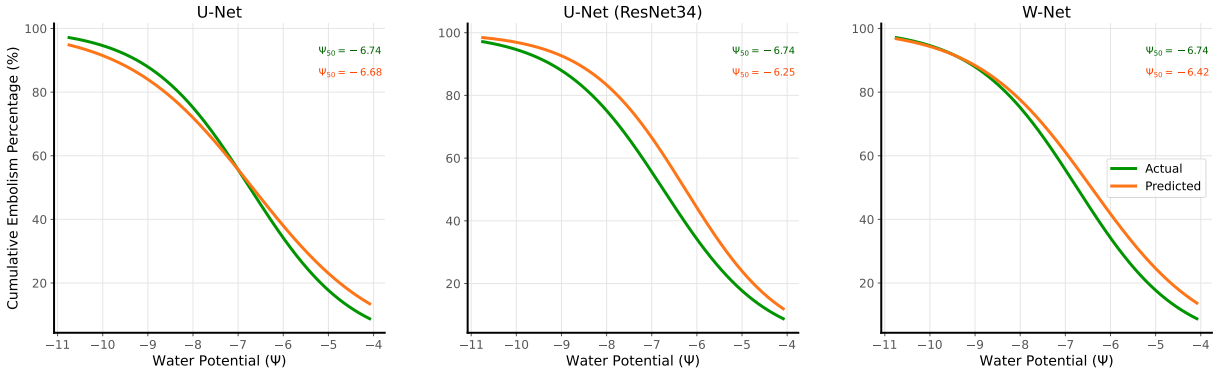


FIGURE D.13: A comparison of the true sigmoid vulnerability curve against the predicted sigmoid vulnerability curve for each model. The curves are created using the across species validation set.

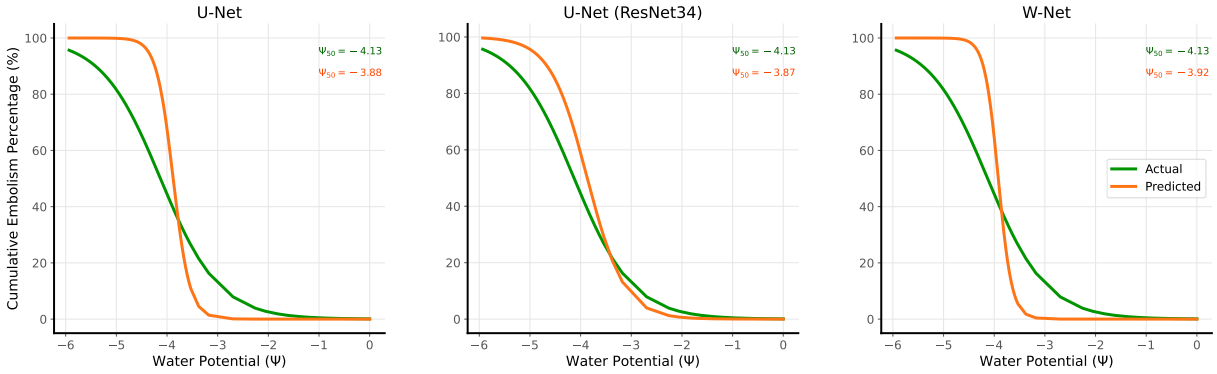


FIGURE D.14: A comparison of the true sigmoid vulnerability curve against the predicted sigmoid vulnerability curve for each model. The curves are created using the across species test set.

Bibliography

Adams, Henry D. et al. (2010). "Climate-Induced Tree Mortality: Earth System Consequences". In: *Eos (Washington, D.C.)* 91.17, pp. 153–154.

Allen, Craig D et al. (2010). "A global overview of drought and heat-induced tree mortality reveals emerging climate change risks for forests". In: *Forest ecology and management* 259.4, pp. 660–684.

Alom, Md Zahangir et al. (2018). "Recurrent residual convolutional neural network based on u-net (r2u-net) for medical image segmentation". In: *arXiv preprint arXiv:1802.06955*.

Anderegg, William RL, Jeffrey M Kane, and Leander DL Anderegg (2013). "Consequences of widespread tree mortality triggered by drought and temperature stress". In: *Nature climate change* 3.1, pp. 30–36.

Anderegg, William RL et al. (2016). "Meta-analysis reveals that hydraulic traits explain cross-species patterns of drought-induced tree mortality across the globe". In: *Proceedings of the National Academy of Sciences* 113.18, pp. 5024–5029.

Araujo, André, Wade Norris, and Jack Sim (2019). "Computing receptive fields of convolutional neural networks". In: *Distill* 4.11, e21.

Askenasy, E (1895). "Ueber das Saftsteigen Verh". In: *Nat. Med. Ver. Heidelb* 5, pp. 325–345.

Badrinarayanan, Vijay, Alex Kendall, and Roberto Cipolla (2017). "Segnet: A deep convolutional encoder-decoder architecture for image segmentation". In: *IEEE transactions on pattern analysis and machine intelligence* 39.12, pp. 2481–2495.

Bentrup, Friedrich-Wilhelm (2017). "Water ascent in trees and lianas: the cohesion-tension theory revisited in the wake of Otto Renner". In: *Protoplasma* 254.2, pp. 627–633.

Bouche, Pauline S. et al. (2016). "Are needles of Pinus pinaster more vulnerable to xylem embolism than branches? New insights from X-ray computed tomography". In: *Plant, cell and environment; Plant Cell Environ* 39.4, pp. 860–870. DOI: 10.1111/pce.12680.

Brochu, Eric, Vlad M. Cora, and Nando de Freitas (2010). "A Tutorial on Bayesian Optimization of Expensive Cost Functions, with Application to Active User Modeling and Hierarchical Reinforcement Learning". In: *CoRR* abs/1012.2599. arXiv: 1012.2599. URL: http://arxiv.org/abs/1012.2599.

Brodersen, Craig R et al. (2010). "The dynamics of embolism repair in xylem: in vivo visualizations using high-resolution computed tomography". In: *Plant physiology* 154.3, pp. 1088–1095.

Brodersen, Craig R. et al. (2013). "In Vivo Visualizations of Drought-Induced Embolism Spread in Vitis vinifera". In: *Plant physiology (Bethesda); Plant Physiol* 161.4, pp. 1820–1829.

Brodribb, Tim J. and Hervé Cochard (2009). "Hydraulic Failure Defines the Recovery and Point of Death in Water-Stressed Conifers". In: *Plant physiology (Bethesda); Plant Physiol* 149.1, pp. 575–584.

Brodribb, Timothy J, Diane Bienaimé, and Philippe Marmottant (2016). "Revealing catastrophic failure of leaf networks under stress". In: *Proceedings of the National Academy of Sciences* 113.17, pp. 4865–4869.

Brodribb, Timothy J., Herve Cochard, and Celia R. Dominguez (2019). "Measuring the pulse of trees; using the vascular system to predict tree mortality in the 21st century". In: *Conservation physiology; Conserv Physiol* 7.1.

Brodribb, Timothy J et al. (2016). "Visual quantification of embolism reveals leaf vulnerability to hydraulic failure". In: *New Phytologist* 209.4, pp. 1403–1409.

Brodribb, Timothy J. et al. (2017). "Optical Measurement of Stem Xylem Vulnerability". In: *Plant Physiology* 174.4, pp. 2054–2061. ISSN: 0032-0889. DOI: 10.1104/pp.17.00552. eprint: http://www.plantphysiol.org/content/174/4/2054.full.pdf. URL: http://www.plantphysiol.org/content/174/4/2054.

Brown, Harvey R. (2013). "The Theory of the Rise of Sap in Trees: Some Historical and Conceptual Remarks". In: *Physics in perspective* 15.3, pp. 320–358. DOI: 10.1007/s00016-013-0117-1.

Canny, Martin J (1997). "Vessel contents during transpiration—embolisms and refilling". In: *American Journal of Botany* 84.9, pp. 1223–1230.

Cardoso, Amanda A. et al. (2018). "Coordinated plasticity maintains hydraulic safety in sunflower leaves: Coordinated plasticity in sunflower leaves". In: *Plant, cell and environment* 41.11, pp. 2567–2576.

Chen, Liang-Chieh et al. (2017). "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs". In: *IEEE transactions on pattern analysis and machine intelligence* 40.4, pp. 834–848.

Choat, Brendan et al. (2012). "Global convergence in the vulnerability of forests to drought". In: *Nature (London); Nature* 491.7426, pp. 752–755.

Choat, Brendan et al. (2016). "Noninvasive measurement of vulnerability to drought-induced embolism by X-ray microtomography". In: *Plant Physiology* 170.1, pp. 273–282.

Choat, Brendan et al. (2018). "Triggers of tree mortality under drought". In: *Nature* 558.7711, pp. 531–539.

Ciresan, D., U. Meier, and J. Schmidhuber (2012). "Multi-column deep neural networks for image classification". In: *2012 IEEE Conference on Computer Vision and Pattern Recognition*, pp. 3642–3649. DOI: 10.1109/CVPR.2012.6248110.

Cochard, H., S. Delzon, and E. Basel (2015). "X-ray microtomography (micro-CT): a reference technology for high-resolution quantification of xylem embolism in trees". In: *Plant, cell and environment; Plant Cell Environ* 38.1, pp. 201–206.

Cochard, Hervé et al. (2013). "Methods for measuring plant vulnerability to cavitation: a critical review". In: *Journal of experimental botany; J Exp Bot* 64.15, pp. 4779–4791.

Creek, Danielle et al. (Oct. 2019). "Xylem embolism in leaves does not occur with open stomata: evidence from direct observations using the optical visualization technique". In: *Journal of Experimental Botany* 71.3, pp. 1151–1159. ISSN: 0022-0957. DOI: 10.1093/jxb/erz474. eprint: https://academic.oup.com/jxb/article-pdf/71/3/1151/31947058/erz474.pdf. URL: https://doi.org/10.1093/jxb/erz474.

Crombie, D. S., M. F. Hipkins, and J. A. Milburn (1985). "Gas Penetration of Pit Membranes in the Xylem of Rhododendron as the Cause of Acoustically Detectable Sap Cavitation". In: *Functional plant biology : FPB* 12.5, p. 445.

Crombie, D.S., J.A. Milburn, and M.F. Hipkins (1985). "Maximum sustainable xylem sap tensions in Rhododendron and other species". In: *Planta; Planta* 163.1, pp. 27–33.

Dalal, Navneet and Bill Triggs (2005). "Histograms of oriented gradients for human detection". In: *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*. Vol. 1. Ieee, pp. 886–893.

Davis, Larry S (1975). "A survey of edge detection techniques". In: *Computer graphics and image processing* 4.3, pp. 248–270.

Dixon, Henry H. and John Joly (1895). "XII. On the ascent of sap". In: *Philosophical transactions of the Royal Society of London.B* 186, pp. 563–576. DOI: 10.1098/rstb.1895.0012.

Dixon, Henry Horatio (1914). *Transpiration and the ascent of sap in plants*. New York: Macmillan and Company, limited, p. 102.

Duchi, John, Elad Hazan, and Yoram Singer (2011). "Adaptive subgradient methods for online learning and stochastic optimization." In: *Journal of machine learning research* 12.7.

Dumoulin, Vincent and Francesco Visin (2018). *A guide to convolution arithmetic for deep learning*. arXiv: 1603.07285 [stat.ML].

Engelbrecht, Bettina M. J. (2012). "Plant ecology: Forests on the brink". In: *Nature (London); Nature* 491.7426, p. 675.

Fontes, Clarissa G. and Jeannine Cavender-Bares (2020). "Toward an integrated view of the 'elephant': unlocking the mysteries of water transport and xylem vulnerability in oaks". In: *Tree physiology; Tree Physiol* 40.1, pp. 1–4.

Fromm, Jörg H. et al. (2001). "Xylem Water Content and Wood Density in Spruce and Oak Trees Detected by High-Resolution Computed Tomography". In: *Plant physiology (Bethesda); Plant Physiol* 127.2, pp. 416–425.

Fu, King-Sun and JK Mui (1981). "A survey on image segmentation". In: *Pattern recognition* 13.1, pp. 3–16.

Fukushima, Kunihiko and Sei Miyake (1982). "Neocognitron: A self-organizing neural network model for a mechanism of visual pattern recognition". In: *Competition and cooperation in neural nets*. Springer, pp. 267–285.

Galdran, Adrian et al. (2020). "The Little W-Net That Could: State-of-the-Art Retinal Vessel Segmentation with Minimalistic Models". In: *arXiv preprint arXiv:2009.01907*.

Ghiasi, Golnaz, Tsung-Yi Lin, and Quoc V. Le (2018). "DropBlock: A regularization method for convolutional networks". In: *NeurIPS*, pp. 10750–10760. URL: http://papers.nips.cc/paper/8271-dropblock-a-regularization-method-for-convolutional-networks.

Girshick, Ross et al. (2014). "Rich Feature Hierarchies for Accurate Object Detection and Semantic Segmentation". In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*.

Goodfellow, Ian et al. (2016). *Deep learning*. Vol. 1. 2. MIT press Cambridge.

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

Guo, Yanming et al. (2018). "A review of semantic segmentation using deep neural networks". In: *International journal of multimedia information retrieval* 7.2, pp. 87–93.

Haines, FM (1935). "Observations on the occurrence of air in conducting tracts". In: *Annals of botany* 49.194, pp. 367–379.

Hastie, Trevor, Robert Tibshirani, and Jerome Friedman (2009). *The elements of statistical learning: data mining, inference, and prediction*. Springer Science & Business Media.

Hatcher, William Grant and Wei Yu (2018). "A survey of deep learning: Platforms, applications and emerging research trends". In: *IEEE Access* 6, pp. 24411–24432.

He, Kaiming et al. (2016). "Deep residual learning for image recognition". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 770–778.

Hochberg, Uri et al. (2017). "Stomatal Closure, Basal Leaf Embolism, and Shedding Protect the Hydraulic Integrity of Grape Stems". In: *Plant physiology (Bethesda); Plant Physiol* 174.2, pp. 764–775.

Holbrook, N Michele et al. (2001). "In vivo observation of cavitation and embolism repair using magnetic resonance imaging". In: *Plant Physiology* 126.1, pp. 27–31.

Hubel, David H and Torsten N Wiesel (1962). "Receptive fields, binocular interaction and functional architecture in the cat's visual cortex". In: *The Journal of physiology* 160.1, pp. 106–154.

Ioffe, Sergey and Christian Szegedy (2015). "Batch normalization: Accelerating deep network training by reducing internal covariate shift". In: *International conference on machine learning*. PMLR, pp. 448–456.

Jégou, Simon et al. (2017). "The one hundred layers tiramisu: Fully convolutional densenets for semantic segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition workshops*, pp. 11–19.

Johnson, Kate M et al. (2020). "Xylem embolism spreads by single-conduit events in three dry forest angiosperm stems". In: *Plant Physiology* 184.1, pp. 212–222.

Kamilaris, Andreas and Francesc X Prenafeta-Boldú (2018). "Deep learning in agriculture: A survey". In: *Computers and electronics in agriculture* 147, pp. 70–90.

Katanforoosh, Kian and Daniel Kunin (2019). *Initializing neural networks*. URL: https://www.deeplearning.ai/ai-notes/initialization/.

Kingma, Diederik P and Jimmy Ba (2014). "Adam: A method for stochastic optimization". In: *arXiv preprint arXiv:1412.6980*.

Kirillov, Alexander et al. (2019). "Panoptic segmentation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pp. 9404–9413.

Kirsch, Russell A et al. (1957). "Experiments in processing pictorial information with a digital computer". In: *Papers and discussions presented at the December 9-13, 1957, eastern joint computer conference: Computers with deadlines to meet*, pp. 221–229.

Klambauer, Günter et al. (2017). "Self-Normalizing Neural Networks". In: *CoRR* abs/1706.02515. arXiv: 1706.02515. URL: http://arxiv.org/abs/1706.02515.

Knipfer, Thorsten et al. (2015). "Patterns of drought-induced embolism formation and spread in living walnut saplings visualized using X-ray microtomography". In: *Tree physiology* 35.7, pp. 744–755.

Krizhevsky, Alex, Ilya Sutskever, and Geoffrey E Hinton (2012). "Imagenet classification with deep convolutional neural networks". In: *Advances in neural information processing systems*, pp. 1097–1105.

Le Cun, Y. et al. (1989). "Handwritten digit recognition: applications of neural network chips and automatic learning". In: *IEEE communications magazine* 27.11, pp. 41–46.

LeCun, Yann, Koray Kavukcuoglu, and Clément Farabet (2010). "Convolutional networks and applications in vision". In: *Proceedings of 2010 IEEE international symposium on circuits and systems*. IEEE, pp. 253–256.

LeCun, Yann et al. (1998). "Gradient-based learning applied to document recognition". In: *Proceedings of the IEEE* 86.11, pp. 2278–2324.

Li, Liangzhi et al. (2020). "Iternet: Retinal image segmentation utilizing structural redundancy in vessel networks". In: *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pp. 3656–3665.

Lin, Min, Qiang Chen, and Shuicheng Yan (2013). "Network in network". In: *arXiv preprint arXiv:1312.4400*.

Lin, Tsung-Yi et al. (2017). "Focal loss for dense object detection". In: *Proceedings of the IEEE international conference on computer vision*, pp. 2980–2988.

Litjens, Geert et al. (2017). "A survey on deep learning in medical image analysis". In: *Medical image analysis* 42, pp. 60–88.

Long, Jonathan, Evan Shelhamer, and Trevor Darrell (2015). "Fully convolutional networks for semantic segmentation". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 3431–3440.

Longuetaud, Fleur et al. (2016). "Within-stem maps of wood density and water content for characterization of species: a case study on three hardwood and two softwood species". In: *Annals of Forest Science* 73.3, pp. 601–614.

Losso, Adriano et al. (2019). "Insights from in vivo micro-CT analysis: testing the hydraulic vulnerability segmentation in Acer pseudoplatanus and Fagus sylvatica seedlings". In: *New Phytologist* 221.4, pp. 1831–1842.

Lowe, David G (2004). "Distinctive image features from scale-invariant keypoints". In: *International journal of computer vision* 60.2, pp. 91–110.

Maier, Jennifer et al. (2020). "Multi-Channel Volumetric Neural Network for Knee Cartilage Segmentation in Cone-beam CT". In: *Bildverarbeitung für die Medizin 2020*. Springer, pp. 67–72.

May, Michael R. et al. (Dec. 2009). "A Pleistocene Clone of Palmer's Oak Persisting in Southern California". In: *PLOS ONE* 4.12, pp. 1–5. DOI: 10.1371/journal.pone.0008346. URL: https://doi.org/10.1371/journal.pone.0008346.

McDowell, Nate G., Timothy J. Brodribb, and Andrea Nardini (2019). "Hydraulics in the 21st century". In: *The New phytologist* 224.2, pp. 537–542. DOI: 10.1111/nph.16151.

McElrone, Andrew J et al. (2013). "Using high resolution computed tomography to visualize the three dimensional structure and function of plant vasculature". In: *JoVE (Journal of Visualized Experiments)* 74, e50162.

Milburn, John A. (1966). "The conduction of sap: I. water conduction and cavitation in water stressed leaves". eng. In: *Planta* 69.1, pp. 34–42. ISSN: 0032-0935.

– (1973a). "Cavitation in Ricinus by Acoustic Detection: Induction in Excised Leaves by Various Factors". In: *Planta; Planta* 110.3, pp. 253–265.

– (1973b). "Cavitation Studies on Whole Ricinus Plants by Acoustic Detection". In: *Planta; Planta* 112.4, pp. 333–342.

Milburn, John A and RPC Johnson (1966). "The conduction of sap". In: *Planta* 69.1, pp. 43–52.

Milburn, John A. and Margaret E. McLaughlin (1974). "Studies of Cavitation in Isolated Vascular Bundles and Whole Leaves of Plantago major L". In: *The New phytologist* 73.5, pp. 861–871.

Ming Liang and Xiaolin Hu (2015). *Recurrent convolutional neural network for object recognition*. DOI: 10.1109/CVPR.2015.7298958.

Mnih, Volodymyr and Geoffrey E Hinton (2010). "Learning to detect roads in high-resolution aerial images". In: *European Conference on Computer Vision*. Springer, pp. 210–223.

Morton, A. G. (1981). *History of botanical science : an account of the development of botany from ancient times to the present day*. London: Academic Press. ISBN: 0125083823.

Nardini, Andrea and Jessica Luglio (2014). "af hydraulic capacity and drought vulnerability: possible trade-offs and correlations with climate across three major biomes". In: *Functional Ecology* 28.4, pp. 810–818. DOI: 10.1111/1365-2435.12246.

Nardini, Andrea et al. (2017). "X-ray microtomography observations of xylem embolism in stems of Laurus nobilis are consistent with hydraulic measurements of percentage loss of conductance". In: *New Phytologist* 213.3, pp. 1068–1075.

Noh, Hyeonwoo, Seunghoon Hong, and Bohyung Han (2015). "Learning deconvolution network for semantic segmentation". In: *Proceedings of the IEEE international conference on computer vision*, pp. 1520–1528.

Nolf, Markus et al. (2017). "Visualization of xylem embolism by X-ray microtomography: a direct test against hydraulic measurements". In: *New Phytologist* 214.2, pp. 890–898.

Pal, Nikhil R and Sankar K Pal (1993). "A review on image segmentation techniques". In: *Pattern recognition* 26.9, pp. 1277–1294.

Parmesan, Camille and Gary Yohe (2003). "A globally coherent fingerprint of climate change impacts across natural systems". In: *Nature (London); Nature* 421.6918, pp. 37–42. DOI: 10.1038/nature01286.

Petruzzellis, Francesco et al. (2020). "A Leaf Selfie: Using a Smartphone to Quantify Leaf Vulnerability to Hydraulic Dysfunction". In: *Plants (Basel); Plants (Basel)* 9.2, p. 234. DOI: 10.3390/plants9020234.

Pickard, William F (1981). "The ascent of sap in plants". In: *Progress in biophysics and molecular biology* 37, pp. 181–229.

Polyak, Boris T (1964). "Some methods of speeding up the convergence of iteration methods". In: *Ussr computational mathematics and mathematical physics* 4.5, pp. 1–17.

Powell, Thomas L. et al. (2017). "Differences in xylem and leaf hydraulic traits explain differences in drought tolerance among mature Amazon rainforest trees". In: *Global Change Biology; Glob Chang Biol* 23.10, pp. 4280–4293. DOI: 10.1111/gcb.13731.

Pratt, R Brandon et al. (2020). "Factors controlling drought resistance in grapevine (Vitis vinifera, chardonnay): application of a new micro CT method to assess functional embolism resistance". In: *American Journal of Botany* 107.4, pp. 618–627.

Preston, RD (1959). "Water stresses in plants: Theoretical and practical implications of the stresses in the water conducting system". In: *Recent advances in botany* 2, pp. 1144–1149.

Rao, Y. et al. (2018). "Roads Detection of Aerial Image with FCN-CRF Model". In: *2018 IEEE Visual Communications and Image Processing (VCIP)*, pp. 1–4. DOI: 10.1109/VCIP.2018.8698718.

Renner, Otto (1911). "Experimentelle beiträge zur kenntnis der wasserbewegung". In: *Flora oder Allgemeine Botanische Zeitung* 103.3, pp. 171–247.

Richter, Hanno and Pierre Cruiziat (2002). *A Brief History of the Study of Water Movement in the Xylem*. URL: http://6e.plantphys.net/essay04.02.html.

Roberts, Lawrence G (1963). "Machine perception of three-dimensional solids". PhD thesis. Massachusetts Institute of Technology.

Ronneberger, Olaf, Philipp Fischer, and Thomas Brox (2015). "U-net: Convolutional networks for biomedical image segmentation". In: *International Conference on Medical image computing and computer-assisted intervention*. Springer, pp. 234–241.

Ruder, Sebastian (2016). "An overview of gradient descent optimization algorithms". In: *CoRR* abs/1609.04747. arXiv: 1609.04747. URL: http://arxiv.org/abs/1609.04747.

Russakovsky, Olga et al. (2015). "Imagenet large scale visual recognition challenge". In: *International journal of computer vision* 115.3, pp. 211–252.

Saito, Takaya and Marc Rehmsmeier (Mar. 2015). "The Precision-Recall Plot Is More Informative than the ROC Plot When Evaluating Binary Classifiers on Imbalanced Datasets". In: *PLOS ONE* 10.3, pp. 1–21.

Savi, Tadeja et al. (2017). "Drought-induced embolism in stems of sunflower: a comparison of in vivo micro-CT observations and destructive hydraulic measurements". In: *Plant physiology and biochemistry* 120, pp. 24–29.

Scholander, Per F et al. (1965). "Sap pressure in vascular plants: negative hydrostatic pressure can be measured in plants". In: *Science* 148.3668, pp. 339–346.

Simonyan, Karen and Andrew Zisserman (2014). "Very deep convolutional networks for large-scale image recognition". In: *arXiv preprint arXiv:1409.1556*.

Skelton, Robert P, Timothy J Brodribb, and Brendan Choat (2017). "Casting light on xylem vulnerability in an herbaceous species reveals a lack of segmentation". In: *New Phytologist* 214.2, pp. 561–569.

Skelton, Robert P et al. (2019). "No local adaptation in leaf or stem xylem vulnerability to embolism, but consistent vulnerability segmentation in a North American oak". In: *New Phytologist* 223.3, pp. 1296–1306.

Skelton, Robert P et al. (2021). "Evolutionary relationships between drought-related traits and climate shape large hydraulic safety margins in western North American oaks". In: *Proceedings of the National Academy of Sciences* 118.10.

Skelton, Robert Paul et al. (2018). "Low Vulnerability to Xylem Embolism in Leaves and Stems of North American Oaks". eng. In: *Plant physiology (Bethesda)* 177.3, pp. 1066–1077. ISSN: 1532-2548.

Smith, Leslie N (2017). "Cyclical learning rates for training neural networks". In: *2017 IEEE winter conference on applications of computer vision (WACV)*. IEEE, pp. 464–472.

Smith-Martin, Chris M et al. (2020). "Lack of vulnerability segmentation among woody species in a diverse dry sclerophyll woodland community". In: *Functional Ecology* 34.4, pp. 777–787.

Snoek, Jasper, Hugo Larochelle, and Ryan P Adams (2012). "Practical Bayesian Optimization of Machine Learning Algorithms". In: *Advances in Neural Information*

Processing Systems. Ed. by F. Pereira et al. Vol. 25. Curran Associates, Inc. URL: https://proceedings.neurips.cc/paper/2012/file/05311655a15b75fab86956663e1819cd-Paper.pdf.

Spanner, D. C. (1951). "The Peltier Effect and its Use in the Measurement of Suction Pressure". In: *Journal of experimental botany* 2.2, pp. 145–168.

Sperry, John S (1985). "Xylem embolism in the palm Rhapis excelsa". In: *IAWA journal* 6.4, pp. 283–292.

Sperry, John S and Melvin T Tyree (1988). "Mechanism of water stress-induced xylem embolism". In: *Plant physiology* 88.3, pp. 581–587.

Stocking, C Ralph (1945). "The calculation of tensions in Cucurbita pepo". In: *American Journal of Botany*, pp. 126–134.

Stojnic, Robert et al. (2021). *Image Classification on ImageNet*. URL: https://paperswithcode.com/sota/image-classification-on-imagenet.

Strasburger, Eduard (1891). *Über den Bau und die Verrichtungen der Leitungsbahnen in den Pflanzen*. 3. G. Fischer.

Sultana, Farhana, Abu Sufian, and Paramartha Dutta (2020). "Evolution of Image Segmentation using Deep Convolutional Neural Network: A Survey". In: *Knowledge-Based Systems* 201-202, p. 106062.

Suuronen, Jussi-Petteri et al. (2013). "Visualizing water-filled versus embolized status of xylem conduits by desktop x-ray microtomography". In: *Plant Methods* 9.1, pp. 1–13.

Szegedy, Christian et al. (2015). "Going deeper with convolutions". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 1–9.

Taiz, Lincoln and Eduardo Zeiger (2010). *Plant physiology*. 5th ed. Sunderland, MA: Sinauer Associates. ISBN: 9780878938667.

Temperley, HNV (1947). "The behaviour of water under hydrostatic tension: III". In: *Proceedings of the Physical Society* 59.2, pp. 199–207.

Tieleman, Tijmen and Geoffrey Hinton (2012). "Lecture 6.5-rmsprop: Divide the gradient by a running average of its recent magnitude". In: *COURSERA: Neural networks for machine learning* 4.2, pp. 26–31.

Torres-Ruiz, José M et al. (2014). "Vulnerability to cavitation in Olea europaea current-year shoots: further evidence of an open-vessel artifact associated with centrifuge and air-injection techniques". In: *Physiologia Plantarum* 152.3, pp. 465–474.

Trenberth, Kevin E. et al. (2013). "Global warming and changes in drought". In: *Nature climate change* 4.1, pp. 17–22. DOI: 10.1038/nclimate2067.

Tyree, M. T. and J. S. Sperry (1989). "Vulnerability of Xylem to Cavitation and Embolism". In: *Annual Review of Plant Physiology and Plant Molecular Biology* 40.1, pp. 19–36. DOI: 10.1146/annurev.pp.40.060189.000315. URL: https://www.annualreviews.org/doi/10.1146/annurev.pp.40.060189.000315.

Tyree, Melvin T. (1997). "The Cohesion-Tension theory of sap ascent: current controversies". In: *Journal of experimental botany* 48.315, pp. 1753–1765.

Tyree, Melvin T. and Michael A. Dixon (1983). "Cavitation Events in Thuja occidentalis L.? Utrasonic Acoustic Emissions from the Sapwood Can Be Measured". In: *Plant physiology (Bethesda); Plant Physiol* 72.4, pp. 1094–1099.

Urli, Morgane et al. (2013). "Xylem embolism threshold for catastrophic hydraulic failure in angiosperm trees". In: *Tree physiology* 33.7, pp. 672–683.

Venturas, Martin D, John S Sperry, and Uwe G Hacke (2017). "Plant xylem hydraulics: what we understand, current research, and future challenges". In: *Journal of Integrative Plant Biology* 59.6, pp. 356–389.

West, D.W. and D.F. Gaff (1976). "Xylem Cavitation in Excised Leaves of Malus sylvestris Mill. and Measurement of Leaf Water Status with the Pressure Chamber". In: *Planta; Planta* 129.1, pp. 15–18.

Woo, Sanghyun et al. (2018). "CBAM: Convolutional Block Attention Module". In: *Proceedings of the European Conference on Computer Vision (ECCV)*.

Xie, Saining and Zhuowen Tu (2015). "Holistically-Nested Edge Detection". In: *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*.

Yang, Xiaofei et al. (2019). "Road detection and centerline extraction via deep recurrent convolutional neural network U-Net". In: *IEEE Transactions on Geoscience and Remote Sensing* 57.9, pp. 7209–7220.

Zeiler, Matthew D and Rob Fergus (2014). "Visualizing and understanding convolutional networks". In: *European conference on computer vision*. Springer, pp. 818–833.

Zhang, Feng-Ping and Timothy J. Brodribb (2017). "Are flowers vulnerable to xylem cavitation during drought?" In: *Proceedings of the Royal Society.B, Biological sciences; Proc Biol Sci* 284.1854, p. 20162642.

Zhang, Yu-Jin (2006). "An overview of image and video segmentation in the last 40 years". In: *Advances in Image and Video Segmentation*, pp. 1–16.

Zhang, Zhengxin, Qingjie Liu, and Yunhong Wang (2018). "Road extraction by deep residual u-net". In: *IEEE Geoscience and Remote Sensing Letters* 15.5, pp. 749–753.

Zhao, H. et al. (2017). "Pyramid Scene Parsing Network". In: *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6230–6239. DOI: 10.1109/CVPR.2017.660.

Zheng, Shuai et al. (2015). "Conditional random fields as recurrent neural networks". In: *Proceedings of the IEEE international conference on computer vision*, pp. 1529–1537.

Zhou, Xinyu et al. (2017). "East: an efficient and accurate scene text detector". In: *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pp. 5551–5560.

Zhuang, Juntang (2018). "Laddernet: Multi-path networks based on u-net for medical image segmentation". In: *arXiv preprint arXiv:1810.07810*.

Zimmermann, Ulrich et al. (2002). "What are the driving forces for water lifting in the xylem conduit?" In: *Physiologia Plantarum; Physiol Plant* 114.3, pp. 327–335. DOI: 10.1034/j.1399-3054.2002.1140301.x.