Instituto Superior
de Engenharia
Politécnico de Coimbra

DEPARTAMENTO DE ENGENHARIA INFORMÁTICA E SISTEMAS

Cork Oak Production Estimation Using a Mask-RCNN

Relatório de Trabalho de Projeto para a obtenção do grau de Mestre em Engenharia Informática

Especialização em Análise Inteligente de Dados

Autor

André Filipe Maia Guimarães

Orientador

Mateus Daniel A. Mendes

Co-orientador

Carlos Manuel Jorge da Silva Pereira

INSTITUTO POLITÉCNICO
DE COIMBRA

INSTITUTO SUPERIOR
DE ENGENHARIA
DE COIMBRA

Coimbra, Junho de 2023

Abstract

Cork is a highly versatile natural material with various applications, including its use as an insulator in construction. To ensure sustainable management of cork oak forests, forest owners must accurately determine when to harvest the cork by periodically calculating the cork volume. However, the traditional method for this calculation is labor-intensive and time-consuming. This study aims to streamline the process of calculating the trunk area of a cork oak, which correlates with the cork production potential. By automating this calculation, it becomes possible to estimate the volume of cork that can be harvested before the stripping process. The research utilizes advanced techniques such as a deep neural network called Mask R-CNN, combined with machine learning algorithms. A dataset of images featuring cork oak trees was created, incorporating targets with known dimensions placed on the tree trunks. The Mask R-CNN model was trained to recognize these targets and accurately identify the regions of cork on the trunks. This allowed for the estimation of the cork area based on the known dimensions of the targets. The results demonstrate the effectiveness of the model in recognizing targets and tree trunks, achieving a mean average precision of 0.96 at an intersection-over-union threshold of 0.7 (mAP@0.7). After obtaining the mask results, three machine learning models were trained to estimate the volume of cork based on the cork area and various biometric parameters of the tree. The results reveal that the best-performing model, utilizing the Support Vector Machine algorithm, achieved an error rate of only 0.15%. The other models utilizing the same algorithm recorded error rates of 8.75%, 2.96%, and 2.74% respectively, all of which are lower than the error margins obtained using traditional methods.

Keywords: Forest management, *Quercus suber*, Cork volume, Machine learning, Mask R-CNN

Resumo

A cortiça é um material natural muito versátil com várias aplicações, incluindo a sua utilização como isolante na construção. Para assegurar uma gestão sustentável dos montados de sobreiro, os proprietários florestais devem determinar com exatidão o momento de extrair a cortiça, calculando periodicamente o volume de cortiça. No entanto, o método tradicional para este cálculo é trabalhoso e demorado. Este estudo tem como objetivo simplificar o processo de cálculo da área do tronco de um sobreiro, que se correlaciona com o potencial de produção de cortiça. Ao automatizar este cálculo, torna-se possível estimar o volume de cortiça que pode ser extraído antes do processo de descortiçamento. A investigação utiliza técnicas avançadas, como uma rede neuronal profunda chamada Mask R-CNN, combinada com algoritmos de *machine learning*. Foi criado um *dataset* de imagens de sobreiros, incorporando alvos com dimensões conhecidas colocados nos troncos das árvores. O modelo Mask R-CNN foi treinado para reconhecer estes alvos e identificar com precisão as regiões onde irá ser retirada a cortiça nos troncos. Os resultados demonstram a eficácia do modelo no reconhecimento de alvos e troncos de árvores, alcançando uma precisão média de 0,96 num *threshold* de intersecção-sobre-união de 0,7 (mAP@0.7). Depois de obter os resultados da máscara, foram treinados três modelos de *machine learning* para estimar o volume de cortiça com base na área da cortiça e em vários parâmetros biométricos da árvore. Os resultados revelam que o modelo com melhor desempenho, utilizando o algoritmo *Support Vector Machine*, obteve uma taxa de erro de apenas 0,15%. Os outros modelos, utilizando o mesmo algoritmo, registaram taxas de erro de 8,75%, 2,96% e 2,74%, respectivamente, todas elas inferiores às margens de erro margens de erro obtidas com os métodos tradicionais.

Palavras-chave: Gestão florestal, Sobreiro, Volume da cortiça, *Machine learning*, Mask R-CNN

Epigraph

The only way to do great work is to love what you do.
Steve Jobs

Dedication

I dedicate this thesis work to all those who have inspired and supported me throughout this challenging journey. Their unwavering belief in my abilities and their encouragement have been the driving force behind my pursuit of knowledge and the completion of this research.

First and foremost, I would like to express my deepest gratitude to my thesis advisor, Prof. Mateus Mendes, and my thesis co-advisor, Prof. Carlos Pereira. Your guidance, expertise and patience have been invaluable in shaping this work. Your commitment to excellence and your passion for research have inspired me to strive for nothing less than my best. Thank you for pushing me to explore new horizons, for your unwavering support and for sharing your knowledge so generously.

I would like to thank all the teachers and non-teaching staff of the Instituto Superior de Engenharia de Coimbra, who have been part of my academic journey, both in my bachelor's and master's degrees, for all the teachings and contributions to my personal growth.

To my closest friends (Diogo, Ema, Sara, Fred, Edgar and Almeida), for always being a refuge and for being there for me in all the important moments of my life.

To my family, thank you for your unconditional love, encouragement, and unwavering belief in me. Your constant support and understanding during the long hours spent studying and researching have been instrumental in my success. I am truly blessed to have you by my side.

To Elisa, for all the unconditional support, patience and understanding that she has shown me since the first day of this thesis.

To all, my sincere thanks.

Acknowledgements

I would like to express my heartfelt gratitude to ESAC (Escola Superior Agrária de Coimbra) for their invaluable support and resources throughout the course of my research. In particular, I extend my sincere appreciation to Professor Raúl Salas-Gonzalez for his remarkable guidance and expertise in the field of cork oaks and forest measurement techniques. His extensive knowledge and unwavering dedication have been instrumental in shaping the direction of my thesis. I am also immensely grateful to Professor Beatriz Fidalgo for her valuable contributions and insightful discussions during the development of this research.

Furthermore, I would like to extend my gratitude to my colleague Maria Valério for her exceptional assistance and collaboration. Her generous sharing of field images and biometric data of the trees has proven invaluable in augmenting the quality and comprehensiveness of this research.

List of Contents

List of Tables

List of Figures

List of Acronyms and Abbreviations

NFI	National Forest Inventory
DBH	Diameter at Breast Height
Mask R-CNN	Mask Region-Based Convolutional Neural Network
Faster R-CNN	Faster Region-Based Convolutional Network
ESAC	Escola Superior Agrária de Coimbra
CNN	Convolutional Neural Network
BA	Basal Area
FPN	Feature Pyramid Network
ANN	Artificial Neural Network
RPN	Region Proposal Network
AP	Average Precision
CVAT	Computer Vision Annotation Tool
MIT	Massachusetts Intitute of Technology
R-CNN	Region-based Convolution Neural Network
FCN	Fully Convolutional Network
SVM	Support Vector Machine
ROI	Region of Interest
Fast R-CNN	Fast Region-Based Convolutional Network
IoU	Interception Over Union
NMS	Non Maximum Suppression
ASPP	Atrous Spatial Pyramid Pooling
mAP	Mean Average Precision
MAPE	Mean Absolute Percentage Error
MSE	Mean Squared Error
RMSE	Root Mean Squared Error

1 Introduction

The purpose of this section is to provide an overview and outline of the study that was undertaken in the course of this project. It starts by setting the stage and providing background information on the existing problem, along with an approach to solve it. Following that, the project's goals will be presented, which are designed to resolve the identified problem. Lastly, a detailed work plan was outlined, explaining the essential steps required to execute the project.

1.1 Background

The cork oak forest in Portugal is highly valued and considered a strategic asset for the country. It is a versatile ecosystem that primarily yields cork, a material with exceptional technological properties such as elasticity, impermeability, and thermal insulation. Throughout history, cork has been widely used in construction due to its low thermal conductivity and impermeability. Nowadays, it is utilized in various construction materials like blocks, cork rubber, insulation sheets, and filling materials. It finds applications indoors and outdoors, including floors, walls, roofs, and even in paint to enhance thermal properties. The longevity of cork materials is typically over 50 years.

According to the latest Portuguese national forest inventory (NFI) conducted in 2013, cork oak covers approximately 23% of the country's forested area [1]. Portugal holds the distinction of being the world's largest exporter of cork, controlling a significant 62.4% share of the global cork trade [2]. The cork oak tree is characterized by its remarkable bark, known as cork, which grows in a continuous layer encompassing the entire trunk and branches. Cork extraction, also known as stripping, occurs periodically on a nine- or ten-year cycle without causing harm to the tree. Subsequently, a new bark layer forms on the exposed stem surface [3].

To estimate cork production, the Portuguese forest inventory employs circular field plots measuring 2000 m^2. These plots involve measuring various dendrometric characteristics of the trees. However, the inventory's sampling intensity only yields national and regional-level results, lacking the necessary detail to guide management decisions at the forest or stand level [4]. Consequently, when it comes to forest exploitation, additional field inventories must be conducted to estimate cork oak production.

1.2 The Problem

Traditional inventory methods involve field measurements of the cork oak's diameter at breast height (DBH) and height. DBH is measured using a caliper or diameter tape, while the hypsometer is used to measure total tree height, crown height, stem height without branches, and the height at which the stem begins to fork. Volume and cork production estimates are then calculated using equations based on DBH and stripped tree height [5]. An alternative non-destructive method involves using the Bitterlich relascope to estimate trunk diameters at different heights. The volume of each trunk section is more accurately estimated using the Smalian or Newton formulae [6]. However, this method is labor-intensive and costly, limiting its widespread application. Therefore, it is essential to develop new non-destructive methodologies that accurately estimate standing tree volume while reducing forest management costs [7].

Advancements in technology have introduced remote sensing methods that can provide valuable information about trees and forest stands [4]. These methods require less time and effort, simplifying measurement work. However, the use of remote sensing technology is still limited due to associated costs, data processing challenges, equipment availability, and the need for specialized personnel [7].

1.3 Approach

To tackle the challenges at hand, the current study proposes an innovative method for automating the estimation of cork volume in cork oak trees. The primary objective is to present a streamlined approach that can accurately determine the volume of cork while being non-destructive and cost-effective, thereby providing a valuable solution for forest management.

The proposed method leverages the power of a Mask R-CNN (Mask Region-Based Convolutional Neural Network), a deep learning model that excels in instance segmentation tasks. The Mask R-CNN extends the capabilities of the Faster R-CNN (Faster Region-Based Convolutional Network) [8], which is specifically designed for such tasks. By utilizing the Mask R-CNN, we can effectively identify the areas of interest within the cork oak tree, which is a crucial step towards estimating the cork volume.

Once the mask is predicted and the trunk area is calculated, a machine learning algorithm is employed to predict the oak volume. This algorithm takes into account the information obtained from the identified areas, as well as other relevant data. By combining the power of deep learning with traditional machine learning techniques, we could achieve more accurate and reliable volume estimations.

The proposed method aims to automate the volume estimation process, reducing the need for extensive manual labor and saving valuable time and resources. By leveraging

machine learning algorithms, we can streamline the entire process, making it more efficient and cost-effective for forest management practices.

1.4 Main Contributions

In partnership with the Coimbra Agriculture School (ESAC), we created a comprehensive dataset specifically tailored for this task. The dataset consists of a collection of images showcasing cork oaks before undergoing the stripping process. The dataset was made available to the public on Kaggle, accessible at [`https://www.kaggle.com/datasets/andreguim/cork-oak-segmentation`], and was released on 25 October 2022.

As part of this research project, we conducted extensive work and studies that led to valuable insights and contributions. These contributions were disseminated through two publications. The first publication, titled "Cork Oak Production Estimation Using a Mask R-CNN," was presented in May 2023 at CongrEGA 2022, the first National Congress in the field of Engineering and Asset Management [8]. The second publication, titled "Cork Oak Production Estimation Using a Mask R-CNN," was published in the journal Energies MDPI in December 2022 [9]. Both articles delve into the methodology, results, and findings obtained during our research, thereby contributing to the scientific literature in the domain of cork oak production estimation.

To bring the project to fruition and make the volume estimation component readily available to users, we integrated it into the larger *Floresta Digital* (Digital Forest) project. The *Floresta Digital* project aims to provide calculation simulators for various tree types, and our volume estimation component plays a crucial role in this endeavor. Users can now access the platform through the official website of the project at [`http://www.floresta.digital.esac.pt`].

1.5 Objectives

The general objectives of this work are as follows:

- Train a model capable of recognising and segmenting, from an image, the area of the trunk of a cork oak before the stripping process, using a deep learning approach.
- Create an area calculation process from the trunk mask obtained by the generated deep learning model.
- Develop a method capable of estimating the volume of cork based on the previously calculated trunk area and biometric data of the tree.

1.6 Document Structure

This Master dissertation is divided into eight chapters and two appendices, with the remaining document structured as follows.

Chapter Two:

In Chapter 2, an in-depth exploration of existing research in the field of volume calculation through image processing and deep learning techniques is presented. This literature review offers a comprehensive overview of various studies and endeavors undertaken to address the precise task of estimating volumes using advanced computational methods.

Within this literature review, significant attention is devoted to elucidating the key findings and methodologies employed in these previous studies. By thoroughly examining the outcomes and techniques employed in the scope of volume calculation, a comprehensive understanding of the existing landscape is achieved.

Chapter Three:

In chapter three, our aim is to explore the cutting-edge methodologies employed in the classification, detection, and segmentation of objects through the utilization of deep learning techniques, predominantly focusing on region-based convolutional neural networks (CNNs). We will familiarize ourselves with the core principles that form the foundation of convolutional neural networks and their pragmatic applications in the field of image recognition. Subsequently, our exploration will delve deeper into the complexities and nuances of region-based convolutional neural networks, specifically engineered to effectively address the challenges posed by object detection and segmentation through the process of partitioning the input image into discernible regions of interest.

Chapter Four:

In Chapter Four, we describe the dataset of images used in the project, highlighting its unique characteristics and shedding light on the challenges faced during its analysis. Additionally, we provide a detailed account of the methodology employed for the implementation of the deep learning model. The step-by-step process followed is presented, encompassing crucial decisions and considerations made along the way.

Moreover, this chapter delves into a thorough examination of the obtained results derived from the implementation of the Mask R-CNN model. By presenting and analyzing these results, we aim to provide a comprehensive understanding of the model's performance and its ability to accurately detect, segment and classify objects within the given dataset.

Chapter Five:

In Chapter Five, the process of determining the size of the trunk area is explained, starting from the mask obtained through the Mask R-CN output. The chapter also presents the results obtained through the utilization of three different machine learning algorithms to estimate the volume of cork.

Chapter Six:

Chapter Six discusses the deployment process of the volume estimation component within the *Floresta Digital* (Digital Forest) project, which aims to integrate calculation simulators for various tree types. The focus is on the development and integration of machine learning models for accurate volume estimations. The chapter highlights the role played in enhancing the functionality and accuracy of the volume estimation component.

Chapter Seven:

The discussion chapter serves as a platform to critically analyze and interpret the findings presented in the preceding research chapters. This section aims to provide a comprehensive understanding of the results, highlight their significance, and offer insights into the broader implications and limitations of the study.

Chapter Eight:

Lastly, Chapter Six summarizes the main findings and contributions of the research. It answers the research questions and provides a final analysis of the data. The following key conclusions were drawn from the study.

2 State of the Art

Some of the studies carried out and the methods proposed aim to facilitate the measurement of the biometric parameters of a tree, using non-destructive methods obtaining quick results. Based on this scope, the following subsections describe previous work based on computer vision methods and on deep learning to estimate biometric parameters of the tree, capable of obtai- ning accurate results. Some traditional methods and the main mathematical formulas used to calculate the volume of a tree are also mentioned below.

2.1 Traditional Methods Used to Estimate Tree Volume

Depending on the degree of detail required in measuring the volume of a tree, it may be necessary to calculate only the volume of the trunk or to include the tree branches. In this project the focus will be only on the trunk volume since it is in this main section of the tree that the cork is extracted.

Currently, there is a significant variety of techniques that are used to measure the volume of a tree. Considering the trunk does not have the same diameter along its height, regardless of technique or method used, it is always necessary to segment the trunk into different sections. The volume is calculated for each section, and the partial volumes of each section will be considered to determine the total volume of the trunk.

Trunk volume can be obtained through non-destructive measurement methods, where it is necessary to measure the diameter along the trunk using the Spiegel-relaskop (4), this professional measuring equipment allows us to carry out the mensuration since the ground level, but this is a very high time-consuming method.

Regarding common measuring instruments, hypsometers are the most used when the objective is to measure the height of a tree. Hypsometers can measure the height of the tree with high precision considering trigonometry and atmospheric pressure. It is possible to estimate the volume of a tree using binary equations based on DBH (diameter at breast height) and the total height of a tree measured by the hypsometer. These binary equations are called hypsometric relations.

The hypsometric relations represent the height-diameter relationship of the tree, using data obtained from a set of trees. Through these data, the relationship between the DBH and the height of the tree is established, and a function is calculated to represent this relationship. To calculate the volume of a tree that can be divided into sections,

there are three widely used formulas, these are Huber's formula, Smalian's formula and Newton's formula. All these formulas are applied to each section of the tree after segmentation using the absolute sectioning method, represented in Figure 2.1.

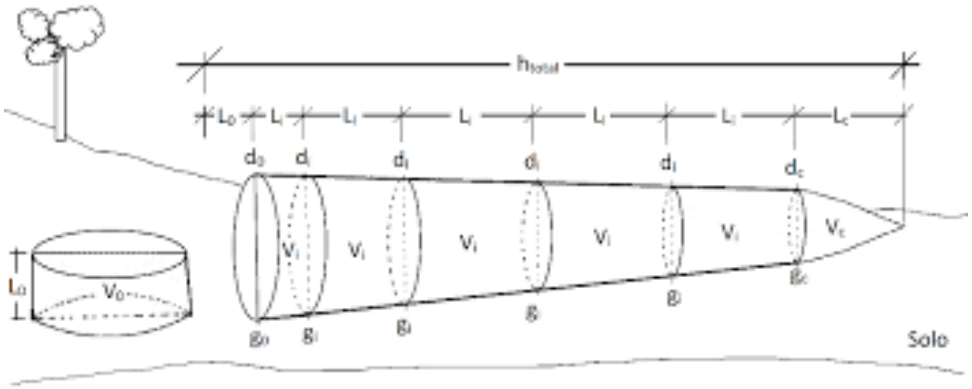

Figure 2.1: Cubing scheme by absolute analytical method.

Smalian's formula [10] states that the volume of a stem section can be estimated by multiplying the average of the cross-sectional area at the lower and upper end of the section by the long of the section. Frequently Smalian's formula, is the easiest to use to calculate the volume of each section. To apply the formula it is necessary that the units used for areas and length are equal, so that the volume is correctly calculated in cubic metres.

The Smalian's formula is presented and described below:

$$v = \frac{L \times (g1 + g2)}{2} \tag{2.1}$$

In which:

v = Trunk volume in cubic metres;

L = Length of the section;

g1 = Cross-sectional area at the lower end of the section;

g2 = Cross-sectional area at the upper end of the section.

Huber's formula [11] is often used, especially in Europe. This formula estimates the volume by multiplying the diameter of the central part of the section by the height of the section.

The Huber's formula is shown below:

$$v = L \times gm \tag{2.2}$$

In which:

v = Trunk volume in cubic metres;

L = Length of the section;

gm = Cross-sectional area at the central part of the section.

Newton's formula [11] is the most complex but gives the most accurate results and is therefore considered the best method. For this formula to be applied, it is necessary to obtain a measurement of the diameter of the lower end of the section, the diameter of the upper end of the section and also the diameter of the central part of the section.

Newton's formula is shown below:

$$v = \frac{L \times (g1 + 4gm + g2)}{6} \tag{2.3}$$

In which:

v = Trunk volume in cubic metres;

g1 = Cross-sectional area at the lower end of the section;

g2 = Cress-sectional area at the upper end of the section;

gm = Cross-sectional area at the central part of the section.

There is also a fourth formula, widely used in the western United States, called the Bruce formula [11]. This formula presents a modification to the Smalian formula, in which a coefficient is added to the calculation of the two ends of the section for a better estimate of the trunk volume

A study by G. de León and L. Uranga-Valencia [12] theoretically evaluated the methods of Huber and Smalian applied to the classical geometry of a tree trunk. The study proved algebraically, that the error of the Huber method is exactly half of the error of the Smalian method.

All three formulae will give an unbiased estimate of the volume of a stem section if the section is cylindrical or shaped as part of what is known as a quadratic paraboloid. Newton's formula will give an unbiased result also if the stem section is shaped as part of a cone (West, 2009).

Smalian's formula is usually the easiest to apply to obtain the section volumes, because the position of the midpoint of each section does not have to be located (Huber formula) (West, 2009).

2.2 Computer Vision Methods

Zhang and Huang [13] present a method of measuring the height of a tree based on image processing. In this method, images of trees were collected, and three red dots were placed on each tree to serve as markers. One of the points was placed at the base of the tree, another one meter from the base and the last one at the maximum possible height of the tree. During image processing, the coordinates of the three marking point were extracted. The photographs were taken perpendicular to the ground, forming a 90-degree angle. In order to extract the coordinates of the top marking point, a model called the HSI colour model was used. This model separates the intensity information from the chromatic information, describing the colours from the human vision point of view and shows advantages over the RGB model when it is used for image segmentation. The authors propose a method in which image segmentation is performed over the three components of the HSI model, managing to segment the tree from all other objects present in the image background. After segmentation, the image is converted to a binary format and the coordinates of the top point are extracted by progressive scanning. The coordinates of the remaining marker points are extracted in a similar way as the coordinates of the top point. Spurious points that compromise the correct extraction of coordinates are removed using mathematical morphology. Finally, the height of the tree is calculated using the similarity theory between triangles. The experimental results indicate that the relative measurement error corresponding to the tree height prediction is about 4%. Therefore, this is a viable method.

The study by D. Han and C.Wang [14] also proposed a method for calculating the height of a tree (7). Overall, this method and the method proposed by Zhang and Huang are similar. The main difference between the two is the fact that this method uses a smartphone to perform target extraction. A marker with two red tips is placed near the tree, parallel to the trunk. Similar to the method presented above, the top point, which represents the maximum height of the tree, and the two tips of the marker are are extracted according to their colour characteristics. The experimental results indicate that the relative measurement error corresponding to the tree height prediction is about 3.6%, slightly below of the method presented above, but still very similar.

B. Putra, N. Ramadhani, D. Soedibyo et al. [15] evaluated the use of optical sensors, including a smartphone camera, which were analysed by an image processing technology in order to estimate tree circumference in homogeneous and production forests, especially rubberwood and albizia plantations, with a real-time measurement approach. The images were captured at a distance of approximately one meter from the tree and with the camera pointed, perpendicular to the ground, to the area of the trunk to calculate the diameter at breast height (DBH). In order to identify the trunk, an HSV-based image segmentation approach was used and the diameter was estimated using computer vision technology. For each measurement, the distance between a tree and the

camera was used as reference. The measurements performed using the camera showed acceptable accuracy with a 95% coefficient of determination and an RMSE of approximately 7.9 cm, which corresponds to a relative measurement error of about 9.4%. Despite the accuracy demonstrated, this method is only applicable to trees with relatively round shapes and there are also several aspects that affect measurement errors by using computer vision, such as the case of the presence of inclined trees, irregular geometric shapes and the computer vision segmentation methods themselves, which are not always the most appropriate.

The study developed by Dianyuan Han [16], also uses image processing to estimate the volume of a tree. Two red coloured marker points were placed on the tree trunk before the photograph was taken. After extraction, both the trunk and the marking points, the edge and the central axis of the trunk were adjusted by constructing a curve so that it represented a better fit to the collected data. The method proposed by this study proved to be viable since it presented relative measurement errors in the order of 5.4%, therefore it is a viable method.

In the study by Coelho et al. [17], the volume of corsican pine trees (Pinus nigra) was estimated using computer vision techniques and classical formulae for volume determination. For the application of this method, a specially designed target was used with the to help in the extraction of the real dimensions of the objects present in an image and also in the calibration of the camera. The target is coloured red, so as to be easily contrasted with the other objects, and consists of a rectangle with four circular holes and four rectangles with a checkerboard pattern. Both the holes present in the target and the target itself have known dimensions. The target is positioned behind the tree. In order to remove the lens distortion and correct the prespective, OpenCV calibration techniques were applied using at least one of the four checkerboard rectangles, calculating the distortion coefficients and rotation vectors. After calibration, the image is converted to HSV and the red target is identified and extracted. After extraction, the target is processed using erosion and dilation methods in order to reduce any noise. The contour of the extracted target is calculated and the values required for the measurement methods previously discussed in this section are obtained. Among the methods developed to estimate the height and volume of a tree, the ones that showed the best results were the method of hypsometric relations, used to calculate height, and Newton's method, used to calculate volume. These methods showed average errors of 12.18% and 10.90%, respectively. Both methods presented, at least, similar error similar when compared to traditional methods.

Summary and analysis of methods results

In order to summarise the results obtained and analyse the main differences of the methods mentioned above, a comparative table is presented.

Table 2.1: Results and main differences.

Method	Relative Error (Height)	Relative Error (Diameter)	Relative Error (Volume)
Zhang and Huang (2009)	4%	Not applicable	Not applicable
D. Han et al. (2012)	3.6%	Not applicable	Not applicable
B. Putra et al. (2021)	Not applicable	9,4%	Not applicable
Dianyuan Han	Not applicable	Not applicable	5.4%
Coelho et al. (2021)	12.18%	Not applicable	10.90%

Regarding the prediction of tree height, by analysing Table 2.1, it is possible to verify that the method proposed by Zhang and Huang and the method of D. Han and C.Wang present the lowest relative errors when compared with the method proposed by J. Coelho et al., which obtained an average error of 12.18%. The difference in the results of these methods can be explained by the image collection conditions. In the case of the first two methods, the images were collected in almost optimal conditions, having been taken at a 90 degree angle, with minimal distortion and from trees representing a not very complex scenario in terms of geometry and surroundings. The last method, on the other hand, although presenting higher errors, is capable of producing results in more complex scenarios, which makes it also a very viable method.

The method proposed by B. Putra et al. focused only on obtaining an estimation of the diameter of a tree and presented a relative error of 9.4%, which makes it a viable method, although the authors estimate that it is very ineffective when applied in more complex scenarios than those presented in the study.

In the case of volume prediction, the methods developed by Dianyuan Han and J. Coelho et al. presented acceptable relative errors although the first method presented about half the error of the second method. The difference in the errors obtained in the two methods can be explained using the same rationale presented in the case of height prediction.

2.3 Deep Learning Method

In the study by Juyal et al. [18] a method was proposed to estimate the diameter and height of a tree using Mask R-CNN neural networks, to calculate biomass, a key indicator of ecological and vegetation management processes. Mask R-CNN are a type of convolutional neural networks used for image segmentation and were used to detect the tree and a reference (white rectangular square held parallel to the tree in the image). As the reference detected by the neural network has known dimensions, it was possible to estimate the circumference and height of the tree and therefore its volume. The volume of a tree is usually calculated by manual measurement made on the ground using measuring equipment where the tree is divided into several segments, the dimensions of each segment are measured and the final volume is calculated. The

method proposed in this paper aims to use deep learning algorithms in order to accelerate this measurement process.

The DBH is a measurement parameter that serves to calculate the cross-sectional area (g) in m^2 of a tree. Basal area of a stand is normally expressed in m^2/ha and provides the degree of occupation of the species in the stand. The tree volume was calculated from the cross-sectional area (g) in m^2, which was then multiplied by the height and by a constant.

About 400 images, including tree instances and reference instances, collected by the Forest Research Institute in Dehradun, were used to train and test the neural network. There is no indication of which tree species were used. The figure 2.2 shows two examples of images used for training the neural networks.

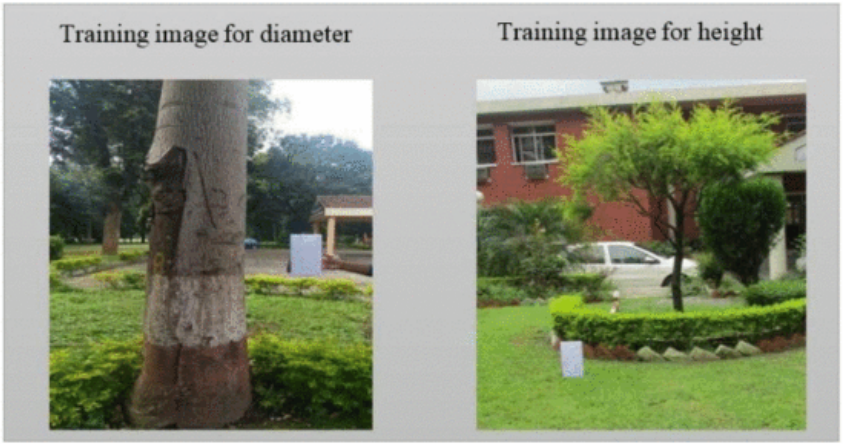

Figure 2.2: Sample images for training.

The use of convolutional neural networks of the Mask R-CNN type allowed not only to detect the object in the image but also to produce a high-quality result regarding the segmentation of the object and, therefore, to obtain a better understanding at the pixel level in order to delimit it with high accuracy. A very important feature of this type of networks is that its backbone is a Feature Pyramid Network (FPN), an artificial neural network (ANN) capable of detecting objects at different scales, which proved to be an essential component for the method proposed in this paper. When the regions of interest are generated, through the application of the RPN, the feature map layer is selected, from the pyramid of feature maps generated by the FPN, with the most adequate scale in order to extract the feature patches.

The proposed method was divided into two stages. The first stage consisted in training a Mask R-CNN to detect the reference and the tree trunk and, using the coordinates returned by the neural network (Xmax and Xmin), it was possible to calculate the trunk diameter. In the second stage, a second ANN was trained to detect the entire tree and the reference, which also returns two coordinates (Ymax and Ymin) to calculate the height of the tree. In both steps a multiplication factor had to be found in order to

be able to calculate both the diameter of the tree and its height. These multiplication factors were calculated from the following formulas:

$$multiplier\_width = \frac{Xmax - Xmin}{WidthOfTheReference} \tag{2.4}$$

$$multiplier\_height = \frac{Ymax - Ymin}{HeightOfTheReference} \tag{2.5}$$

Formula (2.4) corresponds to the multiplication factor used to calculate tree diameter, where Xmax and Xmin are the coordinates returned by the first neural network and width of the reference is the known width of the reference. Formula (2.5) corresponds to the calculation of the multiplication factor used to compute the height of the tree, where Ymax and Ymin correspond to the coordinates returned by the second neural network and height of the reference corresponds to the known height of the reference.

The tree diameter is calculated by multiplying the multiplication factor by the horizontal distance (difference between coordinates Xmax and Xmin). The tree diameter is calculated by multiplying the multiplication factor by the horizontal distance (difference between Xmax and Xmin coordinates). The height of the tree is calculated by multiplying the multiplication factor by the vertical distance (difference between coordinates Ymax and Ymin). After calculating the diameter, it is possible to calculate the cross-sectional area (g) in square meters which corresponds to the following formula:

$$g(m^2) = pi \times \frac{DBH(cm)^2}{40000} \tag{2.6}$$

Subsequently to the calculation of all the variables mentioned above the volume of the tree is calculated, where the cross-sectional area is multiplied by the height of the tree and by a constant that in this case obtained the value of 0.42. Figures 2.3 and 2.4 show examples of diameter and height measurements of a tree using the proposed method, respectively.

Figure 2.3: Diameter measurement.

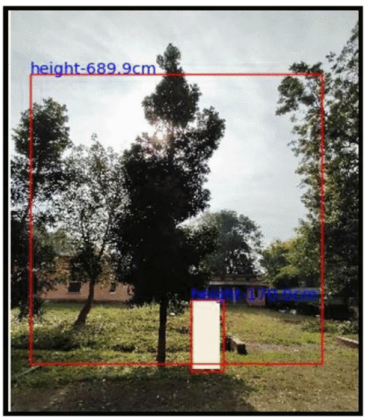

Figure 2.4: Height measurement.

As this is a model that performs object detection and classification tasks, the Average Precision (AP) metric was used to measure the accuracy of the proposed model in both diameter and height prediction. The results show an average accuracy of 0.92 for diameter detection and 0.86 for height detection.

2.4 Tools for Data Annotation

There are some tools available that allow you to annotate images. A comparative study between the different tools was carried out in order to choose the most appropriate one to use in this project.

2.4.1 CVAT

The Computer Vision Annotation Tool (CVAT) [19] is an open source tool supported by Intel. It consists of an online platform that helps with image and video tagging. It supports various types of annotation for object detection, classification and segmentation of images. It offers four formats such as boxes, polygons, poly-lines and points.

With this tool it is possible, through artificial intelligence, to load pre-trained models with the COCO dataset in order to perform semi-automatic annotations of the objects. It is also possible to propagate the annotations to other images by making a copy of the previous ones, which, in a certain way, can help to reduce the time required to perform the following markings.

CVAT has a good usability containing an easy to understand interface that does not require very specialised knowledge.

2.4.2 LabelMe

LabelMe [20] is an online tool used for image annotation tasks. It was created by the Computer Science and Artificial Intelligence Laboratory at the Massachusetts Intitute

of Technology (MIT).

The tool offers a free dataset already annotated with many images and supports 6 annotation types such as polygon, rectangle, circle, line, straight line and point. It is possible to save and export in JSON format.

LabelMe allows to easily import and view annotations and make corrections if necessary. The fact that its interface is offline makes the annotation process very fast.

2.4.3 LabelImg

LabelImg [21] is an image annotation tool, developed in Python, which allows annotation using the delimiting box design. It is possible to export the markings to YOLO and PASCAL VOC formats.

In its base version, this tool only presents the functionality to perform annotation using only rectangle-shaped bounding boxes, which makes it not viable for image segmentation projects.

The tool offers an intuitive and simple interface, which requires almost no learning curve. You can only use this tool if it is installed locally.

MakeSense.AI

The MakeSense.AI tool [22] is free and open source, just like the ones presented above. It does not require any installation since it is a web tool. It supports multiple annotation types, such as lines, lines, points and polygons.

This tool allows extraction of files in YOLO, VOC XML, VGG JSON and CSV formats. It also features automatic annotation functionality using a pre-trained model.

MakeSense.ai, illustrated in figure 2.5, presents a friendly and intuitive interface and offers good zoom control on the image. Since it does not allow project management, the use of this tool is not recommended when you have a large dataset.

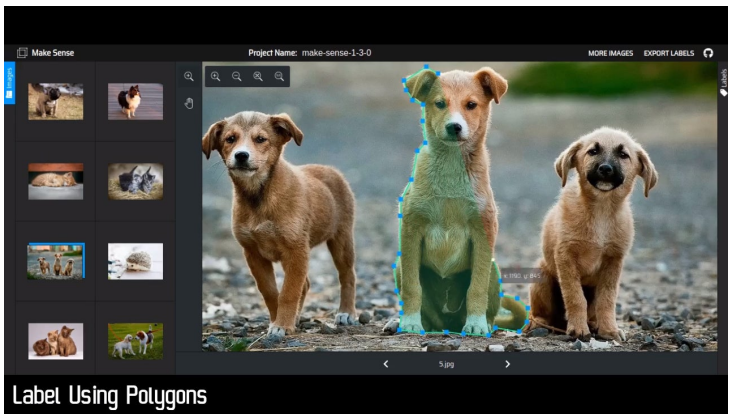

Figure 2.5: MakeSense.ai.[1].

2.4.4 Comparative analysis of annotation tools

After the analysis of each of the tools, the one that was discarded in the initial phase was LabelImg since it only allows the annotation with rectangles, which becomes unfeasible for this project since the goal is to perform image segmentation.

The CVAT tool proved to be quite intuitive, but a little difficult to handle. Moreover, the semi-automatic annotation of the images cannot be applied to the dataset since it is quite different from the COCO dataset with objects not present in that dataset. The functionality to propagate the annotations also did not prove to be effective since the logs present in the dataset have very irregular shapes and with different aspects.

The tools LabelMe and MakeSense.ai proved to be very fast and intuitive and the choice of the tool to be used in this project fell on MakeSense.ai for being less complex and easy to use.

[1]Source: *https://viso.ai/computer-vision/image-annotation/* (last checked on 2023-02-21)

3 Object Classification, Detection and Segmentation using Deep Learning

In this chapter, we will discuss the state-of-the-art techniques for object classification, detection, and segmentation using deep learning and region-based CNNs. We will start by introducing the basic concepts of CNNs and their applications in image recognition. Then, we will delve into the specifics of region-based CNNs, which are designed to efficiently handle object detection and segmentation tasks by dividing the input image into regions of interest.

We will also discuss the popular object detection framework called Region-based CNN (R-CNN) and its variants, such as Fast R-CNN and Faster R-CNN. These frameworks use region proposal algorithms to identify the potential object regions in an image and then use a CNN to classify and localize these objects.

Finally, we will explore the topic of object segmentation, which aims to partition an image into regions corresponding to different objects. We will discuss popular segmentation techniques, such as Fully Convolutional Networks (FCNs) and Mask R-CNN, which combine the advantages of region-based CNNs and pixel-level segmentation.

Overall, this chapter provides a comprehensive overview of deep learning-based techniques for object classification, detection, and segmentation using region-based CNNs.

3.1 Deep Learning

Deep learning has revolutionized the field of computer vision, enabling machines to recognize and interpret visual data with unparalleled accuracy. Through the use of neural networks with multiple layers, deep learning algorithms can automatically learn and extract intricate patterns and features from images, leading to significant advancements in object classification, detection, and segmentation tasks.

Both machine learning and deep learning aim to enable computers to learn and make predictions but deep learning represents a more advanced and complex approach, leveraging deep neural networks to automatically learn hierarchical representations from raw data. Deep learning excels in handling large-scale, unstructured data, but requires more computational resources and careful tuning to achieve optimal performance. Machine learning, on the other hand, offers a broader range of algorithms and can be more easily applied to smaller datasets or problems with simpler patterns.

In recent years, deep learning methods have emerged as the state-of-the-art approach for object classification, detection, and segmentation [23]. These methods leverage the power of convolutional neural networks (CNNs), which are specifically designed to process visual data efficiently [24]. CNNs consist of multiple layers of interconnected neurons that perform operations such as convolution, pooling, and non-linear activation, allowing them to learn hierarchical representations of visual features [25].

One of the key advantages of deep learning approaches in computer vision is their ability to learn high-level features from large-scale datasets. By training on extensive image collections, deep learning models can capture a wide range of visual patterns and variations, enabling them to generalize well to unseen data [26]. This capability has significantly improved the accuracy and robustness of object recognition systems, making them capable of accurately identifying and categorizing objects in complex scenes [27].

The recent advancements in deep learning, particularly in CNNs, have significantly propelled the capabilities of computer vision systems, allowing for more accurate and efficient object recognition, detection, and segmentation. As deep learning continues to evolve, it holds immense potential for further advancements in computer vision and the wide range of applications it encompasses.

3.2 Convolutional Neural Networks

A Convolutional Neural Network (CNN) is a Deep Learning model with the ability to receive an input image, assign an importance (learning weights and biases) to various features/objects of the image and then able to differentiate them from each other. The pre-processing required in the deep neural network ConvNet is much less compared to other classification algorithms. While in primitive methods, like ANNs, filters are handmade, requiring a lot of prior knowledge, CNNs have the ability to autonomously learn these characteristics [28].

Typically, a CNN is composed by three types of layers: convolutional layers, pooling layers and fully connected layers, as it is possible to visualize in figure 3.1.

The convolutional layers are the basis of this type of neural networks and their goal is to produce a two-dimensional representation of the image, also known as feature map. This process allows the CNN to capture local patterns and spatial dependencies within the data. Convolutional layers in CNNs leverage this operation to detect various features, such as edges, textures, and shapes, by convolving filters over the input and producing feature maps. These feature maps are then passed through activation functions to introduce non-linearity and capture higher-level representations. By repeatedly ap-

[1]Source: *https://www.mdpi.com/2073-8994/14/7/1325* (last checked on 2023-02-21)

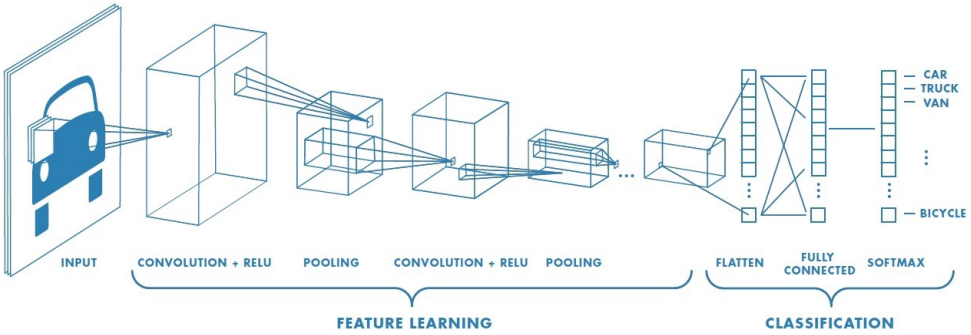

Figure 3.1: CNN Architecture.[1].

plying convolutional layers, CNNs can learn increasingly complex features, enabling them to perform tasks such as image classification, object detection, and semantic segmentation. A feature map is produced by applying filters, called kernels, whose goal is to sweep the image along its length and width in order to generate the scalar product between two matrices. One of the matrices is the kernel itself and the other is the matrix representing the region corresponding to the position of the kernel. Each kernel is used to extract a feature of the image so that it is possible to obtain patterns that can lead to a correct classification of the image. In a convolutional layer the model uses different types of filters of different sizes in order to produce different feature maps. Unlike classical ANNs, the network is able to learn correlations between neighbouring pixels of an image, which makes it invariant to translation.

The pooling layers reduce the dimension of the images by reducing the number of pixels coming from the output of the previous layer. Usually, in this kind of layers, an operation called Max Pooling is performed in order to calculate the maximum value of the weights of the feature map resulting from the application of a filter. This operation results in the so called downsampling of the feature map that helps to decrease the size of the input of the next layer resulting in a decrease in the number of weights that, in turn, results in a decrease in the time required to perform a prediction. In particular, Max Polling is good for eliminating noise and letting the most salient features through. Figure 3.2 illustrates the result of the Max Pooling operation.

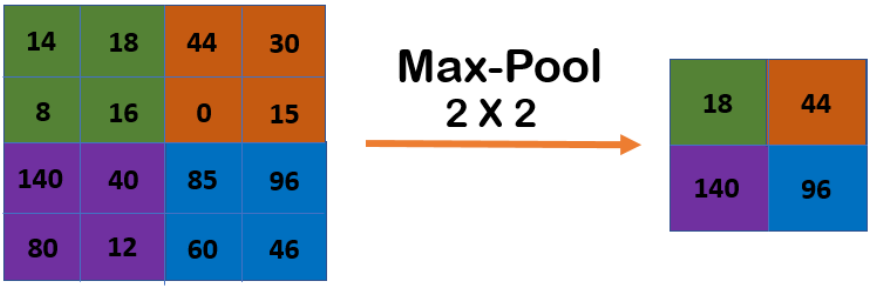

Figure 3.2: Max Pooling Operation.

In addition to Max Pooling, another commonly used operation in pooling layers is Average Pooling. While Max Pooling calculates the maximum value within a specific region of the feature map, Average Pooling computes the average value within that region.

Similar to Max Pooling, Average Pooling, illustrated in Figure 3.3, helps reduce the dimensionality of the input data and downsample the feature map. It divides the input into non-overlapping regions and replaces each region with the average value of the pixels or features within that region. By taking the average instead of the maximum, Average Pooling preserves some information about the overall distribution of values in the region.

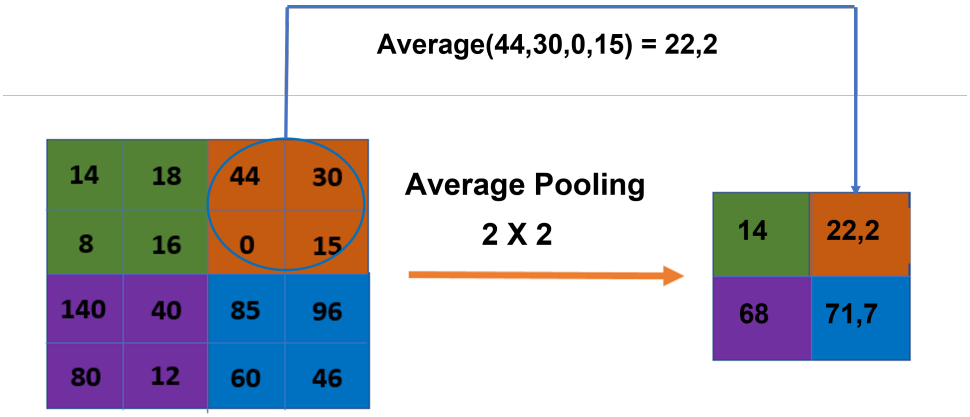

Figure 3.3: Average Polling Operation.

The downsampling achieved through pooling operations such as Max Pooling and Average Pooling brings several benefits. Firstly, it reduces the spatial dimensionality of the feature map, effectively decreasing the number of pixels or features. This reduction simplifies the subsequent layers' computations and reduces the number of parameters required, which can lead to faster training and inference times.

Furthermore, pooling layers help to capture the most salient features in the input data by selecting the maximum or average values. By retaining the most important information while discarding less significant details, pooling contributes to robust feature extraction and can enhance the model's ability to generalize to new data.

3.2.1 Region-based Convolutional Neural Network for Object Detection

Object detection is a computer vision technique that plays the role of detecting instances of objects of a given class in an image or in a video. Making use of this technique it is possible not only to detect and locate objects but also obtain a classification of them.

A standard CNN, where fully connected layers are used after the feature engineering layers, cannot be applied in this kind of problems due to the fact that the network out-

put is variable. A possible approach to solve this type of problem would be to divide the image into different regions of interest and use a CNN to detect the presence of the object within each region. The problem with this type of approach is that the objects that are intended to be identified may be in different locations and have different aspects, that is, to correctly identify the objects in the image, a large number of regions of interest would have to be selected, which would require a high and not viable processing power.

3.2.2 R-CNN

In order to overcome the problem of having to identify a large number of regions of interest, and with the aim of identifying objects more quickly and reducing the necessary processing capacity, several neural network architectures for object detection have been proposed. One of the proposed networks is called Region-based Convolutional Neural Network (R-CNN) and was proposed by Ross Girshick *et al.* [29]. This neural network uses an algorithm called selective search [30]. Selective search makes it possible that, instead of the network trying to classify a huge amount of regions of interest, it starts to classify only 2000 regions. Selective search receives several candidate regions and uses the greedy algorithm, which is based on the principle of similarity, in order to combine several similar regions of interest (in terms of colour, texture and size) into a larger region of interest. When all iterations of the algorithm are completed, the number of regions of interest will be substantially smaller, only remaining the regions of interest called candidate regions. After the candidate regions are identified, they are condensed into squares. These regions will serve as input to the CNN. The feature engineering layers extract the features of each region and the dense layers produce a feature vector that is used as input to the Support Vector Machine (SVM) algorithm, where the presence of the object and the coordinates of the bounding box are detected. Figure 3.4 shows the architecture of the R-CNN neural network.

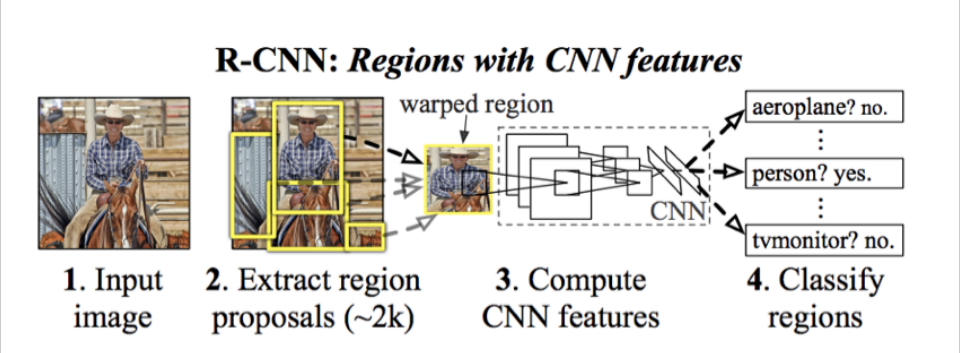

Figure 3.4: R-CNN Architecture.[2].

[2]Source: *https://datascience.eu/pt/visao-computacional/r-cnn-fast-r-cnn-faster-r-cnn-yolo-algoritmos-de-deteccao-de-objetos/* (last checked on 2023-02-21)

Although R-CNN has partially solved some of the issues mentioned above, there are still some problems that this neural network was not able to fully mitigate. In fact, although it has been beneficial to reduce the number of regions of interest to be classified, it still takes a lot of training time to be able to classify the 2000 regions. Another negative factor is the amount of time needed to test an image, which makes it unfeasible to implement for real-time response. Another disadvantage also identified in this type of networks is that the selective search algorithm, which selects candidate regions, does not have the ability to learn, which results in an incorrect identification of some candidate regions.

3.2.3 Fast R-CNN

These disadvantages led to the proposal of a new neural network, based on the previous R-CNN and named Fast R-CNN [31]. Fast R-CNN follows a similar approach to the R-CNN, having the advantage of being faster in detecting objects than the previous proposed network. Instead of offering as input to CNN the 2000 candidate regions, CNN starts receiving the complete image and a set of candidate regions. The same approach of R-CNN is used to identify the candidate regions. The CNN receives the image and, through its convolution and max pooling layers, generates a feature map. Afterwards, a projection of the candidate regions is performed on the resulting feature map using the Region of Interest (ROI) projection. The ROI projection identifies the coordinates of candidate regions in the feature map corresponding to their position in the original image through their sub sampling ratio. Since CNN dense layers do not accept different feature vectors, a layer called ROI pooling is used to transform, for each candidate region, the features vector into a fixed size. Subsequently, each vector serves as input to the CNN dense layers and outputs are generated. The output of the object class is originated by the softmax function and the coordinates of the bounding box are also obtained. Figure 3.5 shows the architecture of the Fast R-CNN network.

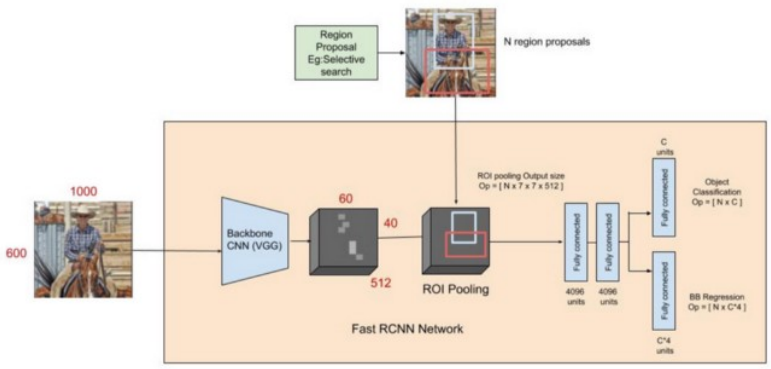

Figure 3.5: Fast R-CNN Architecture.[3].

3.2.4 Faster R-CNN

As time progressed, Gavrilescu *et al.* [32] proposed another network called Faster R-CNN that, according to the published article, reduced the prediction time from 2 seconds to 10 milliseconds and also managed to obtain better results in object detection. This network used the architecture of Fast R-CNN, modifying only the phase of obtaining the candidate regions, where the selective search algorithm was eliminated. Instead of using selective search, which made the processing a little time consuming, it uses a separate network, whose objective is to generate the candidate regions. This network is called Region Proposal Network (RPN).

The architecture of Faster R-CNN can be divided into two parts, as shown in Figure 3.6. The first part consists in applying an RPN in order to generate the candidate regions. The second part uses the previously proposed Fast R-CNN architecture to detect the objects in the regions proposed by the RPN.

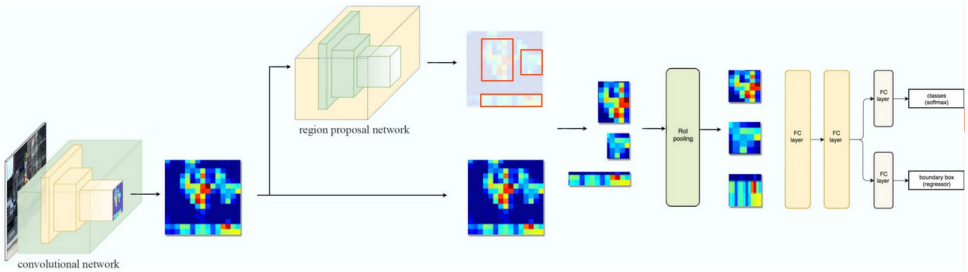

Figure 3.6: Faster R-CNN Architecture.[4].

The R-CNN and Fast R-CNN architectures use as dependency the Selective Search algorithm to generate the candidate regions, regions that serve as input to a pre-trained CNN to produce the classification. The Faster R-CNN introduces the RPN and in this way manages to withdraw some advantages in comparison to the previously proposed models. One of the main advantages comes from the fact that, by using the RPN, it is possible to train it for a specific detection task. In other words, the network will be able to produce better candidate regions since it will be specialized according to the detection objective. The RPN receives as input the feature map resulting from the last convolutional layer of the CNN used for feature extraction (e.g. VGG-16) to produce the candidate regions. The same feature map is shared with the Fast R-CNN.

According to Figure 3.7, a sliding window $n \times n$, with n corresponding to the number of pixels, scans the entire feature map and for each position of the window k (number of anchors for each position of feature map) candidate regions are produced. Each candidate candidate region is parameterized according to a reference box, and here

[3]Source: *https://sumeet-sewate.medium.com/introduction-to-object-detection-and-evolution-rcnn-fast-rcnn-faster-rcnn-yolo-ea409b2c05e2* (last checked on 2023-02-21)

[4]Source: *https://jonathan-hui.medium.com/image-segmentation-with-mask-r-cnn-ebe6d793272* (last checked on 2023-02-21)

the authors introduced the concept called anchor. Each anchor has two parameters: scale and aspect ratio. A candidate region generates k regions that vary in terms of aspect ratio and scale. Generally, 9 regions since 3 scales and 3 aspect ratios are used.

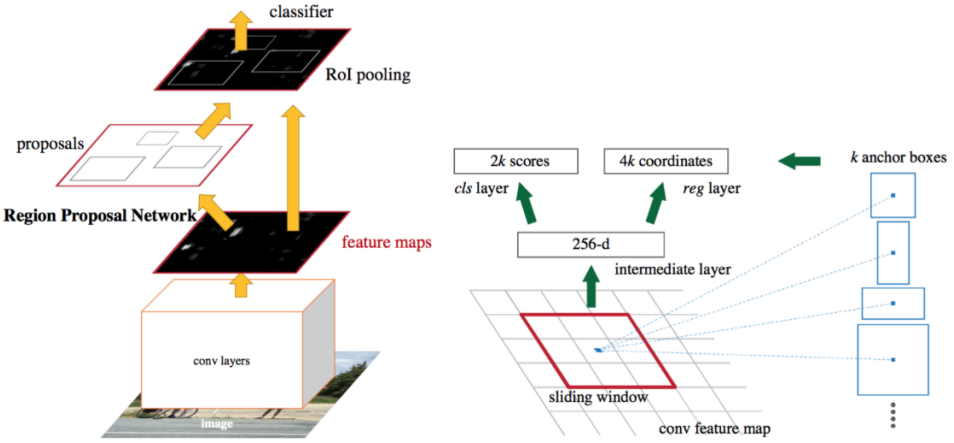

Figure 3.7: RPN Architecture.[5].

The fact that anchors with different scales are provided avoids the use of multiple images or filters, as only one image at one scale is needed.

For each candidate region, a vector is generated and will serve as input to two fully connected layers. The first layer produces a vector of two elements and is responsible for classifying the region. The softmax function will return the probability of the region containing an object or not and the region will be classified as background, if the network predicts that the region does not contain any object or, otherwise, as foreground. The second layer produces a vector of four elements and is responsible for returning the coordinates of the bounding box of that region.

Due to the fact that Faster R-CNN performs thousands of predictions, the target object detection is done by removing the boxes that belong to the backgound class and the rest are filtered according to the confidence score. When two bounding boxes intersect, the metric of the Interception Over Union (IoU) is calculated.

The IoU calculates the interception over union of two bounding boxes, the bounding box predicted by the algorithm with the actual box of the object. An IoU of 1 means that the predicted bounding box perfectly overlaps the real box. In order to detect the object only once in the image, the Non Maximum Suppression (NMS) technique is used to remove all the bounding boxes where IoU is less or equal to 0.7.

The RPN shares candidate regions with Fast R-CNN. Next, a projection of the candidate regions is performed on the feature map that was shared between the two architectures, using the ROI projection. The rest of the process is explained above since it represents the whole Fast R-CNN process.

[5]Source: *https://www.researchgate.net/figure/An-illustration-of-the-Faster-R-CNN-detector-One-network-with-four-losses-The-first_fig5_322262996* (last checked on 2023-02-21)

3.2.5 Mask R-CNN

Mask R-CNN is a powerful architecture that combines object detection with semantic segmentation, allowing for accurate instance-level recognition and pixel-wise mask generation in images. In the study published by Kaiming He*et al.* [33], the authors provide a detailed description of the Mask R-CNN architecture.

The Mask R-CNN architecture, illustrated by Figure 3.8, builds upon the Faster R-CNN model, which is widely used for object detection. Faster R-CNN consists of two main components: a region proposal network (RPN) and a region of interest (ROI) pooling layer. The RPN generates a set of candidate object proposals, while the ROI pooling layer extracts fixed-size feature maps for each proposal. These feature maps are then used for classification and bounding box regression.

In Mask R-CNN, an additional branch is introduced alongside the existing classification and bounding box regression branches in the Faster R-CNN architecture. This new branch, called the mask branch, is responsible for predicting a binary mask for each region of interest. The mask branch is implemented as a fully convolutional network (FCN) with several convolutional layers. This FCN takes the features from the ROI pooling layer and produces a pixel-wise mask for each proposed region.

To obtain accurate pixel-level alignment between the original image and the extracted features, the authors address a misalignment issue in the Faster R-CNN architecture. The ROI pooling layer in Faster R-CNN uses quantization to map the continuous coordinates of the ROI to discrete feature map locations. However, this quantization can result in misaligned regions between the feature map and the original image. To overcome this problem, the authors introduce ROI Align, a technique that performs bilinear interpolation instead of quantization. This ensures that the regions of interest align accurately with the corresponding features in the feature map.

The mask branch in Mask R-CNN produces a binary mask with the same spatial resolution as the input ROI. Each pixel in the mask represents the probability of that pixel belonging to the object instance within the ROI. The mask branch is trained using a pixel-wise binary cross-entropy loss, comparing the predicted masks with ground truth masks.

[6]Source: *https://towardsdatascience.com/computer-vision-instance-segmentation-with-mask-r-cnn-7983502fcad1* (last checked on 2023-05-30)

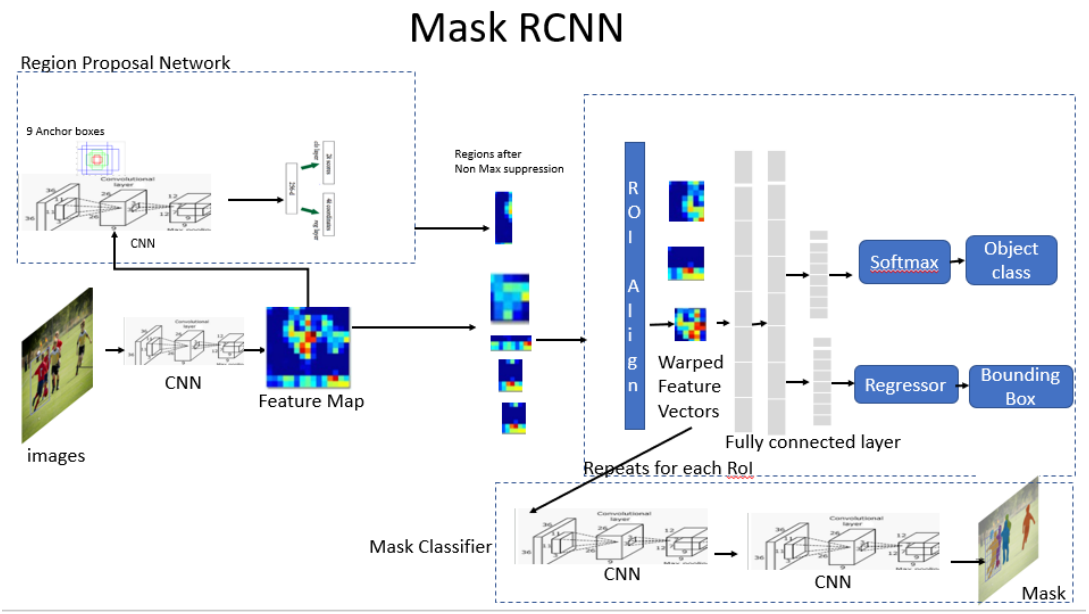

Figure 3.8: Mask R-CNN Architecture.[6].

3.2.6 Comparative Study Between Different Types of Neural Networks for Segmentation

In this subsection, we will compare diferent neural networks for the specific task of segmentation, and explain why Mask R-CNN was chosen.

The goal of this work is to segment the trunks of cork oak trees from the background and accurately measure their area and volume. Trunk segmentation is a challenging task because tree trunks can have complex textures, irregular shapes, and occlusions from other branches or leaves.

U-Net

U-Net is a convolutional neural network (CNN) architecture designed for image segmentation tasks. It was proposed by Olaf Ronneberger, Philipp Fischer, and Thomas Brox in 2015 and has since become one of the most widely used architectures for medical image segmentation [34].

The U-Net architecture is unique in its shape, resembling the letter U, with a contracting path followed by an expansive path. The contracting path consists of convolutional and max-pooling layers, while the expansive path consists of convolutional and upsampling layers. Skip connections are added between the corresponding layers in the contracting and expansive paths to preserve spatial information.

The U-Net architecture has shown superior performance in a variety of image segmentation tasks, especially in medical image segmentation. Its success is attributed to its ability to capture both local and global features and to leverage the contextual infor-

mation from the skip connections [35].

Although U-Net is a popular architecture for medical image segmentation, it may not be well-suited for trunk segmentation because it is designed for small and fine-grained structures. Tree trunks can vary in size, shape, and texture, which may require a more flexible model to accurately segment them.

FCN

FCN (Fully Convolutional Network) [36] is a type of neural network architecture that was developed for the task of semantic segmentation, which involves predicting a pixel-wise classification of an image. Unlike traditional convolutional neural networks (CNNs), which are designed for image classification tasks, FCNs use only convolutional and pooling layers to produce an output that is the same size as the input image. This allows them to perform dense predictions, which are essential for semantic segmentation.

The architecture of FCNs typically consists of two parts: an encoder network and a decoder network. The encoder network is a series of convolutional and pooling layers that downsample the input image to a low-resolution feature map, while the decoder network is a series of deconvolutional and upsampling layers that produce an output segmentation map that is the same size as the input image.

FCNs, like one of its most popular variant (U-Net), have been successfully applied to a wide range of semantic segmentation tasks, including medical image segmentation, object detection, and scene parsing. They have also been used in combination with other neural network architectures, such as CNNs and recurrent neural networks (RNNs), to improve the accuracy of image and video analysis tasks.

Like U-Net, FCNs are not suitable for trunk mask segmentation since it does not handle occlusions or overlapping objects well. Moreover, FCN's are not designed for object detection tasks, which is crucial for segmenting individual trees accurately.

Deep Lab

DeepLab is a deep learning-based algorithm for semantic image segmentation, which aims to label each pixel in an image with its corresponding semantic class. It was first proposed in 2014 by Chen *et al.* [37] and has since undergone several revisions and updates.

DeepLab is based on the use of fully convolutional neural networks (FCNs) which are trained to predict pixel-wise class labels. The original version of DeepLab employed a modified VGG-16 architecture to extract features from input images, followed by atrous (also known as dilated) convolutions to expand the receptive field of the network and capture more contextual information. The output of the network is a probability map

for each class, which is filtered at a threshold to obtain the final segmentation.

Subsequent versions of DeepLab have introduced various improvements, such as the use of the ResNet architecture for feature extraction, multi-scale inputs, and the incorporation of an additional module called the Atrous Spatial Pyramid Pooling (ASPP) module to capture context at multiple scales [38].

DeepLab has achieved state-of-the-art performance on several benchmarks for semantic segmentation, including the PASCAL VOC and COCO datasets. It has also been applied to various applications, such as medical image segmentation and autonomous driving.

DeepLab is a powerful algorithm that uses atrous convolution to capture multi-scale contextual information. While it can handle complex textures and structures, it may struggle with occlusions and irregular shapes of tree trunks.

Comparison with Mask R-CNN

Mask R-CNN is a multi-stage approach that first generates a region proposal and then generates a binary mask for each detected object. This makes it well-suited for trunk segmentation because it can accurately detect and segment tree trunks, even in the presence of occlusions from branches or leaves.

Several studies have compared Mask R-CNN to other segmentation techniques for trunk segmentation and found that Mask R-CNN outperforms them in terms of accuracy and speed. For example, in a study by Zhang *et al.* [39], Mask R-CNN achieved an average intersection-over-union (IoU) of 0.963, compared to 0.895 for U-Net and 0.856 for FCN, when segmenting tree trunks from images captured in a forest. Mask R-CNN also achieved faster inference times than U-Net and FCN.

In another study by Liu *et al.* [40], Mask R-CNN achieved an average IoU of 0.972 when segmenting tree trunks from images captured in a walnut orchard, compared to 0.956 for DeepLab and 0.907 for U-Net. The authors concluded that Mask R-CNN is better suited for trunk segmentation because of its superior accuracy and speed.

In conclusion, Mask R-CNN is a superior segmentation technique for trunk segmentation compared to other methods, as summarised by Table 3.1. It can accurately detect and segment tree trunks, even in the presence of occlusion by branches or leaves, and it achieves faster inference times than other techniques.

Table 3.1: IOU comparation between Mask RCNN and other architectures.

Studies	**U-Net IOU**	**FCN IOU**	**DeepLab IOU**	**Mask R-CNN IOU**
Zhang et al. (2021)	0.895	0.856	Not applicable	0.963
Liu et al. (2020)	0.907	Not applicable	0.956	0.972

4 Trunk Detection and Segmentation Methodology

This chapter discusses the methodology used in the first phase of this project. This stage consists of the detection and segmentation of the trunk of a cork oak tree present in an image, as well as the respective targets that were previously attached to it.

In order to describe the whole implementation flow, this chapter is divided into several sections referring to the different steps carried out during this phase of the project.

In Section 4.1 is described the hardware used in the implementation, training and testing of the deep learning models and the packages used in the development of the models. In section 4.2 the description of the dataset of images is presented, as well as the data collection procedure. This section also discusses the data annotation process and methodology. The data augmentation techniques used in this phase of the project and the evaluation method used to assess the models in the different experiments are also described. The last two subsections of this section present the variations of the diferent experiments, as well as the results obtained and the conclusions.

4.1 Setup

This project was developed using an MSI GL75 Leopard laptop powered by an Nvidia Geforce RTX 2060 GPU with 6GB of dedicated memory and an Intel Core i7 10th generation CPU. Windows 11 was used as the operating system.

In order to simplify package management, a python virtual environment was created using the Anaconda platform. The following packages were used in the development and inference of the model based on the Matterport's implementation[1] of Mask R-CNN:

- numpy (v 1.16.6). Used for efficient numerical computations and array manipulation.
- scipy (v 1.5.3). Numpy arrays, which are commonly used in Mask R-CNN, were directly utilized within Scipy functions, enabling efficient data processing and manipulation.
- pillow (v 9.0.1). Pillow seamlessly integrates with Numpy and Scipy, allowing for easy conversion between Pillow images and Numpy arrays.
- cython (v 0.29.28). Cython library was used in Mask R-CNN implementation to optimize and accelerate the code and integrate with low-level libraries.

[1] `https://github.com/matterport/Mask_RCNN` (last checked on 2023-03-14)

- matplotlib (v 2.2.2). This library was used for visualization of training and evaluation results.
- scikit-image (v 0.14.2). Scikit-image was used for image preprocessing.
- tensorflow (v 1.15.0). TensorFlow enabled efficient model training, inference, and deployment due to its powerful deep learning capabilities and extensive support for neural network training and inference.
- keras (v 2.3.1). Keras simplified the implementation and experimentation process of Mask R-CNN. Keras seamlessly integrates with TensorFlow as its default backend.
- opencv-python (v 4.5.1). Used for image loading and manipulation.
- h5py (v 2.10.0). H5py simplified the handling and management of the dataset in the HDF5 format.
- imgaug (v 0.4.0). This library was used in Mask R-CNN implementation for augmenting and diversifying training data.
- ipython (v 0.33.0). Used to facilitate code exploration and debugging.

Matterports implementation doesn't run with new versions of tensorflow, this is why version 1 of tensorflow was installed. This implementation uses keras high level neural network API to perform some steps of the workflow, like hiperparmeter training and data management.

4.2 Methodology

This section describes the construction of the dataset, including data augmentation methodologies and metrics used to evaluate the deep learning model.

In the next subsections we describe the steps followed in the creation and implementation of the Mask R-CNN model that best suited the detection and segmentation of cork oak trunks and their respective targets.

4.2.1 Data Collection

The dataset, which has been made available to the public at [`https://www.kaggle.com/datasets/andreguim/cork-oak-segmentation` , accessed on 22 May 2023], consists of images of cork oaks obtained before the trees undergo the stripping process. Due to the non-existence of public datasets related to the detection and segmentation of cork oak trunks, a dataset was created in partnership with the Coimbra Agriculture School (ESAC).

On each tree trunk, two or three targets were affixed, depending on the height of the

tree. The targets are detected by the neural network, and then used in the calculation of the segmented trunk area. A tripod was used to fix the camera so that the photographs were taken with the optical axis parallel to the ground. A preliminary dataset had 55 images. It was with this dataset that the first experiments were carried out. Later it was extended to 62 images after more pictures were taken in the field and more experiments were subsequently carried out. Figure 4.1 shows an image of a cork oak with three targets.

Figure 4.1: Sample of a dataset image.

In the creation of the datasets, several steps were considered. The first step was collecting images on different days, so that the surrounding environment underwent the normal changes in terms of sunlight, climate and vegetation. The second step was the annotations that were made in all images and, finally, data augmentation was addressed to increase the dataset size during train. Moreover, the dataset is composed of trees from two herds.

This dataset includes images of several types of trunks characteristic of trees of this species. Namely, there are straighter trunks, more curved trunks, as well as trunks with or without bifurcations. Despite having few instances, the dataset continues to be satisfactory for training models based on the Mask R-CNN network, since this network frequently allows obtaining satisfactory results even in small datasets, as in this circumstance. Figure 4.2 shows examples of images present in the datasets.

Figure 4.2: Set of different images from the datasets.

The first dataset was randomly divided into 80% for training, 10% for validation and 10% for testing. That resulted in 43 images belonging to the training set, 7 to the validation set and 5 to the test set. The second dataset was also randomly divided into 80% for training, 11% for validation and 9% for testing. That resulted in 50 images belonging to the training set, 7 to the validation set and 5 to the test set.

To minimise the problem of working with a small dataset, data augmentation techniques were used. These are described in the following sections.

4.2.2 Trunk Targets

As mentioned in subsection 4.2.1, targets affixed to the trunk are also detected by the neural network.

Checkerboard-patterned targets, also known as calibration targets, consist of alternating black and white squares arranged in a regular grid pattern. These targets are commonly used in computer vision tasks to facilitate camera calibration and 3D reconstruction [41]. By analyzing the deformation and distortion of the squares in the captured images, accurate camera calibration parameters can be obtained, enabling precise measurements and object detection.

When estimating the area of a trunk using Mask R-CNN, the integration of checkerboard-patterned targets can enhance the accuracy and reliability of the results. By placing the

target adjacent to the trunk during image capture, the calibration information extracted from the target can be used to correct any lens distortion, image scaling, or perspective effects that could affect the accuracy of the mask and subsequent area calculation [42].

Furthermore, the checkerboard pattern itself offers a grid-based reference system that can assist in scale estimation. By analyzing the distortion of the squares within the target, the actual size of the trunk can be determined accurately, compensating for any perspective distortion that may occur during image capture [41].

Additionally, if the measurements of the traget are known, it can be used to scale the mask produced by Mask R-CNN to match the actual size of the trunk. This scaling factor ensures that the area calculation derived from the masks corresponds accurately to the true area of the trunk.

In the section referring to volume calculation, the calculation of the trunk area using the target masks will be described in detail.

4.2.3 Data Annotation

Prior to training the model, the images had to be annotated so that the model could have a high-level interpretation of the objects that it is intended to segment and classify.

All images in the dataset were manually annotated using the MakeSense.ai tool, and no automatic or semi-automatic annotation method was used. The annotations are represented in COCO JSON format and correspond to the objects that are intended to classify and segment, in this case the trunk of the cork oak and respective targets.

In each image, only one trunk and its corresponding targets have been annotated, and because each image focuses on a single trunk, the model is only intended to identify the trunk that is focused on in the image. That is, the model should ignore the trunks further away from the camera lens and classify and segment only the nearest trunk and its respective targets.

As mentioned in Subsection 4.2.1, some images were discarded from the dataset, as they were not suitable for training. These images were also not annotated.

The annotation process followed the same criteria for all images, as it is necessary for the model to achieve good performance that the annotations have a high degree of consistency. If the annotations do not present a uniform pattern, the model may have more difficulty in learning the object's patterns and, in turn, correctly segmenting the object.

Some research works argue that the objects that are intended to be segmented should be annotated when these are hidden by other elements of the image, as is the case of the work developed by Golnaz Ghiasi *et al.* [43] These objects are called occluded objects. In the case of this project in some images the targets slightly cover the tree

trunk. However, since the targets belong to a class of objects that is intended to be segmented, it was understood that it might be better for the model if the trunk was annotated without including any part of its targets being included. Figure 4.3 illustrates an example of this annotation criterion.

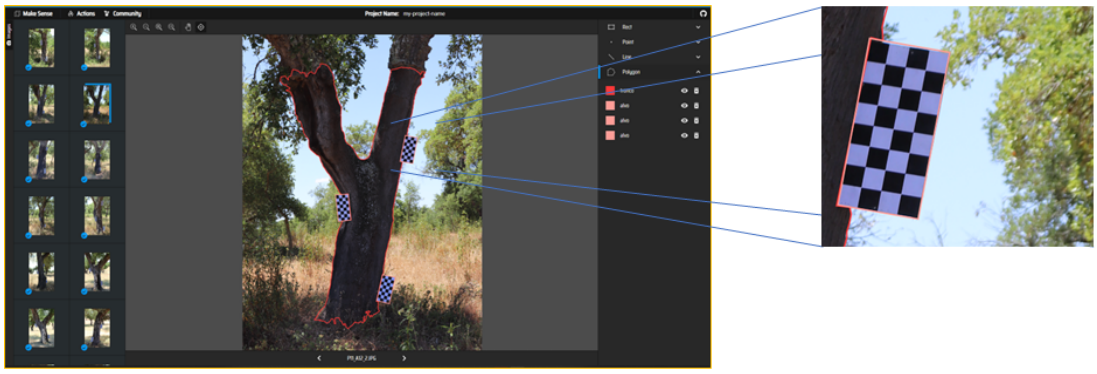

Figure 4.3: Example of a tree with one of the targets slightly overlapping the trunk.

The trunk of a cork oak is not stripped entirely, there is a region from which cork is effectively extracted. It is not intended that the model returns a mask of the complete trunk of the tree, only of the mentioned region. The annotation process followed the same methodology. After the cork extraction region is identified, which is normally easy to identify because the trunk has a different colour and texture, it is annotated ignoring the remaining trunk so that the model can return only the mask of the desired region, as illustrated in Figure 4.4.

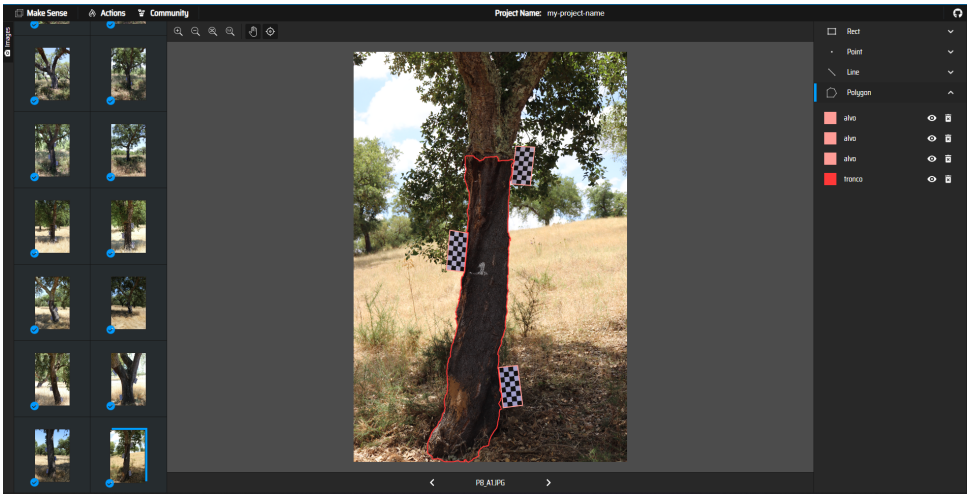

Figure 4.4: Annotated section of the trunk (only the cork extraction region).

In the case of more complex trunks (with bifurcations) or when there were trees very close to the target tree, the annotation criteria were maintained. In the case of bifurcated trunks, all bifurcations were considered as long as they belonged to the cork extraction area, as it is possible to visualise in Figure 4.5.

In case of images with very close trees, which may cause some confusion in the model

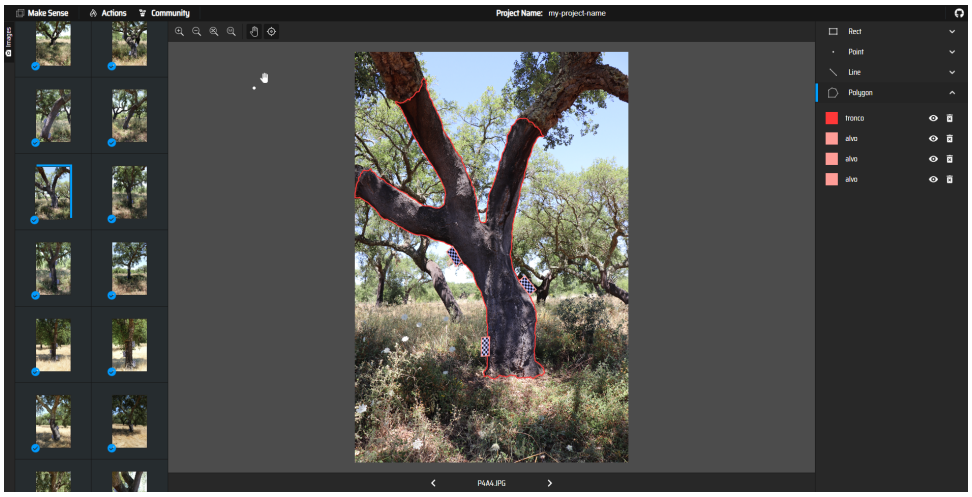

Figure 4.5: Example of complex shaped cork oak (with 3 bifurcations).

classification and segmentation process, it was still decided to annotate only the tree closest to the camera lens, as it is possible to visualise in Figure 4.6.

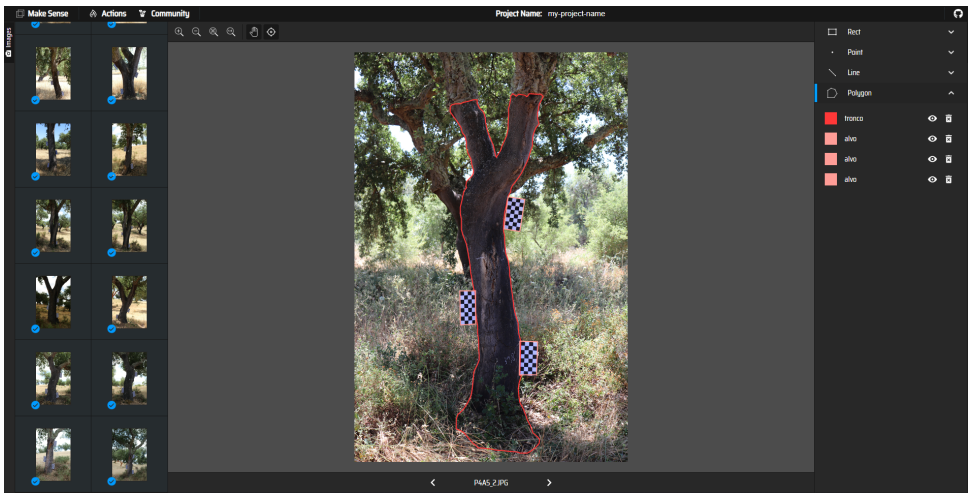

Figure 4.6: Example of an image with two trees in close proximity.

When the annotation process was finished, the annotation_train.json, annotation_val.json and annotation_test.json files were extracted, which correspond to the annotations of the training, validation and test images, respectively.

4.2.4 Data Augmentation

As mentioned before, object detection algorithms based on deep learning require a large amount of data to perform properly. However, the dataset used in this project has a small size. To minimize this problem, data augmentation techniques were used, to increase the number of training instances.

The data augmentation process generates new instances from the original data, using transformation methods such as rotation, translation and resizing. Applying these

techniques minimises the problem of overfitting, which consists of memorising the patterns present in the data, causing the model to fail to generalise its learning and perform poorly in scenarios with unseen data.

A set of several augmentation operations were used. Table 4.1 describes the four techniques used in this project.

Table 4.1: Techniques for data augmentation

Augmentation Technique	**Description**	**Reason for choice**
Image Rotation	Rotation of the image between -10º and 10º	Makes the dataset more accurate in terms of terrain irregularities or trunk inclination
Horizontal Flip (image mirroring)	Mirror input images horizontaly	Since performing the vertical flip is counternatural, it was decided to perform the horizontal flip in order to increase the original dataset.
Translation	Translation of 10% on x axis	More possibility for generalization for the model
Brightness Adjustment	Add -30 to 30 to the brightness-related channels of the image	The change in brightness provides, consequently, a change in the light falling on the trunk and on the targets, thus increasing the possibility of generalisation of the model

These transformations were applied during the training process, using the implementation that is available in the Mask R-CNN network.

It was decided to define that the application of the techniques followed a probability of 83%, that is, each image in the dataset has a probability of 83% of undergoing some modification and being reinserted again in the training set. In addition, the techniques began to be applied randomly and not sequentially, with each one being applied separately to each image, using the imgaug library.

Figure 4.7 shows the code referring to the application of the data augmentation techniques during training using the imgaug library. The augmentation pipeline is defined within the "iaa.Sometimes" augmenter. It includes multiple augmentation operations enclosed within the "iaa.OneOf" augmenter. The "iaa.OneOf" augmenter randomly chooses one of the specified augmenters to apply to the image and mask.

As shown in Figure 4.8, The defined augmentation pipeline was then passed to the "model.train()" function as the augmentation parameter. This ensured that the augmentation operations were applied to the images and masks during the training process. During each training epoch, the "model.train()" function utilized the provided augmentation pipeline (augmentation) to generate augmented versions of the input images and masks, thereby increasing the diversity of the training data.

```
augmentation = iaa.Sometimes(5/6, iaa.OneOf([
    iaa.Affine(rotate=(-10, 10)),
    iaa.Affine(rotate=(-5, 5)),
    iaa.Fliplr(1),
    iaa.Affine(translate_percent=0.1),
    iaa.WithBrightnessChannels(iaa.Add((-30, 30)))
]))
```

Figure 4.7: Implementation code for data augmentation techniques using the imgaug library.

```
model.train(dataset_train, dataset_val,
            learning_rate=config.LEARNING_RATE,
            epochs=100,
            layers='heads',
            augmentation = iaa.Sometimes(5/6,iaa.OneOf(
                [
                    iaa.Affine(rotate=(-10, 10)),
                    iaa.Affine(rotate=(-5, 5)),
                    iaa.Fliplr(1),
                    iaa.Affine(translate_percent=0.1),
                    iaa.WithBrightnessChannels(iaa.Add((-30, 30)))
                ]
            )))
```

Figure 4.8: Mask R-CNN training implementation code.

The "iaa.Sometimes" augmenter with a probability of 5/6 (83%) decided whether to apply the augmentation defined within the "iaa.OneOf" augmenter or not. With a probability of 5/6, one of the specified augmenters was randomly chosen and applied to each image and its corresponding mask. This introduced controlled randomness into the augmentation process, augmenting the data with different transformations each time the augmentation was performed.

By incorporating this approach, the Mask R-CNN model was trained on a varied set of augmented data, enhancing its ability to handle different object orientations, translations, flips, and changes in brightness.

4.2.5 Model Evaluation

In object segmentation, three distinct tasks are performed, one to determine if an object exists in the image, one to find the location of the object, and another to draw a binary mask over the object. Additionally, a typical dataset will have more than one class, whose distribution is not uniform, as is the case with the data used in the present work.

In the present work, the Mean Average Precision (mAP) [44] was used as a criterion to evaluate the models. The global value is determined by calculating the mAP across all classes and at all intersection over union (IoU) boundaries [45]. The IoU metric,

also known as the Jaccard index, allows quantifying the percentage of overlap between the target mask and the mask predicted by the neural network. This metric is closely related to the Dice coefficient, which is often used as a loss function during training.

In a simplified way, the IoU metric measures the number of common pixels between the target and prediction masks divided by the total number of pixels present in both masks, and can be calculated by the following formula:

$$IoU = \frac{TargetMask \cap PredictedMask}{TargetMask \cup PredictedMask} \tag{4.1}$$

The intersection (A ∩ B) is composed of the pixels found in both the predicted mask and the real mask of the object, while the union (A ∪ B) is made up of all the pixels found either in the predicted or target mask. Figure 4.9 ilustrates the IOU between a ground truth mask (in green) and a predicted mask (in red).

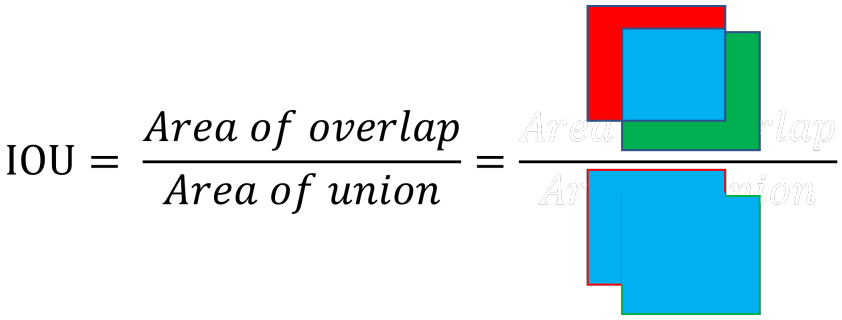

Figure 4.9: Intersection Over Union (IOU).

In the analysis of the results, three mAP limits are considered, mAP@0.5, mAP@0.7 and mAP@0.9. The mAP@0.5 measures the performance of the model with respect to object segmentation when the predicted mask overlaps by at least half of the real box. The mAP@0.7 shows the performance of the model in relation to object segmentation when the predicted mask overlaps at least 70% over the real box. The mAP@0.9 follows the same logic as the previous ones, being that it represents the performance of the model in relation to the segmentation of the object when the predicted mask overlaps by at least 90% of the real box. Figure 4.10 ilustrates the mAP limits, where the predicted mask is shown in yellow and the ground truth mask is shown in red.

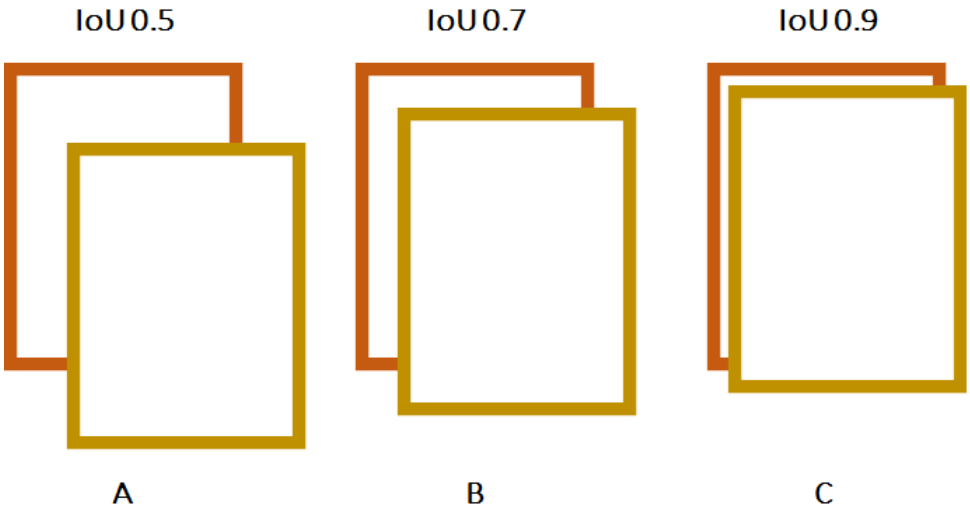

Figure 4.10: mAP limits.

Several experiments were carried out, which allowed the evaluation of the different models. The experiments aimed to improve the performance of the model, through different technical approaches, namely: hyperparameters tunning, increase in the number of images and use of some data augmentation techniques.

4.2.6 Configurations for Mask R-CNN Trainning Phase

During the training phase of Mask R-CNN, several important configurations need to be set in order to optimize the performance and accuracy of the model. By setting these configurations appropriately, the training phase of Mask R-CNN can produce a well-performing model capable of accurately detecting and segmenting objects in images. It is important to experiment and fine-tune these configurations based on the specifics of the dataset and the desired performance goals.

Configurations for the First Experiment

In the first experiment, a dataset of 34 images was used, which was divided into 30 training images and 4 validation images.

Table 4.2 shows the settings defined for the first experiment.

According to Mask R-CNN documentation, the IMAGES_PER_GPU parameter, used to define how many images are trained at once per GPU, consumes a lot of memory. Since the GPU used to train the neural network has only 6GB of dedicated memory, the value used in this experiment was only one image per GPU. It was also decided to keep the default values of learning rate and momentum, in this experiment and in the others carried out later.

The STEPS_PER_EPOCH parameter defines the number of steps in each training epoch.

Table 4.2: Settings of first the experiment.

Parameter	Value
GPU_COUNT	1
IMAGES_PER_GPU	1
STEPS_PER_EPOCH	250
VALIDATION_STEPS	25
BACKBONE	Resnet101
TRAIN_ROIS_PER_IMAGE	50
NUM_CLASSES	3
RPN_ANCHOR_SCALES	(32, 64, 128, 256, 512)
RPN_NMS_THRESHOLD	0.7
RPN_TRAIN_ANCHORS_PER_IMAGE	256
IMAGE_MIN_DIM	800
IMAGE_MAX_DIM	1024

Although the dataset contains few images, this parameter was set to 250 to avoid spending a lot of training time on updates performed by the TensorBoard. The VALIDATION_STEPS parameter defines the number of validations performed at the end of each training period, and a value of 25 was chosen.

The maximum number of regions of interest to be considered in the final layers of the neural network is defined by the TRAIN_ROIS_PER_IMAGE parameter, which obtained a value of 50. This value is relatively low compared to the one defined in the mask article RCNN (512), but also due to the high memory consumption required, it was decided to carry out this experiment using the mentioned value.

The defined batch size is obtained by multiplying the parameters GPU_COUNT and IMAGES_PER_GPU, which in the case of this experiment means that the batch size value is 2. In this project, low batch values are used because the graphics card memory quickly exceeds its limit as this value is increased.

As already mentioned in Subsection 4.2.4, due to the very small size of the dataset, data augmentation techniques were used to increase the amount of training images. In this experiment all the four techniques mentioned in Subsection 4.2.4 were not used, only two of them were used.

- Image Rotation between -10º and +10º;
- Vertical Flip (image mirroring);

The techniques mentioned above were applied sequentially to each image of the dataset throughout each training epoch, using the imgaug library.

Configurations for the Second Experiment

A second experiment was then carried out in an attempt to improve the performance of the previous model. The dataset of this experiment contains more images than the one previously used, more specifically 60 images, which were divided into 50 for training, 7 for validation and 5 for testing.

The settings of the second experiment are shown in Table 4.3. Changes from the settings of the first experiment are highlighted in bold.

Table 4.3: Settings of second the experiment.

Parameter	Value
GPU_COUNT	1
IMAGES_PER_GPU	**3**
STEPS_PER_EPOCH	**500**
VALIDATION_STEPS	**50**
BACKBONE	Resnet101
TRAIN_ROIS_PER_IMAGE	**70**
NUM_CLASSES	3
RPN_ANCHOR_SCALES	(32, 64, 128, 256, 512)
RPN_NMS_THRESHOLD	0.7
RPN_TRAIN_ANCHORS_PER_IMAGE	256
IMAGE_MIN_DIM	**1024**
IMAGE_MAX_DIM	1024

In this experiment, some configuration adjustments were made, changing the parameters IMAGES_PER_GPU, in order to increase the batch size, and TRAIN_ROIS_PER_-IMAGE, in an attempt to improve performance. Both the number of steps per epoch and the number of validation steps were increased, with values 500 and 50 being set, respectively.

One of the main changes from previous experience was the use of more data augmentation techniques and the way in which they were applied. The techniques were applied randomly and not sequentially, with each one being applied separately to each image.

The data augmentation techniques applied were as follows:

- Image Rotation between -10º and +10º;
- Vertical Flip (image mirroring);
- Translation of 10%;
- Brigtness Adjustment.

4.2.7 Model Results

The experiments carried out allowed to evaluate the performance of the different models in relation to the number of images in the dataset, data augmentation techniques and configured parameters.

The models aim to detect and segment the trunk of a cork oak, making it possible to extract the mask from it. The extracted mask will be used to calculate the volume of cork that the cork oak is expected to produce.

Model Results of the First Experiment

The Figures 4.11, 4.12, 4.13 illustrate the segmentation and classification of different types of objects, using the model generated in the first experiment. After 19 epochs it was decided to end the training, as it was being detected a lot of overfitting.

Figure 4.11: First example of segmentation and classification of objects (first experiment).

Figure 4.12: Second example of segmentation and classification of objects (first experiment).

Figure 4.13: Third example of segmentation and classification of objects (first experiment).

Based on the images above, the evaluation of the first experiment's results indicates that the model's performance is not satisfactory. The mean Average Precision (mAP) values at different Intersection over Union (IoU) thresholds are presented in Table 4.4 as follows:

Table 4.4: Results of the first experiment.

mAP 0.5	**mAP 0.7**	**mAP 0.9**
0.579	0.132	0

The mAP metric measures the overall accuracy of object detection by considering the precision and recall trade-off at various IoU thresholds. A higher mAP indicates better performance. In this case, the obtained mAP values are quite low, suggesting that the model struggles to accurately detect objects.

In some cases the presence of false positives, where the model wrongly identifies objects, and the detected mask corresponding to the trunk being lower than expected. These issues suggest that the model may have difficulty accurately identifying objects and distinguishing them from the background or unrelated elements.

Additionally, the results points out that while the model showed some promising results in simpler images, it did not perform well overall. The low mAP values further support the conclusion that the model's performance is unsatisfactory.

These results suggested that further experimentation and fine-tuning improvements was necessary to enhance the model's performance and achieve the desired results.

It should be noted that this experiment served mainly to prepare the next experiment, pending further field data. Expectations in terms of results were low as data were scarce.

Model Results of the Second Experiment

The Figures 4.14, 4.15 and 4.16 illustrates the result of segmentation and classification, using the model generated in the second experiment. The training phase lasted 27 epochs.

Figure 4.14: First example of segmentation and classification of objects (second experiment).

Figure 4.14 shows a forked trunk with three branches, a very common situation in cork oak forests in Portugal. The analysis shows how it was possible to clearly identify the area of the trunk that had already been stripped and that will be stripped again. The image shows a minor problem in the upper zone, on the right branch, where the line separating the stripped and unstripped zone is not clearly visible.

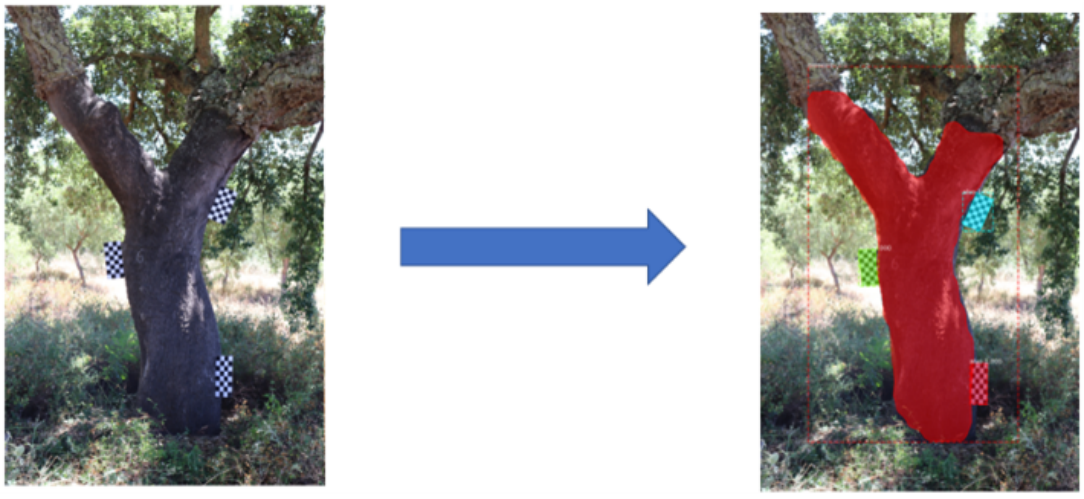

Figure 4.15: Second example of segmentation and classification of objects (second experiment).

The second example, presented in Figure 4.15, shows a cork oak with only two branches and an almost perfect identification of the stripped area. There are still some small problems in the identification of the stripping line and at the base of the trunk, where the natural vegetation is more developed and makes the trunk difficult to see.

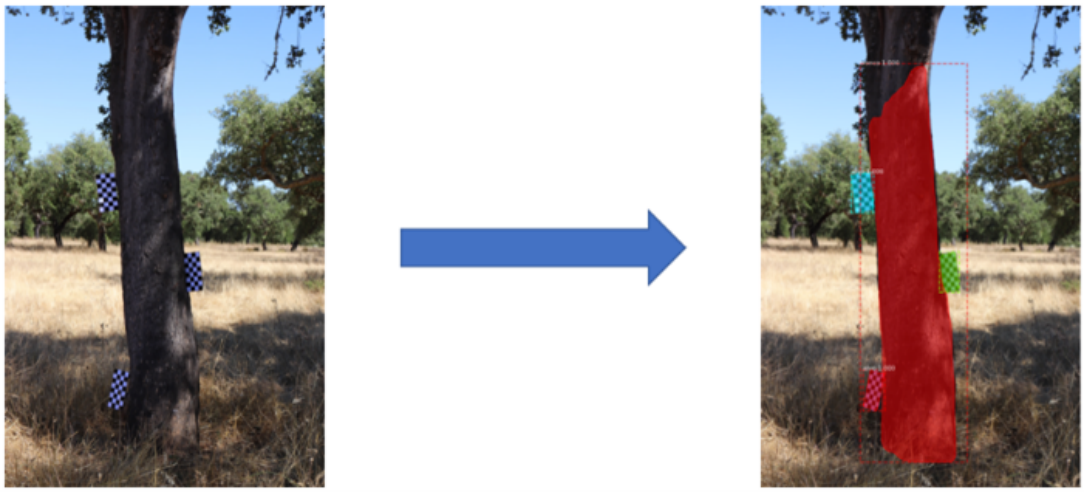

Figure 4.16: Third example of segmentation and classification of objects (second experiment).

The third example is the one that presents a less favorable result since a considerable area in the upper part of the trunk is clearly not classified. This result may be a consequence of the fact that the image was taken in backlight, leaving spots of different luminosities on the trunk. On the other hand, the base of the tree appears very well segmented.

Based on the images, it is evident that the performance of the model was much better than the first experience. Extensive evaluation of the model took place after the training phase, where it was assessed using the aforementioned metrics. The results of this evaluation are summarized in Table 4.5:

Table 4.5: Results of the second experiment.

mAP@0.5	**mAP@0.7**	**mAP@0.9**
0.964	0.964	0.577

These results lead to the conclusion that the model is performing exceptionally well, as its mAP@0.7 exceeds 0.96. The high mAP@0.5 and mAP@0.7 values indicate that this model is suitable for utilization in volume calculations, where precise measurements are crucial. Nevertheless, there remains room for improvement, particularly in enhancing the mAP@0.9 score.

the outcomes of the evaluation emphasize the importance of carefully selecting and collecting images to mitigate interference caused by other forms of vegetation as well as the impact of lighting conditions, including both excess and insufficient light. Addressing these factors is crucial to further enhance the model's performance and ensure accurate results.

4.2.8 Conclusions from the Model Results

The experiments conducted provided valuable insights into the performance of the object detection model. The first experiment revealed that the model's performance was unsatisfactory, with low mean Average Precision (mAP) values indicating difficulties in accurately detecting objects. Issues such as false positives and lower-than-expected detected masks further indicated the model's struggle in accurately identifying and distinguishing objects from the background or unrelated elements.

The second experiment showcased significant improvements compared to the first. The model successfully identified the stripped areas on cork oak trunks in most cases. However, some minor issues persisted, such as difficulties in identifying the stripping line and the base of the trunk when natural vegetation obstructed visibility. Nevertheless, the evaluation metrics, particularly the mAP@0.7, indicated that the model performed exceptionally well, making it suitable for precise volume calculations. Although there was room for improvement, particularly in enhancing the mAP@0.9 score, the results were highly promising.

The evaluation outcomes emphasized the importance of carefully selecting and collecting images to mitigate interference caused by other forms of vegetation and variations in lighting conditions. Both excessive and insufficient light proved to impact the model's performance. Therefore, addressing these factors is crucial to further enhance the model's accuracy and ensure reliable and accurate results.

5 Volume Calculation

This chapter introduces an innovative approach that combines the power of deep learning and machine learning algorithms to streamline the volume estimation process. The proposed method uses the previously generated mask to calculate the area of the trunk region that undergoes the stripping process. The calculated area, together with biometric data of the tree, will be the inputs of a model trained to be able to predict the volume of cork that the cork oak will produce.

The combination of Mask R-CNN for instance segmentation and subsequent volume prediction algorithm holds great promise for revolutionizing the way we assess and manage forest resources, paving the way for more sustainable practices in the future.

5.1 Data Preparation

After training and validating the detection model, the second stage of this work consisted of developing a method for calculating the area of the mask resulting from the output of the neural network. This has been developed with Python 3.7.13, taking advantage of the availability of packages for computer vision and machine learning. For this, firstly, since the images had different dimensions, it was necessary to preprocess them, in which a resize to 1024×1024 was applied so that the detection was performed correctly. Image resizing is a critical step in deep learning detection. It involves scaling an image to a specific size to fit into a neural network model. Since the model was pre-trained with 1024×1024 images this process had to be applied in the detection.

However, resizing an image can also lead to a loss of information, especially if the image is downscaled. To mitigate this, it was important to consider the aspect ratio of an image when resizing. Aspect ratio refers to the proportional relationship between the width and height of an image. If the aspect ratio of an image is not preserved during resizing, the resulting image can appear distorted or stretched.

After obtaining the new width of an image, the aspect ratio can be determined by dividing it by the original width. This ratio can then be used to calculate the new height by multiplying the original height by the ratio. This method ensures that the aspect ratio of the image remains consistent, even after resizing. For this purpose, the scale factor had to be calculated.

The scale factor was determined by the ratio of the new width to the original width, which is equivalent to the ratio of the new height to the original height. Therefore,

to resize an image using a new width, the ratio was calculated by dividing the new width by the original width. Similarly, the ratio using a new height was calculated by dividing the new height by the original height.

To summarize, the formulas for resizing an image while preserving its aspect ratio are presented below:

1. Aspect ratio calculation:

$$AspectRatio = \frac{OriginalWidth}{OriginalHeight} \tag{5.1}$$

2. New height/width calculation:

$$NewHeight = OriginalHeight/Width \times AspectRatio \tag{5.2}$$

3. Scale factor calculation (using the new width):

$$ScaleFactor = \frac{NewWidth}{OriginalWidth} \tag{5.3}$$

4. Scale factor calculation (using the new heigth):

$$ScaleFactor = \frac{NewHeight}{OriginalHeight} \tag{5.4}$$

After the calculation of scale factor, the images were resized using an opencv function (resize()). Figure 5.1 shows examples of masks obtained by applying the resize method and the previously generated model.

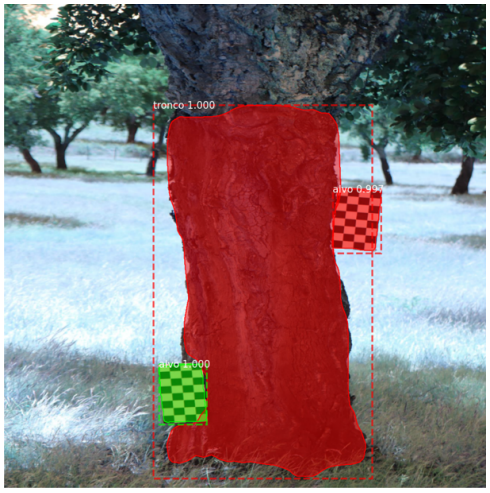

Figure 5.1: Resulting masks (trunk and targets).

5.2 Area Calculation

After detection of the masks (trunk and targets), the area represented by each pixel, in cm^2, was estimated. The estimate was obtained through the ratio between the area in square pixels (px^2) of the targets, calculated from the masks using the minAreaRect method of the OpenCV library, and the actual known area in square centimeters (cm^2). The ratio was calculated for each of the targets present in the image, thus resulting in a list containing the different ratios and the designation of the corresponding targets. The formula for calculating the ratio is shown below:

$$R = \frac{MA}{TA} \tag{5.5}$$

In which:

R = Ratio;

MA = Mask Area in px^2;

TA = Target Area in cm^2.

The ratio variable represents the conversion factor between areas measured in squared pixels (px^2) and squared centimeters (cm^2). The Mask Area denotes the area obtained in px^2 from the analyzed mask, while the Target Area refers to the known area of the target in cm^2. The outcome of the formula described above was rounded off to two decimal places. By leveraging the previously calculated target ratios, we could derive the trunk mask area. This involved determining the trunk area in px^2 through the contour of the trunk mask, using the contour area method once again. However, to convert the area to cm^2, it was necessary to utilize the ratios established earlier from the target masks. Since trees possess unique characteristics and the targets were positioned differently on their trunks, it was necessary to identify the most accurate ratio to use for calculating the trunk area. It was decided to use the average of the ratios of the different targets and to obtain the ratio to be used in the calculation of the trunk area, hereinafter called Factor Ratio (FR). The FR was calculated using the equation below:

$$FR = \frac{R_1 + R_2 + \ldots + R_n}{NT} \tag{5.6}$$

In which:

FR = Factor Ratio;

R_n = Ratio (R) of the target n (n= 1,2,3);

NT = Number of targets in the image.

Figure 5.2 illustrates an example of a tree with two targets.

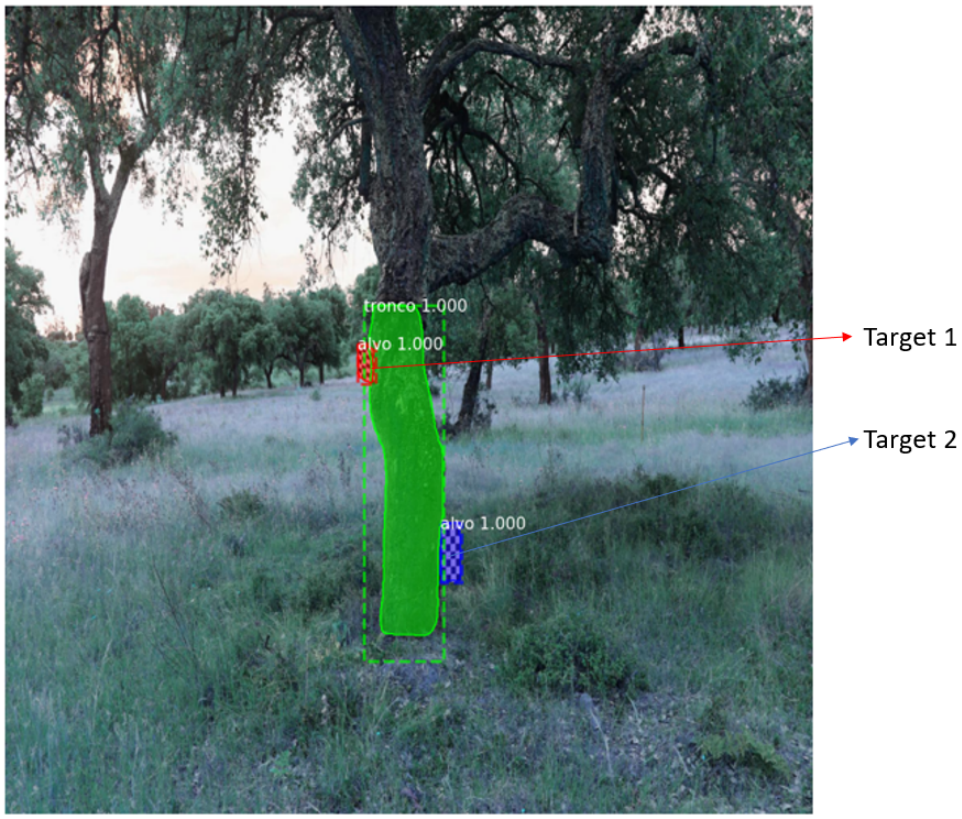

Figure 5.2: Calculation of the Factor Ratio of two targets.

The trunk area, in cm^2, is derived by dividing the calculated area, in cm^2, by the FR. The formula for calculating the trunk area is shown below:

$$TrunkArea(cm^2) = \frac{MaskArea(px^2)}{FR} \tag{5.7}$$

The value resulting from the application of the formula presented above was rounded to two decimal places and later converted into m^2. Figure 5.3 is a flowchart of the procedure used to calculate the area of the trunk.

After calculating the area, three machine learning algorithms were tested to estimate the final volume of cork, which is the third stage of this project. Linear and non-linear regression algorithms were used, namely LinearRegression, Support Vector Regression (SVR) and MLPRegressor, all taken from the Python sk-learn library. All those algorithms were trained and tested using the same dataset. All stages of the model training process are presented below.

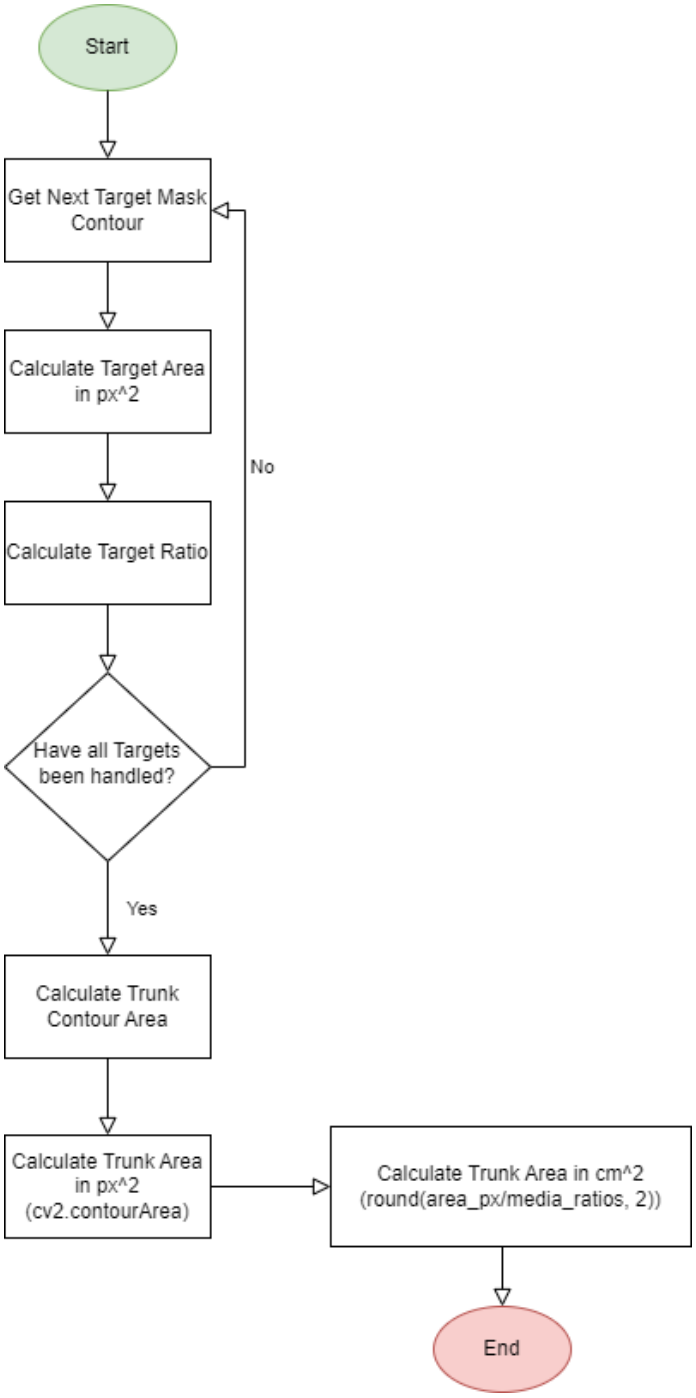

Figure 5.3: Flowchart of the steps to calculate the tree trunk's area (TA).

5.3 Biometric Parameters Dataset

The forest inventory is carried out during the spring-summer crop. Data were collected in the field in circular plots with an area of 1000 m^2. As well as the images collected in the field. The measurements were made by other team members belonging to ESAC.

5.3.1 Data Collection and Manual Calculations

In this inventory, the data was recorded as follows: identification of the farm to which the tree belongs to (property), tree number and the diameter of the tree at the base (D030). The diameter of the trunk was also taken at 1.30 m (D130) and at 2 m (D200) height. The values of cork thickness (Eco030, Eco130; Eco200) were also measured at these heights. The total height of the tree, the height of the first branch, the height of the first fork and the height of the stem without boughs or forks, as well as the diameter of the crown in the north-south and east-west directions were measured. Finally, the type of cork (secundeira or amadia) was recorded. Using these data, it was possible to calculate, manually, the trunk area and the cork volume. Figure 5.4 shows the regions of the trunk from which the dataset values were measured.

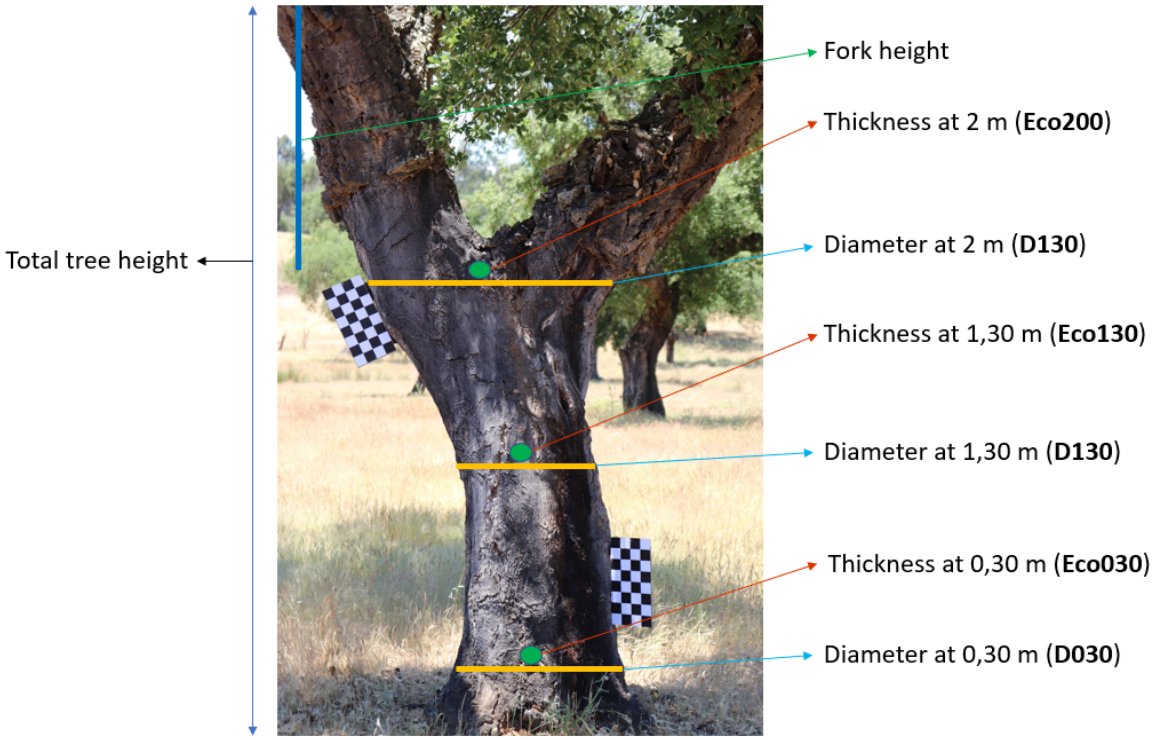

Figure 5.4: Measuring regions.

The first cork stripping usually takes place when the tree is 25 to 30 years old, and the cork stripping occurs between June and August. This cork, sometimes with considerable thickness, is called virgin and differs significantly from the cork stripped at the following stages: it is called secondary cork in the second extraction and amadia in the

subsequent extractions. This dataset does not contain virgin trees since Mask R-CNN would not be able to correctly extract the trunk region corresponding to the extraction of the cork because the trunk of virgin trees does not show any trace of cork stripping.

5.3.2 Data Augmentation

Biometric records of some trees from the dataset of images were used in this study. It was not possible to collect records of all trees and, therefore, the dataset for cork volume estimation contains only data from 18 trees.

Since the dataset has a small size, it was necessary to resort to some data augmentation techniques. For this purpose Roboflow was used, a computer vision developer framework for better data preprocessing and model training techniques, available from [https://roboflow.com, accessed on 20 July 2022]. The data augmentation techniques were applied to images whose data were present in the biometric parameter's dataset. It should be noted that different images of the same tree may result in varying area measurements, but these diverse areas will correspond to the same set of measurements for volume calculation in the model that will be trained after. The following transformations were applied:

- Crop: 0% Minimum Zoom, 10% Maximum Zoom. In this technique, a random portion of the original image is cropped and zoomed in, up to a maximum of 10%. By randomly cropping and zooming in on different portions of the image, the model can learn to recognize objects and patterns from different perspectives and scales, making it more robust and accurate in its predictions. First, a random point is selected within the original image, then a square patch of the image is then cropped around this point, with a size that is between 90% and 100% of the original image size. The cropped patch is then resized to the original image size, effectively zooming in on the selected portion of the image.
- Vertical Flip, which involves flipping an image vertically along the central horizontal axis. This operation creates a mirror image of the original image, where the top and bottom portions of the image are swapped. By flipping the image vertically, new training data that is more representative of the range of camera orientations that may be encountered in the field can be created. It's important to note that vertical flip was a suitable augmentation technique, since the orientation of the trunk in the image has a specific directional bias.
- Brightness: Between -5% and +5%. In this technique, the brightness of an image is adjusted by a certain percentage to create new, slightly different versions of the same image, which can be useful for training machine learning models that need to be robust to changes in brightness levels. When applying a brightness adjustment of -5% to an image, the brightness of the image will be decreased by

5%. This means that the pixel values of the image will be multiplied by a factor of 0.95, resulting in a darker image. On the other hand, when applying a brightness adjustment of +5% to an image, the brightness of the image will be increased by 5%. This means that the pixel values of the image will be multiplied by a factor of 1.05, resulting in a brighter image.

Additionally, crop, vertical flip, brightness adjustment and other machine learning techniques can help reduce overfitting by creating a larger and more diverse training set from a smaller set of original images. By introducing variations in the training data, the model becomes less likely to memorize the training examples and more likely to generalize well to new, unseen images.

After applying the above techniques, the dataset now contains 100 instances since the area calculation algorithm was applied to all images, both transformed and nontransformed. Figure 5.5 illustrates part of the resulting dataset.

	Property	Parcel	No_tree	Type	D030	D130	D200	Eco030	Eco130	Eco200	v (m3)	area (m2)	calculated_area (m2)
0	mouchao	2	1	Sb amadia	40.36	40.23	0.0	3.10	2.80	0.00	0.166	0.52	0.84
1	mouchao	2	1	Sb amadia	40.36	40.23	0.0	3.10	2.80	0.00	0.166	0.52	0.81
2	mouchao	2	1	Sb amadia	40.36	40.23	0.0	3.10	2.80	0.00	0.166	0.52	0.77
3	mouchao	2	1	Sb amadia	40.36	40.23	0.0	3.10	2.80	0.00	0.166	0.52	0.76
4	mouchao	2	1	Sb amadia	40.36	40.23	0.0	3.10	2.80	0.00	0.166	0.52	0.82
...	...	...	...	...	...	...	...	...	...	...	...	...	...
95	mouchao	11	12	Sb amadia	58.30	58.10	0.0	2.77	3.07	2.27	0.346	0.78	1.58
96	mouchao	11	12	Sb amadia	58.30	58.10	0.0	2.77	3.07	2.27	0.346	0.78	1.70
97	mouchao	11	12	Sb amadia	58.30	58.10	0.0	2.77	3.07	2.27	0.346	0.78	1.74
98	mouchao	11	12	Sb amadia	58.30	58.10	0.0	2.77	3.07	2.27	0.346	0.78	1.71
99	mouchao	11	12	Sb amadia	58.30	58.10	0.0	2.77	3.07	2.27	0.346	0.78	1.76

100 rows × 13 columns

Figure 5.5: Part of the Biometric Parameters Dataset.

The application of the area calculation algorithm to augmented images is shown in Figure 5.6.

Once the process of increasing the number of records was concluded, an analysis of the data was carried out in order to identify the features that best correlate with each other and the distribution of the data in the dataset.

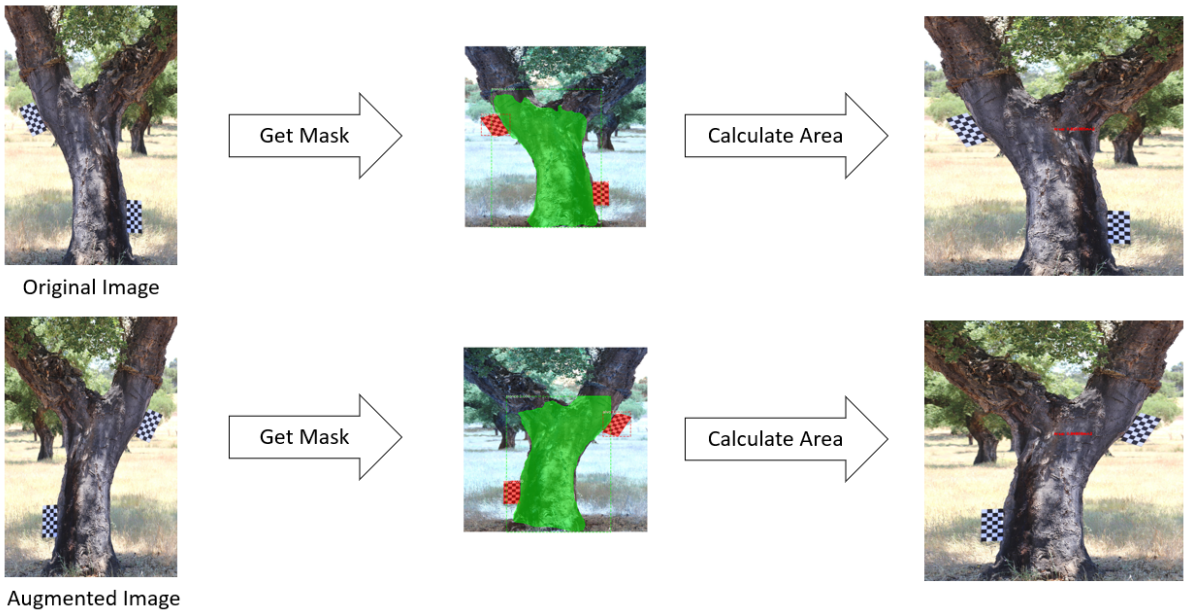

Figure 5.6: Area calculation using augmented images.

5.3.3 Feature Correlation

Before preparing the training dataset, an analysis of the data obtained during the collection phase was performed. For this, a dataset was created composed by the features recorded in data collection phase except the total height of the tree, the height of the first branch, the height of the first fork and the height of the stem without boughs or forks, as well as the diameter of the crown in the north-south and east-west directions. In addition to these features, two more features have been added, one corresponding to the area calculated using the developed methodology described above (area (m^2)) and the other corresponding to the volume (v (m^3)) calculated manually. It should be noted that most of these features were not used for training, but only for dataset analysis, as will be explained in the next section.

The correlation between the dataset features and the extracted cork volume was analyzed. It was found that the base diameter and the diameter at breast height (1.30 m) were the ones that showed the best correlation, 0.98 and 0.99, respectively. This correlation is very high. This is not completely surprising, as the diameter of a tree is known to be highly correlated with its volume..

The correlation between the areas (real and calculated) and the cork volume was also analyzed. It was found that the calculated area and the real area presented a correlation of 0.87 and 0.85, respectively. This reveals that the trunk area is closely linked with the cork volume. Figure 5.7 illustrates the correlation between the different features and the cork volume.

In order to reinforce the high correlation between the D130 feature and the volume, the graph represented in Figure 5.8 was created. By examining the shape and direction of

```
v (m3)                  1.000000
D130                    0.992616
D030                    0.985677
calculated_area (m2)    0.881204
area (m2)               0.868051
D200                    0.532158
Parcel                  0.309632
Eco200                  0.255604
No_tree                -0.202410
Eco130                 -0.252046
Eco030                 -0.321028
Name: v (m3), dtype: float64
```

Figure 5.7: Correlation between features and cork volume.

the points in the plot, we can get an idea of the strength and nature of the relationship between this variables.

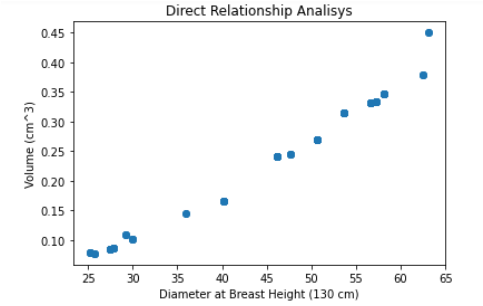

Figure 5.8: Direct Relationship between D130 and Volume.

The following section describes all the experiments carried out to find the best algorithm to estimate the cork volume through the area previously calculated.

5.4 Data preparation for training

Before training the models, the first aproach was to remove some features. This is because, in a real situation, using only the camera, it is not possible to have acess to certain data, such as the diameter and real trunk area, the cork thickness and the volume (target of our model). By removing all the features mentioned, the model would have only three input features, which would be the property, a very specific feature of the trees in this dataset and should be ignored, the area previously calculated through the output of the deep learning model and the tree type (Secundeira or Amadia).

Analysing once again the correlation between the features it is possible to verify that there is indeed a strong correlation between the area previously calculated and the volume of the cork. However, cork volume calculation seems more complex and may require multiple input features to accurately predict more accurately.

With this vision, and to evaluate the performance of different models with different

input features, four datasets were created. The first one (dataset 1) contained the calculated area feature (calculated_area) and the tree type (Type). The second dataset (dataset 2) contained three features, the calculated area (calculated_area), the diameter at breast height (d130) and the tree type (Type). This dataset contains the feature that correlates most with volume, which may provide good results in terms of performance. The third dataset (dataset 3) contains one more feature, corresponding to the diameter of the base (D030). The purpose of the third dataset was to verify if the performance of the models improved when compared to the second dataset. The fourth dataset (dataset 4) was used to evaluate the performance of the models by introducing the feature of cork thickness at breast height (Eco130). Figures 5.9, 5.11 and 5.12 illustrate the flow of inputs and outputs of the generated models using datasets datasets 1, 2 and 3, respectively.

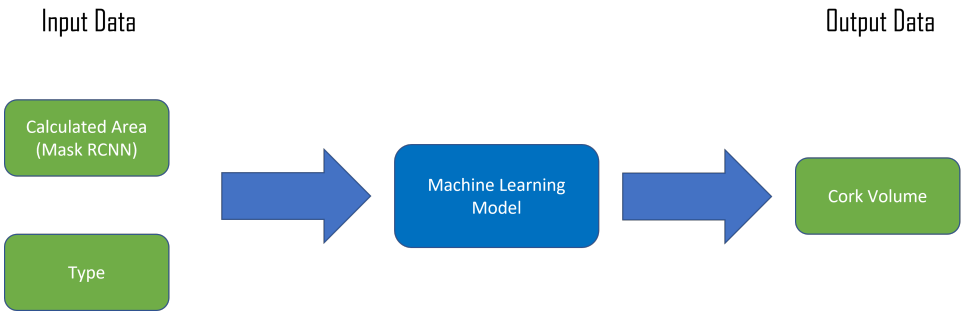

Figure 5.9: Inputs and Output of the Model using dataset 1.

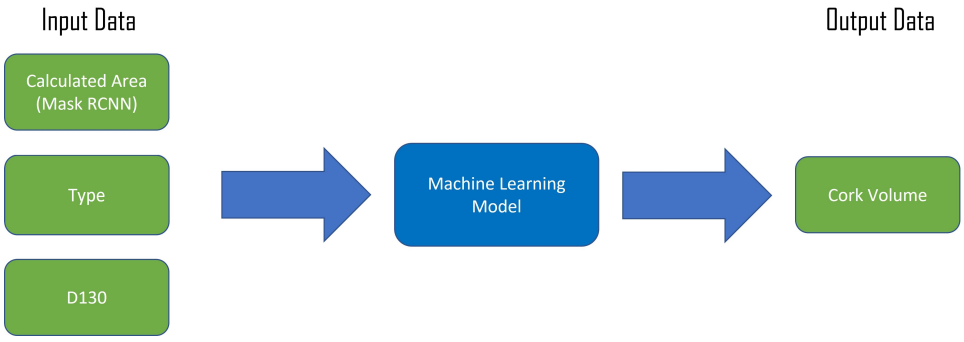

Figure 5.10: Inputs and Output of the Model using dataset 2.

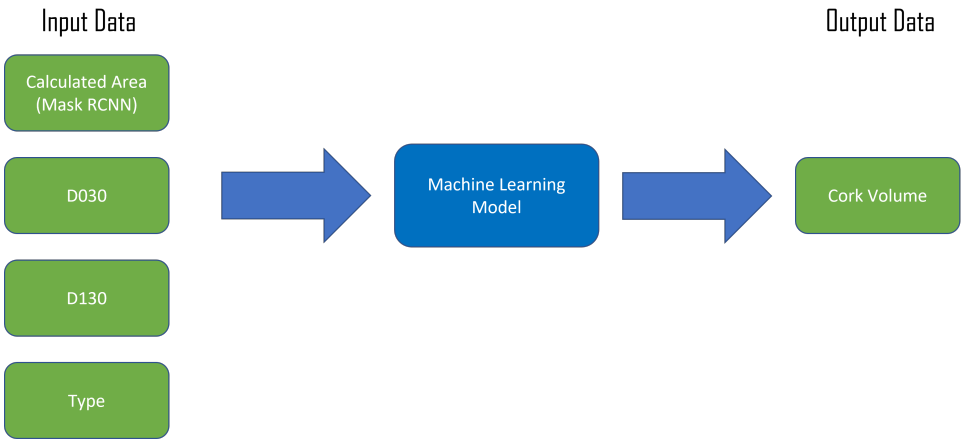

Figure 5.11: Inputs and Output of the Model using dataset 3.

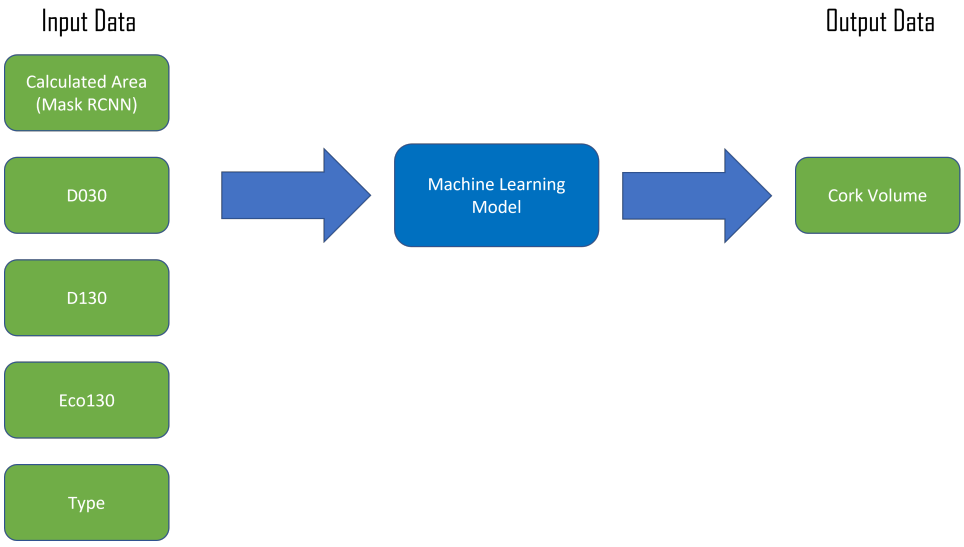

Figure 5.12: Inputs and Output of the Model using dataset 4.

5.5 Machine Learning Models

Three machine learning algorithms were used in the experiments. Linear regression, support vector regression (SVR) and MLP regression were chosen.

5.5.1 Linear Regressor

The linear regression algorithm is a supervised machine learning algorithm used for predicting a continuous outcome variable (also known as the dependent variable) based on one or more independent variables (also known as predictors or features). The goal of linear regression is to find the best fit line or hyperplane that can predict the outcome variable with minimum error.

The simplest form of linear regression is simple linear regression, which involves only one independent variable. In this case, the linear regression model can be expressed as:

$$y = b_0 + b_1 \times x \tag{5.8}$$

where:

- y is the dependent variable (the variable we want to predict)
- x is the independent variable (the variable we use to predict y)
- b_0 and b_1 are the intercept and slope of the line, respectively

The linear regression algorithm estimates the values of b_0 and b_1 by minimizing the sum of squared errors between the predicted and actual values of y. The sum of squared errors (SSE) can be expressed as:

$$SSE = \Sigma(y - y_{pred})^2 \tag{5.9}$$

Where:

- y is the actual value of the dependent variable
- y_{pred} is the predicted value of the dependent variable

The algorithm finds the values of b_0 and b_1 that minimize SSE using a method called ordinary least squares (OLS) regression. Once the values of b_0 and b_1 are determined, the model can be used to predict the value of y for new values of x.

5.5.2 Support Vector Regressor

The SVR algorithm is based on the Support Vector Machine (SVM) algorithm, which was originally developed for classification problems.

In SVR, the goal is to find a function that best fits the data by identifying a hyperplane that maximizes the margin between the predicted values and the actual values. The hyperplane is constructed in such a way that it lies as close as possible to the majority of the points, while still being able to predict the remaining points accurately.

To achieve this, SVR uses a technique called kernel trick. The kernel trick is a method of transforming data into a higher dimensional space, where it is easier to separate the data points using a hyperplane. In other words, it allows the algorithm to fit non-linear data by mapping it into a higher dimensional space.

The SVR algorithm aims to minimize the distance between the predicted values and the actual values, while also controlling the complexity of the model by introducing a penalty for large coefficients. This is done through the use of a loss function, which is optimized using an optimization algorithm such as gradient descent.

SVR is particularly useful in cases where the data is non-linear or has a high degree of noise.

5.5.3 MLP Regressor

The MLP Regressor involves predicting a continuous numerical output based on a set of input features. The term MLP stands for Multi-Layer Perceptron, which refers to the structure of the neural network.

The MLP Regressor consists of an input layer, one or more hidden layers, and an output layer. Each layer contains a set of neurons, which are interconnected and communicate with each other through weighted connections. The neurons in the input layer receive the input features, while the neurons in the output layer produce the predicted output values.

During training, the MLP Regressor adjusts the weights of the connections between the neurons to minimize the difference between the predicted outputs and the true outputs for a given set of training examples. This is done using an optimization algorithm, such as stochastic gradient descent, to find the weights that minimize a cost function, such as mean squared error.

The MLP Regressor is a powerful algorithm that can be used for a wide range of regression tasks. However, it can be prone to overfitting if the model is too complex or if there is not enough data to support the complexity of the model. Regularization techniques, such as L1 and L2 regularization, can be used to prevent overfitting.

5.6 Model evaluation metrics

The metrics of mean absolute percentage error (MAPE), mean squared error (MSE) and root mean squared error (RMSE) were used as performance indicators.

The Mean Absolute Percentage Error (MAPE) is a commonly used evaluation metric in machine learning regression tasks, particularly when dealing with prediction problems. It provides a measure of the accuracy of a model by calculating the average percentage difference between the predicted values and the actual values, as shown in the equation below:

$$MAPE = \sum_{i=1}^{D} \left| \frac{x_i - y_i}{x_i} \right| \times 100 \tag{5.10}$$

where D is the total number of observations, x_i is the actual value of the i-th observation, and y_i is the predicted value of the i-th observation.

The MAPE metric provides a relative measure of the error, as it expresses the prediction accuracy in terms of a percentage of the actual value. This makes it particularly useful when comparing the performance of different models, as is the case of these experiments.

The mean squared error measures the average squared difference between the predicted and actual values in a set of data.

To calculate the MSE, we first take the difference between the predicted value and the actual value for each data point, then square each of these differences, and finally take the average of all the squared differences. The formula for MSE is:

$$MSE = \sum_{i=1}^{D} (x_i - y_i)^2 \tag{5.11}$$

where D is the number of data points, x_i is the actual value of the i data point, and y_i is the predicted value of the i data point.

The higher the MSE, the worst the model's performance is. Ideally, we want the MSE to be as close to 0 as possible, indicating that the model's predictions are very accurate. However, low MSE does not always mean that the model is good, as it could be overfitting the data. Therefore, it's important to use other metrics as well to evaluate the model's performance.

Root mean squared error measures the difference between the predicted values and the actual values of a dataset. The RMSE is calculated by taking the square root of the average of the squared differences between the predicted and actual values, as shown in the equation below:

$$RMSE = \sqrt{\sum_{i=1}^{D}(x_i - y_i)^2} \tag{5.12}$$

Where:

- x_i is the predicted value
- y_i is the actual value
- D is the number of observations in the dataset

RMSE is a widely used metric because it considers the magnitude of the error, and penalizes large errors more than small ones. This means that RMSE is particularly useful when we want to identify models that perform well in predicting values that are far from the mean.

In general, the lower the RMSE value, the better the model's predictive accuracy. However, it's important to note that the interpretation of an RMSE score will depend on the specific problem and dataset being analyzed.

5.7 Configuration and Results of the Machine Learning Models

This section presents the outcomes of the machine learning models used in this study, along with the configurations used to train and test them. This section is crucial for demonstrating the efficacy and reliability of the models and providing insights into their performance.

In our study, we will analyze and present the setup and outcomes of three distinct machine learning models: LinearRegressor, Support Vector Regression (SVR), and MLPRegressor. We will also assess their effectiveness by comparing their performance based on the data gathered during the fieldwork.

5.7.1 Configuration of the models

Machine learning models require configuration settings to be chosen for optimal performance on a given task. In this subsection, we will discuss the configurations of the three regression models. The same model settings have been applied in all experiments performed on the various data sets.

Before training the models, a pipeline was created for data preparation. Since there were no inconsistencies in the data and no missing values, the pipeline only served to encode the categorical variable Type using the OneHotEncoder technique and to scale the numeric variables to a range between 0 and 1 using the MinMaxScaler technique.

The LinearRegressor model is a simple linear regression model that assumes a linear relationship between the input features and the target variable. The hyperparameters set for the LinearRegressor model include:

1. **fit_intercept**: This hyperparameter is a boolean flag that determines whether to fit an intercept term in the model. An intercept term is used to model the bias in the target variable. The hiperparameter was set to True.
2. **normalize**: This hyperparameter is another boolean flag that determines whether to normalize the input features. Normalization is often used to bring all input features to the same scale and is set to False by default. Since the data are highly correlated, normalization was not necessary.;
3. **copy_X**: This hyperparameter is also a boolean flag that determines whether to copy the input features. It is set to True by default, which means that the input features are copied before being processed by the model.
4. **n_jobs**: This hyperparameter specifies the number of CPU cores to use for computation. It was to -1, which means that all available cores are used.
5. **positive**: This hyperparameter is used to enforce positivity constraints on the predicted values. It is set to False by default, which means that the predicted values can be both positive and negative. Since the target variable is strictly positive, setting positive to True improve the accuracy of the predictions.

In order to optimize the performance of an SVR model, a grid search was used to explore different combinations of hyperparameters. Figure 5.13 shows the diferent combinations of hiperparameters used in grid search.

```
svr_param_grid = {'kernel': ['poly', 'rbf', 'sigmoid'], 'degree': [1,2,3,4], 'gamma': ['scale', 'auto'],
                  'coef0': [0.001,0.002, 0.003, 0.005, 0.008, 0.009,0.01,0.02,0.05,0.09,0.1,0.2,0.3,0.4,0.5],
                   'epsilon': [0.0001, 0.0005, 0.001,0.002, 0.003, 0.005, 0.008, 0.009,0.01, 0.02, 0.05, 0.09, 0.1, 0.15, 0.2]}
```

Figure 5.13: Grid Search technique for SVR model optimization.

After the first experiment, using dataset 1, the best hyperparameters resulting from the grid search were:

- coef0: **0.5**. This hyperparameter is used when using the polynomial kernel, which is the case. It controls the independent term in the kernel function. A higher coef0 can lead to a better fit, but can also increase the risk of overfitting.
- degree: **4**. This hyperparameter is also used when using the polynomial kernel. It controls the degree of the polynomial used for transformation. A higher degree may lead to overfitting, while a lower degree may lead to underfitting.
- epsilon: **0.01**. This hyperparameter controls the width of the epsilon-tube, which is the region where no penalty is incurred for errors. A larger epsilon will allow more points to fall within the tube, resulting in a wider margin and a less complex

model. A smaller epsilon will result in a narrower margin and a more complex model.

- kernel: **poly**. The kernel function specifies the type of function that is used to transform the input data into a higher dimensional space where the problem can be linearly separable. In this case, SVR used a polynomial kernel function to transform the input data into a higher-dimensional feature space, where the problem of finding a linear regression function is easier to solve.

The hyperparameters of the MLP Regressor control the architecture and training of the neural network. The hyperparameters set for the MLP Regressor in this study are as follows:

1. **Hiden Layers Sizes**: This hyperparameter controls the number of nodes in each hidden layer of the MLP Regressor. Various sizes were experimented with ranging from 50 to 500 nodes per layer and it was found that a size of 100 nodes per layer provided the best performance.
2. **Activation Function**: This hyperparameter determines the mathematical function used by each node to transform the input data. The ReLU (Rectified Linear Unit) activation function was utilized for all hidden layers and the linear activation function for the output layer. The ReLu function was chosen due to its non-linearity, sparsity, computational efficiency, and robustness to the vanishing gradient problem.
3. **Solver**: This hyperparameter determines the optimization algorithm used to train the MLP Regressor. The Adam solver was used, which is an adaptive learning rate optimization algorithm.
4. **Learning Rate**: This hyperparameter determines the step size used to update the weights of the MLP Regressor during training. The rate was fixed at 0.001.
5. **Number of Epochs**: This hyperparameter controls the number of times the MLP Regressor is trained on the entire dataset. We used 500 epochs, but the early stopping technique was applied.
6. **Batch Size**: This hyperparameter determines the number of samples used to compute the gradient during each iteration of training. A batch size of 10 was used, since represents 10% of the samples.

5.7.2 Results of the Machine Learning models

In this subsection, the results of the machine learning models that were applied to the diferent datasets are presented. The goal of this study was to predict the volume of cork that will be extracted after the stripping process. The resulting datasets (datasets 1, 2, 3 and 4) were divided into 80% for training and 20% for testing. During the training

process, we employed the K-Fold Cross Validation technique with a value of k equal to 10. This approach allowed to assess the performance and generalization ability of my machine learning model more effectively. By dividing the training data into 10 equally-sized folds, the training and evaluation process were performed ten times, ensuring that each fold acted as a validation set once. The utilization of K-Fold Cross Validation with a value of k equal to 10 helped make informed decisions regarding model selection, hyperparameter tuning, and feature engineering, ultimately enhancing the reliability and credibility of my trained model.

After the first experiment, using the dataset 1, with features Type and calculated_area, the results were as follows. Table 5.1 shows the MAPE, MSE and RMSE of each algorithm, with the two features mentioned above.

Table 5.1: Performance results of the machine learning models using dataset 1.

	MSE	**RMSE**	**MAPE**
LinearRegressor	0.00096	0.03095	10.22%
SVR	0.00078	0.02788	8.75%
MLPRegressor	0.00084	0.02903	10.79%

It is possible to verify, through the data presented in Table 5.1, that the SVR algorithm presented the best results, with the average error being 8.75% for the volume estimation. In any case, the other algorithms produced positive results, with the LinearRegressor algorithm showing an average error of 10.22% and the MLPRegressor algorithm showing an error of 10.79%.

In the second experiment, using dataset 3, with features Type, calculated_area, and D130 the results were as follows. Table 5.2 shows the MAPE, MSE and RMSE of each algorithm with the four features mentioned above.

Table 5.2: Performance results of the machine learning models using dataset 2.

	MSE	**RMSE**	**MAPE**
LinearRegressor	0.00009	0.00913	3.17%
SVR	0.00003	0.00592	2.96%
MLPRegressor	0.00007	0.00965	3.56%

It is possible to verify, through the data presented in Table 5.2, that, once again, the SVR algorithm presented the best results, with the average error being 2.96% for the volume estimation. In this experiment, the performance of the models improved significantly, resulting in average errors of less than 3.6%. This can be explained by the high correlation between the feature of trunk diameter at breast height and cork volume. Although the results were better, this case may be more difficult to use in a real scenario since the trunk diameter has to be measured manually.

In the third experiment, using dataset 3, with features Type, calculated_area, D030 and D130 the results were as follows. Table 5.3 shows the MAPE, MSE and RMSE of each algorithm with the four features mentioned above.

Table 5.3: Performance results of the machine learning models using dataset 3.

	MSE	**RMSE**	**MAPE**
LinearRegressor	0.00009	0.00954	3.08%
SVR	0.00003	0.00526	2.74%
MLPRegressor	0.00007	0.00866	3.13%

The results of this experiment show that, despite the slight improvement, the performance of the models did not change significantly leading to the conclusion that the addition of the base diameter feature does not bring very significant improvements. Once again, the SVR algorithm presented the best results, with the average error being 2.74% for the volume estimation.

In the fourth experiment, using dataset 4, with features Type, calculated_area, D030, D130 and Eco130 the results were as follows. Table 5.4 shows the MAPE, MSE and RMSE of each algorithm with the five features mentioned above.

Table 5.4: Performance results of the machine learning models using dataset 4.

	MSE	**RMSE**	**MAPE**
LinearRegressor	0.00009	0.00947	3.15%
SVR	0	0.00049	0.15%
MLPRegressor	0.00003	0.00518	2.39%

After the fourth experiment, it was possible to verify, through the data presented in Table 5.3, that, once again, the SVR algorithm presented the best results, with the average error being 0.15% for the volume estimation. The average error of the SVR algorithm is close to 0% which leads to believe that the cork thickness feature has relevance for the model performance. Again, this case may not be useful for a real scenario as the cork thickness has to be measured manually and the focus is to facilitate the volume estimation process.

A fifth experiment was performed in order to demonstrate the superiority of the SVR algorithm, which was the one that presented the best results, with the direct relationship between the D130 features and the Volume (target). For this, linear regression was used to model the relationship between the dependent variable (target) and one independent variables (D130).

An instance of the linear regression model from scikit-learn was created and fit it to your training data. The linear regression model estimated the coefficients for the relationship between the feature and the target. Table 5.5 shows the MAPE of the algorithm

of Linear Regression compared to the MAPE of the SVR algorithm in the experiments performed.

Table 5.5: Comparation between direct relationship and SVR algorithm performance in the second experiment.

	MAPE
LinearRegressor (Direct Relationship)	3.12%
SVR (First Experiment)	8.75%
SVR (Second Experiment)	2.96%
SVR (Third Experiment)	2.74%
SVR (Fourth Experiment)	0.15%

Analyzing the data of the table it is possible to conclude that, except in the case of the model that received as inputs only the features Type and calculated area, the SVR algorithm presented better results showing itself superior to the direct relationship between the variables.

5.7.3 Summary of the Results and Conclusions

Based on the results presented above, we can draw the following conclusions:

- **Performance of Different Algorithms**: The SVR algorithm consistently outperformed the LinearRegressor and MLPRegressor algorithms in all four experiments. It yielded the lowest average error rates for volume estimation, with errors of 8.75%, 2,96%, 2.74%, and 0.15% in the respective experiments.
- **Improvement in Model Performance**: The second experiment showed a significant improvement in model performance, with all algorithms achieving average errors of less than 3.6%. This improvement can be attributed to the high correlation between trunk diameter and cork volume features, suggesting that these features strongly influence the accuracy of the models.
- **Limitations of Real-World Application**: While the results of the second and third experiments were promising, it is important to note that certain features used for estimation, such as trunk diameter and cork thickness, need to be measured manually. This limitation may restrict the practical applicability of the model in real-world scenarios, as manual measurements can be time-consuming and potentially impractical.
- **Relevance of Cork Thickness**: The fourth experiment demonstrated that the SVR algorithm achieved an average error close to 0%, indicating the relevance of the cork thickness feature for accurate volume estimation. However, the requirement for manual measurement of cork thickness, once again, limits the usefulness of this particular case in practical applications.

The SVR algorithm consistently outperformed other algorithms in estimating cork volume. The strong correlation between certain features and volume estimation contributed to improved model performance. However, the reliance on manual measurements of trunk diameter and cork thickness poses practical challenges for real-world implementation, despite the favorable results obtained in the experiments.

We have previously noted that there is a high correlation between the d130 variable and volume. However, in spite of that, SVR's superiority over a direct relationship with the d130 feature can be attributed to two main factors:

1. **Complex Relationships**: Volume estimation in forestry is a complex task influenced by multiple factors. SVR, being a powerful regression algorithm, can capture intricate relationships among features and the target variable.
2. **Non-Linear Patterns**: SVR is capable of capturing non-linear patterns in the data, which may exist between the input features and volume estimation. This flexibility allows SVR to model complex relationships that a simple direct relationship might not account for.

The inclusion of calculated area, diameter at the base, and especially cork thickness enhanced the model's ability to estimate volume accurately. The performance of SVR suggests its effectiveness in capturing complex relationships and handling high-dimensional data, making it a suitable choice for volume estimation in this context.

6 Deployment

The deployment of the volume estimation component, as part of the larger *Floresta Digital* project that involves the integration of calculation simulators for various tree types, is a crucial aspect of bringing the project to fruition. This chapter focuses on the deployment process of the volume estimation module, including some website details, server configuration, and the functionality of the volume estimation form.

6.1 Project *Floresta Digital*

While the implementation of the deployment platform is handled by another member of the project, our contribution lies in the development and integration of the machine learning models for accurate volume estimations. By actively participating in the project, we have played a crucial role in the successful deployment of the volume estimation component, enhancing the overall functionality and accuracy of the *Floresta Digital* project. Our contribution extended beyond the development of the machine learning models, as we actively collaborated in their implementation, ensuring their seamless integration into the deployment framework. This module provides users with a reliable and efficient tool for estimating the volume of cork in cork oak trees.

The implementation of the deployment platform, including server configuration, falls under the responsibility of another member of the *Floresta Digital* project. The member is in charge of configuring the nginx server, which provides the necessary infrastructure for hosting the volume estimation module. The nginx server ensures efficient handling of requests and enables seamless communication between the users and the deployed models.

Users can access the platform through the project's official website at [`http://www.floresta.digital.esac.pt`]. The website provides an interface for users to interact with the volume estimation form and input the necessary data for predicting the volume of cork in cork oak trees.

6.2 Integration of the Cork Volume Simulator

As a key component of the deployment, the volume estimation form allows users to provide the required data for accurate volume predictions. While certain fields in the form are optional, others are mandatory for precise estimations. The mandatory fields

include the type of cork oak and the image upload. The form has been designed to be user-friendly and intuitive, guiding users through the process of submitting the necessary data. Figure 6.1 shows the form, with the mandatory and optional fields, for filling in the data used as input for the models.

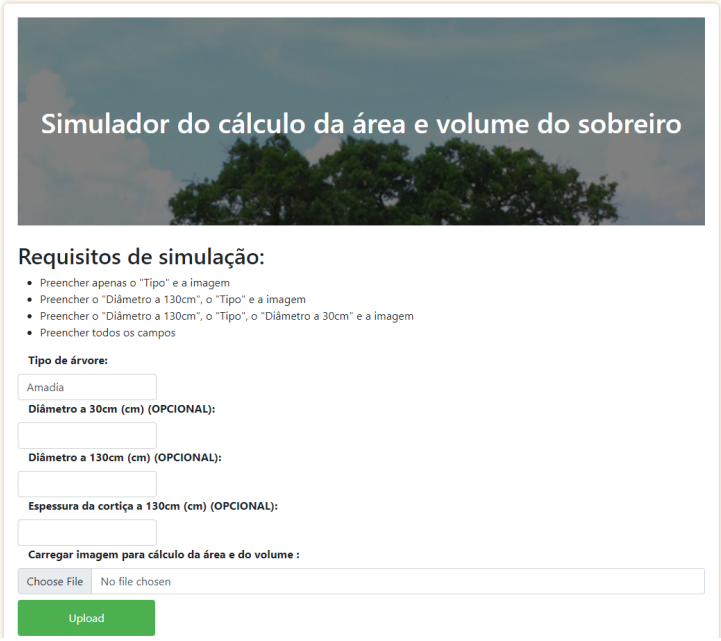

Figure 6.1: Data Input Form.

The volume estimation form incorporates four machine learning models, each designed to address specific scenarios and improve accuracy:

- Model 1 (First Experiment, 91.25% Accuracy): This model is utilized when the user uploads an image and fills in the cork oak type, providing a volume estimation based on these inputs. Figure 6.2 shows the filling of the fields *Tipo de árvore* (Amadia or Secundeira), which corresponds to the type of cork oak, and the uploading of an image of a cork oak with the targets attached to the trunk. The fields mentioned are mandatory and the others are optional, as their completion determines the choice of the volume prediction model.

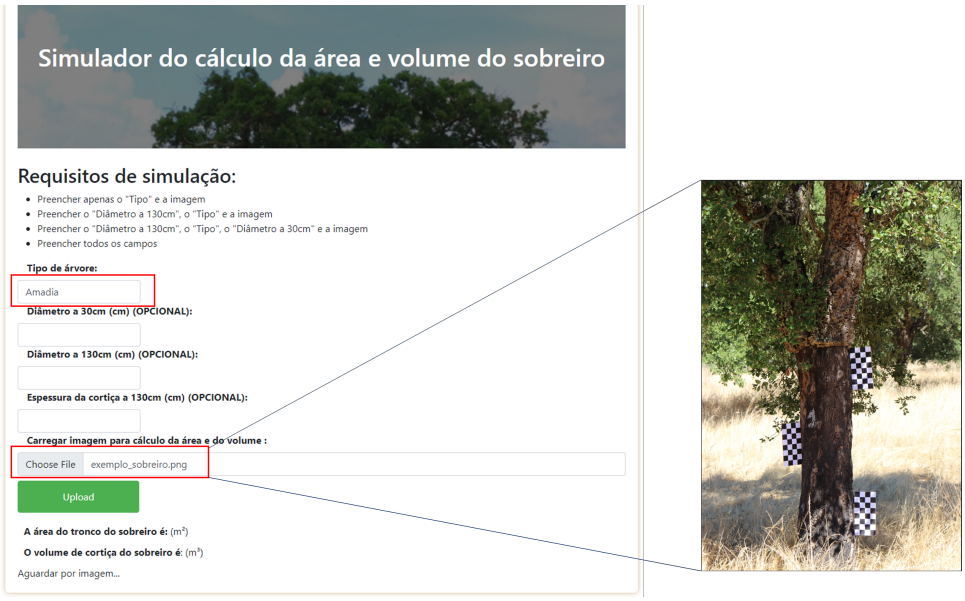

Figure 6.2: Volume Estimation Form for Model 1.

- Model 2 (Second Experiment, 97.04% Accuracy): In addition to the image and cork oak type, when the user provides the trunk diameter at 1.30 cm (diameter at breast height), Model 2 is employed to improve volume predictions. Figure 6.3 shows the completion of the previously mentioned fields with the addition of the field *Diâmetro a 130 cm*, which corresponds to the diameter at breast height.

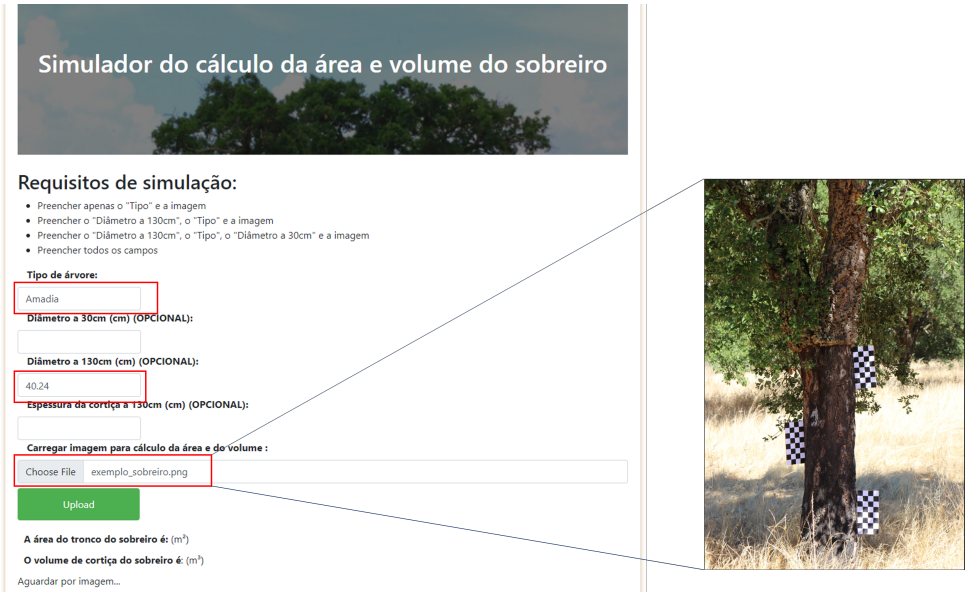

Figure 6.3: Volume Estimation Form for Model 2.

- Model 3 (Third Experiment, 97.26% Accuracy): When the user fills in the diameter of the trunk base along with the previously mentioned fields, Model 3, developed through the third experiment, is utilized for enhanced accuracy. Figure 6.4 depicts the integration of the previously discussed fields, accompanied by

the introduction of a new field called *Diâmetro a 30 cm* (Diameter at 30 cm). This field refers to the measurement of the trunk base diameter.

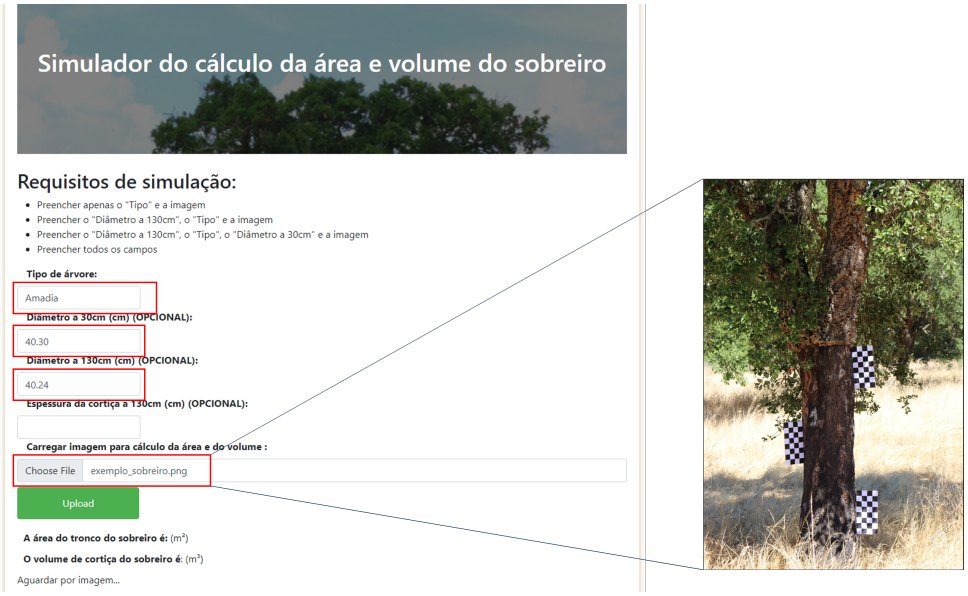

Figure 6.4: Volume Estimation Form for Model 3.

- Model 4 (Fourth Experiment, 99.85% Accuracy): For the most accurate volume predictions, users are encouraged to fill in all the fields in the form, including the thickness of the cork at 130 cm. Model 4, generated in the fourth experiment, demonstrates the highest accuracy. Figure 6.5 illustrates the inclusion of an additional field, namely *Espessura da cortiça a 130 cm* (Cork Thickness at 130 cm), alongside the previously mentioned fields. This particular field pertains to measuring the thickness of the cork at breast height.

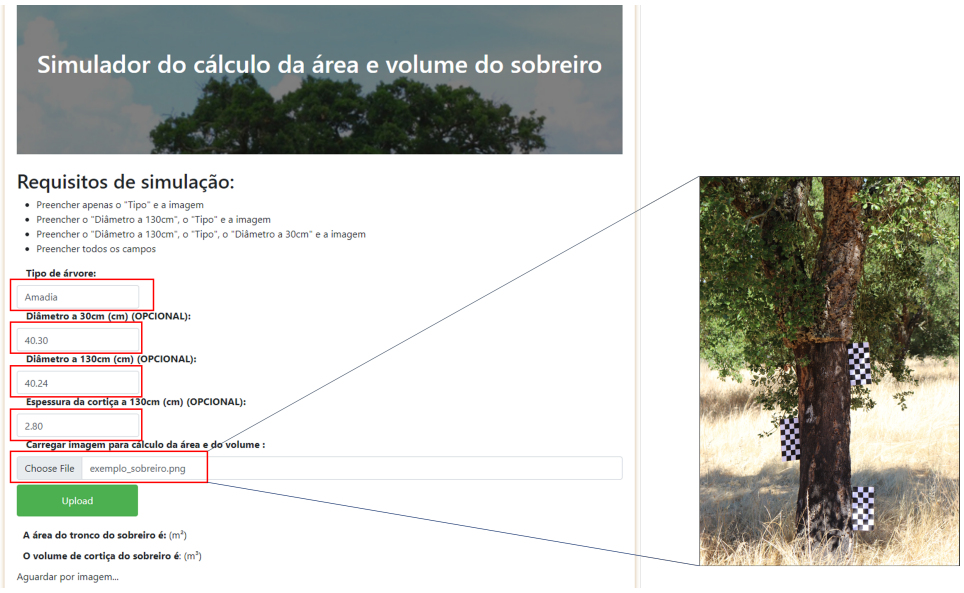

Figure 6.5: Volume Estimation Form for Model 4.

6.3 Results

To obtain accurate volume estimations, it is important for users to provide complete data in the volume estimation form. By including the image, cork oak type, trunk diameter at 1.30 cm, diameter of the trunk base, and thickness of the cork at 130 cm, users can maximize the accuracy of the volume predictions. Despite this, the model that receives only the type of tree (Amadia or Secundeira) and the image is able to make a volume prediction with an error of less than 10%, which is a very acceptable error. The great achievement of this project is to facilitate the calculation of cork volume, that is, the user will be able to use the approach mentioned above and use only one image to obtain the volume prediction, with an error similar to what is obtained from typical manual estimation.

An important note is that before the picture of the cork oak is taken, the targets have to be attached to the trunk in order to allow a correct measurement of the spot area generated by the Mask R-CNN.

After filling in the form fields, the user can click on the "Upload" button in order to submit the data for the volume estimation. Figure 6.6 illustrates an example of cork volume prediction, through the simplest model (Model 1), after data submission (in this case only the tree type and image).

Figure 6.6: Volume Estimation Output.

7 Discussion

In this chapter, we will revisit the research objectives, summarize the key findings, analyze their implications, and identify avenues for future research. By critically evaluating the limitations of the study and integrating the results with existing knowledge, this discussion aims to contribute to the field and provide a comprehensive conclusion to the research endeavor.

7.1 Overview

The project of this master thesis is integrated within the larger framework of the *Floresta Digital* project, which involves a partnership between ISEC and ESAC. The *Floresta Digital* project aims to advance the understanding and management of forest resources through the application of digital technologies.

The main objectives of this project were to train a model capable of recognizing and segmenting the area of the trunk of a cork oak before the stripping process, creating an area calculation process based on the trunk mask obtained, and developing a method for estimating the volume of cork using the trunk area and biometric data of the tree.

7.2 Advantages and Limitations

The successful achievement of the stated objectives in this thesis validates the effectiveness of the integration within the "Digital Forest" project. The developed model for trunk recognition and segmentation, along with the subsequent area calculation process and volume estimation method, demonstrate the feasibility and potential benefits of utilizing digital technologies for cork oak analysis. By leveraging the resources and collaborative opportunities within the *Floresta Digital* project, this master thesis contributes to the advancement of digital forest management practices and provides valuable insights for sustainable utilization of cork oak resources.

The training and performance evaluation of the Mask R-CNN model for recognizing and segmenting the trunk area of cork oak trees provide valuable insights into its capabilities and limitations. The outcomes of the evaluation reveal the model's effectiveness in achieving the main objective of accurately identifying and segmenting the trunk area before the stripping process.

The evaluation results underscore the importance of careful image selection and col-

lection to mitigate interference caused by other forms of vegetation and lighting conditions. Both excessive and insufficient light were identified as factors impacting the model's performance. Addressing these challenges is crucial to further enhance the model's accuracy and ensure reliable and accurate results.

The successful training and performance of the Mask R-CNN model validate its effectiveness in achieving the objectives set forth in this thesis. By accurately recognizing and segmenting the trunk area, the model contributes to advancing digital forest management practices. This application of deep learning techniques in the context of cork oak volume estimation demonstrates the potential for efficient and non-destructive analysis of forest resources.

The evaluation of machine learning models for volume estimation, specifically SVR, LinearRegressor, and MLPRegressor, yielded valuable insights into their performance and potential for practical application. While the SVR algorithm consistently outperformed other algorithms, the reliance on manual measurements of certain features poses challenges for real-world implementation.

Another limitation of accurately calculating the trunk area is the requirement of attaching targets to the tree trunks before capturing the images. This prerequisite poses a practical limitation as it involves additional manual efforts and may not be feasible for large-scale implementation. The dependency on attaching targets introduces an extra step and potential source of error, which can affect the efficiency and reliability of the trunk area calculation process. Finding alternative methods that eliminate the need for physical targets would be beneficial for streamlining the digital forest management practices and ensuring wider applicability of the developed model.

7.3 Main Contributions

In partnership with the Coimbra Agriculture School (ESAC), we created a comprehensive dataset specifically tailored for the task of segmenting the part of the trunk of a cork oak where the cork is extracted. The dataset consists of a collection of images showcasing cork oaks before undergoing the stripping process. The dataset was made available to the public on Kaggle, accessible at [`https://www.kaggle.com/datasets/andreguim/cork-oak-segmentation`], and was released on 25 October 2022.

The findings of this study on the application of the Mask R-CNN model for trunk area recognition and segmentation contribute to the existing literature by expanding upon previous research that utilized the Mask R-CNN model to detect fixed targets on tree trunks.

In the literature review, a study was identified that mentioned the use of the Mask R-CNN model; however, it focused on detecting targets that were fixed to the trunk rather

than specifically addressing trunk area recognition and segmentation [18]. This observation highlights the novel aspect of our study, as we have extended the application of the Mask R-CNN architecture to address the specific objectives of recognizing and segmenting the trunk area of cork oak trees.

While there may be limited direct comparisons with studies that specifically explore trunk area recognition and segmentation, our results align with the previous research on the effectiveness of the Mask R-CNN architecture in object detection and segmentation tasks.

Additionally, our research represents a significant milestone as we are the first, to the best of our knowledge, to develop a model capable of accurately estimating the volume of cork in the trunk of cork oak trees. Previous studies have primarily focused on estimating the volume of wood in trees, and the development of our model specifically targeting cork volume estimation sets our work apart in the field. This groundbreaking achievement expands the applications of the Mask R-CNN architecture and showcases its potential for precise volume estimation in unique contexts such as the cork oak industry.

The implementation of the cork volume prediction models on the website [`http://floresta.digital.esac.pt`], as a component of the *Floresta Digital* project, was a significant step forward in making the models practical and accessible for real-world usage.

Also as part of this project, the work and studies carried out resulted in two publications: "Cork Oak Production Estimation Using a Mask R-CNN", in May 2023, at CongrEGA 2022, the first National Congress in the field of Engineering and Asset Management [8], and "Cork Oak Production Estimation Using a Mask R-CNN" in the journal Energies MDPI, in December 2022 [9].

8 Conclusions and Future Work

This research proposes a novel approach utilizing the Mask R-CNN neural network to develop a model. The primary objective of this model is to identify and isolate the main portion of a cork oak tree known as the area of the cork extraction region, enabling the extraction of a mask specifically for this region. This extracted mask is subsequently employed to estimate the volume of cork expected to be produced by the cork oak.

The initial experimental outcomes reveal that the model demonstrates impressive performance in accurately segmenting the individual instances within the image. The best model achieved mAP@0.5 and mAP@0.7 scores exceeding 0.96, indicating its practicality and effectiveness. However, the mAP@0.9 score reached only 0.58, indicating the potential for further improvement in future iterations.

The inclusion of trunk images with diverse shapes in the dataset, which is a common occurrence in cork oak trunks, has the advantage of enhancing the generalizability of the model. Nevertheless, the dataset size remains relatively small, leaving room for enhancing the achieved results. Future endeavors will focus on augmenting the dataset by annotating and incorporating a larger number of instances.

The cork volume calculation was performed by employing machine learning models that received input from the extracted trunk mask, alongside the trees' biometric data and characteristics. Among the various algorithms tested, the Support Vector Regression (SVR) algorithm demonstrated superior performance for this task, exhibiting an average error of 0.15%. This remarkable accuracy renders the SVR algorithm highly suitable for implementation, especially when compared to alternative volume calculation methods documented in the existing body of research.

The objectives of the research were not only met but exceeded, leading to the creation of a dataset, two scientific publications, and the deployment of a functional model with an error rate below 10% even with only one image of the tree.

Future work should focus on investigating ways to improve the model's ability to accurately identify and segment instances of cork oak trunks, particularly when dealing with more challenging or complex shapes. This could involve exploring different data augmentation techniques or fine-tuning the existing model.

The research highlights the relatively small size of the dataset used for training the model. Future endeavors should prioritize dataset augmentation by annotating and incorporating a larger number of instances, particularly encompassing diverse shapes

and variations commonly observed in cork oak trunks. This would contribute to enhancing the generalizability and robustness of the model's performance across different scenarios.

While the research focuses on cork oak trees, the proposed approach and methodologies could be adapted and applied to other tree species as well. Future work should consider evaluating the generalizability and adaptability of the model to different types of trees. This would involve creating new datasets, modifying the network architecture if necessary, and exploring the performance of the volume estimation models on other tree species.

References

[1] ICNF, "IFN6—Áreas dos usos do solo e das Espécies Florestais de Portugal Continental; Resultados Preliminares," Instituto da Conservação da Natureza e das Florestas: Lisboa, Portugal, 2013, 33p, (Available only in Portuguese).

[2] APCOR. (2020) Cork Yearbook 2020. [Online]. Available: https://www.apcor.pt/en/portfolio-posts/apcor-year-book-2020/

[3] H. Pereira, *Cork: Biology, Production and Uses*. Amsterdam, The Netherlands: Elsevier, 2007.

[4] A. Kangas, R. Astrup, J. Breidenbach, J. Fridman, T. Gobakken, K. T. Korhonen, M. Maltamo, M. Nilsson, T. Nord-Larsen, E. Næsset *et al.*, "Remote sensing and forest inventories in Nordic countries—Road map for the future," *Scand. J. For. Res.*, vol. 33, pp. 397–412, 2018.

[5] ICNF, "IFN6—Relatório Final do Inventário Florestal Nacional Portugal Continental," Instituto da Conservação da Natureza e das Florestas: Lisboa, Portugal, 2015, (Available only in Portuguese).

[6] A. Van Laar and M. Alparslan, *Forest Mensuration*. Dordrecht, The Netherlands: Springer, 2007, vol. 13.

[7] M. I. Marzulli, P. Raumonen, R. Greco, M. Persia, and P. Tartarino, "Estimating tree stem diameters and volume from smartphone photogrammetric point clouds," *Forestry*, vol. 93, pp. 411–429, 2020.

[8] A. Guimarães, M. Valério, B. Fidalgo, R. Salas-Gonzalez, C. Pereira, and M. Mendes, "Estimativa da produção de cortiça usando mask r-cnn," in *1º Congresso Nacional de Engenharia e Gestão de Ativos - CongrEGA*, Coimbra, 2022.

[9] ——, "Cork oak production estimation using a mask r-cnn," *Energies*, vol. 15, no. 24, p. 9593, 2022. [Online]. Available: https://doi.org/10.3390/en15249593

[10] Ministry of Forests, Lands and NRO, "Smalian's Formula," *Paraboloid*, pp. 1–12, 2011.

[11] S. Hajar, M. Mushar, S. Sakinah, and S. Ahmad, "A Comparative study of log volume estimation by using statistical method," *EDUCATUM Journal of Science, Mathematics and Technology*, vol. 7, no. 1, pp. 22–28, 2020.

[12] G. C. de León and L. P. Uranga-Valencia, "Evaluación teórica de los métodos de Huber y Smalian aplicados a las geometrías clásicas de tronco de árbol," *Bosque*,

vol. 34, no. 3, pp. 311–317, 2013.

[13] J. Zhang and X. Y. Huang, "Measuring method of tree height based on digital image processing technology," *2009 1st International Conference on Information Science and Engineering, ICISE 2009*, no. 2006, pp. 1327–1331, 2009.

[14] D. Han and C. Wang, "Tree height measurement based on image processing embeded in smart mobile phone," *2011 International Conference on Multimedia Technology, ICMT 2011*, pp. 3293–3296, 2011.

[15] B. T. W. Putra, N. J. Ramadhani, D. W. Soedibyo, B. Marhaenanto, I. Indarto, and Y. Yualianto, "The use of computer vision to estimate tree diameter and circumference in homogeneous and production forests using a non-contact method," *Forest Science and Technology*, vol. 17, no. 1, pp. 32–38, 2021. [Online]. Available: https://doi.org/10.1080/21580103.2021.1873866

[16] D. Han, "Standing tree volume measurement technology based on digital image processing," in *International Conference on Automatic Control and Artificial Intelligence (ACAI 2012)*, 2012, pp. 1922–1925.

[17] J. Coelho, B. Fidalgo, M. M. Crisóstomo, R. Salas-González, A. P. Coimbra, and M. Mendes, "Non-destructive fast estimation of tree stem height and volume using image processing," *Symmetry*, vol. 13, no. 3, pp. 1–19, 2021.

[18] P. Juyal and S. Sharma, "Estimation of Tree Volume Using Mask R-CNN based Deep Learning," *2020 11th International Conference on Computing, Communication and Networking Technologies, ICCCNT 2020*, 2020.

[19] B. Sekachev, N. Manovich, M. Zhiltsov, A. Zhavoronkov, D. Kalinin, B. Hoff, TOsmanov, D. Kruchinin, A. Zankevich, D. Sidnev *et al.*, "opencv/cvat: V1.1.0," August 2020, accessed on 27 November 2022. [Online]. Available: https://doi.org/10.5281/zenodo.4009388

[20] K. Wada, "labelme: Image polygonal annotation with python," https://github.com/wkentaro/labelme, 2018.

[21] Tzutalin, "Labelimg," Free Software: MIT License, 2015. [Online]. Available: https://github.com/tzutalin/labelImg

[22] "Makesense.ai," https://www.makesense.ai/, accessed on 02 December 2022.

[23] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," *Advances in neural information processing systems*, vol. 25, pp. 1097–1105, 2012.

[24] Y. LeCun, L. Bottou, Y. Bengio, and P. Haffner, "Gradient-based learning applied to document recognition," *Proceedings of the IEEE*, vol. 86, no. 11, pp. 2278–2324, 1998.

[25] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, no. 7553, pp. 436–444, 2015.

[26] O. Russakovsky, J. Deng, H. Su, J. Krause, S. Satheesh, S. Ma, ..., and L. Fei-Fei, "Imagenet large scale visual recognition challenge," *International Journal of Computer Vision*, vol. 115, no. 3, pp. 211–252, 2015.

[27] K. He, X. Zhang, S. Ren, and J. Sun, "Deep residual learning for image recognition," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2016, pp. 770–778.

[28] N. Sharma, V. Jain, and A. Mishra, "An analysis of convolutional neural networks for image classification," *Procedia Computer Science*, vol. 132, pp. 377–384, 1 2018.

[29] R. Girshick, J. Donahue, T. Darrell, J. Malik, U. C. Berkeley, and J. Malik, "1043.0690," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 1, p. 5000, 9 2014. [Online]. Available: http://arxiv.

[30] J. R. Uijlings, K. E. V. D. Sande, T. Gevers, and A. W. Smeulders, "Selective search for object recognition," *International Journal of Computer Vision 2013 104:2*, vol. 104, pp. 154–171, 4 2013. [Online]. Available: https://link.springer.com/article/10.1007/s11263-013-0620-5

[31] R. Girshick, "Fast r-cnn." [Online]. Available: https://github.com/rbgirshick/

[32] R. Gavrilescu, C. Zet, C. Fosalau, M. Skoczylas, and D. Cotovanu, "Faster r-cnn:an approach to real-time object detection," *EPE 2018 - Proceedings of the 2018 10th International Conference and Expositions on Electrical And Power Engineering*, pp. 165–168, 12 2018.

[33] K. He, G. Gkioxari, P. Dollar, and R. Girshick, "Mask R-CNN," in *Proceedings of the IEEE International Conference on Computer Vision*, Venice, Italy, October 2017, pp. 2961–2969.

[34] O. Ronneberger, P. Fischer, and T. Brox, "U-net: Convolutional networks for biomedical image segmentation," in *International Conference on Medical Image Computing and Computer-Assisted Intervention*. Springer, 2015, pp. 234–241.

[35] Z. Zhou, M. M. R. Siddiquee, N. Tajbakhsh, and J. Liang, "Unet++: A nested u-net architecture for medical image segmentation," in *Deep Learning in Medical Image Analysis and Multimodal Learning for Clinical Decision Support*. Springer, 2018, pp. 3–11.

[36] J. Long, E. Shelhamer, and T. Darrell, "Fully convolutional networks for semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 3431–3440.

[37] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, "Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs," *IEEE transactions on pattern analysis and machine intelligence*, vol. 40, no. 4, pp. 834–848, 2018.

[38] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv preprint arXiv:1706.05587*, 2018.

[39] Z. Zhang, Y. Cao, J. Zhang, X. Hu, and J. Wang, "Forest tree trunk detection based on mask r-cnn," *Journal of Intelligent & Fuzzy Systems*, vol. 40, no. 2, pp. 2863–2872, 2021.

[40] X. Liu, X. Ma, Y. Li, and X. Zhao, "Tree trunk segmentation based on improved mask r-cnn algorithm," in *2020 IEEE International Conference on Power, Intelligent Computing and Systems (ICPICS)*. IEEE, 2020, pp. 235–238.

[41] M. Proesmans, L. Van Gool, and A. Oosterlinck, "Checkerboard coding for stereo vision," in *Proceedings of the 2nd European Conference on Computer Vision*. Springer, 1993, pp. 409–413.

[42] Z. Zhang, "A flexible new technique for camera calibration," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 22, no. 11, pp. 1330–1334, 2000.

[43] G. Ghiasi and C. Fowlkes, "Occlusion coherence: Detecting and localizing occluded faces," 06 2015.

[44] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results," Available online: http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html, 2012, accessed on May 16, 2022.

[45] H. Rezatofighi, N. Tsoi, J. Gwak, A. Sadeghian, I. Reid, and S. Savarese, "Generalized intersection over union: A metric and a loss for bounding box regression," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, Long Beach, CA, USA, June 2019, pp. 658–666.

Anexos

Appendix A - Estimativa da produção de cortiça usando Mask R-CNN

Estimativa da produção de cortiça usando Mask R-CNN

André Guimarães[1], Beatriz Fidalgo[2], Raúl Salas-Gonzalez[2], Carlos Pereira[1], Mateus Mendes[1]

1 Politécnico de Coimbra - ISEC, Coimbra, Portugal; a21240003@isec.pt, cpereira@isec.pt, mmendes@isec.pt

2 Politécnico de Coimbra - ESAC, Coimbra, Portugal; bfidalgo@esac.pt, rsalas@esac.pt

Resumo

Este trabalho visa automatizar o processo de cálculo da área do tronco de um sobreiro de onde é extraída cortiça. Através deste cálculo será possível estimar o volume de cortiça produzida, antes do processo de descortiçamento, utilizando uma rede neuronal profunda denominada Mask R-CNN.

Para o desenvolvimento do modelo neuronal foi criado um *dataset* de imagens de sobreiros, aos quais foram fixados nos troncos alvos de dimensões conhecidas. A Mask R-CNN foi treinada para reconhecer tanto os alvos como para criar a máscara do tronco do sobreiro. Assim, conhecido o tamanho dos alvos, é possível estimar a dimensão do tronco do sobreiro e consequentemente o volume de cortiça que poderá ser extraído. Os resultados preliminares obtidos comprovam que o modelo apresenta um bom desempenho no reconhecimento de alvos e troncos, registando um mAP de 0,96.

Palavras-Chave

Gestão florestal; *Quercus suber*; Volume de cortiça; Aprendizagem profunda; Mask R-CNN.

Introdução

O montado de sobro é um ativo de importância estratégica para Portugal. É formado por ecossistemas florestais de uso múltiplo em que o principal produto produzido é a cortiça. A cortiça tem um elevado valor económico devido às suas propriedades tecnológicas, nomeadamente de elasticidade, impermeabilidade e de bom isolamento térmico. De acordo com o último inventário florestal nacional português (IFN), o sobreiro ocupava em 2013, cerca de 23% do território florestal (ICNF, 2013). Portugal é atualmente o maior exportador mundial de cortiça, atingindo 62,4% do comércio mundial (APCOR, 2020). O sobreiro caracteriza-se pelo desenvolvimento de cortiça, ou seja, uma casca proeminente em camada contínua, que envolve todo o tronco e ramos da árvore. A extração da cortiça (descortiçamento) ocorre periodicamente, geralmente num ciclo de nove ou dez anos e não danifica a árvore. Posteriormente, uma nova casca começa a formar-se na superfície exposta do tronco (Pereira, 2007). A altura máxima de descortiçamento é estabelecida por lei e está relacionada com o valor do perímetro do tronco à altura do peito (PBH = 1,30 m), ou seja, a altura máxima de descortiçamento corresponde a três vezes o valor de PBH, aplicado exclusivamente em árvores com três extrações de cortiça (MADRP, 2001).

Para estimar a produção de cortiça o IFN em Portugal utiliza parcelas de campo circulares de 2000 m2 onde se realizam medições de diferentes características dendrométricas das árvores. Contudo, dada a intensidade de amostragem adoptada, os resultados do IFN só produzem resultados a nível nacional e regional não sendo suficientemente detalhados para orientar as decisões de gestão ao nível das florestas ou dos povoamentos (Kangas *et al.*, 2018). Consequentemente, ao nível da exploração florestal, é necessário realizar inventários de campo para estimar a produção de cortiça.

Portanto, os métodos tradicionais de inventário baseiam-se na medição em campo do diâmetro à altura do peito (DAP) e da altura do sobreiro. O DAP é medido usando uma sutaou fita diamétrica. O DAP é medido usando um paquímetro ou fita diamétrica. O hipsómetro é utilizado para medir a altura total da árvore, a altura da copa, a altura do tronco livre de galhos e a altura onde o tronco começa a ser bifurcado. O volume e a produção de cortiça são então estimados através de equações baseadas no DAP e na altura das árvores (ICNF, 2015). Em alternativa, pode ser utilizado outro método não destrutivo, o telerelascópio de Bitterlich, para estimar os diâmetros do tronco a várias alturas. Posteriormente, podem ser aplicadas as fórmulas de Smalian ou Newton para cada secção do tronco separadamente e assim, estimar com mais precisão o volume de cada secção (van Laar e Akça, 2007; West, 2009). No entanto, este método é muito trabalhoso e caro o que impede a sua ampla aplicação. Portanto, é necessário desenvolver novas metodologias de cálculo do volume de árvores em pé que sejam não destrutivas, precisas e que diminuam os custos de gestão florestal (Marzulli et al., 2020).

Com o avanço da tecnologia a deteção remota permite hoje obter já vários atributos das árvores e dos povoamentos florestais (Kangas et al., 2018). Estes métodos exigem menos tempo e facilitam o trabalho de medição. No entanto, o uso desta tecnologia continua limitado pelos custos envolvidos, pelas dificuldades de processamento de dados e pela falta de equipamentos e pessoal especializado (Marzulli et al., 2020).

Perante este problema, o presente trabalho visa desenvolver um método para estimar automaticamente o volume de um sobreiro utilizando a rede neuronal profunda Mask R-CNN. A Mask R-CNN é uma extensão de uma rede convolucional baseada em regiões (Faster R-CNN),

proposta anteriormente (Gavrilescu et al., 2018), que visa resolver tarefas de segmentação de instâncias.

O desempenho do modelo gerado na experiência foi analisado e avaliado de acordo com as métricas estabelecidas. Foi utilizado um *dataset* de imagens, recolhidas em campo, no treino dos modelos, sendo que este *dataset* contém um total de 50 imagens. Diferentes abordagens e técnicas foram usadas para extrair os melhores resultados.

O restante do artigo está organizado da seguinte forma. A secção 1 apresenta uma revisão de literatura, descrevendo alguns trabalhos realizados no âmbito do cálculo de volume utilizando métodos de processamento de imagem e aprendizagem profunda. A secção 2 apresenta a evolução das redes neurais R-CNN. A Secção 3 descreve a rede Mask R-CNN mais detalhadamente. A secção 4 descreve as características do *dataset* e as dificuldades encontradas na realização do trabalho. A secção 5 explica as técnicas de *data augmentation* utilizadas neste trabalho e a secção 6 aborda a metodologia de avaliação do modelo gerado durante a experiência, explicando as métricas que foram consideradas. Na secção 7 apresentam-se e discutem-se os resultados obtidos e na última secção extraem-se conclusões.

1. Revisão da Literatura

As subsecções seguintes descrevem trabalhos anteriores que utilizam métodos de visão computacional e métodos baseados em aprendizagem profunda para estimar parâmetros biométricos da árvore, capazes de obter resultados bastante rápidos.

1.1. Métodos de visão computacional

Zhang e Huang (2009) apresentam um método de medição da altura de uma árvore baseado em processamento de imagens. Neste método foram recolhidas imagens de árvores, tendo sido colocados três pontos vermelhos em cada árvore para servir de marcadores. Um dos pontos foi colocado na base da árvore, outro a um metro da base e o último à altura máxima possível da árvore. As fotografias foram tiradas perpendicularmente ao solo, formando um ângulo de 90 graus. Durante o processamento da imagem, foram extraídas as coordenadas dos três pontos de marcação. Para extrair as coordenadas do ponto de marcação superior, foi utilizado um modelo denominado modelo de cores HSI. Os autores propõem um método no qual a segmentação da imagem é realizada nos três componentes do modelo HSI, conseguindo segmentar a árvore de todos os demais objetos presentes no fundo da imagem. Após a segmentação, a imagem é convertida para um formato binário e as coordenadas são extraídas por meio de varrimento progressivo. Finalmente, a altura da árvore é calculada recorrendo à teoria da semelhança entre triângulos. Os resultados experimentais indicam que o erro de medição relativo correspondente à previsão da altura da árvore é de cerca de 4%.

Han (2012) propôs um método para estimar o volume de uma árvore recorrendo também ao processamento de imagem. Tal como o método proposto por Zhang e Huang (2009), foram colocados dois pontos de marcação vermelhos no tronco da árvore antes da fotografia ser tirada. Após a extração, tanto o tronco quanto os pontos de marcação, a linha exterior do tronco e o seu eixo central foram ajustados através da construção de uma curva, de modo que esta representasse um melhor ajuste aos dados recolhidos. O método proposto por este estudo mostrou-se viável, pois apresentou erros relativos de medição na ordem dos 5,4%.

No estudo de Coelho et al. (2021), foi estimado o volume de árvores de pinheiro bravo utilizando técnicas de visão computacional e fórmulas clássicas de determinação do volume. Foi determinada a área do tronco do pinheiro na imagem usando o algoritmo *Grab Cut*. Dentro dos métodos desenvolvidos para estimar a altura e o volume de uma árvore, os que apresentaram melhores resultados foram a utilização de relações hipsométricas para o cálculo da altura, e o método de Newton para o cálculo do volume da árvore tendo estes métodos apresentado erros médios de 12,18% e 10,90%, respetivamente. Ambos os métodos apresentaram erros semelhantes quando comparados aos métodos tradicionais.

1.2. Método baseado em Aprendizagem Profunda

No estudo de Putra et al. (2021) foi proposto um método para estimar o diâmetro e a altura de uma árvore usando redes neuronais Mask R-CNN com o objetivo de calcular a biomassa, um indicador chave dos processos de gestão ecológica e da vegetação. A rede Mask R-CNN foi utilizada para detetar a árvore e uma referência (quadrado retangular branco mantido paralelo à árvore na imagem). Como a referência detetada pela rede neuronal possui dimensões conhecidas, foi possível estimar a circunferência e a altura da árvore e, consequentemente, o seu volume. Para treinar e testar a rede neuronal, foram utilizadas imagens obtidas pelo Forest Research Institute em Dehradun. Não há indicação das espécies de árvores que foram utilizadas. O uso de redes neuronais convolucionais do tipo Mask R-CNN permitiu não só detetar o objeto na imagem, mas também produzir um resultado de alta qualidade.

Os resultados mostram um erro médio de 8% para deteção de diâmetro e 12% para deteção de altura.

1.3. Evolução das redes R-CNN

Uma Rede Neuronal Convolucional (CNN) é um modelo de aprendizagem profunda que possui a capacidade de receber uma imagem, atribuir uma importância (pesos e bias) a vários aspetos/características da imagem e ser capaz de a diferenciar. O pré-processamento necessário aquando da utilização deste tipo de redes é muito menor em comparação com outros algoritmos de classificação. Enquanto nos métodos primitivos, os filtros são artesanais, exigindo muito conhecimento prévio, as CNNs têm a capacidade de aprender de forma autónoma essas características (Sharma et al., 2018).

Uma CNN padrão não pode ser utilizada neste tipo de problemas porque a saída da rede é variável. Uma possível abordagem para resolver esse tipo de problema seria dividir a imagem em diferentes regiões de interesse e usar uma CNN para detetar a presença do objeto dentro de cada região. O problema deste tipo de abordagem é que os objetos que se pretendem identificar, podem estar em locais diferentes e ter aspetos diferentes, ou seja, para identificar corretamente os objetos na imagem, teria que ser selecionado um grande número de regiões de interesse, o que exigiria um poder de processamento muito elevado e inviável.

Para ultrapassar o problema de terem que ser identificadas muitas regiões de interesse para identificar objetos mais rapidamente e reduzir a capacidade de processamento necessária, foram propostas várias arquiteturas de redes neuronais direcionadas para a deteção de objetos. Uma delas foi denominada de R-CNN e foi proposta por Girshick et al. (2014). Essa rede neuronal utiliza um algoritmo denominado *selective search* (Uijlings et al., 2013) que possibilitou que, ao invés da

rede tentar classificar um grande número de regiões de interesse, esta passasse a classificar apenas 2.000 regiões.

Embora a R-CNN tenha resolvido parcialmente alguns dos problemas mencionados, persistem alguns obstáculos não totalmente mitigados. De facto, embora tenha sido benéfico reduzir o número de regiões de interesse a serem classificadas, continua ainda a ser necessário muito tempo de treino para poder classificar as 2.000 regiões. Outro fator negativo é o tempo necessário para realizar a deteção numa imagem, o que inviabiliza a sua implementação para respostas em tempo real. Outra desvantagem também identificada neste tipo de redes é a de que o algoritmo *selective search*, que seleciona regiões candidatas, não possui capacidade de aprendizagem, o que resulta numa identificação incorreta de algumas regiões candidatas.

Essas desvantagens conduziram a propostas de novas redes neuronais, baseadas na anterior R-CNN e seguindo uma abordagem semelhante, tendo, contudo, a vantagem de serem mais rápidas na deteção de objetos do que a rede proposta anteriormente. Essas redes neuronais são chamadas de Fast R-CNN (Girshick, 2015) e Faster R-CNN (Gavrilescu et al., 2018).

1.4. Mask R-CNN

Doll et al. (2017) propõem uma abordagem capaz de detetar objetos de forma eficiente numa imagem enquanto é gerada uma máscara de segmentação de alta qualidade para cada instância. O método proposto é denominado Mask R-CNN e é uma extensão do método proposto anteriormente denominado Faster R-CNN. Além da camada de reconhecimento da caixa delimitadora presente no modelo Faster R-CNN, foi adicionada outra camada convolucional capaz de prever simultaneamente a máscara do objeto. Esta abordagem revela-se eficaz, simples e sólida, facilitando a implementação de soluções para problemas de reconhecimento ao nível do pixel de uma imagem, algo que o modelo Faster R-CNN não consegue resolver.

O modelo Faster R-CNN possui duas saídas, nomeadamente, a classe à qual o objeto pertence e uma caixa delimitadora do mesmo objeto, enquanto no modelo proposto é adicionada uma terceira saída onde é obtida a máscara binária de cada região de interesse (*Region of Interest* - ROI). Assim, o modelo Mask R-CNN é uma arquitetura criada com o objetivo de integrar a deteção de objetos com a segmentação semântica. Essa integração é designada por segmentação de instâncias. Na prática, a arquitetura Faster R-CNN foi combinada com redes denominadas *Fully Convolutional Networks* (FCNs), criando-se assim a Mask R-CNN.

Como é necessária uma especificação do pixel da imagem, foi ainda necessário fazer alguns ajustes na arquitetura Faster R-CNN. Verificou-se que as regiões do *feature map* resultantes da camada de *ROI Pooling* apresentavam pequenos desvios em relação às regiões da imagem original. Devido a este problema, os autores introduziram o conceito de *ROI Align*.

Neste trabalho, o modelo foi implementado em Python 3.7.11, usando Keras 2.3.1 e Tensorflow 1.15.5.

2. Metodologia

Nesta secção apresenta-se a construção do *dataset*, incluindo as metodologias de *data augmentation* e métricas usadas para avaliação do modelo neuronal.

2.1. Dataset

O *dataset* é constituído por imagens de sobreiros obtidas antes das árvores sofrerem o processo de descortiçamento. Em cada tronco de árvore foram afixados dois ou três alvos, consoante a altura da árvore, que são detetados pela rede neuronal, para calibrar o cálculo da área do tronco segmentado. Foi utilizado um tripé para fixar a câmara para que as fotografias fossem tiradas perpendicularmente ao solo. No total, o conjunto de dados usados neste trabalho possui 55 imagens, uma por cada sobreiro. A Figura 1 apresenta a imagem de um sobreiro e respetivos alvos.

Figura 1 – Exemplo de uma imagem do dataset.

Este *dataset* abrange imagens de vários tipos de troncos característicos das árvores desta espécie, nomeadamente troncos mais retos, mais curvos e contendo ou não bifurcações. Apesar de possuir poucas instâncias, o *dataset* continua a ser útil para o treino de modelos baseados na rede Mask R-CNN, uma vez que esta rede permite a obtenção de resultados satisfatórios mesmo nesta circunstância.

Com o objetivo de treinar o modelo de forma supervisionada, anotaram-se as imagens de treino e teste, contornando o tronco e os respetivos alvos como ilustrado na Figura 2. Para tal, foi utilizada a ferramenta web MakeSense.AI. Além de permitir marcar as imagens, esta ferramenta permite a extração de ficheiros em formato COCO JSON, utilizado no treino de redes Mask R-CNN.

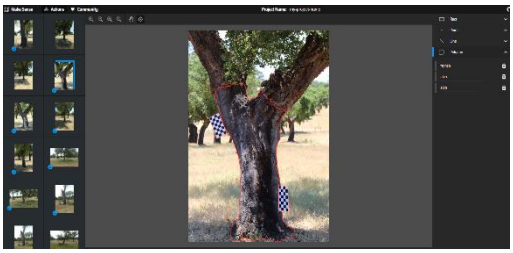

Figura 2 – Anotação de uma imagem utilizando a ferramenta MakeSense.AI.

Após a marcação, o *dataset* resultante foi dividido de forma aleatória em 80% para treino, 11% para validação e 9% para teste, sendo que 43 imagens pertenceram ao conjunto de treino, 7 ao conjunto de validação e 5 ao conjunto de teste. Para minimizar o problema de se trabalhar com *um dataset* de reduzida dimensão, foram aplicadas técnicas de *data augmentation*, que se descrevem em seguida.

2.2. **Data Augmentation**

Como mencionado anteriormente, a deteção de objetos recorrendo a algoritmos baseados em *deep learning* requer que seja utilizada uma grande quantidade de dados de forma a ser possível realizar um treino adequado. No entanto, o conjunto de dados aqui utilizado possui um tamanho reduzido. Para minimizar este problema, foram utilizadas técnicas de *data augmentation* com o objetivo de aumentar o número de instâncias de treino e validação.

O processo de *data augmentation* gera novas instâncias a partir dos dados originais, usando métodos de transformação tais como rotação, translação e redimensionamento (Roth et al., 2016), minimizando o problema de *overfitting*.

Em particular, as técnicas utilizadas foram: Rotação da imagem; Espelho da imagem; Translação; Ajuste de brilho. Estas técnicas foram utilizadas aquando do processo de treino, pois a implementação da rede Mask R-CNN assim o permite.

2.3. **Avaliação do Modelo**

Na segmentação de objetos são realizadas três tarefas distintas, uma para determinar se um objeto existe na imagem, outra para localizar o objeto na imagem e outra para desenhar uma máscara binária sobre o objeto. Adicionalmente, um *dataset* típico terá mais do que uma classe, cuja distribuição não é uniforme, o que acontece de facto nos dados utilizados neste estudo.

Como critério de avaliação dos modelos utilizou-se a Precisão Média (*Mean Average Precision - mAP*) (Everingham et al., 2012). O valor global é determinado pelo cálculo do mAP em todas as classes e em todos os limites de interseção sobre a união (*Intersection Over Union - IoU*) (Rezatofighi, 2019). A métrica IoU, também conhecida como índice Jaccard, permite quantificar a sobreposição percentual entre a máscara alvo e a máscara predita pela rede neuronal. Esta métrica está intimamente relacionada com o coeficiente *Dice*, que é frequentemente utilizado como função de *loss* durante o treino.

De forma simplificada, pode dizer-se que a métrica IoU mede o número de pixéis comuns entre as máscaras real e predita, dividido pelo número total de pixéis presentes em ambas as máscaras, e pode ser calculada pela seguinte fórmula:

$$IoU = \frac{MáscaraReal \cap MáscaraPredita}{MáscaraReal \cup MáscaraPredita}$$

A interseção (A∩B) é composta pelos pixéis encontrados tanto na máscara predita quanto na máscara real do objeto, enquanto a união (A∪B) é composta por todos os pixéis encontrados na máscara predita e na máscara real do objeto.

Na análise dos resultados são considerados três limites de mAP, o mAP@0,5, o mAP@0.7 e o mAP@0.9. O mAP@0.5 traduz o desempenho do modelo em relação à segmentação de objetos aquando da máscara predita se sobrepõe em pelo menos metade da máscara real. O mAP@0.7 mostra o desempenho do modelo em relação à segmentação de objetos quando a máscara predita

se sobrepõe em pelo menos 70% sobre a máscara real. O mAP@0.9 segue a mesma lógica dos anteriores, sendo que traduz o desempenho do modelo em relação à segmentação do objeto quando a máscara predita se sobrepõe em pelo menos 90% da máscara real.

Foram realizadas várias experiências, o que permitiu a avaliação dos diferentes modelos. As experiências tiveram como objetivo melhorar o desempenho do modelo, por meio de diferentes abordagens técnicas, nomeadamente: modificação de valores de determinados hiperparâmetros, aumento da quantidade de imagens do *dataset* e uso de algumas técnicas de *data augmentation.*

3. Resultados e Discussão

Nesta seção são descritas as configurações do treino do modelo que produziu os melhores resultados. São também apresentados os resultados dos treinos e testes, utilizando os modelos baseados na rede neuronal Mask R-CNN. São analisados os valores das métricas de avaliação.

3.1. Resultados do modelo baseado na Mask R-CNN

Para o treino foi utilizado um *dataset* que contém 50 imagens, que foram divididas em 43 para treino e 7 para validação.

As configurações que produziram melhores resultados são apresentadas na tabela 1.

Tabela 1 – Configurações da experiência.

Parametro	Valor
GPU_COUNT	1
IMAGES_PER_GPU	3
STEPS_PER_EPOCH	500
VALIDATION_STEPS	50
BACKBONE	Resnet101
TRAIN_ROIS_PER_IMAGE	70
NUM_CLASSES	3
RPN_ANCHOR_SCALES	(32, 64, 128, 256, 512)
RPN_NMS_THRESHOLD	0.7
RPN_TRAIN_ANCHORS_PER_IMAGE	256
IMAGE_MIN_DIM	1024
IMAGE_MAX_DIM	1024

De acordo com a documentação do Mask R-CNN, o parâmetro IMAGES_PER_GPU, usado para definir quantas imagens são treinadas em simultâneo por GPU, consome muita memória. Como a GPU utilizada para treinar a rede neuronal possui apenas 6 GB de memória dedicada, o valor utilizado nesta experiência foi de apenas uma imagem por GPU. Também se optou por manter os valores padrão de taxa de aprendizagem e *momentum*.

O parâmetro STEPS_PER_EPOCH define o número de etapas em cada época de treino. Embora o *dataset* contenha poucas imagens, esse parâmetro foi definido como 500 para evitar gastar muito tempo de treino nas atualizações realizadas pelo *TensorBoard*. O parâmetro VALIDATION_STEPS define o número de validações realizadas no final de cada época de treino, e foi escolhido o valor de 50.

O número máximo de regiões de interesse a serem consideradas nas camadas finais da rede neuronal é definido pelo parâmetro TRAIN_ROIS_PER_IMAGE, que foi parametrizado com o valor de 70. Este valor é relativamente baixo comparado com o definido no artigo de máscara RCNN

(512), porém, dado o alto consumo de memória necessário, decidiu-se realizar esta experiência utilizando o valor mencionado.

O *batch size* definido é obtido pela multiplicação dos parâmetros GPU_COUNT e IMAGES_PER_GPU, que no caso desta experiência resulta num valor de *batch size* igual a 3. Refira-se que neste projeto, são utilizados valores de *batch* baixos porque a memória da placa gráfica excede rapidamente o seu limite à medida que esse valor vai aumentando.

Como já foi referido, dado o tamanho reduzido do *dataset*, foram aplicadas três técnicas de *data augmentation* para tentar aumentar a quantidade de imagens de treino:

- Rotação da imagem entre -10º e +10º;
- Rotação da imagem entre -5º e +5º;
- Flip vertical;
- Translação de 10%;
- Ajuste de brilho.

Optou-se por definir que a aplicação das técnicas mencionadas acima seguia uma probabilidade de 83%, ou seja, cada imagem do *dataset* tem uma probabilidade de 83% de sofrer alguma modificação. Além disso, as técnicas passaram a ser aplicadas de forma aleatória e não sequencial, sendo que cada uma foi aplicada separadamente a cada imagem, utilizando a biblioteca imgaug. Imgaug é uma biblioteca que permite aumentar o número de imagens do *dataset*.

As Figuras 3, 4 e 5 ilustram a segmentação e classificação dos diferentes tipos de objetos, utilizando o modelo gerado nesta experiência.

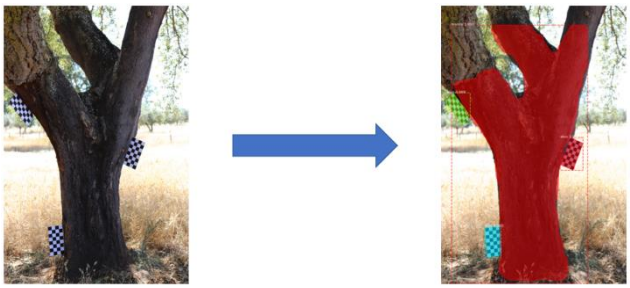

Figura 3 – Exemplo 1 de segmentação e classificação dos objetos.

A figura acima apresentada mostra um tronco bifurcado com três pernadas, situação muito comum nos montados de sobro em Portugal. A sua análise mostra como foi possível identificar claramente a área do troco já descortiçada e que irá ser descortiçada novamente. A imagem mostra alguns problemas de classificação na zona superior, na pernada localizada à direita onde a linha que separa a zona descortiçada e não descortiçada não é claramente visível.

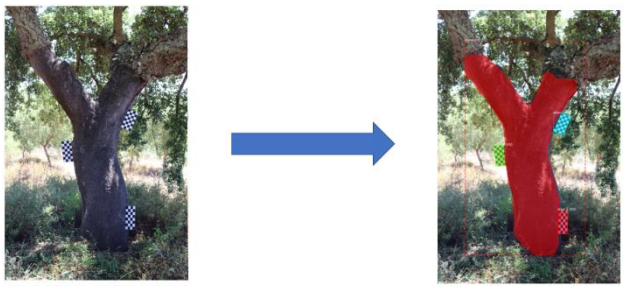

Figura 4 – Exemplo 2 de segmentação e classificação dos objetos.

O Exemplo 2, apresentado na Figura 4 mostra um sobreiro apenas com duas pernadas e a identificação da área descortiçada quase perfeita. Continuam alguns pequenos problemas na identificação da linha de descortiçamento, e na base do tronco, onde a vegetação natural se apresenta mais desenvolvida e dificulta a visibilidade do tronco.

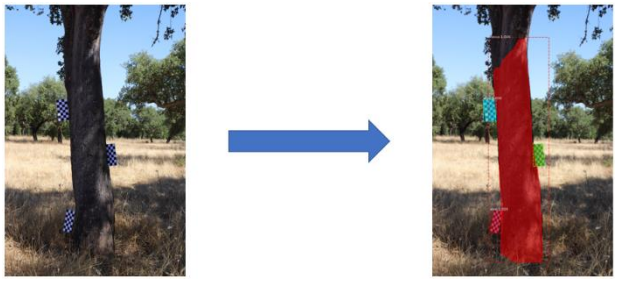

Figura 5 – Exemplo 3 de segmentação e classificação dos objetos.

O Exemplo número três é de todos aquele que apresenta um resultado menos favorável, uma vez que claramente não é classificada uma zona considerável na parte superior do tronco. Este resultado pode ser consequência do facto de a imagem ter sido tirada em contraluz, deixando manchas de diferentes luminosidades no tronco. Em contrapartida a base da árvore aparece bastante bem segmentada.

Com base nas imagens acima, é possível verificar o bom desempenho do modelo.

Após a fase de treino, o modelo foi avaliado de acordo com as métricas mencionadas. Os resultados obtidos são apresentados na Tabela 2.

Table 2 – Resultados de desempenho da experiência.

mAP@0.5	mAP@0.7	mAP@0.9
0.964	0.964	0.577

Os resultados permitem concluir que o modelo apresenta bom desempenho, pois o seu mAP@0,7 está acima de 96%.

Os efeitos do aumento do *dataset*, da mudança de configurações e do uso de técnicas *data augmentation* foram visíveis, uma vez que, em experiências anteriores, os resultados não se mostraram muito satisfatórios. Os valores de mAP@0,5 e mAP0,7 tornam este modelo viável para ser utilizado no cálculo de volume. No entanto, existe algum espaço para progressão, pois o mAP@0,9 pode ser melhorado.

Os resultados mostram também a necessidade de cuidar a colheita das imagens por forma a minimizar interferências de outra vegetação e do efeito de luz e contraluz.

Conclusão e Trabalho Futuro

Este estudo propõe um modelo baseado na rede neuronal Mask R-CNN. O modelo visa detetar e segmentar o tronco de um sobreiro, possibilitando extrair a máscara deste. A máscara extraída será posteriormente utilizada para estimar o volume de cortiça que o sobreiro poderá produzir.

Os resultados experimentais preliminares demonstram que o modelo apresenta um bom desempenho na segmentação das instâncias na imagem. Os mAP@0,5 e mAP@0,7 do melhor modelo apresenta um valor superior a 0,96, o que torna viável a utilização do mesmo. O mAP@0,9 registou um valor de 0,58, o que demonstra que ainda há espaço para evolução futura.

O facto de o *dataset* possuir imagens de troncos de diversas formas, algo que acontece com frequência em troncos de sobreiros, pode ser benéfico para aumentar a capacidade de generalização do modelo. No entanto o tamanho do *dataset* é ainda reduzido, existindo assim margem para melhorar os resultados obtidos, sendo em trabalhos futuros pretende-se anotar e incluir um maior número de instâncias.

Em trabalho futuro, será calculado o volume de cortiça, utilizando a máscara do tronco que foi extraída e os dados biométricos das árvores. Para isso, será treinado um algoritmo de *machine learning,* que irá receber os dados biométricos das árvores e os dados relativos à máscara do tronco.

Agradecimentos

Os autores gostariam de agradecer à Herdade do Mouchão por permitir a recolha de fotografias e medições de sobreiros da Herdade, dados esses que foram utilizados no âmbito do presente trabalho.

Referências Bibliográficas

APCOR. Cork yearbook 2020. Available at: capa_web_v3 (apcor.pt). [accessed 04.04.2022].

Coelho, J.; Fidalgo, B.; Crisóstomo, M..M.; Salas-González, R.; Coimbra, A.P.; Mendes, M. Non-Destructive Fast Estimation of Tree Stem Height and Volume Using Image Processing. Symmetry 2021, 13, 374. https://doi.org/10.3390/sym13030374

Gavrilescu, R., Zet, C., Foşalău, C., Skoczylas, M., and Cotovanu, D. (2018). Faster r-cnn: an approach to real-time object detection. In 2018 International Conference and Exposition on Electrical And Power Engineering (EPE), pages 0165–0168.

Girshick, R. (2015). Fast r-cnn. In Proceedings of the IEEE International Conference on Computer Vision (ICCV).

Girshick, R., Donahue, J., Darrell, T., and Malik, J. (2014). Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR).

Han, D. (2012). Standing tree volume measurement technology based on digital image processing. In International Conference on Automatic Control and Artificial Intelligence (ACAI 2012), pages 1922–1925.

He, K., Gkioxari, G., Dollar, P., and Girshick, R. (2017). Mask r-cnn. In Proceedings of the IEEE International Conference on Computer Vision (ICCV).

ICNF., IFN6—Áreas dos usos do solo e das espécies florestais de Portugal continental. Resultados preliminares, Instituto da Conservação da Natureza e das Florestas. Lisboa, 33 pp, 2013, (available only in Portuguese).

ICNF. IFN6 - Relatório final do Inventário Florestal Nacional Portugal Continental, Instituto da Conservação da Natureza e das Florestas. Lisboa, 2015, (available only in Portuguese).

Kangas, A.; Astrup, R.; Breidenbach, J.; Fridman, J.; Gobakken, T.; Korhonen, K.T.; Maltamo, M.; Nilsson, M.; Nord-Larsen, T.; Næsset, E.; et al. Remote sensing and forest inventories in Nordic countries—Road map for the future. Scand. J. For. Res. 2018; 33, 397–412.

Litjens, G., Kooi, T., Bejnordi, B. E., Setio, A. A. A., Ciompi, F., Ghafoorian, M., van der Laak, J. A., van Ginneken, B., and Sánchez, C. I. (2017). A survey on deep learning in medical image analysis. Medical Image Analysis, 42(December 2012):60–88

Marzulli M. I., Raumonen P., Greco R., Persia M., and Tartarino P., Estimating tree stem diameters and volume from smartphone photogrammetric point clouds, Forestry 2020; 93, 411–429, doi:10.1093/forestry/cpz067

Ministério da Agricultura, do Desenvolvimento Rural e das Pescas (MADRP)., Diário da República de Portugal— I Série - A., Decreto-Lei n.º 169/2001 de 25 de Maio, (available only in Portuguese).

M. Everingham, L. V. Gool, C. Williams, J. Winn and Zisserman, A. (2012). The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. 2012:1– 32.

Pereira, H., Cork: biology, production and uses. Elsevier Publications, Amsterdam. https //doi.org/10.1016/B978-044452967-1/50013-3, 2007.

Putra, B. T. W., Ramadhani, N. J., Soedibyo, D. W., Marhaenanto, B., Indarto, I., and Yualianto, Y. (2021). The use of computer vision to estimate tree diameter and circumference in homogeneous and production forests using a non-contact method. Forest Science and Technology, 17(1):32–38.

Rezatofighi, H., Tsoi, N., Gwak, J., Sadeghian, A., Reid, I., and Savarese, S. (2019). Generalized intersection over union: A metric and a loss for bounding box regression. Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, 2019-June:658–666.

Roth, H. R., Lu, L., Liu, J., Yao, J., Seff, A., Cherry, K., Kim, L., and Summers, R. M. (2016). Improving Computer-Aided Detection Using Convolutional Neuronal Networks and Random View Aggregation. IEEE Transactions on Medical Imaging, 35(5):1170–1181.

Sharma, N., Jain, V., and Mishra, A. (2018). An Analysis of Convolutional Neuronal Networks for Image Classification. Procedia Computer Science, 132(Iccids):377–384.

Uijlings, J. R., Van De Sande, K. E., Gevers, T., and Smeulders, A. W. (2013). Selective search for object recognition. International Journal of Computer Vision, 104(2):154–171.

Van Laar A., & A. Alparslan., Forest Mensuration, Volume 13, Springer Dordrecht, The Netherlands, 2007.

West, P.W., Tree and Forest Measurement, 2nd edition, DOI: 10.1007/978-3-540-95966-3_5, © Springer-Verlag Berlin Heidelberg, 2009.

Zhang, J. and Huang, X. Y. (2009). Measuring method of tree height based on digital image processing technology. 2009 1st International Conference on Information Science and Engineering, ICISE 2009, (2006):1327–1331.

Appendix B - Cork Oak Production Estimation Using a Mask R-CNN

energies

MDPI

Article

Cork Oak Production Estimation Using a Mask R-CNN

André Guimarães [1,*], Maria Valério [2], Beatriz Fidalgo [2], Raúl Salas-Gonzalez [2], Carlos Pereira [1,3] and Mateus Mendes [1,4,*]

[1] Instituto Superior de Engenharia de Coimbra, Polytechnic of Coimbra—ISEC, 3030-199 Coimbra, Portugal
[2] School of Agriculture of Coimbra, Polytechnic of Coimbra—ESAC, 3045-093 Coimbra, Portugal
[3] Departamento de Eng. Informática, CISUC—Centre for Informatics and Systems of the University of Coimbra, Pólo II, Rua Sílvio Lima, 3030-290 Coimbra, Portugal
[4] Departamento de Eng. Eletrotécnica e Computadores, ISR—Institute of Systems and Robotics of the University of Coimbra, Pólo II, Rua Sílvio Lima, University of Coimbra, 3030-194 Coimbra, Portugal
* Correspondence: a21240003@isec.pt (A.G.); mmendes@isec.pt (M.M.)

Abstract: Cork is a versatile natural material. It can be used as an insulator in construction, among many other applications. For good forest management of cork oaks, forest owners need to calculate the volume of cork periodically. This will allow them to choose the right time to harvest the cork. The traditional method is laborious and time consuming. The present work aims to automate the process of calculating the trunk area of a cork oak from which cork is extracted. Through this calculation, it will be possible to estimate the volume of cork produced before the stripping process. A deep neural network, Mask R-CNN, and a machine learning algorithm are used. A dataset of images of cork oaks was created, where targets of known dimensions were fixed on the trunks. The Mask R-CNN was trained to recognize targets cork regions, and so the area of cork was estimated based on the target dimensions. Preliminary results show that the model presents a good performance in the recognition of targets and trunks, registering a mAP@0.7 of 0.96. After obtaining the mask results, three machine learning models were trained to estimate the cork volume based on the area and biometric parameters of the tree. The results showed that a support vector machine produced an average error of 8.75%, which is within the error margins obtained using traditional methods.

Keywords: forest management; *Quercus suber*; cork volume; machine learning; mask R-CNN

Citation: Guimarães, A.; Valério, M.; Fidalgo, B.; Salas-Gonzalez, R.; Pereira, C.; Mendes, M. Cork Oak Production Estimation Using a Mask R-CNN. *Energies* **2022**, *15*, 9593. https://doi.org/10.3390/en15249593

Academic Editor: Paul Stewart

Received: 31 October 2022
Accepted: 14 December 2022
Published: 17 December 2022

1. Introduction

The cork oak forest is an asset of strategic importance for Portugal. It is formed by multiple-use forest ecosystems in which the main product is cork. Cork has a high economic value due to its technological properties, namely elasticity, impermeability, and good thermal insulation. Due to its low thermal conductivity and impermeability, it has been used in construction since pre-historical times. Nowadays, it is transformed and used in a variety of construction materials such as blocks, cork rubber or insulation sheets. It can also be transformed into grains and incorporated into filling materials. It can be used indoors and outdoors, in floors, walls or under tiles as a roof insulator. It can also be transformed and incorporated into paint, thus improving the paint's thermal properties. Cork materials in general have service life of over 50 years.

According to the last Portuguese national forest inventory (NFI), in 2013 the cork oak occupied approximately 23% of the forest territory [1]. Portugal is currently the largest cork exporter in the world, controlling a 62.4% share of the world trade in cork [2]. Cork oak is characterized by the development of cork, that is, a prominent bark in a continuous layer which involves the entire trunk and branches of the tree. Cork extraction (stripping) occurs periodically, usually on a nine- or ten-year cycle, and does not damage the tree. Afterwards, a new bark begins to form on the exposed stem surface [3]. The maximum height of cork stripping is established by law and is determined by the cork harvesting coefficient (HC),

which must not exceed the value of 3.0 and is defined as the ratio between stem harvesting height (H) and stem perimeter over cork, at breast height [4].

To estimate cork production, the IFN in Portugal uses circular field plots of 2000 m^2, where measurements of different dendrometric characteristics of the trees are carried out. However, given the sampling intensity adopted, the IFN results only produce results at the national and regional level and are not sufficiently detailed to guide management decisions at the forest or stand level [5]. Consequently, at the level of forest exploitation, new stand field inventories must be made to estimate cork oak production.

Traditional inventory methods are based on field measurements of diameter at breast height (DBH) and height of the cork oak. DBH is measured using a caliper or diameter tape. The hypsometer is used for measuring total tree height, crown height, the stem height free from branches and the height where stem begins to fork. The volume and cork production are then estimated using equations based on the DBH and tree height [6]. Alternatively, another non-destructive method, the Bitterlich relascope, can be used to estimate trunk diameters at various heights. Subsequently, the Smalian or Newton formulae can be applied to each section of the trunk separately to estimate more precisely the volume of each section [7,8]. However, this method is very laborious and expensive, issues which prevent its wider application. Therefore, it is necessary to develop new methodologies for calculating the volume of standing trees that are non-destructive, accurate and reduce forest management costs [9].

With the advancement of technology, remote sensing allows us to obtain several attributes of trees and forest stands [5]. These methods require less time and make measurement work easier. Nevertheless, the use of this technology remains limited by the costs involved, the difficulties of data processing, the lack of equipment and the requirement of specialized personnel [9].

Faced with this problem, the present work aims to propose a method to automatically estimate the volume of cork in a cork oak tree, using a Mask R-CNN deep neural network to find the area of interest and another machine learning algorithm to predict the volume of oak from the area and other data. The Mask R-CNN is an extension of the previously proposed Faster Region-Based Convolutional Network (Faster R-CNN) [10], which solves instance segmentation tasks.

A dataset of images collected in the field was used in the training process, containing a total of 62 images.

The article is organized as follows. Section 2 presents a literature review, describing some works carried out in the scope of volume calculation using image processing and deep learning methods. Section 3 presents the evolution of R-CNN neural networks. Section 4 describes the Mask R-CNN architecture in more detail. Section 5 describes the characteristics of the dataset and the difficulties encountered and addresses the methodology followed in the implementation of the deep learning model. Section 6 presents the obtained results of the Mask R-CNN model. Section 7 describes the methodology for calculating the trunk area after obtaining the mask from the output of Mask R-CN. Section 8 shows the results obtained with three different machine learning algorithms to estimate the final volume of cork. The last section presents the conclusions drawn from this study, as well as the analysis through the results obtained in the experiment.

2. Literature Review

The following subsections describe previous work based on computer vision methods and on deep learning to estimate the biometric parameters of the tree. The techniques used are capable of obtaining accurate results.

2.1. Traditional Methods

Depending on the degree of detail required in measuring the volume of a tree, it may be necessary to calculate only the volume of the trunk or to include the tree branches. In

this project, the focus will be only on the trunk volume since it is in this section of the tree that the cork is extracted.

Currently, there is a significant variety of techniques that are used to measure the volume of a tree. Considering that the trunk does not have the same diameter along its height, regardless of technique or method used, it is always necessary to segment the trunk into different sections. The volume is calculated for each section, and the partial volumes of each section will be considered to determine the total volume of the trunk.

Trunk volume can be obtained through direct measurement by a destructive method, where it is necessary to cut the tree and measure the diameter along the trunk using a measuring tape [11], or by remote measurement methods, performed with the support of professional measuring equipment that allows us to carry out a ground-level measurement. With the advancement of technology, the use of remote measurement methods allows us to spend less time measuring and make the measurement work easier, although the acquisition of this type of equipment can be very expensive.

Regarding common measuring instruments, hypsometers are the most used when the objective is to measure the height of a tree. Hypsometers can measure the height of the tree with high precision considering trigonometry and atmospheric pressure. It is possible to estimate the volume of a tree using binary equations based on DBH (diameter at breast height) and the height calculated by the hypsometer. These binary equations are called hypsometric relations. The hypsometric relations represent the height-diameter relationship of the tree, using data obtained from a set of trees. Through these data, the relationship between the DBH and the height of the tree is established, and a function is calculated to represent this relationship.

2.2. Computer Vision Methods

Zhang and Huang [12] present a method of measuring the height of a tree based on image processing. In this method, images of trees were collected, and three red dots were placed on each tree to serve as markers. One of the points was placed at the base of the tree, another one a meter from the base and the last one at the maximum possible height of the tree. The photographs were taken perpendicular to the ground, forming a 90-degree angle. During image processing, the coordinates of the three marking points were extracted. In order to extract the coordinates of the upper marking point, a model called HSI color model was used. The authors propose a method in which the image segmentation is performed on the three components of the HSI model, managing to segment the tree of all other objects present in the background of the image. After segmentation, the image is converted into a binary format and the coordinates are extracted through progressive scanning. Finally, the height of the tree is calculated using the theory of similarity between triangles. Experimental results indicate that the relative measurement error corresponding to the tree height prediction is about 4%.

Han [13] proposed a method to estimate the volume of a tree also using image processing, such as the method proposed by Zhang and Huang [12]. Two red marking points were placed on the tree trunk before the photograph was taken. After extraction, both the trunk and the marking points, the edge and the central axis of the trunk, were adjusted through the construction of a curve, so that it represented a better fit to the collected data. The method proposed by this study proved to be viable since it presented relative measurement errors in the order of 5.4%.

Another approach to calculating the volume was proposed by Putra et al. [14]. Here, the use of optical sensors, namely a smartphone camera, was evaluated. They also used image processing methods to estimate the circumference of trees in homogeneous and productive forests, especially seringueira and albizia plantations, with a real-time measurement approach. The images were captured at approximately one meter from the tree and with the camera pointed, parallel to the ground, at the trunk area, which allows us to calculating the diameter at breast height. Measurements performed using the camera showed acceptable accuracy, with a coefficient of determination of 95% and an RMSE of

approximately 7.9 cm. These correspond to a relative measurement error of about 9.4%. Despite the accuracy, this method is only applicable to trees with relatively circular shapes, and there are also several aspects that affect measurement errors such as the presence of inclined trees, irregular geometric shapes and the computer vision segmentation methods themselves, which sometimes are not the most suitable.

In the study by Coelho et al. [15], the volume of Corsican pine trees was estimated using computer vision techniques and classical formulae for volume determination. The area of the pine tree in the image was determined using a Grab Cut algorithm. Among the methods developed to estimate the height and volume of a tree, the ones that presented the best results were the use of hypsometric relations to calculate the height, and the Newton method to calculate the volume of the tree. These methods showed average errors of 12.18% and 10.90%, respectively.

This last study presents a viable volume calculation method for use as it shows similar errors when compared with traditional methods, allowing the reduction of time spent in the field and the high associated costs. The error of this method serves as a reference for this study.

Several studies have been carried out using LiDAR data to estimate measurements of trees, such as the DBH, height and volume. The methods used vary, but the more common ones use tree reconstruction or automatic methods to detect the tree and estimate its parameters.

Gonzalez et al. [16] used this method when scanning 29 trees (the specimen was not named, only that 9 were from Peru, 10 from Indonesia and 10 from Guyana). The scan was performed using a RIEGL VZ-400 3D terrestrial laser scanner. This scanner has a horizontal field of view of 360º and a vertical one of 100º, with a scan resolution of 0.06º. According to the authors, this method showed an RMSE of 3.29 m^3 when estimating the volume of the referred trees.

Heurich M. [17] tested the use of airborne LiDAR for delineating individual trees. Airborne LiDAR are LiDAR systems mounted on vehicles such as airplanes, helicopters, and drones. These systems have been more commonly used for large-area retrieval of forest structural parameters due to the benefits of cost-efficiency. In this study, a cohort of 2584 trees, comprised of *Picea abies* and *Fagus sylvatica*, were scanned, and 76.9% of the trees could be recognized. Then, regression equations were used to determinate tree height, DBH and volume of single trees, resulting in a coefficient of determination (R^2) of 0.97 for tree height and of 0.90 for volume.

Although LiDAR techniques show relatively good results, they are expensive and time consuming.

2.3. Deep Learning Methods

In the study by Juyal et al. [18] a method was proposed. The aim was to estimate the diameter and height of a tree using Mask R-CNN neural networks, and to calculate biomass, a key indicator of ecological and vegetation management processes. The Mask R-CNN network was used to detect the tree and a reference (white rectangular square held parallel to the tree in the image). As the reference detected by the neural network has known dimensions, it was possible to estimate the circumference and height of the tree and, consequently, its volume. To train and test the neural network, images obtained by the Forest Research Institute in Dehradun were used. There is no indication of which species of trees were used. The use of Convolutional Neural Networks of the Mask R-CNN type allowed us not only to detect the object in the image, but also to produce a high-quality result and, therefore, to obtain a better understanding at the pixel level and thus to delimit the object with high accuracy.

The results show an average error of 8% for diameter detection and 12% for height detection.

This method is in part similar to the present approach. However, the present approach is applied to cork oak trees and aims to estimate the area of the trunk and the volume of cork.

3. Evolution of R-CNN

A Convolutional Neural Network (CNN) is a deep learning model with the ability to receive an input image, assign an importance (learning weights and biases) to various features/objects of the image and then able to differentiate them from each other. The pre-processing required in the deep neural network ConvNet is much less than that in other classification algorithms. While filters in primitive methods are handmade, requiring a lot of prior knowledge, CNNs have the ability to autonomously learn these characteristics [19].

A standard CNN cannot be used in this type of problem because the network output is variable. A possible approach to solve this type of problem would be to divide the image into different regions of interest and use a CNN to detect the presence of the object within each region. The problem with this type of approach is that the objects that are intended to be identified may be in different locations and have different aspects. That is, to correctly identify the objects in the image, a large number of regions of interest would have to be selected, something which would require a high and not viable processing power.

In order to overcome the problem of having to identify many regions of interest, with the aim of identifying objects quicker and reducing the necessary processing capacity, several neural network architectures have been proposed to aim at object detection. One of them is called R-CNN and was proposed by Girshick et al. [20]. This neural network uses an algorithm called selective search [21]. This can, instead of trying to classify a huge number of regions of interest, start by classifying just 2000 regions.

Even if R-CNN has partially solved some of the issues mentioned above, there are still some problems that this neural network was not able to fully mitigate. In fact, although it has been beneficial to reduce the number of regions of interest to be classified, it still takes a lot of training time to be able to classify the 2000 regions. Another negative factor is the amount of time needed to test an image, which makes the method unfeasible to implement for a real-time response. Another disadvantage also identified in this type of network is that the selective search algorithm, which selects candidate regions, does not have the ability to learn, which results in an incorrect identification of some candidate regions.

These disadvantages led to the proposal of new neural networks, based on the previous R-CNN and following a similar approach, having the advantage of being faster in object detection. These new networks are called Fast R-CNN [22] and Faster R-CNN [10].

4. Mask R-CNN

In the study published by Kaiming He et al. [23], an approach is proposed which is capable of efficiently detecting objects in an image while also generating a high-quality segmentation mask for each instance. The proposed method is called Mask R-CNN and it is an extension of the previously proposed method called Faster R-CNN. In addition to the bounding box recognition layer present in the Faster R-CNN model, another convolutional layer, capable of simultaneously predicting the object mask, was added. This approach proves to be effective, simple, and solid, facilitating the implementation of solutions for recognition problems at the pixel level of an image, something that the Faster R-CNN model is unable to solve.

The Faster R-CNN model has two outputs, the class to which the object belongs and a bounding box of the same object. Conversely, in the proposed model a third output, was added where the binary mask of each region of interest is obtained (Region of Interest—ROI). Thus, the Mask R-CNN model is an architecture created with the aim of integrating object detection with semantic segmentation. In practice, the Faster R-CNN architecture was combined with networks called Fully Convolutional Networks (FCNs), creating the Mask R-CNN.

Since an image pixel specification is required, it was necessary to make some adjustments to the Faster R-CNN architecture. It was verified that the regions of the feature map, resulting from the ROI Pooling layer, presented slight deviations from the regions of the original image. Due to this problem, the authors introduced the concept of ROI Align.

In the present work, the Mask R-CNN model was implemented in Python 3.7.11, using Keras 2.3.1 and Tensorflow 1.15.5.

5. Methodology for Mask R-CNN Model

This section describes the construction of the dataset, including data augmentation methodologies and the metrics used to evaluate the neural model.

The objective of this work is to create a model that follows a three-step approach in order to be able to predict the cork volume of a cork oak. The first stage, described in this section, is to create a deep learning model that receives as input an image of a cork oak and generates as output the mask of the part of the trunk from which the cork is extracted.

5.1. Dataset

The dataset, which has been made available by the authors to the public at [https://www.kaggle.com/datasets/andreguim/cork-oak-segmentation, accessed on 25 October 2022], consists of images of cork oaks obtained before the trees undergo the stripping process. Two or three targets were affixed on each tree trunk, depending on the height of the tree. The targets are detected by the neural network, and then used to calibrate the calculation of the segmented trunk area. A tripod was used to fix the camera so that the photographs were taken with the optical axis parallel to the ground. A preliminary dataset had 55 images, and it was later extended to 62 images after more pictures were taken in the field. Figure 1 shows an image of a cork oak with three targets.

Figure 1. Sample image of the dataset.

This dataset includes images of several types of trunks characteristic of trees of this species. Namely, there are straighter trunks, more curved trunks, as well as trunks with or without bifurcations. Despite having few instances, the dataset continues to be satisfactory for training models based on the Mask R-CNN network, since this network frequently allows us to obtain satisfactory results even in small datasets, as in this circumstance.

To train the model in a supervised way, the training and test images were annotated. The contours of the trunks and the respective targets were marked using the MakeSense.AI web tool. In addition to marking the images, this tool allows the extraction of files in the COCO JSON format, used for training Mask R-CNN networks. After annotation, the resulting dataset was randomly divided into segments of 80% for training, 11% for validation and 9% for testing. That resulted in 50 images belonging to the training set, 7 to the validation set and 5 to the test set. To minimize the problem of working with a small dataset, data augmentation techniques were applied, which are described below.

In order to reduce the error in obtaining the mask from the photograph, the photograph should be taken by a camera with good resolution (the better the resolution, the better the results will be), with favorable climatic conditions for a good perception of the tree and at a time of day when there is enough light to be able to clearly identify the part of the trunk from which the cork will be extracted.

It is also advisable that the images are taken close to the tree and that the camera is pointed as parallel to the ground as possible.

Therefore, each tree was photographed twice in different positions with respect to the tree at a distance of 5 m, and two more times at a distance of 10 m. The camera used was a Canon EOS 200D with an EFS 18–55 mm lens. All the fieldwork took place during the spring–summer season with days of good sunlight, i.e., under open sky conditions. Additionally, we tried to take the photographs without shade, i.e., with direct exposure to the sun (light), and in a position in which it avoids having another tree behind it that could confuse the surface of the working tree with another tree.

5.2. Data Augmentation

As mentioned before, object detection algorithms based on deep learning require a large amount of data to perform properly. However, the dataset used in this project has a small size. To minimize this problem, data augmentation techniques were used to increase the amount of training instances.

The data augmentation process generates new instances from the original data, using transformation methods such as rotation, translation and resizing [24]. The application of those techniques minimizes the overfitting problem.

In particular, the techniques used were: image rotation; image mirror; translation; and brightness adjustment. These transformations were applied during the training process, using the implementation that is available in the Mask R-CNN network.

5.3. Model Evaluation

In object segmentation, three distinct tasks are performed: one to determine if an object exists in the image, another to find the location of the object, and another to draw a binary mask over the object. Additionally, a typical dataset will have more than one class, whose distribution is not uniform, which is also the case in the data used in the present work.

In the present work, the Mean Average Precision (mAP) [25] was used as a criterion to evaluate the models. The global value is determined by calculating the mAP across all classes and at all intersection over union (IoU) boundaries [26]. The IoU metric, also known as the Jaccard index, allows us to quantify the percentage of overlap between the target mask and the mask predicted by the neural network. This metric is closely related to the Dice coefficient, which is often used as a loss function during training.

In a simplified way, the IoU metric measures the number of common pixels between the target and prediction masks, divided by the total number of pixels present in both masks, and can be calculated by the following formula:

$$IoU = \frac{TargetMask \cap PredictedMask}{TargetMask \cup PredictedMask}$$

The intersection (A∩B) is composed of the pixels found in both the predicted mask and the real mask of the object, while the union (A∪B) is made up of all the pixels found in either the predicted or target mask.

In the analysis of the results, three mAP limits are considered: mAP@0.5, mAP@0.7 and mAP@0.9. The mAP@0.5 measures the performance of the model with respect to object segmentation when the predicted mask overlaps by at least half of the real box. The mAP@0.7 shows the performance of the model in relation to object segmentation when the predicted mask overlaps by at least 70% over the real box. The mAP@0.9 follows the same logic as the previous ones, being that it represents the performance of the model in relation

to the segmentation of the object when the predicted mask overlaps by at least 90% of the real box.

Several experiments were carried out which allowed the evaluation of the different models. The experiments aimed to improve the performance of the model, through different technical approaches, namely: a modification of values of certain hyperparameters, an increase in the amount of dataset images and the use of some data augmentation techniques. The best model is presented in this paper.

6. Results of the Mask R-CNN Model

In this section, the configurations of the best models are described, and the results are presented and analyzed.

For the model construction, as referred above, a dataset containing 62 images was used. This was divided into 50 images for training and 7 for validation, leaving the other 5 for testing.

The settings defined for this experiment are shown in Table 1:

Table 1. Settings of the experiment.

Parameter	Value
GPU_COUNT	1
IMAGES_PER_GPU	3
STEPS_PER_EPOCH	500
VALIDATION_STEPS	50
BACKBONE	Resnet101
TRAIN_ROIS_PER_IMAGE	70
NUM_CLASSES	3
RPN_ANCHOR_SCALES	(32, 64, 128, 256, 512)
RPN_NMS_THRESHOLD	0.7
RPN_TRAIN_ANCHORS_PER_IMAGE	256
IMAGE_MIN_DIM	1024
IMAGE_MAX_DIM	1024

According to Mask R-CNN documentation, the IMAGES_PER_GPU parameter, used to define how many images are trained at once per GPU, consumes a large amount of memory. Since the GPU used to train the neural network has only 6 GB of dedicated memory, the value used in this experiment was only one image per GPU. It was also decided that the default values of learning rate and momentum should be kept.

The STEPS_PER_EPOCH parameter defines the number of steps in each training epoch. Although the dataset contains few images, this parameter was set to 500 to avoid spending a lot of training time on updates, performed by the TensorBoard, for providing the measurements and visualizations needed during the workflow [27]. The VALIDATION_STEPS parameter defines the number of validations performed at the end of each training period, and a value of 50 was chosen.

The maximum number of regions of interest to be considered in the final layers of the neural network is defined by the TRAIN_ROIS_PER_IMAGE parameter, which was parameterized with a value of 70. This value is relatively low compared to the one defined in the mask article RCNN (512). However, due also to the high memory consumption required, it was decided to carry out this experiment using the mentioned value.

The defined batch size is obtained by multiplying the parameters GPU_COUNT and IMAGES_PER_GPU, which in the case of this experiment means that the batch size value is 3. In this project, low batch values are used because the graphics card memory quickly exceeds its limit as this value is increased.

As already mentioned, due to the very small size of the dataset, four data augmentation techniques were used to increase the amount of training images:

- Image Rotation between −10° and +10°;
- Vertical Flip (image mirroring);
- Translation of 10%;
- Brightness Adjustment (Add −30 to 30 to the brightness-related channels of the image).

It was decided to define the application of the techniques as following a probability of 83%, that is, each image in the dataset has an 83% probability of undergoing some modification and being reinserted again into the training set. In addition, the techniques began to be applied randomly and not sequentially, with each one being applied separately to each image, using the *imgaug* library. *Imgaug* [28] is a library for image augmentation in machine learning experiments.

Figure 2 illustrates the result of segmentation and classification, using the model generated in this experiment and shows a forked trunk with three legs, a very common situation in cork oak trees in Portugal. The analysis shows how it was possible to clearly identify the area of the trunk that had already been stripped (left image) and that will be stripped again (right image). The image shows a minor problem in the upper zone on the right leg, where the line separating the stripped and unstripped zone is not clearly visible.

Figure 2. Example of segmentation and classification of objects.

Based on the image described, it is possible to verify that the performance of the model seems quite good.

After the training phase, the model was evaluated according to the aforementioned metrics. The results obtained are presented in Table 2.

Table 2. Performance results of the experiment.

mAP@0.5	mAP@0.7	mAP@0.9
0.964	0.964	0.577

The results allow us to conclude that the model is performing well, since its mAP@0.7 is above 96%.

The mAP@0.5 and mAP@0.7 values make this model viable for use in the volume calculation. Even so, there may be some space for progression, as mAP@0.9 may still be improved.

The results also show the need to carefully collect the images to minimize interference from other vegetation and the effect of light and against light.

7. Calculation of the Area of Cork to Be Stripped

After training and validating the detection model, the second stage of this work begins by creating a method for calculating the area of the mask resulting from the output of the

neural network. This was developed using Python 3.7.13, leveraging the availability of packages for computer vision. For this, firstly, since the images had different dimensions, it was necessary to preprocess them, in which a resize to 1024 × 1024 was applied so that the detection was performed correctly. Figure 3 shows examples of masks obtained by applying the previously generated model.

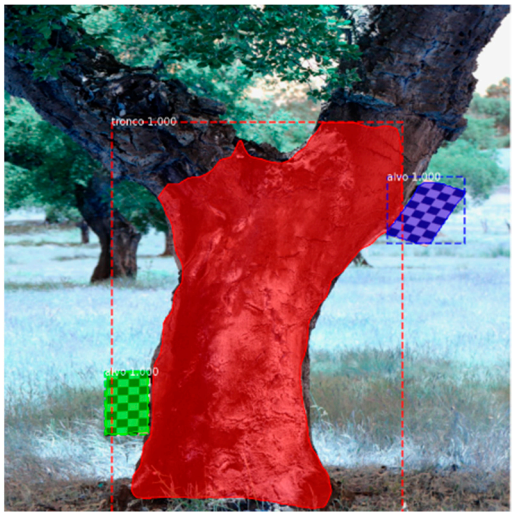

Figure 3. Resulting masks (trunk and targets).

After the detection of the masks (trunk and targets), the area represented by each pixel, in cm^2, was estimated. The estimate was obtained through the ratio between the area in square pixels (px^2) of the targets, calculated from the masks using the minAreaRect method of the OpenCV library, and the actual known area in square centimeters (cm^2). The ratio was calculated for each of the targets present in the image, thus resulting in a list containing the different ratios and the designation of the corresponding targets. The formula for calculating the ratio is shown below:

$$Ratio = \frac{MaskArea\left(px^2\right)}{RealArea(cm^2)}$$

The variable ratio corresponds to the ratio between areas (px^2 to cm^2), the MaskArea is the calculated area, in px^2, through the obtained mask and the RealArea is the known area of the target, in cm^2. The value resulting from the application of the formula presented above was rounded to two decimal places.

With the previously calculated target ratios, it became possible to obtain the trunk mask area. For this, the trunk area in px^2 was obtained, which was calculated through the contour of the trunk mask using the contour area method again. However, to calculate the area in cm^2, it was necessary to use the ratios previously calculated using the masks of the targets. Due to the natural characteristics of the trees and the positioning of the targets on their trunks, it was necessary to find a solution for identifying the most correct ratio to be used in the calculation of the trunk area. It was decided to use the average of the ratios of the different targets and to obtain the ratio to be used in the calculation of the trunk area, hereinafter called FRatio.

The trunk area, in cm^2, is derived by dividing the calculated area, in cm^2, by the FRatio. The formula for calculating the trunk area is shown below:

$$Area\left(cm^2\right) = \frac{CalculatedTrunkArea\left(px^2\right)}{FRatio}$$

The value resulting from the application of the formula presented above was rounded to two decimal places and later converted into m^2.

8. Volume Estimation

After calculating the area, three machine learning algorithms were tested to estimate the final volume of cork, which is the third stage of this project. Linear and non-linear regression algorithms were used, namely LinearRegression, Support Vector Regression (SVR) and MLPRegressor, all taken from the Python sk-learn library. All those algorithms were trained and tested using the same dataset. All stages of the model training process are presented below.

8.1. Biometric Parameters Dataset

A forest inventory was conducted in the spring–summer season. The data were collected in the field in circular plots of 1000 m^2. In this inventory, the data recorded were the following: identification of the farm to which the trees belonged (property), tree number, tree diameter at the base. These were taken at 1.30 m and at 2 m height. At these heights of the trunk, the thickness of the cork (Eco030, Eco130; Eco200), the total height of the tree, the height of the first bough, the height of the first fork and the height of the stem without boughs or forks, as well as the diameter of the crown in the north-south and east-west directions were measured. Finally, the type of cork (segundeira or ama-dia) was recorded. Using these data, it was possible to estimate the trunk area and its volume. In addition to these features, another feature was added that corresponds to the area calculated using the developed methodology described above (area (m^2)). It should be noted that most of these features were not used for training, but only for dataset analysis, as will be explained in the next section.

Biometric records of some trees from the dataset of images were used in this study. It was not possible to collect records of all trees and, therefore, the dataset for volume estimation contains only data from 18 trees.

Since the dataset has a small size, it was necessary to resort to some data augmentation techniques. For that we used Roboflow, a computer vision developer framework for better data preprocessing and model training techniques, available from [https://roboflow.com, accessed on 20 July 2022]. The data augmentation techniques were applied to images whose data were present in the biometric parameter's dataset. The following transformations were applied:

- Crop: 0% Minimum Zoom, 10% Maximum Zoom;
- Flip: Vertical;
- Brightness: Between −5% and +5%.

After applying the above techniques, the dataset now contains 100 instances since the area calculation algorithm was applied to all images, both transformed and non-transformed. Figure 4 illustrates part of the final dataset.

Once the process of increasing the number of records was concluded, the correlation between the dataset features and the extracted cork volume was analyzed. It was found that the base diameter and the diameter at breast height (1.30 m) were the ones that showed the best correlation, 0.98 and 0.99, respectively. Despite the high correlation, this fact can be explained by the fact that there are few records in the dataset.

The correlation between the areas (real and calculated) and the cork volume was also analyzed. It was found that the calculated area and the real area presented a correlation of 0.87 and 0.85, respectively. This reveals that the trunk area is closely linked with the cork volume. Figure 5 illustrates the correlation between the different features and the cork volume.

	Property	Parcel	No_tree	Type	D030	D130	D200	Eco030	Eco130	Eco200	v (m3)	area (m2)	calculated_area (m2)
0	mouchao	2	1	Sb amadia	40.36	40.23	0.0	3.10	2.80	0.00	0.166	0.52	0.84
1	mouchao	2	1	Sb amadia	40.36	40.23	0.0	3.10	2.80	0.00	0.166	0.52	0.81
2	mouchao	2	1	Sb amadia	40.36	40.23	0.0	3.10	2.80	0.00	0.166	0.52	0.77
3	mouchao	2	1	Sb amadia	40.36	40.23	0.0	3.10	2.80	0.00	0.166	0.52	0.76
4	mouchao	2	1	Sb amadia	40.36	40.23	0.0	3.10	2.80	0.00	0.166	0.52	0.82
...	...	...	...	...	...	...	...	...	...	...	...	...	...
95	mouchao	11	12	Sb amadia	58.30	58.10	0.0	2.77	3.07	2.27	0.346	0.78	1.58
96	mouchao	11	12	Sb amadia	58.30	58.10	0.0	2.77	3.07	2.27	0.346	0.78	1.70
97	mouchao	11	12	Sb amadia	58.30	58.10	0.0	2.77	3.07	2.27	0.346	0.78	1.74
98	mouchao	11	12	Sb amadia	58.30	58.10	0.0	2.77	3.07	2.27	0.346	0.78	1.71
99	mouchao	11	12	Sb amadia	58.30	58.10	0.0	2.77	3.07	2.27	0.346	0.78	1.76

100 rows × 13 columns

Figure 4. Part of the Biometric Parameters Dataset.

```
v (m3)                  1.000000
D130                    0.992616
D030                    0.985677
calculated_area (m2)    0.881204
area (m2)               0.868051
D200                    0.532158
Parcel                  0.309632
Eco200                  0.255604
No_tree                -0.202410
Eco130                 -0.252046
Eco030                 -0.321028
Name: v (m3), dtype: float64
```

Figure 5. Correlation between features and volume.

The following section describes all the experiments carried out to find the best algorithm to estimate the cork volume through the area previously calculated.

8.2. Results of Machine Learning Models

In this section, we consider various regressors and compare their performance in terms of the data collected in the field. We used the mean absolute error (MAE), the mean squared error (MSE) and root mean squared error (RMSE) as performance indicators. The MAE refers to the mean error between the predictions made and the actual values of these observations, taking the average over all observations. The MSE measures the average squared difference between the estimated values and the actual value. Therefore, values closer to zero are better. Due to the square, large errors are emphasized and have a relatively greater effect on the value of the performance metric. Finally, the RMSE is a metric that tells the square root of the average squared difference between the predicted values and the actual values in a dataset. The lower the RMSE, the better a model fits a dataset. All these metrics are useful to analyze the performance of regression models.

Before starting the training of the models, some features were removed. This is because, in a real situation, using only the camera, it is not possible to have access to certain data, such as the diameter and real area of the trunk, the cork thickness and the volume (target of our model). Therefore, the mentioned features were removed, as well as the tree number and the plot. The final model includes three features, the property, the type of cork of the tree and the area calculated using the developed method. The resulting dataset was randomly divided into 80% for training and 20% for testing.

Three machine learning algorithms were used, a LinearRegressor, Support Vector Regression (SVR) and MLPRegressor.

Table 3 shows the MAE, MSE and RMSE of each algorithm with the three features mentioned above.

Table 3. Performance results of the machine learning models.

	MSE	RMSE	%MAE
LinearRegressor	0.00096	0.03095	10.22%
SVR	0.00078	0.02788	8.75%
MLPRegressor	0.00084	0.02903	10.79%

It is possible to verify, through the data presented in Table 3, that the SVR algorithm presented the best results, with the average error being 8.75% for the volume estimation. In any case, the other algorithms produced positive results, with the LinearRegressor algorithm showing an average error of 10.22% and the MLPRegressor algorithm showing an error of 10.79%.

9. Conclusions and Future Work

This study proposes a model based on the Mask R-CNN neural network. The model aims to detect and segment the trunk of a cork oak, making it possible to extract the mask from it. The extracted mask will be used to calculate the volume of cork that the cork oak is expected to produce.

The preliminary experimental results demonstrate that the model presents a good performance in the segmentation of the instances in the image. The mAP@0.5 and mAP@0.7 of the best model have values greater than 0.96, which makes this model viable to be used. The mAP@0.9 registered a value of 0.58, which shows that there is still room for future evolution.

The fact that the dataset has images of trunks of different shapes, something that often happens in cork oak trunks, can be beneficial to increase the generalization of the model. However, the size of the dataset is still small, so there is scope to improve the results obtained, and in future works we intend to annotate and include a greater number of instances.

The cork volume was calculated, using machine learning models that received data from the mask of the trunk that was extracted, the characteristics and biometric data of the trees. The SVR algorithm proved to be the best algorithm for this problem, presenting an average error of 8.75%, which makes this model perfectly viable to be used. This error is also very good when compared to other methods for volume calculation, described in the state of the art.

Future work includes obtaining more images and also enrich the biometric parameters dataset with other significant features.

Author Contributions: Methodology, A.G, B.F., R.S.-G., C.P. and M.M.; Software, A.G.; Data curation, M.V. and R.S.-G.; Writing—original draft, A.G.; Writing—review & editing, M.V., B.F., R.S.-G., C.P. and M.M. All authors have read and agreed to the published version of the manuscript.

Funding: This research received no external funding.

Data Availability Statement: The dataset and scripts have been made available to the public at https://www.kaggle.com/datasets/andreguim/cork-oak-segmentation.

Acknowledgments: The authors would like to thank Herdade do Mouchão for allowing the collection of photographs and measurements of cork oaks from the Herdade, data that were used in the scope of this work.

Conflicts of Interest: The authors declare no conflict of interest.

References

1. ICNF. *IFN6—Áreas dos usos do solo e das Espécies Florestais de Portugal Continental*; Resultados Preliminares; Instituto da Conservação da Natureza e das Florestas: Lisboa, Portugal, 2013; 33p, (Available only in Portuguese).
2. APCOR. Cork Yearbook 2020. Available online: https://www.apcor.pt/en/portfolio-posts/apcor-year-book-2020/ (accessed on 4 April 2022).
3. Pereira, H. *Cork: Biology, Production and Uses*; Elsevier: Amsterdam, The Netherlands, 2007.
4. Ministério da Agricultura, do Desenvolvimento Rural e das Pescas (MADRP)., Diário da República de Portugal—I Série—A., Decreto-Lei n.º 169/2001 de 25 de Maio, (Available only in Portuguese). Available online: https://data.dre.pt/eli/dec-lei/169/2001/05/25/p/dre/pt/html (accessed on 4 April 2022).
5. Kangas, A.; Astrup, R.; Breidenbach, J.; Fridman, J.; Gobakken, T.; Korhonen, K.T.; Maltamo, M.; Nilsson, M.; Nord-Larsen, T.; Næsset, E.; et al. Remote sensing and forest inventories in Nordic countries—Road map for the future. *Scand. J. For. Res.* **2018**, *33*, 397–412. [CrossRef]
6. ICNF. *IFN6—Relatório Final do Inventário Florestal Nacional Portugal Continental*; Instituto da Conservação da Natureza e das Florestas: Lisboa, Portugal, 2015; (Available only in Portuguese).
7. Van Laar, A.; Alparslan, A. *Forest Mensuration*; Springer: Dordrecht, The Netherlands, 2007; Volume 13.
8. West, P.W. *Tree and Forest Measurement*, 2nd ed.; Springer: Berlin/Heidelberg, Germany, 2009. [CrossRef]
9. Marzulli, M.I.; Raumonen, P.; Greco, R.; Persia, M.; Tartarino, P. Estimating tree stem diameters and volume from smartphone photogrammetric point clouds. *Forestry* **2020**, *93*, 411–429. [CrossRef]
10. Gavrilescu, R.; Zet, C.; Foșalău, C.; Skoczylas, M.; Cotovanu, D. Faster r-cnn:an approach to real-time object detection. In Proceedings of the 2018 International Conference and Exposition on Electrical and Power Engineering (EPE), Iasi, Romania, 18–19 October 2018; pp. 0165–0168.
11. Koirala, A.; Montes, C.R.; Bullock, B.P.; Wagle, B.H. Developing taper equations for planted teak (*Tectona grandis* L.f.) trees of central lowland Nepal. *Trees For. People* **2021**, *5*, 2666–7193. [CrossRef]
12. Zhang, J.; Huang, X.Y. Measuring method of tree height based on digital image processing technology. In Proceedings of the 2009 1st International Conference on Information Science and Engineering, ICISE, Nanjing, China, 26–28 December 2009; pp. 1327–1331.
13. Han, D. Standing tree volume measurement technology based on digital image processing. In Proceedings of the International Conference on Automatic Control and Artificial Intelligence (ACAI 2012), Xiamen, China, 3–5 March 2012; pp. 1922–1925.
14. Putra BT, W.; Ramadhani, N.J.; Soedibyo, D.W.; Marhaenanto, B.; Indarto, I.; Yualianto, Y. The use of computer vision to estimate tree diameter and circumference in homogeneous and production forests using a non-contact method. *For. Sci. Technol.* **2021**, *17*, 32–38. [CrossRef]
15. Coelho, J.; Fidalgo, B.; Crisóstomo, M.M.; Salas-González, R.; Coimbra, A.P.; Mendes, M. Non-Destructive Fast Estimation of Tree Stem Height and Volume Using Image Processing. *Symmetry* **2021**, *13*, 374. [CrossRef]
16. Gonzalez de Tanago, J.; Lau, A.; Bartholomeus, H.; Herold, M.; Avitabile, V.; Raumonen, P.; Martius Ch Goodman, R.C.; Disney, M.; Manuri, S.; Burt, A.; et al. Estimation of above-ground biomass of large tropical trees with terrestrial LiDAR. *Methods Ecol. Evol.* **2018**, *9*, 223–234. [CrossRef]
17. Heurich, M. Automatic recognition and measurement of single trees based on data from airborne laser scanning over the richly structured natural forests of the Bavarian Forest National Park. *For. Ecol. Manag.* **2008**, *255*, 2416–2433. [CrossRef]
18. Juyal, P.; Sharma, S. Estimation of Tree Volume Using Mask R-CNN based Deep Learning. In Proceedings of the 2020 11th International Conference on Computing, Communication and Networking Technologies, ICCCNT 2020, Kharagpur, India, 1–3 July 2020.
19. Sharma, N.; Jain, V.; Mishra, A. An Analysis of Convolutional Neural Networks for Image Classification. *Procedia Comput. Sci.* **2018**, *132*, 377–384. [CrossRef]
20. Girshick, R.; Donahue, J.; Darrell, T.; Malik, J. Rich feature hierarchies for accurate object detection and semantic segmentation. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Columbus, OH, USA, 23–28 June 2014.
21. Uijlings, J.R.; Van De Sande, K.E.; Gevers, T.; Smeulders, A.W. Selective search for object recognition. *Int. J. Comput. Vis.* **2013**, *104*, 154–171. [CrossRef]
22. Girshick, R. Fast r-cnn. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015.
23. He, K.; Gkioxari, G.; Dollar, P.; Girshick, R. Mask r-cnn. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017.
24. Roth, H.R.; Lu, L.; Liu, J.; Yao, J.; Seff, A.; Cherry, K.; Kim, L.; Summers, R.M. Improving Computer-Aided Detection Using Convolutional Neural Networks and Random View Aggregation. *IEEE Trans. Med. Imaging* **2016**, *35*, 1170–1181. [CrossRef] [PubMed]
25. Everingham, M.; Van Gool, L.; Williams, C.K.I.; Winn, J.; Zisserman, A. The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results. 2012. Available online: http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html (accessed on 16 May 2022).

26. Rezatofighi, H.; Tsoi, N.; Gwak, J.; Sadeghian, A.; Reid, I.; Savarese, S. Generalized intersection over union: A metric and a loss for bounding box regression. In Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition, Long Beach, CA, USA, 16–17 June 2019; pp. 658–666.
27. Get Started with TensorBoard. 2022. Tensorflow. Available online: https://www.tensorflow.org/tensorboard/get_started (accessed on 3 May 2022).
28. Imgaug 0.4.0 Documentation. 2020. Imgaug. Available online: https://imgaug.readthedocs.io/en/latest (accessed on 16 May 2022).

Instituto Superior de Engenharia
Politécnico de Coimbra