# Development of an innovative and efficient urban vegetation monitoring system for sustainable urban ecology

A thesis

submitted in fulfilment

of the requirements for the degree

of

**Master of Philosophy in School of Engineering**

at

**The University of Waikato**

by

**Pejman Alighaleh**


THE UNIVERSITY OF
WAIKATO
*Te Whare Wānanga o Waikato*

2024

# Abstract

In the rapidly urbanizing environment of New Zealand, monitoring and preserving urban green spaces are crucial for ecological balance and sustainability. Urban vegetation significantly contributes to the ecological and social fabric of city life, making its distribution, health, and diversity essential for effective urban management and environmental protection. This research addresses a significant gap in current methodologies for monitoring urban greenery by introducing a more efficient, accurate, and less labor-intensive approach using deep learning technology.

Extensive data collection, preprocessing, and training were key components of this research. Image collection routes were meticulously planned to ensure a comprehensive dataset representing diverse urban vegetation. A major challenge was the quality and privacy concerns within the dataset. Continuous image capturing in urban areas led to issues like blurry, repetitive, and non-vegetative images. To mitigate these issues, diverse filters were introduced to enhance the dataset's quality and ensure privacy protection. Focusing on advanced computer vision systems, this thesis employed the YOLOv5 and YOLOv8 models for automated urban vegetation detection using RGB images from car-mounted cameras. The models were chosen for their effectiveness in image recognition, with various image augmentation techniques and annotation tools enriching the training process. The thesis also explored transfer learning and layer freezing techniques in model training, along with a unique method of using larger bounding boxes for tree annotations to enhance the detection of specific tree species.

The models' performance in vegetation detection was evaluated through precision and recall rates, highlighting their reliability and efficiency, with YOLOv5x demonstrating the highest precision (0.93) and recall (0.90) for urban vegetation detection. In tree species detection, the YOLOv5m with Transfer Learning model achieved a strong balance of precision (0.95), recall

(0.77), and accuracy (0.86), while YOLOv8m with Transfer Learning excelled in recall (0.93), proving useful for applications requiring high true positive identification. For specific tree detection, the YOLOv8m_Bigger_size model led with high precision (0.95), recall (0.98), and the top mAP50-95 score (0.79) in validation. These results underscore the models' applicability in diverse urban scenarios and reinforce the feasibility of this approach for large-scale urban vegetation monitoring.The research findings indicate that deep learning technology, particularly the YOLOv5m and YOLOv8m models, significantly enhances the efficiency and accuracy of urban vegetation monitoring. The innovative use of car-mounted cameras for data collection marks a significant advancement over traditional methods, enabling the capture of large-scale urban vegetation data with minimal intrusion and high efficiency.

In summary, this study demonstrates that applying advanced computer vision systems can markedly improve urban vegetation monitoring. The integration of YOLOv5 and YOLOv8 models in New Zealand's urban landscapes offers a novel approach to address the challenges of urbanization. The models are adaptable for use in other New Zealand cities with access to similar urban vegetation datasheets, crucial for evaluating model accuracy and enhancing precision in future models. This thesis advances our understanding of urban vegetation monitoring using computer vision and highlights the potential of deep learning models in enhancing environmental conservation efforts. The insights and methodologies presented provide valuable tools for urban environmental management and suggest new directions for future research and applications in this critical area.

# Declaration of AI Assistance

This thesis has utilized ChatGPT (version 4) for grammar checking, translation assistance, and creating summaries at the end of each chapter. This tool was employed solely to support the linguistic quality and readability of the work, with no AI involvement in content generation or analysis.

# Acknowledgements

Foremost, I express my deepest gratitude to my supervisors, Dr. Melanie Ooi and Dr. Ye Chow Kuang, for their invaluable guidance, encouragement, and insightful perspectives throughout my research journey. Dr. Ooi's clear articulation of the main research idea and Dr. Kuang's detailed, step-by-step guidance have been instrumental in shaping both this thesis and my academic growth. Their unwavering belief in my abilities and their consistent support have been key motivators, playing a significant role in the successful completion of this project. I am truly thankful for their mentorship and dedication throughout this enlightening journey.

Special thanks are extended to Dr. Steve Pawson for his ongoing assistance and collegiality at various stages of this research. I am equally grateful to Benson Chang for his invaluable support in data collection, which has been crucial to the study.

I would also like to express my gratitude to New Zealand’s Biological Heritage National Science Challenge for awarding me the MPhil scholarship, enabling me to undertake this research.

Lastly, but most importantly, I extend my heartfelt thanks to my loving family. In particular, I thank my mother, Sorour, for her inspiration and unconditional love throughout my life, and my brother, Saeed, for his endless support.

**Contents**

# List of Figures

**List of Figures**

# List of Tables

# 1 Introduction

## 1.1 Context

Urban areas in New Zealand host a diverse range of native tree species, including Kahikatea (Dacrycarpus dacrydioides), Grey Willow (Salix cinerea), and members of the Myrtaceae family [1, 2]. With over 87% of the country's population residing in cities [3], the significance of urban vegetation in maintaining ecosystem health cannot be overstated. Urban vegetation has positive influences on such as ecosystem functions [4, 5], improved human health and quality of life [6, 7], and safe havens for indigenous biodiversity [8, 9]. Trees in Urban Areas can suffer from chronic abiotic stresses (such as changes in the growing season and circadian rhythm due to urban thermal discomfort, disorders caused by air pollution, and droughts) [10], invasive alien plant species [11], insect pests and diseases [12]. To control them, continuous inspection and monitoring programs are needed.

Continuous monitoring of urban vegetation proves challenging, time-consuming, and labor-intensive [13]. To address these limitations, researchers have explored innovative methods to rapidly assess plant health [14, 15]. These investigations indicated two main steps: (i) image acquisition and (ii) analysis and classification [11, 16, 17]. Various technologies, including satellite imagery [18-20], Unmanned Aerial Vehicles (UAVs) [21, 22], and vehicle-mounted cameras [11, 23], have been utilized for image collection. However, each of these technologies presents limitations. Satellites are oriented towards large-scale surveys and might not precisely identify individual trees or sparse clusters. UAVs encounter issues of limited coverage, proficiency prerequisites, and regulatory constraints [22]. Furthermore, drone flight duration constraints due to frequent recharging or refueling introduce time-related challenges [11].

Recent research has dedicated efforts to advancing techniques for urban vegetation identification, classification, and modeling, reflecting the increasing recognition of urban

vegetation's ecological importance and the necessity for innovative monitoring approaches. Notably, a study explored automated urban tree detection via RGB image analysis, combining Faster R-CNN and FPN architecture to achieve remarkable precision [24]. Additionally, research extended to identifying and classifying invasive alien plant species using RGB images, utilizing YOLOv3 for object detection and ResNet50V2 for accurate classification, underscoring the potential for precise categorization [11]. In parallel, investigations were conducted to detect and classify a substantial number of tree images using three distinct methodologies: the Region-based Convolutional Neural Network (Faster R-CNN), YOLOv3, and RetinaNet [17].

It's worth noting that while these deep learning models hold promise for urban vegetation analysis, they are not without limitations. Faster R-CNN's limitation in accurately detecting smaller objects is linked to the coarse resolution of its CNN feature block [25]. YOLOv3, though known for its efficiency, can compromise accuracy in detecting small objects and handling imbalanced class distributions [26]. RetinaNet faces challenges in feature loss due to resolution reduction, channel compression, and up-sampling, particularly affecting smaller defects and feature fusion [27]. ResNet-50, although impactful, has drawbacks related to overfitting and accuracy fluctuations [28].

As urban vegetation monitoring gains importance in the context of ecological equilibrium and sustainability, it is imperative to navigate these limitations while harnessing the potential of advanced techniques to effectively manage urban green spaces.

## 1.2 Objective

This research thesis aims to develop and deploy an advanced computer vision system for the automated detection of urban vegetation, with a specific focus on identifying a particular species of tree and individual trees using RGB images captured by a car-mounted camera. The primary objective of this study is to design a robust methodology that leverages the YOLOv5 model to achieve accurate and efficient tree detection in urban environments. The research encompasses multiple stages, each contributing to the successful realization of the proposed objectives.

### 1.2.1 Scope of work

#### 1.2.1.1 Image preprocessing and enhancement

The research begins by preprocessing the collected RGB images to enhance their quality and utility for subsequent model training. This includes the implementation of filters to remove blur, repetition, and privacy-sensitive information, ensuring a clean and focused dataset.

#### 1.2.1.2 Data annotation and augmentation

A critical step involves the annotation of the prepared images to facilitate supervised learning. Accurate labeling of target vegetation, especially the specific species under study, is crucial. Augmentation techniques are applied to expand the dataset, enriching the model's understanding of various scenarios and improving its generalization capability.

#### 1.2.1.3 Model architecture selection and customization

The choice of the YOLOv5 model architecture serves as the foundation for achieving effective object detection. Customization of the architecture is carried out to optimize it for the detection of urban vegetation, with a particular emphasis on the target species and individual trees.

#### 1.2.1.4 Species-specific model development

The research involves training the initial YOLOv5 model to detect all trees within the urban landscape. A subsequent refinement process is executed to specialize the model's capabilities in recognizing a specific species, such as magnolia trees.

#### 1.2.1.5 Individual tree identification

Building upon the species-specific model, the thesis extends to the development of a model capable of identifying individual trees with a high degree of accuracy. This requires fine-tuning the existing model and integrating advanced object recognition techniques.

### 1.2.2 Research questions and expected outcomes

The research endeavors to answer the following key questions:

- How effective are the applied preprocessing filters in enhancing the quality of the training dataset and subsequently improving model performance?
- Can the customized YOLOv5 model accurately detect the target species (e.g., magnolia trees) within urban landscapes, surpassing generic tree detection models?

- How well does the specialized model perform in identifying individual trees, and what level of accuracy can be achieved?

### 1.2.3 Significance and implications

The outcomes of this research hold significant potential for urban planning, environmental monitoring, and species conservation. An accurate and efficient computer vision system for tree detection, particularly for specific species and individual trees, can provide valuable insights into urban green spaces and contribute to informed decision-making.

## 1.3 Thesis outline

The contents of each chapter of the thesis are outlined below, covering key areas of the research topic and model development.

Chapter 2 details the comprehensive literature review undertaken for this research. This chapter explores various monitoring platforms, including Satellite-Based Monitoring, Unmanned Airborne Vehicle (UAV) Monitoring, and vehicle-based approaches. It also examines different imaging techniques in vegetation analysis like Hyperspectral, Multispectral, and RGB Imaging, followed by a discussion on image processing techniques.

Chapter 3, 'System design and methodology', outlines the methodological framework of the research. It covers the data collection process with a focus on the infrastructure of the imaging system, data collection strategy, and preprocessing steps such as removal of non-vegetation images and privacy protection. This chapter also delves into the training of deep learning models, particularly YOLOv5 and YOLOv8, discussing aspects like dataset preparation, image augmentation, annotation, and the training process.

Chapter 4, 'Results and Discussion,' presents the findings and analysis. It revisits the dataset preprocessing steps, evaluates the annotation tools used, and focuses on the outcomes of urban vegetation detection, specific species detection, and specific tree detection with variable bounding box sizes. It concludes with a summary of the key findings.

Chapter 5, 'Conclusion,' offers a comprehensive overview of the research outcomes, encapsulating the major findings and their implications. It also highlights future research pathways and opportunities, suggesting potential areas for further investigation and development in urban vegetation surveillance using advanced computer vision techniques.

# 2 Literature review

The thesis has thus far introduced the needs for monitoring urban vegetation, emphasizing its critical role in enhancing environmental quality, public health, and urban planning. The central aim of this research is to explore the potential of RGB images and deep learning techniques in achieving efficient and effective monitoring of urban vegetation, with the end goal of providing a state-of-the-art tool for managing urban ecosystems.

This chapter presents a comprehensive literature review, covering a range of key aspects central to this research. These include the broader framework of urban vegetation management, diverse monitoring methods, and specific technical domains such as imaging techniques and image processing, extending to machine and deep learning. While these topics may appear disparate, each is integral to the conceptual and methodological fabric of this thesis.

Urban vegetation monitoring methods are far from standardized, necessitating a thorough examination of existing approaches to establish a foundational understanding. The evolution from traditional image processing methods to more advanced machine and deep learning techniques offers a myriad of methodological choices. This review not only guides the selection of the most appropriate methods but also identifies research gaps that this thesis aims to fill.

Each primary section of this chapter will discuss general theories found in the literature, followed by an examination and critique of prior research in these areas. Through this approach, the literature review seeks to integrate diverse yet crucial areas of research, laying a robust intellectual groundwork for employing RGB images and deep learning in urban vegetation monitoring.

## 2.1 The importance of urban vegetation surveillance

Rapid urbanization is a defining characteristic of the modern era, bringing about significant social, economic, and environmental repercussions [29, 30]. As of 2011, over half of the world's population resided in urban areas, a number projected to increase to 86% in the developed world and 64% in the developing world by 2050 [30, 31]. Increasing urban population is likely to lead to an array of environmental problems in cities such as increasing air pollution and climatic perturbations [32].

Against the mentioned potential problems, trees and urban green spaces (UGSs) emerge as a unique and sustainable solution [30, 33]. Trees, an integral part of the urban ecosystem, can alleviate environmental issues in both direct and indirect ways [34, 35]. Directly, trees absorb gaseous pollutants like sulfur dioxide (SO2), nitrogen dioxide (NO2), and ozone (O3) through leaf stoma and intercept particulate matter in the air [36]. Indirectly, they can lower atmospheric temperatures by providing shade and transpiring, thereby reducing the rate of chemical reactions that lead to secondary pollutants [37, 38]. Moreover, trees offer other ecological benefits such as carbon sequestration through photosynthesis, noise reduction, and the mitigation of the urban heat island effect—increasingly crucial due to ongoing climate change [39-43]. The benefits of urban vegetation are not just ecological but also extend to the physical and psychological well-being of city dwellers [44, 45].

On the other hand, urban vegetation faces a wide range of challenges that endanger its health and longevity. One of these challenges is the effects of climate changes. In urban area, the stress on trees from climate change is worsened due to factors like the heat island effect, poor air quality, hard surfaces, and changed soil conditions, all of which can significantly impact tree growth and survival [46]. Increased storm intensity and limited available space are causing higher rates of street tree mortality [47]. Moreover, increased occurrences of drought and heatwaves are causing significant damage to tree canopies [48]. Changes in soil chemistry and microbial life due to urbanization have also been shown to slow the growth of trees compared to those in natural settings [49].

Equally pressing are the threats from invasive species, which pose a unique risk by outcompeting and displacing native vegetation [11]. These species are a significant factor in the erosion of biodiversity and rank among the top five global threats to biological diversity [50].

Beyond damaging native species and their ecosystems, these invasive plants can also have harmful repercussions on human health and economic well-being [11]. It is crucial to have effective monitoring systems in place to pinpoint where these species are most prevalent, their potential for spreading, or where they can be most effectively contained [51]. Channels of transportation like roads, rail lines, and footpaths play a pivotal role in the dissemination of these invasive species, as they allow for the long-distance transport of seeds [52, 53].

In addition to the threat of invasive plant species, another natural factor significantly impacting the health and survival of urban vegetation is the presence of diseases and pests [54]. The rise in global commerce and the accompanying transfer of plant materials across countries and continents have escalated the risks associated with pests and diseases on urban flora considerably in recent decades [55]. Plant pathogens and insect infestations can significantly harm urban plant ecosystems, leading to both gradual weakening and, in severe cases, the death of trees and other vegetation [56].

A particularly alarming projection comes from a recent study, which estimates that by the year 2050, roughly 1.4 million street trees in urban communities are likely to be destroyed due to invasive insect pests [57]. This data highlights the need for more thorough monitoring and management strategies focused on mitigating these harmful effects. The decline of a substantial number of trees can substantially impact various facets of urban life, from urban ecosystems [58, 59] and urban biodiversity [60-62] to human and wildlife health [63] and well-being [64, 65]. Moreover, this decline adversely affects the quality of urban soil [66], urban water cycles [67], and atmospheric conditions [68], resulting in issues like soil degradation, increased flooding, and elevated urban temperatures [69, 70].

In conclusion, the necessity for robust urban vegetation surveillance is immediate and enduring, as confirmed by a multitude of studies. The consequences of inadequate monitoring are far-reaching, affecting everything from soil quality to human well-being, thereby highlighting its pervasive impact on urban ecology. While the projected adverse scenarios are alarming, they also accentuate the importance of prompt and precise monitoring as a means to mitigate these undesirable outcomes. Consequently, due to the urgent and critical nature of this issue, the subsequent section will provide a thorough examination of the existing methodologies used in

urban vegetation surveillance, scrutinizing their effectiveness in terms of both the technologies employed and the strategies implemented.

## 2.2 Monitoring platforms in urban vegetation surveillances

The Effective and continuous urban vegetation surveillance plays a crucial role in understanding and managing the intricate urban ecosystems. While the monitoring of vegetation has traditionally been labour-intensive and costly, recent advancements in technology have enabled more efficient data collection and monitoring [13, 71]. This section aims to delve into various monitoring platforms, exploring their capabilities and application in urban vegetation management.

In the realm of urban vegetation monitoring, image acquisition has been pivotal, wherein different technologies offer varied degrees of effectiveness and practicality. Platforms such as satellites [18, 20], Unmanned Aerial Vehicles (UAVs) [21, 22], and vehicular-mounted cameras [11, 23] have been employed to cater to diverse monitoring needs and contexts. These platforms utilize imaging techniques such as RGB, multispectral and hyperspectral imaging to capture data that is critical to understanding vegetation health and dynamics [72-75].

While each platform provides valuable data, choosing an appropriate method involves a trade-off between spatial resolution, area of coverage, cost, and operational feasibility. The ensuing subsections will present and assess studies that have leveraged these platforms, focusing on the practicalities and outcomes of their application in urban vegetation monitoring.

### 2.2.1 Satellite-based monitoring

Satellite-based monitoring of vegetation has roots dating back to the early 1970s when the first Earth-observing satellites such as the Landsat program were launched by NASA [76, 77]. Initially, these technologies were focused on broader land use and land cover applications, often lacking the resolution and sensitivity required for more localized, urban assessments [78]. With advancements in satellite technology and imaging software, more recent systems can now capture data at much finer spatial resolutions, making them more applicable for urban vegetation surveillance. [79]. Moreover, offering many advantages such as large-scale data acquisition, high temporal and spatial resolution, and the ability to monitor inaccessible areas, satellites have become a crucial tool for understanding and managing urban green spaces [80, 81].

Several studies highlight the utility of satellite-based monitoring in urban vegetation surveillance [33, 82-84]. For instance, a study conducted by [85] employed Very High Resolution (VHR) Pleiades imagery for vegetation mapping in Kuala Lumpur. The researchers utilized the Normalized Difference Vegetation Index (NDVI) in combination with a Maximum Likelihood classification technique to differentiate between areas of low vegetation, high vegetation, and non-vegetated land cover. A distinctive feature of the study was its use of varying NDVI threshold values for specific vegetation classification. The methodology achieved an overall accuracy rate of about 71%, backed by a kappa coefficient of 0.5. While confirming the utility of VHR Pleiades imagery and NDVI for urban vegetation surveillance, the study also emphasized the necessity for further research to adapt this approach to different geographic areas and varying types of land use [85].

In a 2019 study, [86] utilized RGB and 10-m resolution near-infrared and red bands from Sentinel-2 to accurately calculate the Normalized Difference Vegetation Index (NDVI) for urban areas in the Pearl River Delta. The NDVI estimates were cross-validated with high-resolution Google satellite images, achieving a strong correlation of 0.97. The study confirmed that Sentinel-2 data can effectively complement Urban Canopy Parameters (UCPs) from the World Urban Database and Access Portal Tools (WUDAPT) [86].

In the other study was done by [86] on the application of U-Net-like Convolutional Neural Networks (CNNs) for individual tree recognition using very high-resolution RGB satellite images, specifically focusing on northern temperate mixed forests in Russia's Primorsky Region. Utilizing GeoEye-1 pan-sharpened RGB imagery with a spatial resolution of 0.46 m/pixel, the researchers trained the U-Net-like CNN to perform image segmentation. The study found that this deep learning approach outperformed standard pixel-based machine learning methods, such as random forests and k-nearest neighbor classifiers, particularly in minimizing false-positive identifications of green-colored objects. Furthermore, the U-Net-like CNN recognized trees not just as a collection of pixels, but as spatial objects with specific geometries, achieving a mean accuracy score of up to 0.96. Importantly, the study also emphasized the significance of capturing satellite images during appropriate phenological seasons for effective tree species recognition [86].

While satellites offer valuable advantages for monitoring urban vegetation, it’s important to point out that using satellite data to monitor urban areas is difficult due to the complex variety of

land features, especially when observed at finer scales [87-89]. Temporal restrictions are another drawback, as satellites aren't always available to capture images at the specific times they may be needed; additionally, long waiting periods can occur between the acquisition and reception of these images [90]. Environmental factors like cloud cover further complicate the situation by obstructing clear views, making satellite imagery less reliable for continuous monitoring [90]. These challenges are especially noticeable when satellite data are compared or integrated with ground-level or localized aerial data.

Moving forward, the subsequent section will delve into the emerging realm of Unmanned Airborne Vehicle (UAV) Monitoring. This approach serves as an intermediate solution, bridging the gap between ground-based and satellite observations.

### 2.2.2 Unmanned airborne vehicle (UAV) monitoring

Unmanned Airborne Vehicles (UAVs), commonly known as drones, have undergone a significant transformation since their inception. Originating from military applications, by the early 2000s, they began to find their place in commercial and research sectors [91, 92]. Initially, these drones were constrained by limited flight durations and imaging capabilities; however, the rapid advancement of technology has greatly expanded their potential [93]. Today, UAVs strike a perfect balance between cost-efficiency and high-resolution imaging. This unique combination makes them especially well-suited for detailed, timely, and versatile research and planning, particularly in the domain of urban vegetation monitoring [91, 94].

Several studies show UAVs' role in urban vegetation monitoring [95-98]. [99] used deep fully convolutional networks (FCNs) for segmenting a specific tree species in an urban context using UAV-captured RGB images. Methodologically, they assessed five FCN models, including SegNet, U-Net, FC-DenseNet, and two DeepLabv3+ variants. They also evaluated the benefits of using fully connected conditional random fields (CRFs) for post-processing. The research pinpointed the presence of the endangered tree species, Dipteryx alata Vogel. Results showcased overall accuracies between 88.9% and 96.7%, with CRFs improving performance but at a higher computational cost.

[100] in their study titled "Tree species classification using deep learning and RGB optical images obtained by an unmanned aerial vehicle," investigated urban tree species identification leveraging RGB imagery from UAVs. Methodologically, they presented a dataset encompassing

10 distinct tree species with tree canopy images set against both simple and complex backgrounds. They assessed the effectiveness of deep learning models, such as AlexNet, VGG16, and ResNet-50, and juxtaposed their performance with traditional techniques, notably K-nearest neighbor (KNN) and BP neural network. Remarkably, their findings highlighted ResNet-50's superior capability, which achieved an impressive 92.6% overall accuracy on the dataset.

In the other article, [101] utilized training and test data from three temperate forest sites in Japan, covering 56 tree species. Using UAV RGB images combined with deep learning for tree identification, the model's performance yielded a high Kappa score of 0.97 when tested on identical conditions to the training set. However, the score declined to 0.47 when evaluated on data from different times and locations. The results highlighted challenges in identifying closely related species and those with similar environmental preferences, suggesting the need for further optimization in real-world applications.

UAVs have undeniable potential, but there are limitation such as limited battery life, administrative and legal requirements, privacy concerns, limited mission time and distance [91, 92]. Moreover, by increasing the complexity of urban landscapes, the necessity for ground-level monitoring becomes more important. This leads us to consider the role of vehicles equipped with advanced monitoring systems. The upcoming subsection will explain monitoring urban vegetation using vehicles.

## 2.3 Monitoring using vehicles

Monitoring urban vegetation using vehicles, specifically through street-level imagery, offers an intimate perspective on urban vegetation analysis. Such an approach serves as a valuable data source to precisely and accurately detect and evaluate urban greenery along the streets [102]. Vehicles equipped with cameras empower researchers to gather images from specific areas, delivering detailed, street-level insights that are often unattainable through satellites and UAVs [103].

In a 2022 study, [102] conducted in Tai’an city, China, researchers utilized multi-year Baidu Street View images to monitor urban street-side vegetation greenery changes. To distinguish green vegetation pixels based on the images' spectral information, the study employed both the ISODATA (an unsupervised classification method) and the SVM (a supervised classification method). However, when combined with segmentation, the Vegetation Greenery Index (VGI)

showcased superior classification accuracy compared to both ISODATA and SVM methods. Specifically, classifications using VGI yielded the lowest commission, omission, and overall errors. While all three methods overestimated green vegetation, with some misclassifications like green building surfaces and shades, the overestimation was least pronounced with the VGI-based approach [102]. Several similar studies were conducted using street-view images [103-105].

In another study, [11] developed a system to monitor Invasive Alien Plant Species (IAPS) along roadsides using a high-speed camera mounted on a moving vehicle. The study collected images of seven specific IAPS and employed three deep convolutional neural networks, ResNet50V2, MobileNetV2, and YOLOv3, for classification and detection. The performance of these networks was influenced by both the input image size and the size of the IAPS within the images. Binary classification, which differentiated between IAPS and non-IAPS, exhibited enhanced accuracy compared to the classification of individual plant species. The results indicated a recall of 92% on training and validation, with an average precision of 77% for detected species. While ResNet50V2 was more efficient in detecting Rosa rugosa, YOLOv3 was superior in identifying the harder-to-detect species, Lupinus polyphyllus and Pastinaca sativa [11].

Vehicle-based street-level monitoring offers a distinct advantage in capturing the detailed characteristics of urban greenery along road networks, addressing information gaps that satellite and UAV imagery often overlook [102, 103]. This method demonstrates particular efficacy in mapping street-level vegetation with high accuracy in densely populated urban areas [102]. However, its application is confined to road-accessible zones and is reliant on vehicle availability, providing a more limited scope in comparison to aerial methods [106]. Studies indicate that street-level imagery is especially valuable for identifying invasive species and evaluating urban green coverage along transportation corridors [103].

Table 2.1 provides a comparative analysis of the three predominant urban vegetation monitoring methods: Satellite-Based Monitoring, Unmanned Airborne Vehicle (UAV) Monitoring, and Vehicle-Based Monitoring, and concisely outlines their respective advantages and disadvantages, offering a clear perspective on their applicability and limitations.

**Table 2.1:** Comparative analysis of monitoring methods.

| Monitoring Method | Advantages | Disadvantages |
|---|---|---|
| **Satellite-Based Monitoring** | - Wide coverage and remote area access<br>- High-resolution data<br>- Useful for broad landscape analysis | - Challenges in detailed urban mapping<br>- Time delays in data capture<br>- Weather-related image interference |
| **UAV Monitoring** | - Detailed, high-resolution imaging<br>- Flexible and timely data collection<br>- Cost-effective for small areas | - Limited flight time and range<br>- Regulatory and privacy concerns<br>- Weather dependency |
| **Vehicle-Based Monitoring** | - Precise street-level detail<br>- Direct observation of urban greenery<br>- Effective in urban settings | - Limited to road-accessible areas<br>- Dependent on vehicle availability<br>- Narrower scope compared to aerial methods |

As we delve into the advantages of data collection platforms like satellites, UAVs, and vehicle-based monitoring, it's essential to understand that the imaging techniques employed significantly influence the outcomes. After discussing the common platforms and their respective strengths and weaknesses, we will transition to our subsequent section, "Imaging Techniques in Vegetation Analysis" Here, we will explore various imaging methods, including RGB Imaging, Multispectral Imaging, and Hyperspectral Imaging, to understand their roles and capabilities in urban vegetation analysis.

## 2.4 Imaging techniques in vegetation analysis

In this subsection, various imaging techniques that are commonly used in monitoring urban vegetation will be discussed. We will discuss three primary types of imaging techniques: Hyperspectral Imaging, Multispectral Imaging, and RGB Imaging.

### 2.4.1 Hyperspectral imaging

Hyperspectral imaging synthesizes spectroscopy and imaging to concurrently capture spatial and spectral data, enabling detailed spectral analysis and precise material identification in diverse scientific applications [107]. Hyperspectral images have a wide range of applications in environmental [108, 109], military [110], mining [111], and medical fields [112]. Unlike traditional imaging systems, hyperspectral images are rich in spectral information, and this spectral information can reflect the physical structure and chemical composition of the object of interest, which is helpful for image classification [113, 114]. A dozen of research showed that the hyperspectral data obtained from satellites ([115, 116]), UAVs [117-119] and ground-based platforms [120, 121] have been widely used for species identification and vegetation monitoring.

A study conducted by [122] utilized hyperspectral imaging to classify urban tree species in Hong Kong. In this study, a hyperspectral library was established, classifying 19 tree species. A

total of 450 bi-monthly images were acquired by a terrestrial hyperspectral camera (SPECIM-IQ) from November 2018 to October 2019. The accuracy of the Deep Neural Network classification model, developed to identify tree species from hyperspectral imagery, ranged from 85% to 96%, depending on the season [122].

In another study, [123] leveraged hyperspectral imaging to classify different vegetation types in highly fragmented planting areas of southern China. A commercial hyperspectral imaging sensor (S185), mounted on a UAV platform, was used to capture images, which were then aggregated to various spatial resolutions (from 0.025 m to 2.5 m) to analyze the impact on classification accuracy. The method employed was the Object-Based Image Analysis (OBIA), with a thorough examination of different segmentation scale parameters and feature numbers. Results indicated that the classification accuracies ranged from 84.3% to 91.3% [123].

While hyperspectral imaging provides substantial benefits for monitoring vegetation, it is not without its challenges. Firstly, hyperspectral data is characteristically nonlinear, and the spectral information is subject to loss or interference from noise, attributed to a myriad of uncontrollable external factors including environment, atmosphere, illumination, and data transmission [124]. Secondly, hyperspectral data, inherently voluminous and with a pronounced correlation between adjacent bands, readily induces information redundancy and overlap, presenting challenges in effective data management and analysis [125]. Thirdly, the procurement of usable training samples proves to be a considerable obstacle, leading to the ill-posed problem in certain methodologies, diminishing the generalization capability of classifiers, and subsequently limiting the widespread application of hyperspectral data [113].

Given the pros and cons of hyperspectral imaging, the next section will transition to multispectral imaging, which offers another approach for vegetation monitoring and other applications

### 2.4.2 Multispectral imaging

Multispectral imaging works with images in select narrow bands across a specific range of the spectrum and can create characteristic wavelengths for each pixel in a focused object [126, 127]. Pictures obtained through a multispectral camera can glean a considerable volume of extra data, which neither the human eye nor a conventional camera can discern, thereby having vital applications in specific fields like precision agriculture, forestry, and medicine, as well as in

identifying and classifying objects [127-132]. This technology also offers practical benefits, including lower acquisition time and complexity compared to hyperspectral imaging, along with the capability for real-time application in certain domains [133, 134].

In a recent study by [135], a methodology that integrates an object-oriented approach and a random forest algorithm, utilizing UAV multispectral images, was developed to enhance the identification accuracy of urban trees. The process initiated with the segmentation of images using a multi-scale segmentation algorithm, informed by the scale determined through the Estimation of Scale Parameter 2 (ESP2) tool and visual discrimination. Then, various features, including spectral, index, texture, and geometric, were amalgamated into schemes S1–S8, and S9, which consisted of features chosen by the recursive feature elimination (RFE) method. The classification of urban trees was executed based on these schemes, employing random forest (RF), support vector machine (SVM), and k-nearest neighbor (KNN) classifiers. Results showcased that the RF classifier outperformed SVM and KNN, with the highest accuracy being achieved in scheme S9 (contains all the preferred features), marking an overall accuracy (OA) of 91.89% and a Kappa coefficient (Kappa) of 0.91 [135].

In the next study, the use of multispectral Sentinel-2 imagery for urban land cover and land use (ULCLU) classification was explored, particularly focusing on a challenging and complex densely populated urban area in the South-West district of Delhi, India [136]. Utilizing a 10 m spatial resolution imagery and involving five pivotal ULCLU classes - roads, water, buildings, vegetation, and barren land - three distinct classification algorithms were applied: Support Vector Machine (SVM), Artificial Neural Network (ANN), and Maximum Likelihood Classifier (MLC), each with a training sample size of 150 pixels per class. Of the methods employed, SVM showcased superior classification performance, achieving an impressive 98.05% accuracy, whereas all methods surpassed 90% accuracy. The findings affirm the substantive potential of Sentinel-2 multispectral imagery and advanced classification algorithms in deriving accurate and critical information for ULCLU, particularly in densely urbanized contexts [136].

In another study, [137] utilized WorldView-2 multispectral images, spanning a spectral range of 427–908 nm and offering a 2 m spatial resolution, to classify six commercial forest species in KwaZulu-Natal, South Africa. The classification technique applied was partial least squares discriminant analysis (PLS-DA), which achieved an overall accuracy of 85.42% and a kappa

statistic value of 0.83, with accuracies for individual species fluctuating between 63% and 100%. Additionally, the variable importance in the projection (VIP) method was employed to identify the most effective wavebands for species discrimination, notably coastal blue (427 nm), blue (478 nm), green (546 nm), and red (659 nm), achieving an overall accuracy of 84.38% and a kappa value of 0.81 with the critical bands. Individual species accuracies using VIP bands ranged from 69% to 100%, highlighting the notable influence of the visible spectrum in forest species discrimination [137].

However, multispectral imaging comes with certain limitations that require attention and ongoing research to enhance its utility and accessibility. These challenges include the need for careful calibration to avoid potential misinterpretations [138], relatively low spectral resolution compared to hyperspectral imaging [139], and a higher cost compared to RGB cameras [140]. As we transition from this discussion, the ensuing subsection will pivot towards exploring RGB imaging, illuminating its distinct attributes, applications, and how it juxtaposes with the hyperspectral and multispectral imaging technologies detailed above.

### 2.4.1 RGB imaging

Red, Green, and Blue (RGB) imaging serves as a fundamental pillar within the realm of monitoring and remote sensing [141]. RGB cameras, characterized by their lightweight construction, offer a cost-effective and straightforward operation, maintaining functionality under various atmospheric conditions, whether sunny or cloudy [142, 143]. With high spatial resolution and broad accessibility in everyday devices like smartphones and digital cameras, RGB imaging is widely adopted across numerous applications, making it a practical choice in remote sensing [144-152]. They are emerging as an attractive alternative to utilizing multispectral or hyperspectral images, gaining increasing adoption in the realm of remote sensing [147, 152]. Many studies have been conducted on RGB-based vegetation monitoring, such as estimating leaf chlorophyll content [153], leaf area index (LAI) estimation [154], evaluating plant height and biomass [155], diagnosing nutrient deficiencies [156], monitoring plant growth [157], detecting stress [158], monitoring plant diseases [159], crop genotyping [160], etc. Specifically, in the field of monitoring urban vegetation using RGB images, some research has been done, a few instances of which will be referred to in the following.

In a 2022 study regarding tree species monitoring and classification via high-spatial-resolution RGB images, a deep learning-based mobile application tool was proposed and developed [161]. Through the evaluation of various deep learning architectures, including both mobile networks and traditional models, the research utilized a dataset of 2,349 images, which comprised Dipteryx alata species and other local species. The images, manually annotated, were systematically divided into training, validation, and testing subsets, aligning with the methodology of five-fold cross-validation. The assessment of both accuracy and speed (across GPU and CPU) of the implemented deep learning architectures revealed that traditional networks offered superior performance in F1 score metrics, albeit with mobile networks being more rapid in operation. Specifically, the Inception V3 model yielded the highest accuracy, attaining an F1 score of 97.4%, whereas MobileNet, although registering the lowest F1 score of 83.84%, delivered the most optimal classification speed for both CPU and GPU units [161].

In another 2021 study exploring the utilization of CNN-based methods for individual tree detection in urban locales using high-resolution RGB images, three state-of-the-art methods were evaluated: Faster-RCNN, RetinaNet, and Adaptive Training Sample Selection (ATSS) [162]. A comprehensive dataset, comprising 220 manually labeled images, reflecting 3382 trees, was employed to discern the efficacy of these methods in recognizing single trees within the urban environment. The study identified ATSS as the most accurate method, achieving an $AP_{50}$ of 69.2% and an $AP_{75}$ of 28.4%, albeit with a general observation that Faster-RCNN and RetinaNet may perform qualitatively better in certain contexts. Despite all metrics remaining below 70%, a qualitative analysis suggested satisfactory performance of the model, particularly notable in areas of less density and for smaller tree crowns. Nonetheless, a noticeable decrement in performance was observed in regions of greater tree crown size and density [162].

In other study that conducted in 2021, concentrated on monitoring urban forests by evaluating various features on a single-tree scale [163]. The approach involved extracting spectral data, vegetation morphological parameters, texture information, and vegetation indexes from ultrahigh-resolution RGB images, which were captured using UAVs and subsequently applied to develop an object-oriented-based random forest (RF) classifier. The research primarily focused on four tree species: *Michelia chapensis*, *Salix babylonica*, *Cedrus deodara*, and *Prunus cerasifera*. The findings illustrated that the RGB images from UAVs effectively detected surface fragments and

achieved a notable accuracy of 91.3% in RF classification, exceeding the outcomes from using high-resolution street view maps by 20.5% [163].

Another notable 2021 study employed a machine vision system that amalgamated UAV-derived RGB images and a convolutional neural network (CNN) for tree classification, achieving over 90% accuracy by utilizing color and 3D information for segmentation and object-based CNN classification [164]. Similarly, a 2020 study introduced a tree species classification approach using high-resolution RGB images from automated UAV flights and a lightweight CNN, attaining an average classification accuracy of 92% under varied illumination and phenological stages across four tree species [165].

However, RGB images have some limitation such as being confined to the visible spectrum and displaying a susceptibility to environmental factors, notably, fluctuations in illumination which may impair image quality, especially during low-light conditions like nighttime [148, 166, 167].

After a brief exploration of various imaging modalities such as hyperspectral, multispectral, and RGB imaging, each presenting its own set of advantages, constraints (Table 2.2), and specific applications in research scenarios, the ensuing section shifts the focus toward an array of image processing techniques.

**Table 2.2:** Comparative overview of imaging technologies.

| Imaging Type | Advantages | Disadvantages |
|---|---|---|
| **Hyperspectral Imaging** | - Detailed spectral analysis<br>- Precise material identification<br>- Useful across various fields like environmental, military, mining, and medical | - Nonlinearity in data<br>- Susceptible to noise and interference<br>- Challenges in data management and analysis |
| **Multispectral Imaging** | - Faster acquisition and less complexity than hyperspectral<br>- Directly applicable in real-time scenarios<br>- Important in fields like agriculture and forestry | - Need for calibration<br>- Lower spectral resolution<br>- Higher cost compared to RGB cameras |
| **RGB Imaging** | - Cost-effective and lightweight<br>- High spatial resolution<br>- Widely accessible and easy to process and interpret | - Limited to the visible spectrum<br>- Sensitive to environmental light changes<br>- May not capture as much detail as hyperspectral or multispectral |

## 2.5 Image processing techniques

In this subsection, we will concentrate on methods relevant to image processing, particularly with respect to urban vegetation monitoring. Our exploration will initially delve into conventional machine learning algorithms and subsequently into Deep Learning techniques. Each method's key features, along with their pros and cons in different uses, will also be explored.

### 2.5.1 Conventional machine learning algorithms

Machine learning, a branch of artificial intelligence (AI), encompasses the precise development and optimization of algorithms that learn from designated datasets to make predictions about unknown data, utilizing statistical analysis to discern patterns within large datasets and derive inferences about the dataset's output [168, 169]. Numerous algorithms have gained popularity in this domain for their utility in image processing, including K-nearest neighbor (KNN), random forest (RF), support vector machine (SVM), self-organizing maps (SOM), multilayer perceptron (MLP), radial basis function neural network (RBF), learning vector quantization (LVQ), and principal component analysis (PCA) [72]. Over the past decade, the application of these algorithms has permeated various scientific disciplines, finding utility in areas such as robotics [170, 171], agriculture [172, 173], medicine [174, 175], and aquaculture [176, 177]. It is particularly noteworthy that much research has been undertaken in the surveillance, identification, and classification of urban plant species employing some of the mentioned algorithms. Subsequently, a number of these studies will be discussed.

In a 2023 study, very high-resolution WorldView-2 satellite imagery was meticulously explored for its capability to detect, delineate, and classify urban plant species across both public and private domains [178]. A selection of four spectral bands (blue, green, yellow, red) and four texture features (namely, energy, entropy, inverse difference moment, Haralick correlation) were identified as the most efficient attributes for object-based classification from the WV-2 images. Following this, a classification of plant species was executed using a Random Forest classifier. 22 dominant plant species, along with grassland, were successfully identified and classified, achieving an overall accuracy of 84% and 83% respectively. Notable peaks in classification accuracy were observed for *Pinus spp* specie whereas the lowest classification accuracies were noted for *Quercus spp* specie[178].

The next study published in 2022 concerning the application of machine learning algorithms for urban tree species classification, specifically between deciduous (broadleaf) and coniferous tree species utilizing 3D raw LiDAR data distinct methodologies was explored [179]. Initially, ground, building, and varying vegetation classes (low, medium, and high) were derived from raw LiDAR data through a hierarchical-rule-based classification method. Subsequently, individual tree crowns were segmented employing a mean shift clustering algorithm from high vegetation points. For the discrimination of deciduous and coniferous tree species within the urban area, a total of 25 spatial- and intensity-based features were utilized across three machine learning classifiers: support vector machine (SVM), random forest (RF), and multi-layer perceptron (MLP). The resulting overall accuracies of the machine learning-based classifications were 80%, 83.75%, and 73.75% for the SVM, RF, and MLP classifiers, respectively, in a 70/30 training/testing split. Notably, the SVM and RF algorithms consistently outperformed the MLP algorithm in terms of providing superior classification results for identifying the urban tree species[179].

Another study exploring urban tree species classification utilizing UAV-based multi-sensor data and machine learning, a multi-sensor data fusion technique was developed to enhance classification accuracy with limited training samples [168]. This research employed various UAV-based sensors, including multispectral, hyperspectral, LiDAR, and thermal infrared imagery, collected over an urban area to classify 96 trees spanning seven different species. Two supervised machine learning classifiers, Random Forest (RF) and Support Vector Machine (SVM), were evaluated, focusing on their capability to handle and analyze high-dimensional datasets derived from multiple sensors. When using hyperspectral-derived spectral features with RF, the fusion of all features (spectral, LiDAR, thermal) from all sensor types achieved the highest overall classification accuracy (OA) of 83.3% and a kappa of 0.80. In contrast, while multispectral reflectance bands alone secured a significantly lower OA of 55.2%, the combination of all dataset types (spectral, LiDAR, thermal) using multispectral-derived spectral features nonetheless achieved an OA of 81.3% and a kappa of 0.77 with RF [168].

Lastly, Pu et al. focused on the utility of multi-seasonal high-resolution Pléiades satellite imagery for urban tree species mapping in Tampa, FL, USA, the authors explored the potential to enhance classification through the exploitation of seasonal variations [180]. The study assessed and compared the capabilities of Pléiades images acquired in different seasons for classifying seven species and groups of trees to discern the impact of seasonal variations on mapping accuracy.

The employed species and groups included various oaks, pine, palm, camphor, and magnolia. A multi-level classification system, utilizing a suite of classifiers – Random Forest, Support Vector Machine, and Linear Discriminant Analysis – was applied to classify image objects of the tree species and groups with features derived from the images. Notably, the results illustrated a substantial improvement in tree species mapping accuracies using the late spring season (April) image compared to other seasonal images, with combined dry-wet season images performing even better ($p < 0.01$). Furthermore, the findings revealed a significant seasonal effect on tree species classification and established that the Random Forest classifier notably outperformed the other classifiers for tree species classification, highlighting the criticality of selecting apt seasonal remote sensing data for mapping tree species [180].

Conventional machine learning algorithms showcases some advantages in image processing, including they can train and work on low-end machines [181], often requires less data to train a model effectively [182], and they have lower computational complexity [183]. However, one cannot overlook these algorithms' drawbacks. One prominent limitation of conventional machine learning algorithms is that they cannot learn straight from unprocessed data; they need a careful preprocessing process to extract features from raw data and classify them, and then they use these classified features to find specific patterns [184]. The next drawback is that the predictive accuracy of these algorithms can be compromised and diminished due to extended training durations, gradient explosions, and susceptibility to overfitting [185]. Moreover, conventional machine learning algorithms, while excelling with small to medium-sized datasets, manifest limitations in accuracy when applied to larger datasets [186]. Another limitation is that the issue of noisy and imprecise data, along with inaccurate labels, significantly challenges the accuracy and reliability of machine learning algorithms [187]. In light of these challenges and limitations with conventional machine learning, the subsequent section will delve into deep learning models, exploring their capabilities and nuances.

### 2.5.2 Deep Learning techniques

Deep Learning (DL), a vital component of machine learning (ML) (Figure 2.1), has experienced remarkable growth in recent years, particularly highlighted by its increasing implementation in research and industry [188-191]. A major turning point came in 2012 when, fueled by advancements in computer vision tasks like image categorization, object identification and tracking, and semantic segmentation, deep learning techniques began to be widely utilized for

tasks such as land cover monitoring and classification [192]. Its role has since expanded into vegetation remote sensing, showcasing its versatility across various research domains [193-196]. DL surpasses traditional machine learning algorithms by autonomously learning pertinent features from annotated datasets, offering an end-to-end learning approach that eliminates the need for manual expertise in feature extraction [164, 197, 198].

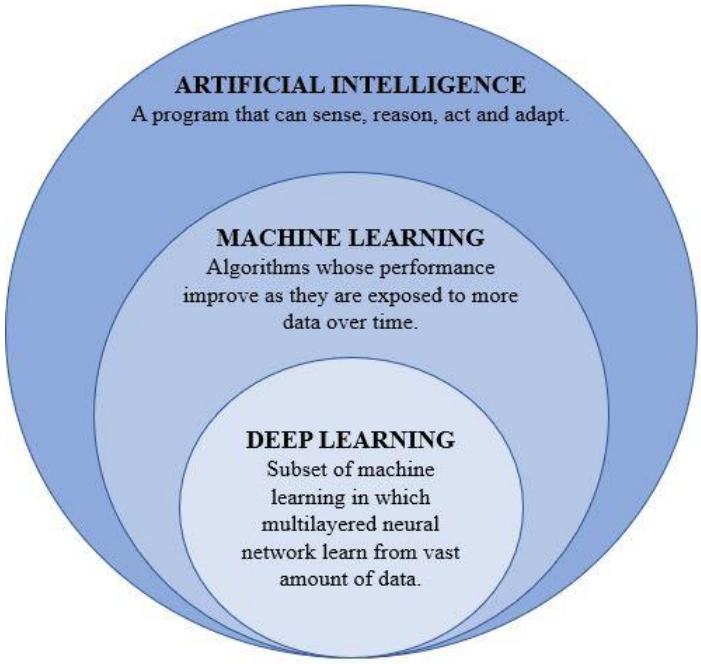



**Figure 2.1:** Deep learning family

Inspired by the human brain's information processing patterns, DL utilizes multiple layers of algorithms or artificial neural networks (ANNs) for varied data interpretations [199, 200]. Unlike conventional ML techniques, which involve a series of steps including preprocessing and feature selection, DL streamlines the process by integrating learning and classification in a single step (Figure 2.2) [200, 201]. This efficiency has propelled DL to prominence in big data fields and is continuously evolving to enhance performance in various ML tasks [202-205]. Its applications range from image super-resolution [206] to object detection [207, 208], and image recognition [203, 209], with its performance in some areas surpassing human capabilities.

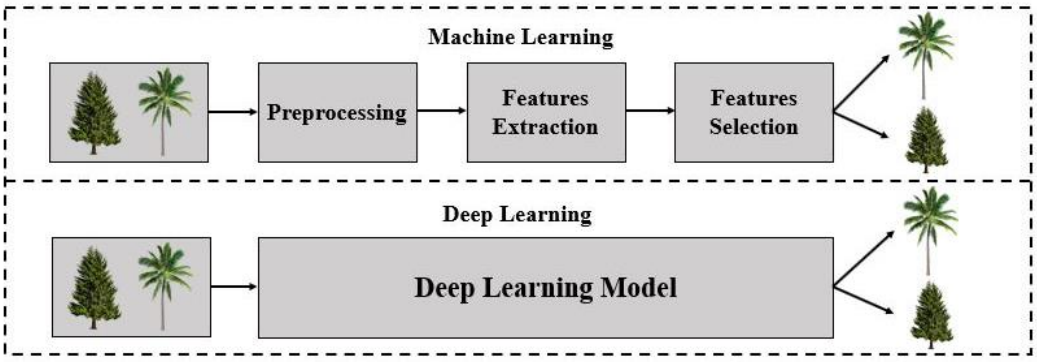



**Figure 2.2:** The difference between deep learning and traditional machine learning.

Building on this momentum, DL has more recently been incorporated into vegetation remote sensing, marking its expanding influence in diverse research domains [193-196]. While conventional machine learning algorithms necessitate manual expertise for feature extraction and involve intricate procedures like feature selection, deep learning approaches enable end-to-end learning from annotated datasets, autonomously ascertain the most pertinent features, and have demonstrably surpassed traditional algorithms in the realm of computer vision tasks [164, 197, 198].

DL's impact extends across numerous scientific fields and industries, significantly transforming them. Major technology companies worldwide are continually striving to advance DL, which has already surpassed human-level performance in various applications such as predictive analytics in logistics, loan approvals, and entertainment [210]. Despite its remarkable achievements, DL's potential for further advancements remains vast, particularly in enhancing human lives. Its precision and accuracy hold promise for natural disaster prediction [211], new drug discoveries [212], and cancer diagnosis [213-215].

In vegetation monitoring, detecting and classifying research, several deep learning architectures have been used containing: DeepLabV3+ [192], Res-UNet [216], ResNet [11, 217, 218], VGG16 [164, 219], LeNet [220], U-net [117, 221], DenseNet [222], AlexNet [164, 219], YOLOv3 [11], MobileNet [11], EfficientNet [101], etc. In the domain of urban vegetation monitoring, deep learning has paved the way for different research. Some of these studies are briefly reviewed below.

In a 2018 study, publicly available aerial and street view images from Google Maps™ were harnessed to develop a fully automated pipeline for tree detection and species recognition [223].

Utilizing VGG16 convolutional neural networks (CNN), the researchers processed and analyzed these images for tree detection and subsequent species classification. The integrated system displayed a proficiency to detect >70% of street trees and achieve a classification accuracy of >80% for 40 different species. Specific trees, for instance, certain species, might exhibit higher or lower classification accuracies [223].

In the next study, a deep learning model, specifically the U-net, was applied for urban tree canopy mapping employing high-resolution aerial photographs [224]. The U-net's efficacy in tree canopy mapping was assessed across four spatial scales—16 cm, 32 cm, 50 cm, and 100 cm. During the training phase, the data was apportioned with 85% allocated for training and 15% for validation. Performance was gauged using four quantitative metrics: Dice, Intersection over Union, Overall Accuracy, and Kappa Coefficient. Notably, the U-net's application with 32-cm input images showcased exemplary results, recording an overall accuracy of 0.9914 and an Intersection over Union of 0.9638 [224].

In another study, a DeepLabv3 model was introduced for the purpose of tree species mapping in tropical urban settings [225]. The network architecture processes aerial RGB images and yields two distinct outputs: a semantically segmented image and a distance map transform. The segmented image provides labeled pixel positions, whereas the distance map indicates the Euclidean distance to the crown boundary for each pixel. A subsequent post-processing approach was devised that fuses these outputs, leading to the classification of nine and five tree species with average F1-scores of $79.3 \pm 8.6\%$ and $87.6 \pm 4.4\%$ respectively. Noteworthy is the post-processing method's ability to craft a detailed tree species composition map, exclusively labeling pixels of the target species with elevated class membership probabilities [225].

In a 2022 study, the YOLOV3 model employed to detect and classify five urban street tree species and estimate their profiles utilizing Google Street View images [226]. The YOLOv3 model was rigorously trained on 5,480 pertinent images, allocating 80% for training and the remaining 20% for testing. After extensive training, up to 1,000,000 epochs, the optimal model was pinpointed at 980,000 epochs. This model showcased a maximum mAP of 0.564, accompanied by precision and recall rates of 0.727 and 0.634, respectively [226].

Deep learning boasts several distinct advantages in the realm of data processing and interpretation. Primarily, it possesses the capability to autonomously extract profound features

directly from images, thereby ensuring an end-to-end analytical process [227-229]. Unlike traditional methods that rely on pre-designed features, deep learning gleans patterns directly from raw data based on established extraction rules [230]. By initially comprehending simple constructs and subsequently harnessing multi-layered neural networks, it is able to represent intricate structures within high-dimensional data more abstractly, culminating in superior classification and recognition accuracy [231]. Central to deep learning's philosophy is the empowerment of the model itself to discern critical features, thereby minimizing human intervention. In essence, the deep learning paradigm manages both feature representation and classifier learning in the training phase. Its prowess is further substantiated by its remarkable successes in areas like classification [232, 233] and target detection [198, 234]. Despite its prowess, deep learning presents challenges. A primary concern is the High Computational Requirement. Implementing deep learning demands substantial computational power, particularly during training, emphasized by the dependence on specialized hardware like GPUs and TPUs [235]. Equally significant is the Dependence on Image Resolution. In vision tasks, the quality of outcomes hinges on the image's resolution, making high-quality inputs essential and, as a result, amplifying the requirements for data processing and storage [198].

## 2.6 Summery

- Modern urbanization leads to environmental challenges, including air pollution.
- Urban green spaces mitigate environmental issues, absorbing pollutants and providing ecological, psychological, and physical benefits.
- Monitoring urban green spaces is tedious and costly using traditional methods, but image acquisition technologies, like satellites, UAVs, and vehicle-mounted cameras, help streamline the process.
- Satellites offer extensive data capture and reach remote areas but face challenges in urban monitoring due to varied terrains, availability, and issues like cloud cover and ground data inconsistencies
- UAVs provide cost-effective, high-resolution monitoring of urban vegetation but grapple with battery, legal, privacy challenges, and urban complexities.
- Vehicle-based street-level imagery provides a detailed and accurate view of urban vegetation, offering insights not always achievable with satellites and UAVs.

- Hyperspectral imaging offers detailed data for applications like urban vegetation monitoring but faces challenges like data nonlinearity and obtaining valuable training samples.
- Multispectral imaging offers detailed data quickly but faces challenges such as calibration needs, lower resolution, and higher costs than RGB cameras.
- RGB imaging is a cost-effective and accessible approach in remote sensing, preferred for its lightweight equipment, higher spatial resolution, and ease of processing. With cameras available in everyday devices, their widespread use is facilitated. However, they are limited to the visible spectrum and can be affected by environmental factors, impacting image quality in conditions like low-light.
- Conventional machine learning, a component of AI, has several benefits for image processing. However, its significant drawbacks include the need for detailed data preprocessing, reduced accuracy over extended training periods, potential overfitting, and limitations in handling larger datasets and noisy data.
- Deep Learning offers an autonomous, end-to-end approach to data analysis, excelling in computer vision tasks by automatically identifying critical features. Its strengths lie in enhanced accuracy and reduced human intervention, but it demands high computational power and quality image inputs.

# 3 System design and methodology

## 3.1 Introduction

This research thesis, forming an integral part of a broader study, addresses critical ecological challenges, particularly the identification of non-native and invasive species, and the early detection of diseases in urban vegetation. Figure 3.1 illustrates this through a systematic and comprehensive flowchart, which visually differentiates the completed stages of this research from those planned for future exploration. Colored cells in the flowchart represent the steps achieved within this thesis, while uncolored cells indicate the phases intended for subsequent research.

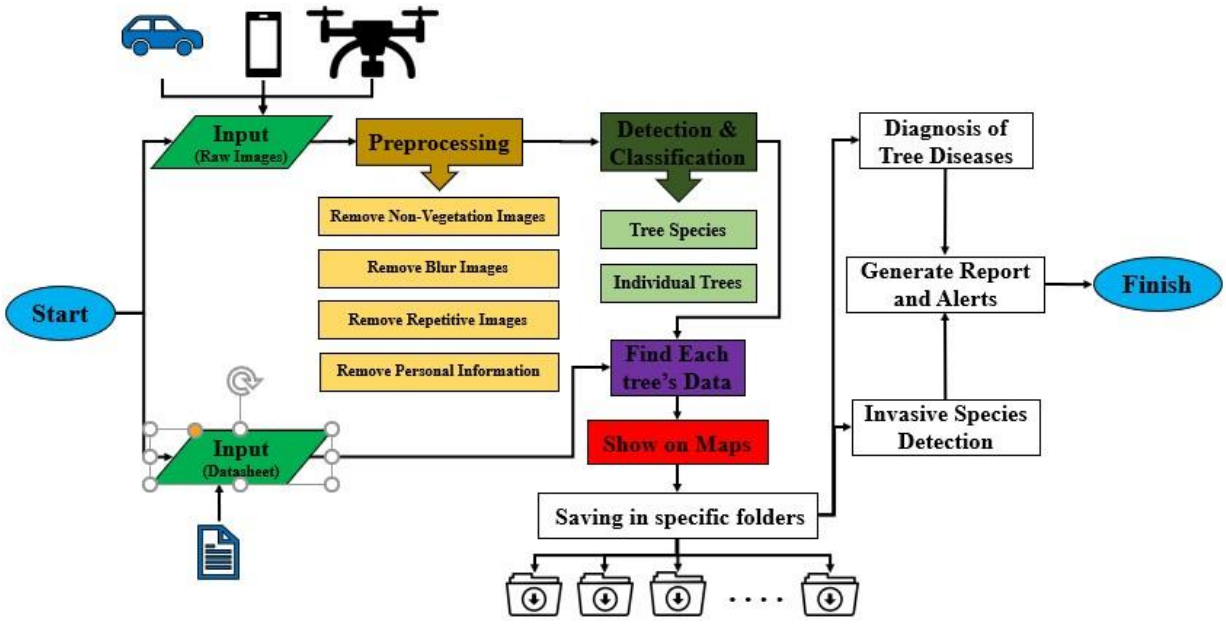



**Figure 3.1:** Overview of the broader study's framework;
colored cells show completed sections and non-colored sections represent planned for future works

Central to this thesis is the use of advanced computer vision techniques, employing RGB images from a car-mounted camera and deep learning models, specifically YOLOv5 and YOLOv8, to identify various tree species and individual trees within urban landscapes. The

research commenced with detailed data collection, focusing on camera setup and image capture, followed by a critical preprocessing stage where filters were applied to improve image quality and address privacy by removing non-vegetative and blurry images.

A key aspect of this study was the training of deep learning models. This involved preparing datasets, annotating images, and conducting a rigorous training process, which successfully led to the effective detection and classification of tree species. An important achievement of this thesis was the integration of this data with information from datasheets and the subsequent mapping of tree locations, which significantly enhanced the understanding of urban vegetation.

Looking forward, as indicated in the flowchart, the research aims to expand its scope by incorporating drone and smartphone imagery and organizing tree images into specific folders dedicated to individual trees. These steps will be crucial in future studies for monitoring tree health, detecting diseases early, and identifying invasive species. The ultimate goal is to generate comprehensive reports on each tree's status, thereby contributing to more effective urban environmental management.

The System design and methodology chapter is meticulously structured, providing a comprehensive view of the research's technical foundations. It encompasses several key areas, starting from the initial data collection and extending through preprocessing, deep learning model training, and vegetation type detection. This chapter highlights the thoroughness of the research process, from the initial setup and image capture to the deployment of advanced YOLO models for data analysis, underscoring the robustness and replicability of the study.

## 3.2 Data collection

The quality of the dataset—specifically, the visual images—plays a pivotal role in the accuracy of recognition algorithms within computer vision systems [236]. Thus, the data collection process becomes a cornerstone for the success of such systems. For the purpose of this research, data was captured through a combination of common hardware and meticulously planned capture sessions. To ensure the accuracy of models, the data was required to be of high quality, representative of varied urban vegetation, and well-documented. Ensuring this level of quality necessitates a detailed examination of the technical aspects involved in data acquisition. The following subsections will therefore explore the infrastructure of the imaging system (Section 1.2.1) and the image capture process (Section 1.2.2). These sections are integral to understanding the systematic

approach adopted to collect a dataset capable of training a computer vision system with high precision and reliability in urban vegetation recognition.

### 3.2.1 The infrastructure of the imaging system

The infrastructure of the imaging system serves as the technical backbone of the data collection process. This encompasses all the physical and software components that, in concert, capture the visual data required for analysis. The design and integration of these elements are crucial in creating a reliable and efficient data acquisition system. High-fidelity cameras, robust computing hardware, precise GPS modules, and tailored frame designs constitute the hardware suite, while sophisticated software ensures these components work seamlessly together. The hardware is tasked with capturing high-resolution images under various urban conditions, and the software synchronizes image capture with location data and system controls. Together, they form a comprehensive system engineered to meet the demanding requirements of urban vegetation mapping through computer vision. The following subsections provide an in-depth look at each component's role and specifications, as well as the rationale behind their selection and configuration for this research project

#### 3.2.1.1 Hardware

The hardware deployed for data collection forms the cornerstone of the imaging infrastructure. Figure 3.2 illustrates the complete setup of the imaging system hardware, encompassing the camera (A) with its lens (B), computer (C), battery (D), GPS module (E), GPS antenna (F), and Wi-Fi router (G). Each of these components plays a pivotal role in the system's functionality and will be discussed in their respective sections. The last part of the hardware section is related to the Frame design which is crucial for the physical setup, ensuring that all hardware components are securely mounted and aligned for optimal data capture.

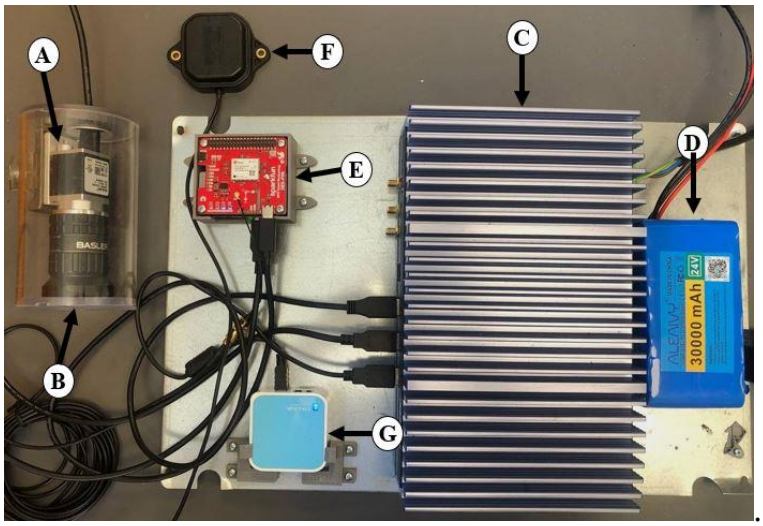



**Figure 3.2:** Overview of the complete imaging system hardware;
A. the camera with B. lens, C. computer, D. batteries, E. GPS module with F. antenna, and G. Wi-Fi router

3.2.1.1.1 Camera and lens

The Basler a2A5320-23ucBAS RGB camera (Figure 3.2.A), with a 16.1-megapixel Sony IMX542 CMOS sensor, captures high-resolution images (5328 x 3040 pixels) at 23 frames per second (Figure 3.3). This frame rate is effective for capturing stationary details from a moving vehicle, essential for urban data collection. The adjustable settings of this camera allow it to efficiently adapt to the dynamic lighting and motion conditions typical of urban environments. This capability is helpful in meeting the research objectives, which include compiling a detailed visual inventory of urban vegetation.


**Figure 3.3:** The Basler a2A5320-23ucBAS RGB camera.

In the pursuit of capturing the most informative images possible, three Basler lenses with different focal lengths—50mm, 25mm, and 12mm—were rigorously evaluated (Figure 3.4.A). The selection of the lens was crucial to maximizing the amount of detail and information in each captured image. Following extensive testing, the 12mm lens was chosen for its wider field of view, both horizontally and vertically, making it particularly suitable for the urban environment. To visually demonstrate the differences in the capabilities of these lenses, individual sample output images for each lens are presented (Figure 3.4.B for the 50mm lens; Figure 3.4.C for the 25mm

lens; Figure 3.4.D for the 12mm lens). Additionally, a comparative image (Figure 3.4.E) has been created to juxtapose the field of view and angle of view vertically for each lens, providing a clear and direct comparison of their respective focal lengths and imaging characteristics.

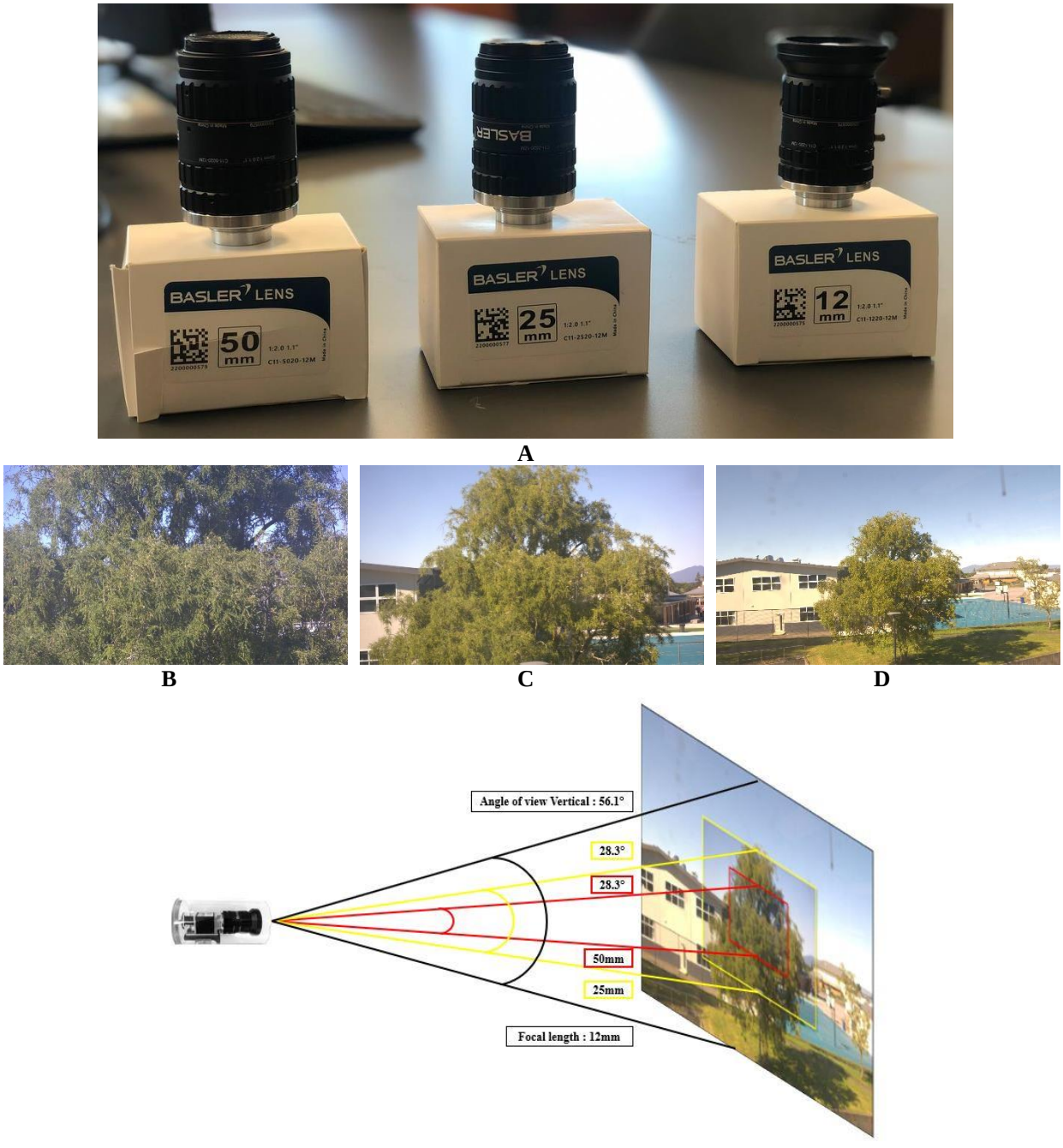



**Figure 3.4:** Comprehensive display and comparison of tested Basler lenses.
A. shows the 50mm, 25mm, and 12mm lenses. B. is a sample image captured with the 50mm lens, C. with the 25mm lens, and D. with the 12mm lens, all taken from a fixed distance to illustrate the output variation. E. provides a comparative analysis of the lenses, highlighting differences in focal length and vertical angle of view.

An essential aspect of the camera setup was determining the optimal mounting angle (α) on the vehicle to ensure effective data acquisition (Figure 3.5.A). Through several trial runs, it was observed that a camera angle of less than 45 degrees often resulted in many images being blurred (Figure 3.5.B). This blurring was likely due to the motion of the vehicle and the speed of passing objects. On the other hand, a camera angle greater than 45 degrees frequently captured multiple trees in a single frame, which posed challenges for the process of individual tree identification and analysis (Figure 3.5.D). Consequently, a 45-degree angle was identified as the optimal orientation for the camera (Figure 3.5.C). This decision aligned with findings from other research, which similarly recommended this angular setup for effective imaging [11].

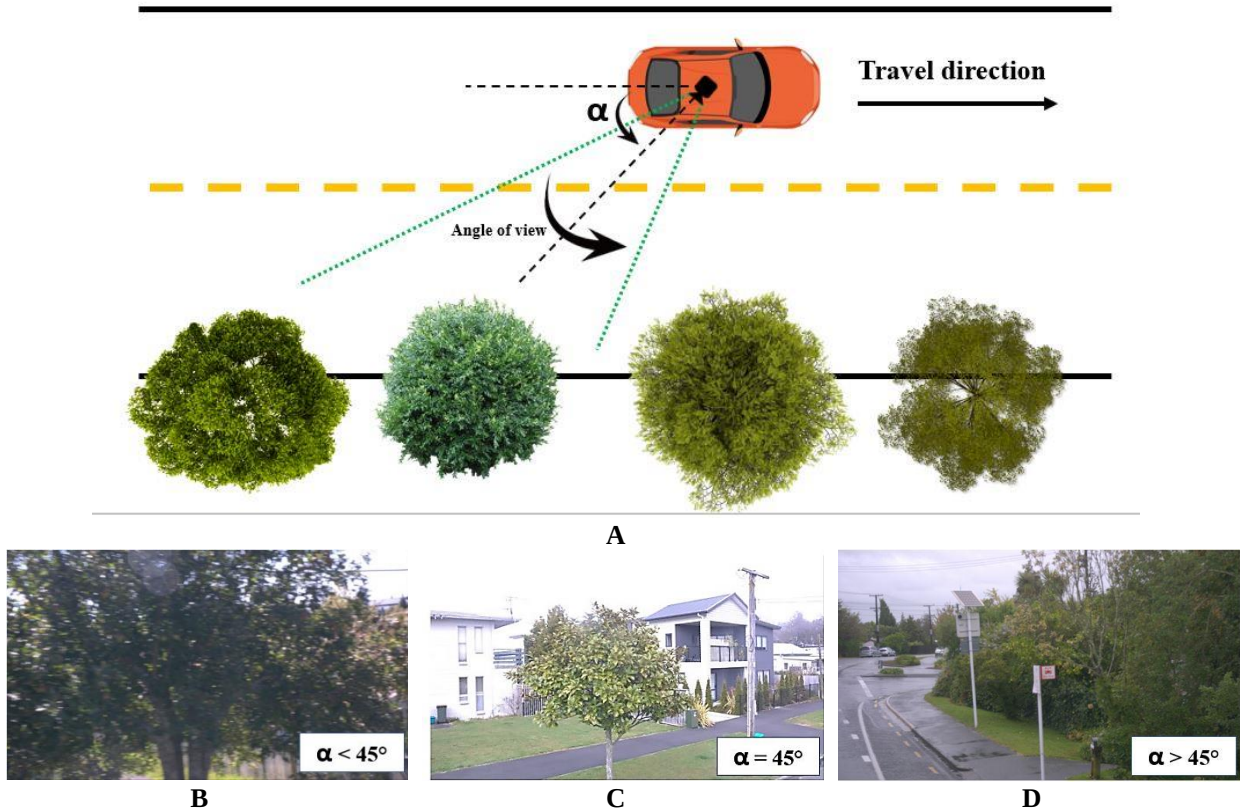



**Figure 3.5:** Camera angle on the car roof and its effects on images.
A. Depicts the camera's placement and its 45-degree angle on the car roof. B. Shows a sample image captured with the camera angle set to less than 45 degrees, C. Presents a sample image captured at the optimal 45-degree angle, D. Illustrates a sample image captured with the camera angle set to more than 45 degrees.

3.2.1.1.2 Computer

The computational core of the data collection system is the BOXER-6641-A1, a fanless industrial computer from AAEON (Taiwan) (Figure 3.2.C). For reliable and continuous operation during field data acquisition, the system was powered by two batteries, each with a capacity of

30000 mAh (Figure 3.2.D). The computer is powered by an Intel(R) Core (TM) i7-8700T CPU @ 2.40GHz and is equipped with 32 GB of installed memory (RAM), alongside an Intel(R) UHD Graphics 630 GPU. This setup provides substantial processing power necessary for the imaging tasks. Its fanless design contributes to reduced mechanical wear and enhanced longevity. Furthermore, the system is capable of operating across a wide voltage range from 10V to 35V and can withstand a broad temperature spectrum from -20°C to 55°C, which is essential for maintaining operational stability in diverse urban environments (Figure 3.6).

**Figure 3.6:** The fanless AAEON BOXER-6641-A1 computer.

3.2.1.1.3 GPS module and antenna

In addition to the importance of the imaging system, analyzing the collected images in conjunction with the information recorded by the city's council for each tree is crucial. This information includes each tree's scientific and common names, area (such as urban, rural, park, etc.), and latitude and longitude. It is important to note that using geographical coordinates (latitude and longitude) is the most precise and efficient method for matching each tree's image with its recorded information. Therefore, alongside the hardware dedicated to imaging, a GPS module (Figure 3.2.E) is necessary to record the geographical coordinates of each photograph. In this study, the u-blox ZED-F9R-00B module (Figure 3.7) and a GPS antenna (Figure 3.2.F) was employed for this purpose.

**Figure 3.7:** The u-blox ZED-F9R-00B module.

3.2.1.1.4 Wi-Fi router

The last component of the imaging system's hardware is the TP-Link TL-WR703N (China) (Figure 3.2.G), a compact wireless router. This device serves as a critical link between the computer and the operator's laptop. The TL-WR703N is configured to create a local Wi-Fi

network, to which the laptop can connect (Figure 3.8). Once this wireless connection is established, the laptop gains access to the imaging system computer's interface, enabling remote operation and monitoring of the imaging system. This setup is particularly advantageous for fieldwork, as it allows for flexibility and mobility, eliminating the need for direct physical interaction with the imaging system's computer for system adjustments or data retrieval.


**Figure 3.8:** The TP-Link TL-WR703N Wi-Fi router.

3.2.1.1.5 Frame design

As mentioned in the previous sections, the basis of this research involves using images collected by cameras mounted on vehicles. Consequently, the imaging system requires a frame that can be attached to a vehicle. A key feature considered in the frame's design is its ability to be mounted on the roofs of various vehicles (specifically those at Waikato University) and then unmounted after data collection is completed. This frame was meticulously developed in SolidWorks software version 2018, and the final design was constructed following approval. The precise dimensions of the frame are detailed in the appendix section. As shown in Figure 3.9, this frame consists of three parts. Part 'A' corresponds to the main frame, which is mounted on the roof of the car. The other two sections, labeled 'B' and 'C', are respectively related to the box that houses the computer and the camera mounting location.

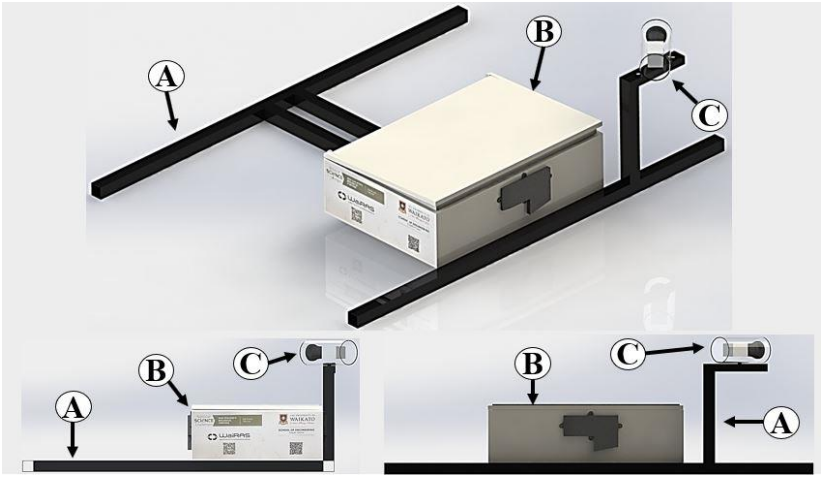



**Figure 3.9:** Designed frame in Solidworks 2018;
A. main-frame, B. Computer box and C. Camera.

#### 3.2.1.2 Software

In the software domain, the designed system hinges on a Python-based framework, which capitalizes on the ROS (Robot Operating System) for efficient message passing and process management. Central to software infrastructure is the ability to process and save images from two different sources. Scripts utilize OpenCV for image processing and cv_bridge for converting between ROS image messages and OpenCV image formats. It maintains a real-time link with GPS data, employing a subscriber model to receive updated coordinates. These coordinates are then used to tag captured images with their respective geographical locations, effectively mapping images to their real-world coordinates.

Furthermore, Scripts demonstrates a sophisticated approach to file management and data recording. Two separate image streams are handled concurrently, with images being saved to designated file paths and their metadata (including GPS coordinates and timestamps) being logged in a CSV file. This meticulous data logging is crucial for later analysis or review. Additionally, scripts include functionality to dynamically update the file paths for image storage, allowing for greater flexibility during operation. The system's shutdown procedure is designed to ensure all data is correctly saved before termination, reflecting the emphasis on data integrity and robust operation in diverse environments. This software suite not only exemplifies integration of various Python libraries and ROS capabilities but also underlines the system's adaptability and reliability in field operations. The scripts are available online on GitHub[1].

[1] GitHub link: https://github.com/Sami3610/BioVison/tree/main/Imaging_system/Drivers

With the assembly and preparation of the imaging system now complete, encompassing both the hardware and software configurations, the focus shifts to the practical application of this sophisticated setup. The next step involves the selection of specific routes for data collection, tailored to target specific tree species. This route creation is not just about navigating through urban spaces; it involves an approach considering various factors such as tree distribution, species prevalence, and accessibility. The process and considerations behind designing these routes will be discussed in the following subsection.

## 3.3 Data collection strategy

In this section, the comprehensive data collection strategy employed in the research is outlined. This strategy is twofold, encompassing both the meticulous planning of routes for effective data capture and a detailed overview of the tree species that were the focus of the study.

### 3.3.1 Route planning for data collection

Given the system's battery limitations, routes were designed to maximize species coverage. The most important data used in this research for creating these imaging routes were the datasheets provided by the city councils (Auckland, Hamilton, and Cambridge datasheets are available). As mentioned in previous sections, these datasheets contain comprehensive information about each tree, such as its scientific and common names, the area where the tree is located, its geographical coordinates, etc. Figure 3.10, which is a flowchart, illustrates how these datasheets were utilized to create the routes for image capture.

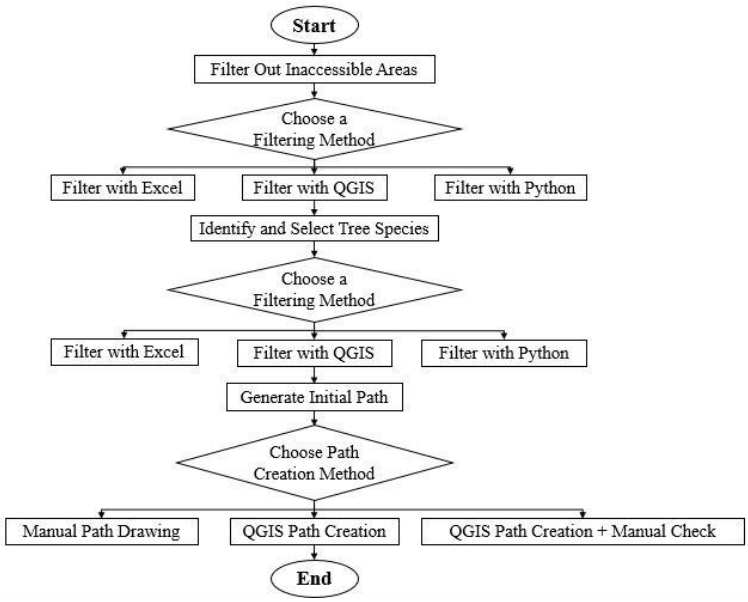



**Figure 3.10:** Simplified workflow for creating a tree photography route from council's datasheets.

As illustrated in the flowchart, inaccessible areas such as parks, lakes, and protected zones were filtered out, reducing the Cambridge dataset from 15,319 trees to 6,693 (43.7%) accessible trees, achieved using Excel, QGIS, or Python (Figure 3.11), Following this filtering, species with the highest frequency in accessible areas were selected as target species. Non-target species, about 54% of the remaining dataset, were excluded, ensuring a focus on the most prevalent species aligned with research objectives.

**Figure 3.11:** Examples of areas that cannot be accessed by the vehicle-mounted camera in Cambridge; colored dots represent trees.

The final step in the process is Route Creation. This involved two methods: a manual approach and an automated approach using QGIS software. The manual approach is accurate but time-consuming (Figure 3.12.A). Conversely, the automated approach, created in QGIS with the ROS tools plugin (Figure 3.12.B), allows for the quick development of complex routes, albeit with a higher likelihood of errors. Combining these methods provided optimal results by generating preliminary routes in QGIS and manually refining them for accuracy (Figure 3.12.C). During data

collection, vehicle speed was regulated between 20–40 km/h to optimize image quality and reduce motion blur.


**Figure 3.12:** Path creation methods;
A. Illustration of a manually drawn route in Cambridge, B. Automated route generation using QGIS and ROS tools, C. Example of manual verification and optimization of an algorithm-generated route.

### 3.3.2 Overview of selected tree species

The primary factor for selecting tree species was their abundance, as a sufficient number of images are needed for training and validation of the model. Given that most tree species in New Zealand are well-known [237], the selection was based solely on the count of each species that was accessible. For example, in the Cambridge datasheet, about 135 different tree species are recorded as being accessible for the imaging system, with their frequencies ranging from 1969 for Japanese Maple to just 1 for Purple Birch. After meticulous analysis, the final selection of eight tree species was made, specifically: *Prunus Species* (Flowering Cherry), *Cornus Florida* (Flowering Dogwood), *Acer Palmatum* (Japanese Maple), *Platanus X Acerifolia* (London Plane), *Liquidambar Styraciflua* (Liquidambar), *Magnolia Grandiflora* (Bull Bay), *Quercus Palustris* (Pin Oak), and *Betula Pendula* (Silver Birch). These species were deemed most suitable for the aims of this research. Images of these selected species are presented in Figure 3."

**A**

**B**

**C**

**D**


**Figure 3.13:** Collection of selected tree species;
A. *Prunus Species* (Flowering Cherry), B. *Cornus Florida* (Flowering Dogwood), C. *Acer Palmatum* (Japanese Maple), D. *Platanus X Acerifolia* (London Plane), E. *Liquidambar Styraciflua* (Liquidambar), F. *Magnolia Grandiflora* (Bull Bay), G. *Quercus Palustris* (Pin Oak), and H. *Betula Pendula* (Silver Birch).

After each data collection session, a wide range of images is acquired, which comprises a mix of redundant, blurry, and vegetation-free images among the captured data. This assortment is referred to as the 'raw dataset.' If these irrelevant images are not removed, they can significantly increase the time required for image analysis. Consequently, the following section, dedicated to preprocessing, will discuss the methods employed to eliminate such images, thereby streamlining the dataset into a more effective and useful collection for analysis.

## 3.4 Preprocessing

In this section, the focus is placed on the important steps undertaken to transform the raw dataset into a form suitable for detailed analysis. An initial review of the raw datasets obtained from the imaging system revealed that in a single data collection session, covering a distance of 10 kilometers at the specified speed mentioned in the previous section, approximately 6,000 images are captured within a span of 45 minutes. These images are the result of the system's continuous operation from activation to deactivation. The preprocessing stage, therefore, is essential in managing this large volume of data, ensuring that only the most relevant and high-quality images are retained for further analysis.

The preprocessing process involves a series of steps, each critical in transforming the raw dataset into a format optimized for analysis. According to Figure 3.14 that outlines all the steps in this section, the process begins with the removal of non-vegetation images. This step is crucial to ensure focus on pertinent data, with examples illustrated in Figure 3.15.A. Following this, the elimination of blurry images (Figure 3.15.B) is undertaken, where different techniques were tested

to identify and remove images lacking clarity. The next crucial step is the identification and removal of repetitive images (Figure 3.15.C). A variety of methods were trialed to efficiently reduce data redundancy. Additionally, privacy protection is a key aspect of preprocessing, involving the blurring of personal information within the images, as demonstrated in Figure 3.15.D. These steps are integral to refining the dataset, ensuring it is prepared for analysis and aligned with the research's goals.

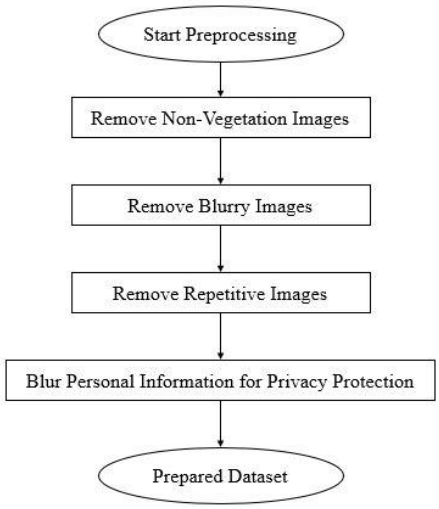



**Figure 3.14:** Flowchart of the preprocessing steps

C

E

**Figure 3.15:** Examples of images from raw datasets that should be removed or edited; A. non-vegetation images, B. blur images, C. repetitive images, and D. images with privacy information such as car plate number, human faces, etc.

### 3.4.1 Remove non-vegetation images

While capturing images in urban areas with the imaging system, a variety of non-target elements such as buildings, vehicles, and urban infrastructure are inevitably included (Figure 3.15.A). These non-vegetation components can significantly clutter the dataset, leading to inefficiencies in processing and analysis. To tackle this issue, a specialized method was employed involving the use of a trained YOLOv5 model, designed to detect urban vegetation. This approach, detailed in the 'Training a model for detecting urban vegetation' section, enables the precise identification of images containing vegetation. After training this model, each image from the data collection sessions is analyzed, and only those with urban vegetation, irrespective of species, are retained. Non-vegetation images, which do not contribute to the research objectives, are removed from the raw dataset. This selective filtering process is important for maintaining a dataset that is both relevant and manageable, thereby enhancing the overall efficiency and accuracy of the subsequent analytical phases of the study.

### 3.4.2 Remove blur images

Blur in an image is a prevalent issue in digital image processing, often challenging to rectify due to factors like environmental interference and camera movement [238, 239]. In various professional fields such as remote sensing, astronomy, microscopy, and medical imaging, blurry images can significantly impede research and diagnostics [238]. The primary reason behind blurry images is typically the dynamic movement of the lens during capture, where the lens fails to set a proper angle and focus [240, 241]. Digital images, composed of pixels in a grid formation, lose clarity when these pixels do not represent the intended tone due to such disturbances [238].

In the context of this research, where the imaging system is mounted on a vehicle, similar challenges are encountered. The vehicle's movements, influenced by potholes (Figure 3.16.A), speed bumps (Figure 3.16.B), and uneven road surfaces, lead to significant camera instability, resulting in blurred images (Figure 3.15.A). Blurry images lack the clarity needed for precise analysis, so they must be filtered from the dataset. To achieve this, two techniques were used: Fast Fourier Transform and Laplacian Variance, each further explained below.

**A** **B**

**Figure 3.16:** Common urban road conditions leading to camera shake;
A. pothole and B. speed bump.

3.4.2.1 Fast fourier transform technique

The Fast Fourier Transform (FFT) is one of the most common techniques in the identification of blurred images. FFT, a powerful tool in digital image processing, transforms a signal from its original domain (often time or space) into a representation in the frequency domain [242].

In the context of image preprocessing, FFT is utilized to analyze the frequency components of an image. Blurred images typically display a distinct pattern in the frequency domain, characterized by a scarcity of high-frequency components, which are indicative of sharp edges and fine details [243]. By applying FFT to each image, it becomes possible to differentiate between clear and blurred images based on their frequency spectra. This differentiation is particularly evident when assessing the amount of high-frequency content; a low quantity of high-frequency components usually signifies that the image is blurry [242].

Building upon the foundational principles of FFT in distinguishing between clear and blurred images, a Python script was developed to operationalize this technique. The script follows a four-step process to identify blurred images effectively. Initially, after converting RGB images to grayscale, FFT is applied to convert each image from the spatial domain to the frequency domain, where low frequencies are positioned at the corners and high frequencies at the center (Figure 3.17). The script then employs "np.fft.fftshift" function to shift the zero-frequency component to the center, simplifying the manipulation of the central high-frequency components.

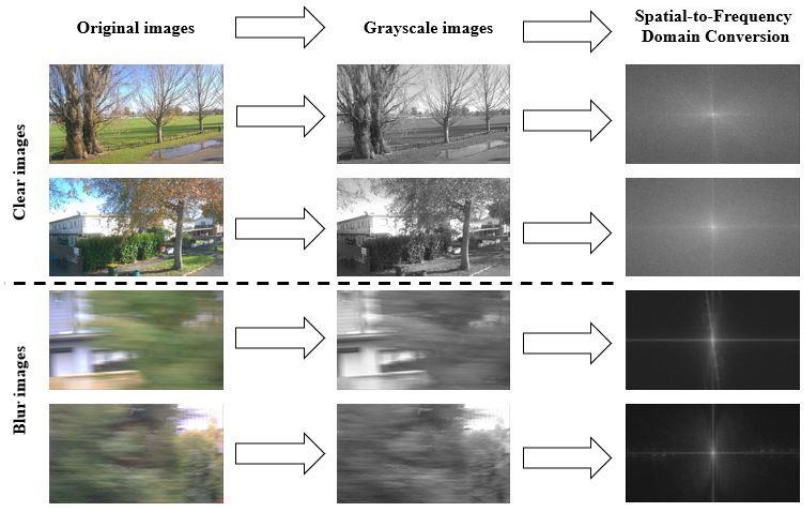



**Figure 3.17:** Original, grayscale, and frequency domain images;
Left: Original images featuring two clear and two blurred examples. Middle: Grayscale versions of the original images. Right: Spatial-to-frequency domain conversions of each grayscale image, illustrating their frequency domain representation.

Subsequently, a rectangular filter is applied within this frequency domain: a specified rectangular area at the center, where the high frequencies are located, is set to zero. This step is crucial in selectively filtering out certain high-frequency components from the image. The size of this rectangular area, which is changeable, is defined by the 'size' parameter in the script (Figure 3.18). After applying this filter, an inverse FFT "np.fft.ifft2" function is executed to revert the modified frequency domain back to the spatial domain (Figure 3.19).

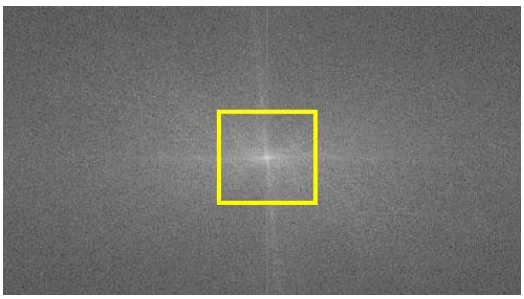


**Figure 3.18:** Frequency domain with rectangular filter which shows the frequency domain after FFT; highlighting a central rectangular area set to zero to filter out specific high frequencies.

The final step involves calculating the mean of the magnitude values from the transformed image and comparing it against a predefined threshold to determine if an image is blurry. In the written script, if this calculated value is less than the threshold, the image is classified as blurry, and vice versa. It’s noteworthy that the threshold value used was determined through trial and error, optimizing it for the most accurate blur detection in our dataset.

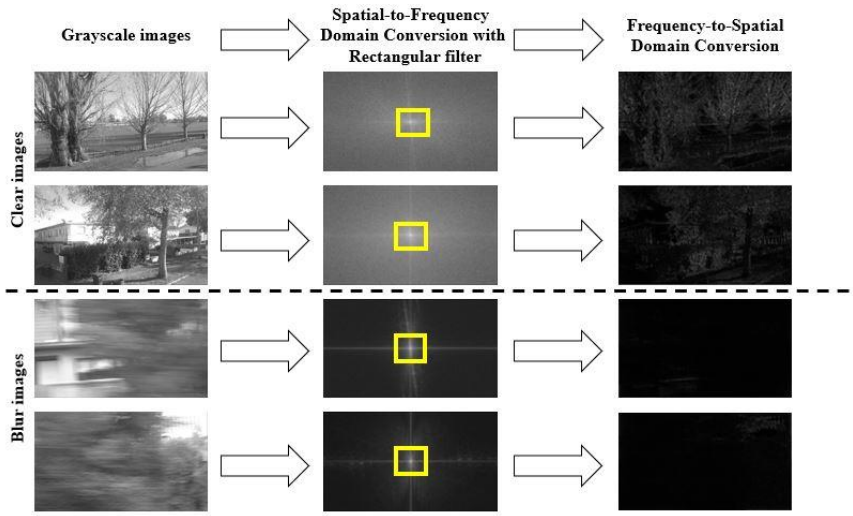



**Figure 3.19:** Grayscale, frequency domain with rectangular filter and inverse FFT Images; Left: Grayscale images featuring two clear and two blurred examples. Middle: Spatial-to-frequency domain conversions of each grayscale image with a rectangular filter that shows the area where is set to zero. Right: Inverse FFT images from frequency to spatial domain.

For finding the optimum threshold value and rectangular size, a selection of 100 blurry and 100 clear images were manually chosen and placed in a folder. These images were then separated using the written script. After each separation, the accuracy of the differentiation was evaluated based on the threshold level. Ultimately, the threshold that yielded the highest accuracy in separation was selected. In a similar manner, the most appropriate size for the rectangular filter was also determined. The script is available online on GitHub[1].

In summary, the Fast Fourier Transform technique provides a method for identifying blur in images by analyzing their frequency spectra. Next, the Laplacian Variance technique will be discussed, which is another pivotal tool for finding blur images. The Laplacian Variance technique offers a different perspective by focusing on the spatial domain rather than the frequency domain, evaluating image sharpness through edge detection. The subsequent section will explore the technicalities of the Laplacian Variance technique.

### 3.4.2.2 Laplacian Variance technique

Laplacian variance is used as a metric to assess the quality of an image in terms of blur. To calculate the variance of the Laplacian, the Laplacian operator, a second-order derivative method, is applied to the image; this operator effectively highlights areas of rapid intensity change, which

[1] GitHub link: https://github.com/Sami3610/BioVison/tree/main/Filters/Blur_images_filter/FFT

are indicative of edges [239]. Mathematically, for a two-dimensional image I (x, y), the Laplacian L (x, y) is given by ([244]):

$$L(x,y) = \frac{\partial^2 I}{\partial x^2} + \frac{\partial^2 I}{\partial y^2} \quad (1)$$

The equation (1) represents the sum of the second derivatives of the image intensity I with respect to the x and y coordinates. It calculates the difference in intensity for a pixel compared to its surrounding neighbors in both the horizontal and vertical directions.

| | | | | | | | | |
|---|---|---|---|---|---|---|---|---|
| **0** | **1** | **1** | **2** | **2** | **2** | **1** | **1** | **0** |
| **1** | **2** | **4** | **3** | **3** | **3** | **4** | **2** | **1** |
| **1** | **4** | **5** | **3** | **0** | **3** | **5** | **4** | **1** |
| **2** | **3** | **3** | **12** | **24** | **12** | **3** | **3** | **2** |
| **2** | **3** | **0** | **24** | **40** | **24** | **0** | **3** | **2** |
| **2** | **3** | **3** | **12** | **24** | **12** | **3** | **3** | **2** |
| **1** | **4** | **5** | **3** | **0** | **3** | **5** | **4** | **1** |
| **1** | **2** | **4** | **3** | **3** | **3** | **4** | **2** | **1** |
| **0** | **1** | **1** | **2** | **2** | **2** | **1** | **1** | **0** |

**Figure 3.20:** Laplacian operator applied to a 9x9 pixel matrix [244]

Figure 3.20 shows the effect of applying the Laplacian operator to an image section. Each cell in the matrix corresponds to a pixel, with the number indicating the pixel's intensity value post-Laplacian application. The central region, with higher values, depicts an area of high intensity change, characteristic of sharp edges or significant texture contrast. This matrix visually demonstrates how the Laplacian operator accentuates regions with rapid intensity changes, aiding in the detection and analysis of image sharpness and texture.

By calculating the variance of the Laplacian, one can determine the extent of blur in the image. A higher variance indicates less blur, while a lower variance indicates more blur. The variance of the Laplacian L can be computed as:

$$Var(L) = \frac{1}{N}\sum_{i=1}^{N}(L_i - \mu)^2 \quad (2)$$

Where, $L_i$ represents each pixel value in the Laplacian image, μ is the mean of all Laplacian values, N is the total number of pixels, and the summation is over all pixels. Equation (2) calculates

the average of the squared differences from the mean, providing a measure of the image's edge sharpness.

Based on the explanations provided, a Python script was developed to identify blurred images in the dataset. According to the script, initially, each image is converted into grayscale, which simplifies the data for analysis. This preprocessing step is crucial as it facilitates a simpler computation of the Laplacian variance.

Subsequently, the Laplacian operator, an edge detection tool, is applied to each image, followed by the computation of its variance. This variance acts as an indicator of the image's sharpness, quantifying the presence of clear edges. Based on a predetermined threshold, the script categorizes each image as either blurry or clear. This threshold can be changed according to the characteristics of the dataset. Images that exhibit Laplacian variance values lower than this threshold are classified as blurred. The script is available online on GitHub[1].

Having effectively addressed the challenge of blur images using the Fast Fourier Transform and Laplacian variance techniques, the thesis focus now shifts to another crucial aspect of image preprocessing: the removal of repetitive images. This next section will delve into techniques employed to identify and eliminate duplicate or near-duplicate images from datasets. This step is essential to ensure the uniqueness and quality of the data, further refining our analysis and contributing to the robustness of the study's findings.

### 3.4.3 Remove repetitive images

Continuous imaging in urban areas often results in repetitive images (Figure 3.15.C) with similar content, primarily when the vehicle stops due to traffic lights or traffic rules. Removing these duplicates significantly reduces analysis time. To address this, three methods were evaluated for identifying and eliminating repetitive images: Mean Squared Error, Normalized Cross-Correlation, and GPS-based filtering, each discussed in the subsequent subsections.

#### 3.4.3.1 Mean Squared Error technique

Mean Squared Error (MSE) is established as an effective signal fidelity measure in image analysis. It provides a quantitative score to assess the similarity or error between two images [245].

[1] GitHub link: https://github.com/Sami3610/BioVison/tree/main/Filters/Blur_images_filter/Laplacian_Variance

Typically, one image is considered the original, while the other is a distorted or altered version. In the context of images, MSE is defined as [245, 246];

$$MSE(x,y)=\frac{1}{N}\sum_{i=1}^{N}(x_i-y_i)^2 \tag{3}$$

Where $x=\{x_i|i=1,2,...,N\}$ and $y=\{y_i|i=1,2,...,N\}$ represent the pixel values of the two images and N is the total number of pixels.

The MSE's attractiveness lies in its simplicity and effectiveness. It is parameter-free, inexpensive to compute, and memoryless, allowing the squared error to be evaluated independently at each pixel [245].

In this research, a Python script was employed to detect and remove similar images from collected datasets using the Mean Squared Error (MSE) technique. To identify duplicate images, a specific MSE threshold was initially established through a process of trial and error, where a lower MSE value indicated greater similarity between images. Subsequently, each image underwent a systematic process: they were loaded, resized to a uniform size, and converted into a grayscale pixel array. This standardization was essential for the accurate calculation of MSE. The core functionality of the script involved calculating the MSE for each pair of images by comparing their pixel intensity values, thus determining the average squared differences. When the MSE value of a pair fell below the predetermined threshold, the images were identified as duplicates. These duplicates were then relocated to a designated output folder, significantly streamlining the image dataset by eliminating redundant images. The script is available online on GitHub[1].

Based on the experiences gained in this research, it was observed that the Mean Squared Error (MSE) technique is sensitive to changes in brightness and contrast. This means that variations in lighting conditions between two images can result in high MSE values, even if the images are structurally similar, as demonstrated in Figure 3.21, which provides an example of this limitation. Consequently, another technique, the “Normalized Cross-Correlation technique”, which is not sensitive to differences in brightness, was also tested for detecting and removing similar images. This technique and its application will be discussed in the subsequent subsection.

[1] GitHub link: https://github.com/Sami3610/BioVison/tree/main/Filters/Similar_image_filter/similarity.detection_MSE

**Figure 3.21:** Three similar images with varying brightness levels; unrecognized as similar by the MSE technique.

### 3.4.3.2 Normalized cross-correlation technique

Normalized cross-correlation (NCC), a robust mathematical tool in the fields of signal and image processing, is employed for matching features, analyzing similarity, tracking motion, and recognizing objects [247-249]. It is an effective method for measuring similarity between images, with its main advantage being its insensitivity to variations in brightness and contrast [250]. For comparing two images based on NCC, the equation (4) is used in this context is as follows [250]:

$$NCC(A,B) = \frac{1}{N}\sum_{x,y}\left(\frac{(A(x,y)-\mu_A)\times(B(x,y)-\mu_B)}{\sigma_A \times \sigma_B}\right) \quad (4)$$

Where A and B represent the two images being compared. The pixel values at any given position (x, y) in these images are denoted by A(x, y) and B(x, y), respectively. Additionally, $\mu_A$ and $\mu_B$ indicate the mean pixel values of images A and B, while $\sigma_A$ and $\sigma_B$ represent the standard deviations of the pixel values in these images, and the total number of pixels in the images is represented by N. This equation involves a summation over all pixel positions (x, y) in the overlapping region of the images, ensuring a comprehensive comparison.

A Python script was utilized for employing the Normalized Cross-Correlation (NCC) technique. Firstly, a threshold should be considered which is selected using trial and error. According to this threshold, images with an NCC value above this threshold would be considered similar. Images are systematically resized to 256x146 for computational efficiency and converted to grayscale. The script then calculated NCC values between pairs of images using "cv2.matchTemplate", recording the maximum NCC value for each pair to determine similarity. Pairs with NCC values exceeding the set threshold are identified as similar and removed from datasets. The script is available online on GitHub[1].

[1] GitHub link: https://github.com/Sami3610/BioVison/tree/main/Filters/Similar_image_filter/similarity.detection_NCC

The NCC technique, in practical evaluations, demonstrated that it does not possess the limitations inherent in the Mean Squared Error technique. However, this research also investigated another approach, which was based solely on using GPS coordinates recorded in the names of each image. This method will be discussed in the following subsection.

#### 3.4.3.3 Using GPS coordinates

Utilizing GPS information to identify similar images, unlike the previous methods, does not involve any computations on the pixel values of each image, resulting in significantly higher computational speed. Generally, the process involves subtracting the geographic longitude and latitude of two points. If the resulting value is less than a specified amount, it indicates that the two images were taken from the same location. Considering the camera's fixed angle and the consistent positioning of the camera on the vehicle, it can be concluded that the two images are similar.

For this, a Python script was designed to extract latitude and longitude data directly from the filenames of each image. The script's primary function is to identify images with extremely close GPS coordinates. This is accomplished by determining whether the absolute differences in latitude and longitude between any two images are less than or equal to a very fine threshold of 0.000001 (this value is changeable) degrees. Such small differences indicate that the images were taken in close proximity to each other. Upon identifying images that meet this criterion, the script then proceeds to remove the second image from the dataset.

Despite the high speed of this method, it also comes with fundamental limitations. The first limitation relates to the accuracy of the recorded GPS coordinates. It was sometimes observed in dataset analyses that the GPS data could be highly erroneous, preventing this method from correctly identifying similar images (Figure 3.22). Another potential issue arises with routes that have a single lane. When imaging is conducted to and from both sides of a street, two entirely different images may have very close GPS coordinates. In such cases, this method mistakenly removes one of the images automatically.


**Figure 3.22:** Comparative GPS coordinates from two datasets have been collected at different times from the same routes in Cambridge; A. Blue points represent GPS coordinates that are completely correct, and B. Blue points represent GPS coordinates that are completely wrong.

After thoroughly examining various methods for identifying and removing similar images from datasets, such as the Mean Squared Error (MSE) technique, Normalized Cross-Correlation, and the use of GPS data, the focus of this study extends beyond mere image processing. An equally critical aspect that emerges is the consideration of privacy concerns. The upcoming section, titled 'Privacy Protection: Blurring Personal Information,' will explore the specific approach adopted in this research to ensure the blurring of personal information within urban imagery. This is in line with adhering to privacy protection standards and integrating ethical considerations into data handling, reflecting a comprehensive approach to urban imaging studies.

### 3.4.4 Privacy protection: blurring personal information

Images often contain sensitive information [251-253] such as faces and vehicle license plates, raising significant privacy concerns. In this research, focused primarily on imaging urban vegetation, the inadvertent inclusion of such private details alongside pictures of trees is inevitable. Recognizing the critical need for privacy protection, these privacy-sensitive elements must be carefully removed during the final phase of dataset preparation. Given the impracticality of manually processing a large volume of images, the deployment of an automated model becomes essential. The YOLOv5 model, which has been pre-trained to identify over 80 types of objects, including humans, buses, trucks, personal vehicles, motorcycles, and bicycles, was selected for its robust capability. Consequently, this model was effectively employed to detect and remove personal information from the images, ensuring adherence to privacy standards.

For protecting privacy information in images, a custom script was developed to process images using a pre-trained YOLOv5 model, specifically designed for object detection, with an added functionality to selectively blur certain objects. The script includes a 'blur' function, which applies

a Gaussian blur to designated regions within the images. These regions are determined by bounding boxes that encapsulate detected objects. When objects are identified in an image, their bounding boxes and class labels, such as 'person,' 'car,' 'truck,' 'motorcycle,' 'bicycle,' and 'bus,' are extracted. The script then strategically blurs these objects using the Gaussian blur function to obscure them (Figure 3.23). The script is available online on GitHub[1].

A B

**Figure 3.23:** Example of privacy information filter's results; A. Original image, and B. Filtered image.

Throughout the 'Preprocessing' section, various filters and techniques were meticulously applied to refine the raw dataset. This included the removal of non-vegetation images, the elimination of blurred images using Fast Fourier Transform and Laplacian Variance techniques, the deletion of repetitive images through Mean Squared Error, Normalized Cross-Correlation methods, and GPS coordinate analysis, as well as ensuring privacy protection by blurring personal information. As a result of these comprehensive preprocessing steps, the dataset has been effectively modified and prepared, laying a solid foundation for the subsequent phase. The study now progresses to the crucial 'Deep Learning Models Training' section, where these curated datasets will be utilized to train sophisticated deep learning models, marking a significant step towards achieving the research objectives.

## 3.5 Deep Learning model training

After refining the dataset in the preprocessing stage, the next crucial step is Deep Learning Model Training, aiming to accurately detect and analyze urban vegetation. In this project, the YOLO (You Only Look Once) frameworks have been utilized, predominantly due to their

[1] GitHub link: https://github.com/Sami3610/BioVison/tree/main/Filters/Privacy_filter

efficiency as single-stage detectors. While two-stage object detectors, such as the R-CNN series, have been popular in the research community [234, 254, 255], single-stage detectors like the YOLO series [256-259] are known for their faster performance, as they simultaneously predict the bounding box and the class of objects [260].

The YOLO series, in particular, has played an important role as a single-stage detector in object detection and action detection tasks [260]. The YOLO series has evolved from YOLOv1 in 2015 through to the latest version, YOLOv8, in 2023 [261]. Each version has been instrumental in various studies for detecting, monitoring, or counting vegetation [261-273]. In this research, YOLOv5 and YOLOv8 were selected for their specific strengths: YOLOv5 has demonstrated strong performance in preliminary studies, while YOLOv8 offers state-of-the-art features. The following subsections explore each model's unique characteristics and capabilities.

### 3.5.1 YOLOv5

YoloV5 is one of the most famous detection algorithms due to its fast speed and high accuracy [260]. YOLOv5, a Part of the YOLO (You Only Look Once) series, builds upon the developments from YOLOv1 to YOLOv4 [274] and has achieved top performances on major object detection datasets like Pascal VOC [275] and Microsoft COCO [276]. The network architecture of YOLOv5 is shown in Figure 3.24. The architecture of YOLOv5 is meticulously designed for efficiency in both speed and accuracy, making it particularly suitable for applications requiring rapid and precise detection [274].

As depicted in Figure 1.23, YOLOv5's framework is composed of three distinct parts: the Backbone, the Neck, and the Head. The Backbone of the model, featuring CSPDarknet, is created by incorporating the Cross Stage Partial Network (CSPNet) [277] into Darknet and is responsible for the initial feature extraction [274]. This design helps avoid redundant gradient information, reducing the model's size—an advantage for using YOLOv5 on devices with limited resources

The Neck, utilizing the Path Aggregation Network (PANet) [278], enhances the information flow between the layers and improves the object location accuracy. PANet's structure allows low-level features to be passed up efficiently, a benefit for applications like urban environmental analysis where precise object detection is important. This aspect of the architecture is particularly crucial for urban environmental applications, where precise detection and localization are key [274].

Lastly, the Head of the architecture, namely the Yolo Layer, is tasked with generating the final detection outputs. It produces feature maps of three different sizes to facilitate multi-scale prediction, enabling the model to handle objects of varying sizes[274].

Furthermore, YOLOv5 stands out in the single-stage detector category for its regression-based approach to object detection, allowing for faster processing compared to two-stage detectors. This approach is significant for projects where speed is as important as accuracy, making YOLOv5 an ideal choice for urban vegetation analysis. The model's CSP backbone significantly reduces inference time, while the PANet structure ensures both bottom-up and up-down pathways for enhancing detection capabilities [274].

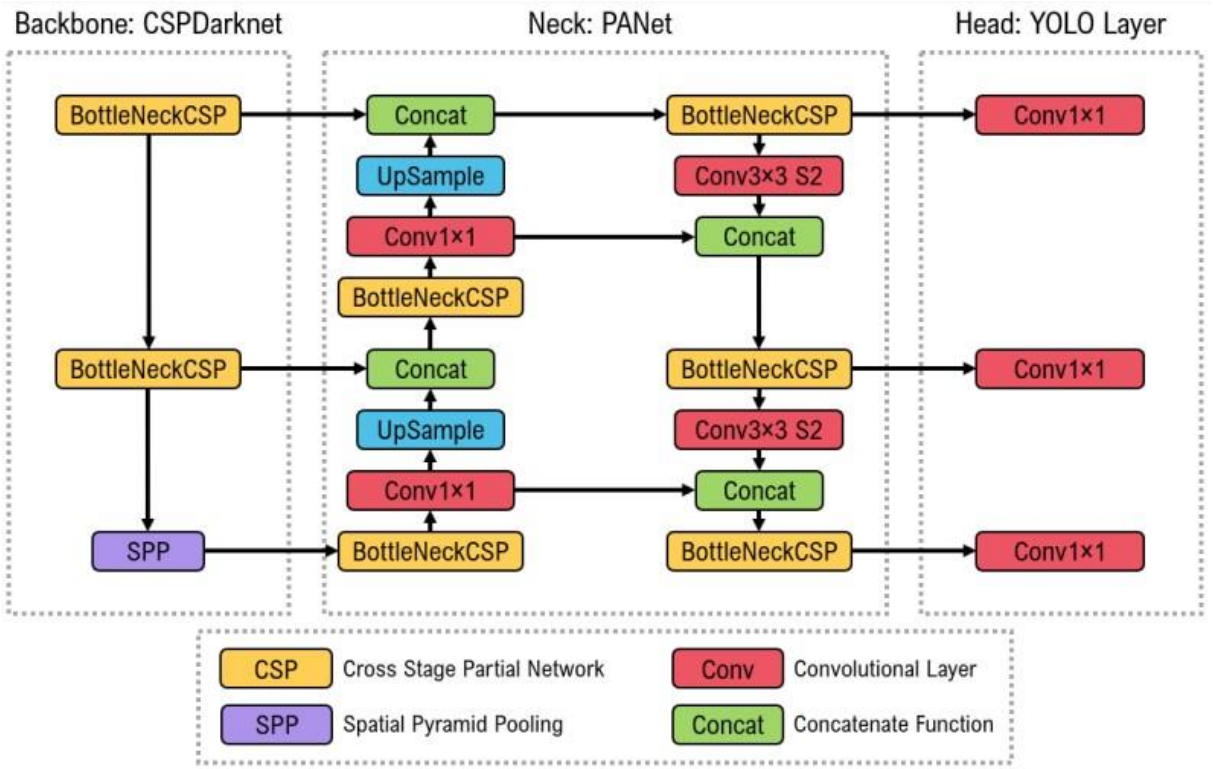



**Figure 3.24:** YOLOv5 network architecture.
This architecture is divided into three main components: (1) Backbone, featuring CSPDarknet for initial feature extraction; (2) Neck, utilizing PANet for the fusion of extracted features; and (3) Head, comprising the Yolo Layer for the final output of detection results including class, score, location, and size of the detected objects. The process begins with data input to the CSPDarknet, followed by feature integration in PANet, culminating in the Yolo Layer's detection output [274].

The diverse range of YOLOv5 sub-models is concisely summarized in Table 3.1, which presents a comparative analysis of their performance metrics. These sub-models vary in size, speed, and accuracy, offering a spectrum of options to suit different computational requirements

and detection scenarios. The smallest variant, YOLOv5n, provides the fastest detection speed at the cost of lower accuracy, making it an ideal choice for resource-constrained environments or applications where real-time processing is critical. On the other end of the spectrum, YOLOv5x, the largest model, offers the highest accuracy but requires significantly more computational resources and processing time. This model is better suited for complex scenarios where precision is paramount.

Intermediate models like YOLOv5s, YOLOv5m, and YOLOv5l offer balanced solutions, catering to scenarios that require a compromise between speed and accuracy. The table also details the models' performance in terms of mean Average Precision (mAP) across various input resolutions, inference speeds on different hardware configurations (CPU and NVIDIA V100), and computational complexity measured in millions of parameters (M) and floating-point operations per second (FLOPs). This detailed breakdown aids in selecting the most appropriate YOLOv5 variant for specific tasks in urban vegetation analysis, ensuring optimal performance tailored to the unique demands of each study.

**Table 3.1:** Comparison of YOLOv5 sub-models

| **Model** | **Size** (pixels) | **mAP$^{val}$** 50-95 | **mAP$^{val}$** 50 | **Speed** CPU b1 (ms) | **Speed** V100 b1 (ms) | **Speed** V100 b32 (ms) | **Parameters** (M) | **FLOPs** @640 (B) |
|---|---|---|---|---|---|---|---|---|
| **YOLOv5n** | 640 | 28.0 | 45.7 | 45 | 6.3 | 0.6 | 1.9 | 4.5 |
| **YOLOv5s** | 640 | 37.4 | 56.8 | 98 | 6.4 | 0.9 | 7.2 | 16.5 |
| **YOLOv5m** | 640 | 45.4 | 64.1 | 224 | 8.2 | 1.7 | 21.2 | 49.0 |
| **YOLOv5l** | 640 | 49.0 | 67.3 | 430 | 10.1 | 2.7 | 46.5 | 109.1 |
| **YOLOv5x** | 640 | 50.7 | 68.9 | 766 | 12.1 | 4.8 | 86.7 | 205.7 |

The exploration of YOLOv5's architecture, sub-models, and their respective performance metrics has provided a comprehensive understanding of its capabilities and versatility in addressing various object detection challenges. With the analysis of YOLOv5 now concluded, attention is shifted to the next subsection, where YOLOv8 will be the focus. In this section, the features of YOLOv8, the latest iteration in the YOLO series, will be explored.

### 3.5.2 YOLOv8

YOLOv8, released in January 2023, represents a significant evolution in the YOLO series, offering enhanced detection accuracy and speed [279]. This model represents the latest advancement in the YOLO object detection series, building upon the successes of previous iterations such as YOLOv5. It has been optimized for a wide range of vision tasks, including object detection, segmentation, pose estimation, tracking, and classification [280]. It should be noted that,

similar to YOLOv5, YOLOv8 offers five scaled versions: YOLOv8n (nano), YOLOv8s (small), YOLOv8m (medium), YOLOv8l (large), and YOLOv8x (extra-large). The architecture of YOLOv8, comprising a backbone, neck, and head, introduces several innovative changes for improved performance. This advanced structure is illustrated in Figure 3.25.

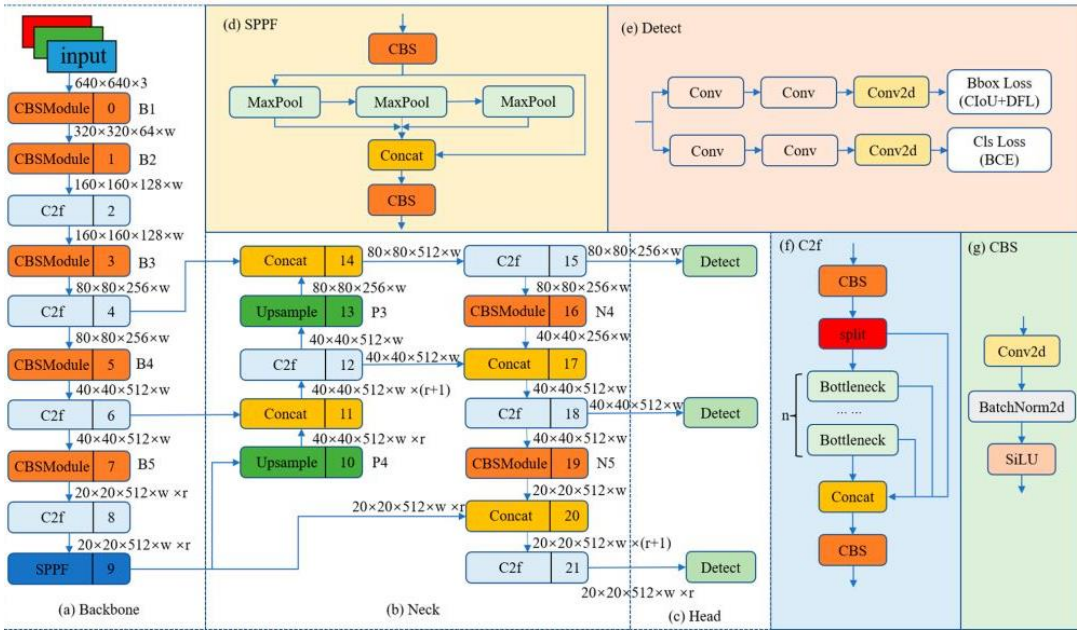



**Figure 3.25:** YOLOv8 network architecture.
This figure showcases the key components of YOLOv8: a. The Backbone with modified CSPDarknet53 and C2f module for feature extraction, b. The Neck with a PAN-FPN structure for enhanced feature fusion, and c. The Head with a decoupled structure for classification and bounding box regression [279].

YOLOv8 employs a modified CSPDarknet53 [257] as its backbone, down-sampling input features to five different scales (B1–B5), illustrated in Figure 3.25a. It replaces the original CSP module with the C2f module (Figure 3.25f), which improves information flow and enables efficient feature extraction [279, 280]. The CBS module, which combines convolution, batch normalization, and activation, produces the final output (Figure 3.25g). The backbone also includes a faster version the Spatial Pyramid Pooling Fast (SPPF) module (Figure 3.25d) to reduce processing demands and latency compared to the traditional version [279].

YOLOv8's neck features a PAN-FPN structure, inspired by PANet [281] and displayed in Figure 3.25b. This setup combines features from two different scales, improving how well the model can capture object details. The FPN (top-down) adds deeper feature information, while the PAN (bottom-up) enhances spatial accuracy, making it suitable for detecting objects at different scales [279].

YOLOv8's detection utilizes a decoupled head structure (Figure 3.25e), with separate branches for object classification and bounding box regression, using binary cross-entropy loss (BCE) and distribution focal loss (DFL) [282] with CIoU [283], respectively. This structure enhances accuracy and speeds up model convergence. As an anchor-free model, YOLOv8 precisely designates sample types and employs the Task-Aligned Assigner [284] to dynamically improve detection accuracy and robustness.

The range of YOLOv8 sub-models is succinctly detailed in Table 3.2, showcasing a comparative analysis of their performance across various metrics. These variants, including YOLOv8n, YOLOv8s, YOLOv8m, YOLOv8l, and YOLOv8x, are designed to accommodate a wide array of computational needs and detection scenarios, differing in size, speed, and accuracy.

At one end of the spectrum, YOLOv8n (nano), the smallest model, offers the fastest detection with a mean Average Precision (mAP) of 37.3, suited for scenarios where speed is crucial, particularly in resource-limited settings. In contrast, YOLOv8x (extra-large), the largest model in the series, achieves the highest mAP of 53.9, demanding more computational power and time, ideal for complex tasks where accuracy is critical. The intermediate models, YOLOv8s (small), YOLOv8m (medium), and YOLOv8l (large), present balanced options. They cater to use cases that necessitate a middle ground between the swift processing of YOLOv8n and the high precision of YOLOv8x. The table provides insights into each model's mAP, inference speeds on different hardware such as CPU and A100 TensorRT, and computational complexity, measured in millions of parameters and billions of FLOPs.

**Table 3.2:** Comparison of YOLOv8 sub-models

| **Model** | **Size** (pixels) | **mAP$^{val}$** 50-95 | **Speed** CPU ONNX (ms) | **Speed** A100 TensorRT (ms) | **Parameters** (M) | **FLOPs** (B) |
|---|---|---|---|---|---|---|
| **YOLOv8n** | 640 | 37.3 | 80.4 | 0.99 | 3.2 | 8.7 |
| **YOLOv8s** | 640 | 44.9 | 128.4 | 1.20 | 11.2 | 28.6 |
| **YOLOv8m** | 640 | 50.2 | 234.7 | 1.83 | 25.9 | 78.9 |
| **YOLOv8l** | 640 | 52.9 | 375.2 | 2.39 | 43.7 | 165.2 |
| **YOLOv8x** | 640 | 53.9 | 479.1 | 3.53 | 68.2 | 257.8 |

An exploration of the YOLOv5 and YOLOv8 models, along with their intricate structures and diverse sub-models, has been conducted, establishing a solid foundation for understanding these advanced object detection systems. With the architectural complexities and capabilities of these models thoroughly examined, attention is now shifted to the important aspect of dataset

preparation for training. In the forthcoming section, the meticulous processes of data augmentation and annotation, essential for tailoring the dataset to maximize the potential of the YOLOv5 and YOLOv8 models, will be delved into.

### 3.5.3 Dataset preparation for training

After selecting the models, the preprocessed data must be prepared for training. As illustrated in Figure 3.26, data preparation involves image augmentation, a critical step that will be discussed in the following subsection. This process includes diversifying the dataset to enhance the model's robustness, which is essential for ensuring accurate and reliable object detection across varied scenarios.

Following image augmentation, the next crucial part of the model training is image annotation. It should be noted that for this research, three datasets were annotated to train the models for identifying vegetation coverage (irrespective of species type), selected species, and specific trees of a particular species, all of which will be explained in detail in the subsequent sections. So, four annotation tools, namely ADAMs, Labelimg, Label Studio, and Roboflow.com, were evaluated in this research.

The training and evaluation of the model commence post-annotation. Should the results prove satisfactory, the model is approved for further use. However, if the results are not up to the expected standard, the process is repeated, beginning anew from the annotation phase.

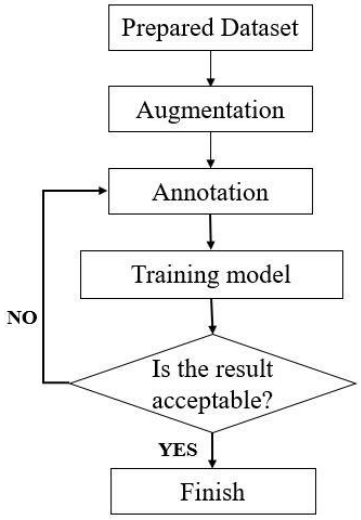



**Figure 3.26:** Flowchart of the preparing dataset for training models.

#### 3.5.3.1 Image augmentation

The rapid advancements in deep learning for computer vision are largely due to improvements in computational power, innovative algorithms, and the availability of extensive image datasets [285]. However, collecting large, labeled datasets remains challenging, especially in specialized fields like agricultural imaging [286]. To address this, image augmentation is employed to increase both the size and diversity of datasets, enhancing model performance and reducing overfitting [287, 288].

In this research, a diverse array of image augmentation techniques was applied to enhance both the size and diversity of the training dataset, a critical step in refining the performance of deep learning models. Blurring, achieved through kernel filters like Gaussian blur [289], played a pivotal role Figure 3.27B. This approach not only simulated motion blur but also conditioned the models to withstand such common image distortions in real-world scenarios.

Furthermore, meticulous adjustments in brightness (Figure 3.27C and Figure 3.27D) and exposure (Figure 3.27E and Figure 3.27F)were implemented by manipulating the RGB values in the color channels of the images. This technique, akin to photo editing, effectively recreated various lighting conditions, essential for outdoor imaging applications. Alongside, image cropping (Figure 3.27G) addressed the challenges posed by differing image dimensions, serving dual purposes: standardizing the image size and introducing translation effects to the dataset [289].

Noise injection (Figure 3.27H), involving the addition of random Gaussian-distributed values to images, was another significant augmentation, enhancing the robustness of CNNs against visual noise [289]. Complementing this, horizontal flipping (Figure 3.27I) provided the models with mirror-image variations of objects for improved recognition capabilities. Additionally, the cutout technique, which randomly removes parts of images, encouraged the model to focus on less prominent features, aiding in the detection of partially obscured objects (Figure 3.27J).

Moreover, rotational augmentations (Figure 3.27K and Figure 3.27L), carefully calibrated in degrees, were applied to ensure the integrity of the labels post-transformation, thus preserving the quality of the training data [289]. Lastly, the mosaic technique (Figure 3.27M), combining several images into one, trained the models to recognize multiple objects in a single frame, thereby mimicking complex real-world scenarios.

Overall, these augmentation methods collectively not only expanded the training dataset but also infused it with a broad spectrum of realistic variations and complexities. Such a comprehensive approach was instrumental in robustly training the models, equipping them with the capability to accurately detect and classify objects in a wide range of conditions and scenarios.


**Figure 3.27:** Examples of augmentation techniques applied to the dataset;
A. Original image, B. Image with Gaussian blur filter applied, C and D. Brightness adjustments from +20% to -20%, respectively, E and F. Exposure adjustments from -10 to +10, respectively, G. Cropped image, H. Image with noise injection, I. Image with horizontal flipping, J. Image with cutout technique, K and L. Images rotated by -10 and +10 degrees, respectively, and M. Image showcasing the mosaic technique.

With the completion of the image augmentation process, where the diversity and complexity of the dataset were effectively enhanced, the focus now shifts to the subsequent crucial step in dataset preparation: annotation. The upcoming subsection will delve into the detailed process of labeling and annotating the augmented images. This phase is crucial, as it provides the models with accurate and comprehensive information about the objects within these images, which is essential for ensuring precise object detection and classification.

#### 3.5.3.2 Image annotation

In the realm of computer vision, especially in object detection and recognition, the process of annotation is crucial. It plays a central role in building comprehensive datasets with ground truth labels, essential for effective model training and evaluation [290]. Annotation involves detailing the shape, location, identity, and other potential attributes of objects within images, such as their pose. However, creating a detailed annotated dataset is often resource-intensive, both in terms of time and cost, particularly given the need to label a multitude of objects across a wide range of images. This challenge often results in existing datasets being limited in their scope of object classes [291].

In the process of annotation, objects are typically outlined with bounding lines that capture their properties [292]. Depending on the nature of the object and the requirements of the dataset, various annotation methods are used. These methods include bounding boxes (Figure 3.28A) for defining object perimeters and polygon segmentation (Figure 3.28B) for detailed, shape-specific marking [290]. The precision and quality of these annotations are critical to the success of computer vision projects, as they directly impact the performance of object detection models. Therefore, incorrect or misleading annotations can result in misidentification and flawed outcomes from the models [290].

**A** **B**

**Figure 3.28:** Annotation methods;
A. Bounding box and B. Polygon segmentation

The selection of annotation tools and their format—be it JSON, XML, or plain text—depends on the specific needs of the project, as demonstrated in well-known datasets such as COCO, PASCAL VOC, and YOLO. Annotation typically involves a collaborative effort, bringing together annotators, inspectors, and examiners who contribute to the meticulous creation and verification of the dataset [293].

Given that the models selected for this research are from the YOLO series, the annotation tools must generate outputs in plain text files, formatted to be compatible with YOLO's specific requirements. Annotations are saved in a 'label.txt' file, typically containing information such as class identifiers and object bounding box coordinates in a format standardized for YOLO. Specifically, the annotation tool creates text files with the format [294]:

$$< object\_class >< x\_center >< y\_center >< width >< height > \quad (5)$$

where <object_class> represents the object's identity, an integer number ranging from 0 to <classes-1>. The bounding box specifications <x_center> <y_center> <width> <height> are expressed as floating-point numbers relative to the width and height of the image, ranging from 0.0 to 1.0. This format ensures precise location and classification data for each object within the images, which is crucial for the accurate training and functioning of the YOLO models.

Considering the diversity of objects and their characteristics, there has been a development of various annotation tools, some of which are freely available while others require a premium license. The following sections of this research will focus on exploring several free annotation

tools, discussing their functionalities, output formats, and examining their advantages and disadvantages.

#### 3.5.3.2.1 ADAM

ADAMS (Advanced Data mining And Machine learning System), that developed at the University of Waikato, introduces a significant change in workflow engines [295]. Moving away from the usual methods of manually linking operators on a canvas, ADAMS uses a tree-like structure (Figure *3.29*). This design uses control actors and sub-actors within actor-handlers to manage data flow, removing the need for direct connections.

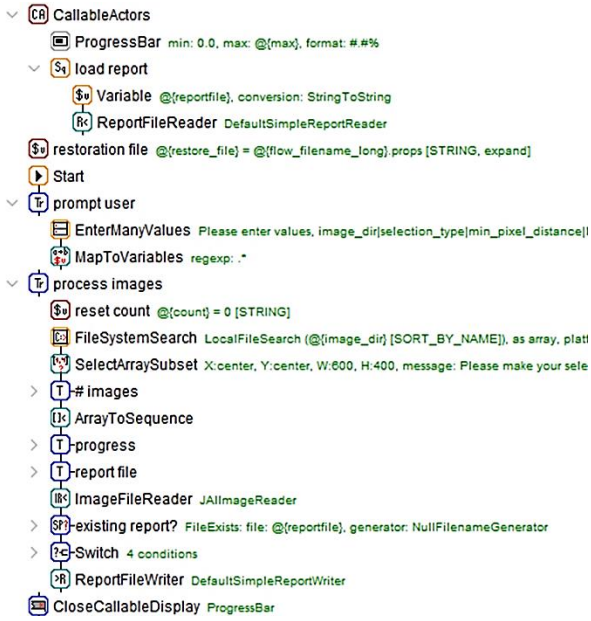


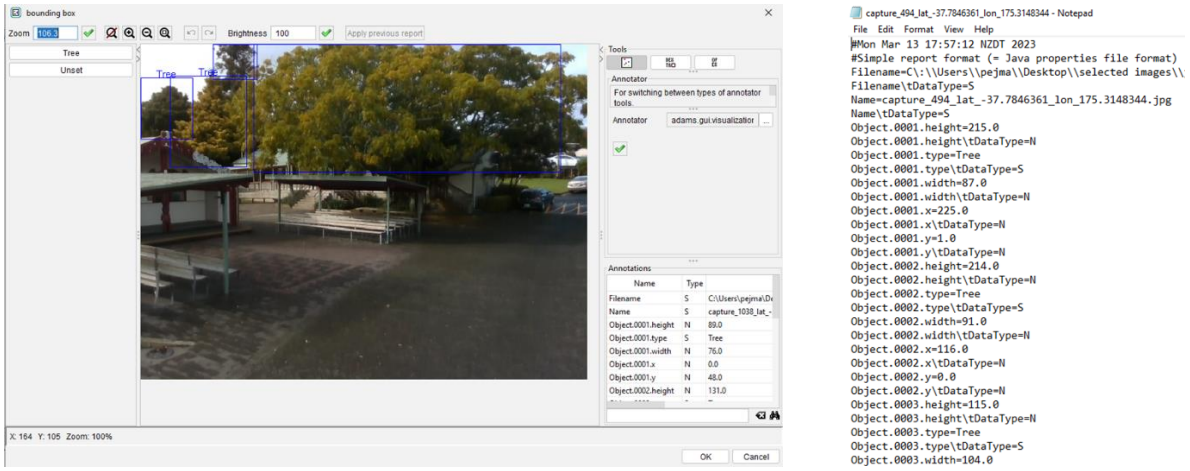

**Figure 3.29:** ADAMS structure with control actors and sub-actors for annotating objects.

The software has a variety of features for different parts of data management and analysis. It is especially good in areas like machine learning and data mining. ADAMS has flexible data processing abilities, supporting formats such as XML, HTML, JSON, and YAML.

For database interactions, ADAMS seamlessly works with systems like MySQL, SQLite, and PostgreSQL. It also has features for imaging and graphics, supporting formats such as BMP, JPG, PNG. Additionally, ADAMS allows scripting with languages like Groovy and Python, meeting a broad range of needs in data analysis and workflow automation.

Moreover, a comprehensive set of ready-made structures for annotating images, such as 'annotate objects' and 'image segmentation annotation', is included in ADAMS. These structures are prepared for use in their tree-like structure when any of them is selected. Once the 'run'

command is initiated, access to the specific environment for the selected structure is enabled. In this research, the 'annotate objects' structure was utilized. The annotation environment of the software, following the execution of the structure, is observable in Figure 3.30A. Following the image annotation process, a text file serving as a report for each picture is generated in the output section. The contents of each file are depicted in Figure 3.30B.




**A** **B**

**Figure 3.30:** ADAM software;
A. The annotation environment of the ADAMS software, illustrating the interface and tools used for image annotation, and B. The contents of a sample text file generated as a report for each annotated image, demonstrating the output format and data structure.

ADAMS operates offline, an advantage in limited connectivity environments. However, it is not designed for multiple users and requires a converter for YOLO compatibility, leading to some workflow limitations.

#### 3.5.3.2.2 LabelImg

LabelImg is an open-source annotation tool developed with Python and QT for its graphical interface, available on GitHub for both Windows and Mac operating systems. Widely used in research [296-300], it allows users to annotate objects with bounding boxes and saves these annotations as XML files in PASCAL VOC format, as well as in YOLO format for easy compatibility with object detection models(Figure 3.31).

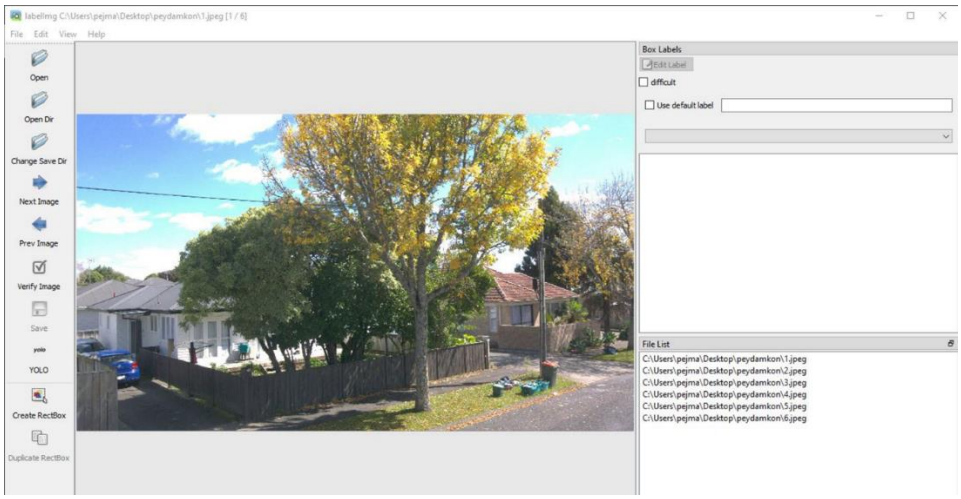



**Figure 3.31:** The GUI of LabelImg

Advantages of Labelimg include its focus on a single task (object detection), which streamlines its use for specific annotation needs [301]. It offers direct usability with an executable for Windows, though manual installation is required for other operating systems. Moreover, LabelImg integrates a functionality to mark inspected annotations with a flag, enhancing its utility in annotation review processes [302].

However, LabelImg has its limitations. It does not support pre-annotation natively, nor does it consider image pre-processing, which may require additional steps for users working with raw images; the tool also lacks the integration of functions for post-annotation processing based on prior knowledge [302]. Furthermore, LabelImg is designed for straightforward image annotation and does not offer functionalities for more complex tasks like semantic segmentation or automated annotation inspection approaches [303]. It is also noteworthy that LabelImg, being a lightweight tool, is not designed to directly generate models using the annotated dataset [302].

Given these limitations, the next tool to be explored in the subsequent subsection is 'Roboflow', which may offer additional features and capabilities that address some of the shortcomings of LabelImg.

#### 3.5.3.2.3 Roboflow.com

Roboflow, accessible at roboflow.com [304], is the other tool in the field of image annotation and dataset management for machine learning and computer vision applications. This web-based platform supports a variety of annotation types, including bounding boxes, polygons, and key-points, vital for a range of object detection and image segmentation tasks [305].

A key feature of Roboflow is its compatibility with numerous input formats such as JPG, PNG, BMP, MOV, MP4, and AVI, which greatly eases the importation of images for dataset creation; This platform is equally versatile in its output capabilities, offering a variety of export formats like coco JSON, VGG, Vott JSON, Marmot XML, YOLO PyTorch, YOLO Darknet TXT, and Kaggle CSV [305].

Roboflow's robust data organization and management capabilities allow for efficient handling of large datasets. The platform also enhances collaboration and team management, offering online access and shared project workflows, which are particularly beneficial for teams distributed across various locations. Moreover, its extensive options for data augmentation and preprocessing enhance the quality and diversity of datasets. The next advantage of Roboflow is its intuitive interface and intelligent defaults, enabling quick and accurate image annotation without the need for extensive training or specialized knowledge (Figure 3.32).

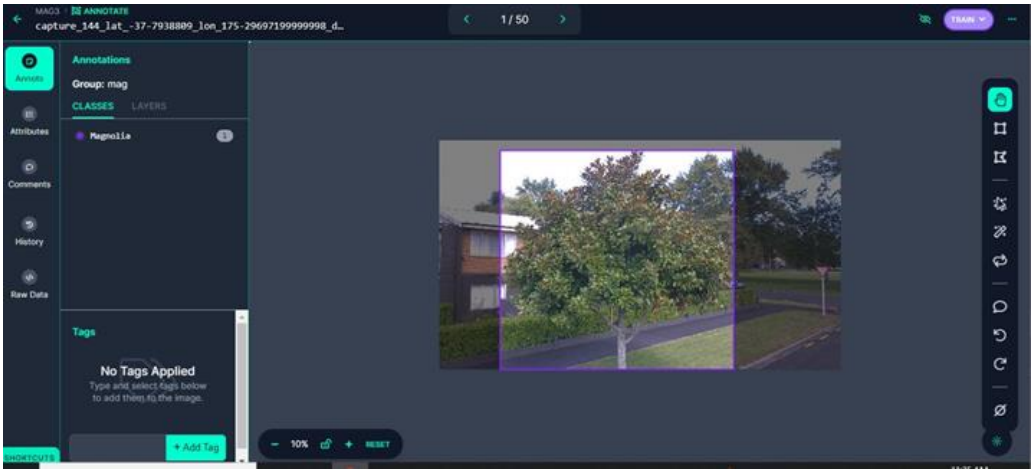



**Figure 3.32:** The annotation interface of Roboflow

However, the platform has its limitations. Users are required to upload images to Roboflow, which might be a constraint when dealing with sensitive or large-scale data. Additionally, being a cloud-based tool, Roboflow relies on stable internet connectivity, which could be a significant drawback in areas with poor internet access. The free version of Roboflow limits the number of images that can be annotated, and advanced features are available only through paid subscriptions. Moreover, the platform may offer limited customization options compared to other annotation tools, which could be a constraint for projects requiring highly tailored workflows.

In light of these limitations, the next tool to be evaluated is Label Studio. This evaluation will determine if Label Studio offers features that address some of the shortcomings identified in Roboflow.

#### 3.5.3.2.4 Label studio

Label Studio [306] is a web application platform, designed to cater to a wide range of data labeling needs across multiple data types including text, images, videos, audio, time series, and multi-domain data types. Available both as an open-source solution and with advanced features under a commercial license, Label Studio stands out for its accessibility and functionality in machine learning applications [307]. The platform offers a comprehensive

According to the Label Studio documentation, a notable advantage of Label Studio is its versatility in annotation types. It supports annotations for text, images, and videos, among others, thereby serving a wide array of data labeling tasks. Its workflow is customizable, allowing users to tailor the platform according to their specific project needs. Furthermore, Label Studio facilitates collaboration and team management, making it an ideal choice for projects that involve multiple annotators or teams. Another significant feature is its ability to run offline, providing flexibility for users without constant internet access. Additionally, it can connect to online storage services such as AWS S3, Google Cloud Storage, Microsoft Azure, and Redis, ensuring easy data management and accessibility.

However, Label Studio does come with its own set of limitations. The initial setup and infrastructure requirements can be complex, potentially posing a challenge for users without technical expertise. It also has limitations in input formats, specifically to CSV and JSON, which might restrict its use in scenarios requiring more diverse file types.

One of the most remarkable aspects of Label Studio is its high degree of customization. According to its documentation, users can modify almost every element within the software interface, such as import and export options, user management, and annotation tools like bounding boxes and polygons. These modifications can be made by altering the HTML, CSS, and JavaScript files of the frontend to meet specific project requirements (Figure 3.33).

Additionally, using the Django framework in the Label Studio backend, users can integrate additional services or external systems. This includes capabilities like reading Excel files, further extending the platform's utility and adaptability in various data processing scenarios.

Given Label Studio's capability for customization, a version of the software was specifically customized to meet the requirements of this project. The following section will explain the details of this customized version and its applications.

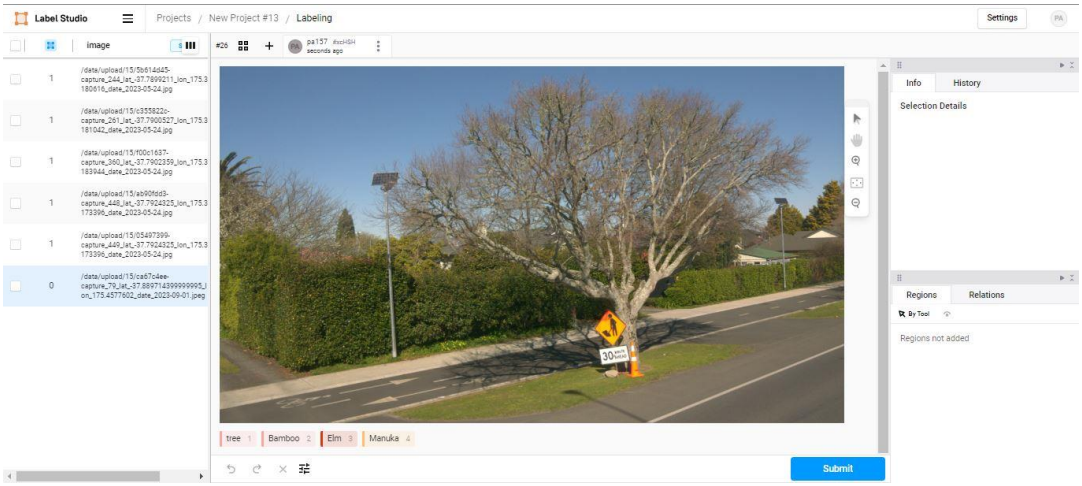



**Figure 3.33:** The user interface of Label Studio.

3.5.3.2.5 Customized Label Studio

Given the evaluations conducted in the annotation section, only Label Studio offers significant customization capabilities. Hence, a version tailored to the specific needs of this research was developed.

One of the key challenges in annotating images of vegetation cover lies in accurately identifying the plant species for use as labels. This task is facilitated by the data collected by city councils, which have been comprehensively recorded in Excel files. These files contain GPS coordinates for various tree species. By matching these coordinates with those embedded in the names of the images, the software can accurately determine and display the nearest registered species to the tree shown in the image. Notably, this system is designed to identify species within a 10-meter radius of the location recorded in the photograph, and this radius is adjustable to suit different research needs (Figure 3.34).

```
PS C:\Python-environments\label_studio>  c:; cd 'c:\Python-environments\label_studio'; & 'C:\Users\pejm
a\AppData\Local\Programs\Python\Python311\python.exe' 'c:\Users\pejma\.vscode\extensions\ms-python.pyth
on-2023.20.0\pythonFiles\lib\python\debugpy\adapter/../..\debugpy\launcher' '11086' '--' 'C:\Python-env
ironments\label_studio\tree_coords.py'
Enter the filename of the spreadsheet: Council.xlsx
Loaded 6798 trees, waiting for Label-studio input...



Image #29
          Name  Distance (m)           ID
0  London Plane      9.674123  London Plane
Press Ctrl+C to stop.
```

**Figure 3.34**: Display of identified tree species in customized Label Studio.

This process was implemented through modifications made in the backend of the software. Leveraging Label Studio's backend, which is built on the Django framework, allowed for the integration of this tree species identification feature, enhancing the accuracy and relevance of the annotations for this research.

After the annotation of images, the data preparation phase is completed, and the selected models must be trained with it. The process of training these models will be explained in detail in the following section.

### 3.5.4 Models training process

The model training process in this project is a comprehensive and intricate journey, pivotal to developing accurate and efficient machine learning models for vegetation detection and classification. It begins with the foundational setup of training parameters, involving the careful calibration of variables such as learning rate, batch size, and epochs. These parameters are crucial in defining the models' learning path and overall effectiveness.

In developing a model for urban vegetation detection, with a focus that extends beyond species identification, the creation of a robust and diverse dataset is paramount. This step involves meticulous collection, augmentation, and annotation of urban vegetation data, ensuring that the models are trained on rich and representative samples. The quality and diversity of this dataset are fundamental in laying the groundwork for the models' ability to accurately identify various plant types within urban environments.

A special emphasis is placed on training models for the specific detection of Magnolia trees. This task requires a more targeted approach, incorporating specialized datasets with variable bounding box sizes. The training approaches for these models are varied, ranging from building new models from scratch to employing advanced techniques like transfer learning.

Furthermore, the training process expands to encompass models for other specific species. This phase involves creating targeted datasets that cater to the unique characteristics of each species. The precision and relevance of these datasets are critical in ensuring that the models are effectively trained to detect and differentiate between various plant species, thereby enhancing the models' utility in diverse vegetation analysis scenarios.

#### 3.5.4.1 Configuration of training parameters

The configuration of training parameters is essential for optimizing model performance, especially for urban vegetation detection using YOLOv5 and YOLOv8 sub-models. Table 3.3 summarizes the key training parameters for both YOLOv5 and YOLOv8.

**Table 3.3:** Comparative overview of training parameters for YOLOv5 and YOLOv8 models.

| Parameter | YOLOv5 | YOLOv8 |
|---|---|---|
| Learning Rate | Default | Default |
| Batch Size | 8 | 8 |
| Number of Epochs | 100 | 100 |
| Optimizer | Default | Default |
| Loss Function | Default | Default |
| Image Size | 640 | 640 |

- **Learning rate:** The learning rate in training CNNs is crucial, as it dictates the adjustments made to the model's weights during backpropagation; this parameter significantly impacts the speed and overall effectiveness of the learning process [308]. Generally ranging from 0.001 to 0.01, an optimal learning rate is essential to avoid extremes [309]; a rate that is too low can result in very slow learning or an inability to learn at all, while a rate that is too high might cause the model's performance to oscillate or even worsen over the training epochs, leading to suboptimal final performance [308]. In this project, the learning rate was kept at the default setting for both models. This approach was chosen to utilize pre-optimized rates that are apt for various data scenarios.
- **Batch size:** The next key hyperparameter in training is Batch size, determining how many samples are processed before updating the model's parameters [310]. A balance is essential: too high, and the network may converge slowly, while too low can lead to unstable training [311]. The choice often varies from 2 to 64, influenced by dataset complexity and computational capacity. In this project, a batch size of 8 was chosen to balance efficient learning with computational constraints, ensuring stable training progress without overburdening the system.

- **Number of epochs:** the number of epochs, which dictates how many times the entire dataset is passed through the model, is critical for effective learning. Choosing the right number balances underfitting and overfitting: too few epochs can lead to incomplete learning, while too many might cause the model to overfit to training data. This number often lies between 50 and 300. For this project, 100 epochs were selected, offering a balanced approach that allows thorough learning without overfitting.
- **Optimizer:** Optimizers play a crucial role in training deep neural networks by guiding the minimization of error functions and enhancing convergence [312]. The choice of an optimizer affects the network's learning rate and generalization ability. Broadly, optimizers fall into two categories: the classic stochastic gradient descent (SGD) with a static learning rate [313] and adaptive algorithms like ADAM, which adjust the learning rate dynamically [314]. While SGD is known for its reliable and straightforward approach, adaptive optimizers like ADAM are favored for their flexibility across various problems. However, the perfect optimizer choice often depends on the specific task at hand, with no one-size-fits-all solution. In this research, the default optimizer, SGD, was used.
- **Loss Function:** In deep learning, the loss function is responsible for quantifying the difference between the predicted output of the model and the actual ground truth; it effectively measures the accuracy, similarity, or goodness of fit, playing a pivotal role in guiding the training process toward optimal performance [315]. A well-designed loss function is tailored to address specific challenges and requirements unique to deep learning tasks. In the context of this project, among various loss functions like Mean Squared Error (MSE), Cross-Entropy, and others, the Complete Intersection over Union (CIOU) loss function was chosen as the default.
- **Image size:** The chosen image size is important, impacting both accuracy and computational load. Larger sizes can enhance detail and accuracy but increase computational demands, while smaller sizes are less resource-intensive but might lose essential details. For this project, an image size of 640x640 pixels was selected, striking a balance between providing adequate detail for effective vegetation detection and classification, and maintaining computational efficiency.

Having detailed the crucial training parameters and their specific configurations for the YOLOv5 and YOLOv8 models, the next aspect to consider is the system setup that facilitated this training process. The following section will delve into the computational environment used.

#### 3.5.4.2 System configuration for model training

For the training of neural network models, Google Collaboratory (Colab) is widely utilized in numerous research projects due to the extensive resources it offers [316-320]. This research also employed Google Colab as the primary computational platform, leveraging its convenient and powerful cloud-based environment. Google Colab allows the use of high-performance computing resources without the need for local setup, making it an ideal choice for demanding deep learning tasks. Each training session in this project involved uploading the dataset to Google Colab and utilizing its computational capabilities for efficient model training. The system used was equipped with an NVIDIA Tesla T4 GPU, featuring a driver version of 525.105.17 and CUDA Version 12.0. With 15360 MiB of memory, the Tesla T4 GPU provided a robust platform for training, ensuring rapid processing and substantial computational power. This was especially beneficial for the extensive computations required in tasks like vegetation detection and classification using YOLOv5 and YOLOv8 models. The accessibility and strength of Google Colab's GPU resources significantly enhanced the efficiency and effectiveness of the model training process.

After a thorough examination of the model parameters and the configuration of the system used, the next step is to apply these foundational elements to develop models tailored to specific objectives. The following sections will delve into the details of these models.

#### 3.5.4.3 Training a basic model for urban vegetation detection

The initial model trained in this research focused on distinguishing urban vegetation from other elements typically found in cityscapes, such as walls, cars, traffic signs, billboards, and other urban features (Figure 3.35). The primary expectation from this basic model was to accurately identify vegetation coverage amidst these various urban elements. The success of this initial model was critical, as it laid the groundwork for subsequent phases involving specific tree identification and classification of particular species. The next subsection details assembling and preparing the dataset instrumental in training this initial model.

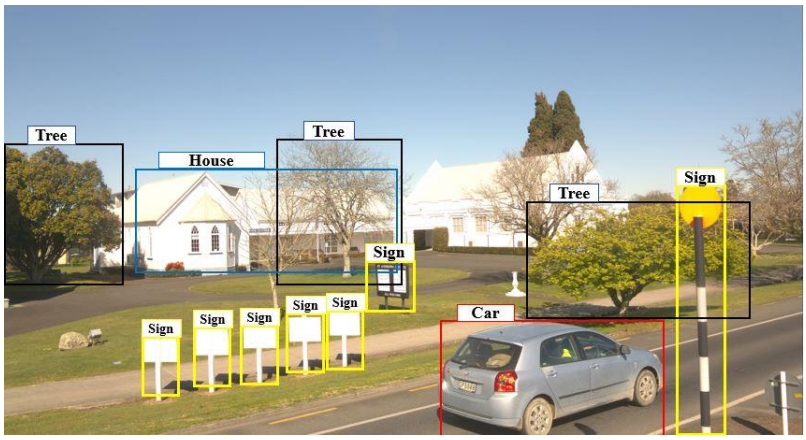



**Figure 3.35:** A typical urban scene with various objects; like cars, traffic signs, houses, advertisement signs, and trees.

3.5.4.3.1 Creation of a dataset for urban vegetation

For training this model, data collected on January 20th, March 17th, and March 23rd of 2023 from Hamilton city were utilized. According to Table 3.4, from 7155 images gathered, 450 containing at least one tree were manually selected. These images were then augmented using techniques described in previous sections and subsequently annotated, resulting in a total of 1213 images for the dataset (Figure 3.36A). The dataset was divided in an 80-20 ratio for training and validation purposes. Additionally, to enhance the model's ability to differentiate between vegetation and non-vegetation elements, about 10% of the dataset, approximately 120 images, comprised background images without any trees (Figure 3.36B). Following the preparation and annotation of this dataset, the models were trained to identify and classify urban vegetation effectively in Google Colab.

**Table 3.4:** Dataset summary for urban vegetation model training.

| Date | Number of images | Number of selected images | Number of images after augmentation | Training-Validation split (%) |
|---|---|---|---|---|
| 2023-01-20 | 2295 | 140 | 377 | 80-20 |
| 2023-03-17 | 3630 | 240 | 646 | 80-20 |
| 2023-03-23 | 1230 | 70 | 190 | 80-20 |
| Total | 7155 | 450 | 1213 | ---------- |

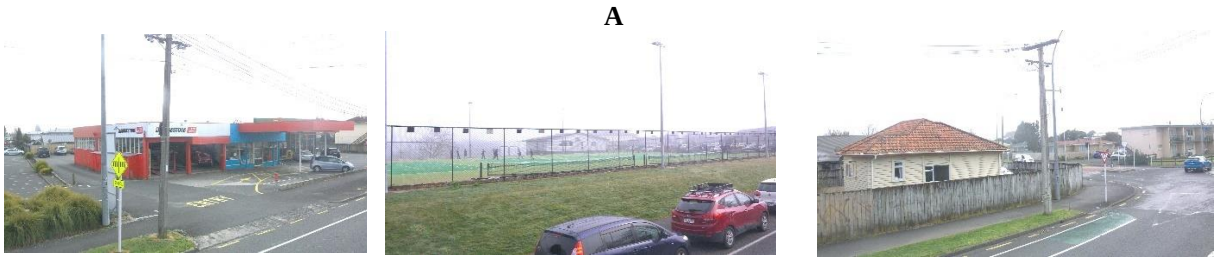


**Figure 3.36:** Example of annotated and background images in the prepared dataset; A. Annotated images and B. Background images without any trees.

3.5.4.4 Training models for specific species

Following the development of the basic model for identifying trees without considering their species, the next step involved evaluating the selected models' ability to recognize and classify different species. This phase required a dataset specifically curated for species identification.

3.5.4.4.1 Creation of a dataset for specific species

To prepare the dataset for training models on specific species, the initial step involved identifying the target tree species. This selection process, as elaborated in section "Overview of Selected Tree Species" utilized images collected from Cambridge City on September 1st and September 22nd, 2023. The choice of species was guided by the tree information's datasheet provided by the Cambridge City Council. The key criterion for selecting a species was its frequency in easily accessible areas, which is crucial to ensure a diverse range of images for effective model training. As a result of this process, eight distinct species were chosen for inclusion in the study which contains; *Prunus Species* (Flowering Cherry), *Cornus Florida* (Flowering Dogwood), *Acer Palmatum* (Japanese Maple), *Platanus X Acerifolia* (London Plane), *Liquidambar Styraciflua* (Liquidambar), *Magnolia Grandiflora* (Bull Bay), *Quercus Palustris* (Pin Oak), and *Betula Pendula* (Silver Birch).

The collected images were then manually reviewed to separate images corresponding to each species. According to Table 3.5, a total of 9547 images were gathered on the aforementioned dates. From these, 100 images per species were selected. After augmenting these selected images, the total count rose from 800 to 2400. Furthermore, 240 background images were added, bringing the final dataset to 2640 images.

**Table 3.5:** Dataset summary for training models on specific species.

| Species | Total images (Selected images) | | Total of selected images | Number of images after augmentation | Training-Validation split (%) |
|---|---|---|---|---|---|
| | 2023-09-01 | 2023-09-22 | | | |

| | | | | | |
|---|---|---|---|---|---|
| *Prunus Species* (Flowering Cherry) | 180 (65) | 73 (35) | 100 | 300 | 80-20 |
| *Cornus Florida* (Flowering Dogwood) | 97 (60) | 55 (40) | 100 | 300 | 80-20 |
| *Acer Palmatum* (Japanese Maple) | 311 (70) | 135 (30) | 100 | 300 | 80-20 |
| *Liquidambar Styraciflua* (Liquidambar) | 55 (55) | 50 (45) | 100 | 300 | 80-20 |
| *Platanus X Acerifolia (London Plane)* | 235 (50) | 163 (50) | 100 | 300 | 80-20 |
| *Magnolia Grandiflora* (Bull Bay) | 70 (60) | 63 (40) | 100 | 300 | 80-20 |
| *Quercus Palustris* (Pin Oak) | 93 (65) | 50 (35) | 100 | 300 | 80-20 |
| *Betula Pendula* (Silver Birch) | 101 (80) | 77 (20) | 100 | 300 | 80-20 |
| Others* | 5199 | 2540 | -------- | -------- | --------- |
| Total | 6341 | 3206 | 800 | 2400 | --------- |

*Non-target species + without vegetation

#### 3.5.4.4.2 Training approaches for specific trees species detection

After preparing two datasets with varying bounding box sizes, the next phase of training begins. In this section, one of the training approaches adopted is the conventional training method, previously utilized in other model training processes. This approach entails executing the entire model training process in the Google Colab environment, starting from dataset uploading through to the final stages of model development, with each step methodically followed. For training models to identify selected Magnolia trees, besides this standard approach, two additional techniques of transfer learning were also employed: retrain a trained model and freezing layers which will be discussed in the following subsections. These methods represent advanced techniques in machine learning, designed to enhance the model's learning efficiency and accuracy in identifying specific tree characteristics.

##### 3.5.4.4.2.1 Applying transfer learning for training models

Transfer learning, a technique that mirrors the human vision system's ability to leverage prior knowledge in related domains for new tasks [321]. Typically, transfer learning is employed to fine-tune a previously trained model using a dataset that may not have a sufficient number of images for each class; this approach leverages the model's pre-existing knowledge, adapting it to new, limited datasets for improved accuracy and efficiency [322]. Various other research endeavors have successfully implemented transfer learning, demonstrating its effectiveness across different domains and applications [323-326]. Consistent with the principles of transfer learning, a pre-trained model that had been previously trained for tree identification was selected for this project. It is worth noting that the transfer learning approach is adaptable for both YOLOv5 and YOLOv8 models. This adaptability ensures that regardless of the version, the model can efficiently utilize the prior knowledge encapsulated in the pre-trained model.

3.5.4.4.2.2 Implementing layer freezing in transfer learning

Freezing layers in deep learning is a fundamental technique in transfer learning [327]. This method involves 'freezing' certain layers of the model, meaning their weights and biases remain unchanged during the training process [328]. This strategy is beneficial when adapting a pre-trained model to a new but related task, as it preserves previously learned features that are still pertinent. Furthermore, freezing layers, often regarded as a subsequent method in transfer learning, are widely used in various research projects to enhance model performance without extensive retraining [329-332].

In this research, the initial focus was placed on freezing the backbone layers of the model. Typically composed of the initial layers responsible for basic feature extraction, the backbone's freezing allowed for the leveraging of core patterns already learned by the model. This was followed by an experiment involving the freezing of all layers of the model. The aim was to understand the impact of limiting the model's entire adaptability to the specific characteristics of the dataset, with a particular focus on identifying specific tree species. However, it should be noted that according to the YOLOv8 documentation, freezing layers in this model is not yet an available option.

3.5.4.5 Training models for specific Magnolia trees

One of the key objectives of this research is to identify specific trees, with a particular focus on developing a model that can recognize individual trees within a species. After thorough evaluations, the research team selected 30 Magnolia trees located in the Hamilton-East area of Hamilton City. It is important to note that identifying individual trees within a specific species poses a significant challenge for the model, mainly when the differences in images are limited to the size of the trees and their background. Such subtle distinctions demand a high level of precision from the model to accurately distinguish between the trees. This requirement adds complexity to the dataset preparation process, necessitating modifications that differ from those used in previous models. The subsequent subsections will explore these necessary adjustments in detail.

3.5.4.5.1 Creation of specialized datasets with variable bounding box sizes

The imaging of selected trees, crucial for creating a dataset to train the model, was carried out on May 24th, June 9th, and July 26th, 2023. Following the collection, images specifically featuring the selected Magnolia trees were isolated. Given the importance of annotation in model training,

these trees were uniquely labeled from 'Magnolia-1' to 'Magnolia-32', and their GPS coordinates were recorded (Figure 3.37). Upon reviewing the images, 30 images per tree were manually chosen, and after augmentation, the total number of images increased from 960 to 3840 (Table 3.6). In addition, 390 background images were added, bringing the final dataset to 4230 images.

**Table 3.6:** Dataset summary for training models on specific species.

| Specific Tree's name | Total images (Selected images) | | | Total of selected images | Number of images after augmentation | Training-Validation split (%) |
|---|---|---|---|---|---|---|
| | 2023-05-24 | 2023-06-09 | 2023-07-26 | | | |
| Magnolia-1 | 25 (16) | 19 (10) | 20 (4) | 30 | 120 | 80-20 |
| Magnolia-2 | 13 (13) | 14 (10) | 7 (7) | 30 | 120 | 80-20 |
| Magnolia-3 | 17 (15) | 15 (5) | 21 (10) | 30 | 120 | 80-20 |
| Magnolia-4 | 17 (12) | 16 (7) | 16 (11) | 30 | 120 | 80-20 |
| Magnolia-5 | 21 (18) | 25 (12) | 9 (0) | 30 | 120 | 80-20 |
| Magnolia-6 | 21 (10) | 17 (7) | 23 (13) | 30 | 120 | 80-20 |
| Magnolia-7 | 53 (20) | 32 (5) | 33 (5) | 30 | 120 | 80-20 |
| Magnolia-8 | 15 (15) | 10 (10) | 15 (5) | 30 | 120 | 80-20 |
| Magnolia-9 | 23 (12) | 19 (10) | 12 (8) | 30 | 120 | 80-20 |
| Magnolia-10 | 7 (7) | 21 (15) | 12 (8) | 30 | 120 | 80-20 |
| Magnolia-11 | 18 (10) | 15 (10) | 19 (10) | 30 | 120 | 80-20 |
| Magnolia-12 | 15 (10) | 10 (10) | 17 (10) | 30 | 120 | 80-20 |
| Magnolia-13 | 19 (11) | 9 (6) | 23 (13) | 30 | 120 | 80-20 |
| Magnolia-14 | 17 (14) | 15 (8) | 12 (8) | 30 | 120 | 80-20 |
| Magnolia-15 | 31 (10) | 32 (11) | 31 (9) | 30 | 120 | 80-20 |
| Magnolia-16 | 27 (15) | 20 (5) | 23 (10) | 30 | 120 | 80-20 |
| Magnolia-17 | 15 (0) | 18 (18) | 15 (12) | 30 | 120 | 80-20 |
| Magnolia-18 | 16 (14) | 10 (4) | 12 (12) | 30 | 120 | 80-20 |
| Magnolia-19 | 16 (10) | 16 (13) | 19 (7) | 30 | 120 | 80-20 |
| Magnolia-20 | 16 (6) | 16 (16) | 15 (8) | 30 | 120 | 80-20 |
| Magnolia-21 | 20 (18) | 9 (2) | 17 (10) | 30 | 120 | 80-20 |
| Magnolia-22 | 24 (9) | 13 (7) | 19 (14) | 30 | 120 | 80-20 |
| Magnolia-23 | 14 (10) | 12 (11) | 12 (9) | 30 | 120 | 80-20 |
| Magnolia-24 | 10 (10) | 14 (14) | 6 (6) | 30 | 120 | 80-20 |
| Magnolia-25 | 17 (12) | 14 (5) | 18 (13) | 30 | 120 | 80-20 |
| Magnolia-26 | 15 (6) | 15 (12) | 16 (12) | 30 | 120 | 80-20 |
| Magnolia-27 | 23 (14) | 5 (5) | 23 (11) | 30 | 120 | 80-20 |
| Magnolia-28 | 20 (17) | 13 (3) | 21 (10) | 30 | 120 | 80-20 |
| Magnolia-29 | 21 (10) | 19 (13) | 14 (7) | 30 | 120 | 80-20 |
| Magnolia-30 | 20 (10) | 19 (6) | 14 (14) | 30 | 120 | 80-20 |
| Magnolia-31 | 15 (13) | 9 (7) | 12 (10) | 30 | 120 | |
| Magnolia-32 | 13 (9) | 17 (11) | 20 (10) | 30 | 120 | |

| | | | | | | |
|---|---|---|---|---|---|---|
| Others* | 5530 | 5009 | 4094 | ------- | -------- | -------- |
| Total | 6144 | 5517 | 4640 | 960 | 3840 | -------- |

*Non-target trees + without vegetation images

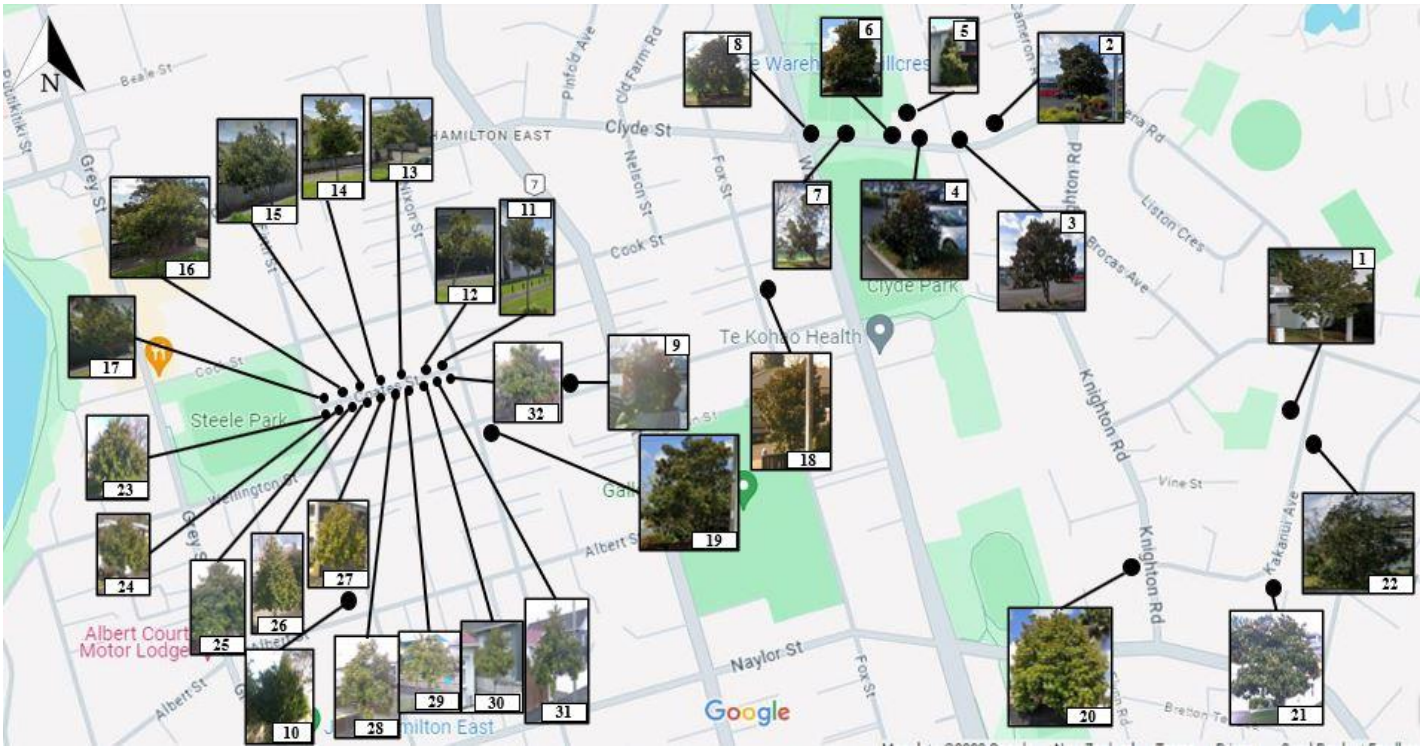



**Figure 3.37:** Map of Hamilton East area in Hamilton city*;*
highlighting the locations of selected magnolia trees.

The next crucial step involved annotating these images, for which two methods were explored: the standard bounding box annotation and a novel method devised for this project. These approaches aimed to evaluate and enhance the model's accuracy in tree identification which will be discussed in the following subsections.

3.5.4.5.1.1 Standard bounding box Annotation

In the first method of annotation, which was performed in a standard manner, the bounding box identifying the object (trees, in this research) was drawn precisely so that the tree's boundaries were aligned with the edges of the rectangle [333] (Figure 3.38). This approach ensured that the maximum portion of the tree's image was contained within the bounding box while minimizing the inclusion of other objects (noise) that were not part of the tree. This standard annotation method, which had been effectively used in previous models for identifying general vegetation and different species, faced no significant challenges.

However, when it comes to recognizing a specific tree, it is crucial to consider factors such as the tree's growth, seasonal changes like leaf shedding, and regular pruning, which may alter its appearance over time. As a result, a model trained with standard bounding box annotation might struggle to maintain accuracy as the physical characteristics of the tree change. Therefore, another annotation method was explored to address this challenge.


**Figure 3.38:** Example of selected magnolia trees with the standard bounding box.

3.5.4.5.1.2 Extended bounding box annotation

In this approach, a larger-than-usual bounding box was chosen to encompass not only the tree but also a portion of its background (Figure 3.39). This strategy was based on the observation that the background elements, such as buildings, various types of walls (differing in material and color), parking areas, shops, etc., often differ significantly for each tree. The rationale behind this extended bounding box method was to enable the model to recognize and analyze not only the tree but also its unique background context. By doing so, the model would be trained to identify and classify the tree with greater accuracy, reducing the likelihood of misclassification due to changes in the tree's appearance or its surrounding environment over time.


**Figure 3.39:** Example of selected magnolia trees with the extended bounding box.

Datasets, after annotating, were divided into training and validation sets, and subsequently uploaded to Google Colab for model training.

With the methodologies for annotating phase, the next critical aspect of this research involves evaluating the effectiveness of these approaches. The upcoming section, 'Performance Metrics and

Evaluation Criteria,' will detail the metrics and standards used to assess the performance of the models. This section will delve into how the models' accuracy, precision, recall, and other key performance indicators were measured, providing a comprehensive understanding of their efficacy in correctly identifying and classifying targets. This evaluation is essential to validate the methodologies employed and to gauge the models' practical applicability in real-world scenarios.

## 3.6 Performance metrics and evaluation criteria

To ensure a system's effectiveness, key performance indicators (KPIs) are crucial. Their role is to connect strategies with results. This section discusses the identified key performance index for trained models. In the study, KPIs such as confusion matrix, Precision, Recall and mAP (mean average precision) were used to evaluate the performance of the trained models.

### 3.6.1 Confusion matrix

The confusion matrix summarizes the number of correct and incorrect predictions and is broken down by the different classes. For a binary classifier, four cells of the confusion matrix quantify the frequency of every combination of a predicted class and the actual class [334]. According to Figure 3.40, the target variable comprises two values: Positive and Negative. The actual values of the target variable are represented in the columns, while the predicted values are shown in the rows. This figure illustrates four different outcomes: True Positive (TP), False Positive (FP), False Negative (FN), and True Negative (TN).

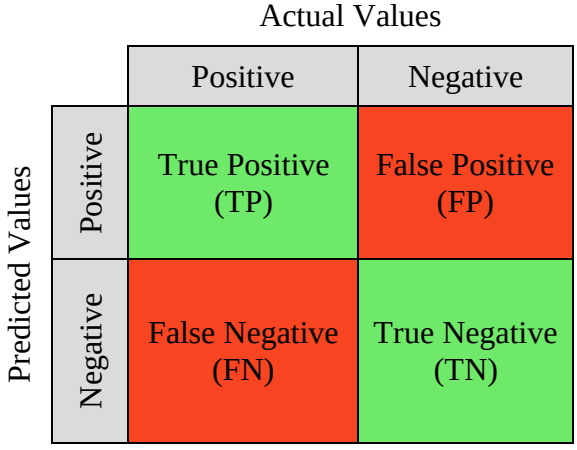



**Figure 3.40:** Confusion matrix

**True Positive (TP):**

- There is a perfect match between the predicted and the actual values.
- The actual value is positive, and the model correctly predicts a positive result.

**True Negative (TN):**

- The prediction aligns precisely with the actual value.
- The actual value is negative, and the model accurately predicts a negative result.

**False Positive (FP):**

- The prediction is incorrect.
- Despite the actual value being negative, the model incorrectly predicts a positive result.

**False Negative (FN):**

- The prediction is incorrect.
- Although the actual value is positive, the model mistakenly predicts a negative result.

### 3.6.2 Precision

Precision measures the model's ability to correctly identify relevant objects, essentially answering the question: 'How frequently is the model correct when it makes a guess?' It represents the proportion of accurate positive detections, calculated by dividing the number of true positive cases by the total of both true positive and false positive cases, as detailed in Equation (6) [332]

$$\Pr ecision = \frac{\sum TP}{\sum TP + \sum FP} \tag{6}$$

### 3.6.3 Recall

Recall assesses the model's capability to identify all actual cases, determined by the proportion of ground truth cases correctly identified. This is calculated by dividing the number of true positive cases by the combined total of true positive and false negative cases, as outlined in Equation (7). Essentially, it addresses the question: 'Does the model make a prediction whenever it's supposed to'? [335]

$$\mathrm{Re}\, call = \frac{\sum TP}{\sum TP + \sum FN} \tag{7}$$

### 3.6.4 Mean Average Precision

The mean average precision (mAP), as defined in Equation (8), represents the balance between precision and recall. It is widely recognized as a crucial parameter for evaluating the performance of object detection models [336].

$$mAP = \frac{1}{N}\sum_{q=1}^{q} Average \quad \Pr ecision(q) \tag{8}$$

where N represents the total number of objects, and q refers to a specific object. This concept ties in with the Intersection over Union (IoU), calculated as the proportion between the intersecting and combined areas of the predicted and actual bounding boxes, illustrated in Figure 3.41. The IoU metric is crucial for determining whether a bounding box is a true positive, false positive, or false negative.

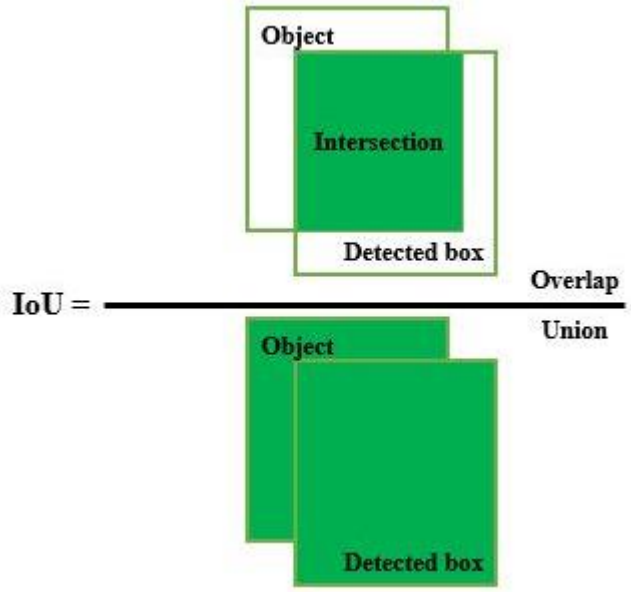



**Figure 3.41:** IoU visualization *[337]*.

Diverse IoU thresholds, such as 0.5:0.05:0.95, 0.5, and 0.95, are utilized for calculating mAP. The first threshold allows for gradual adjustments during evaluation, aiding in the assessment of localization accuracy [338]. The latter thresholds focus solely on calculating the aforementioned percentages. In this research, mAP@0.5 and mAP@0.5:0.95 are the metrics employed for evaluation.

## 3.7 Summery

- This chapter focuses on creating a computer vision system for urban vegetation detection, aimed at identifying various tree species and individual trees with RGB images using YOLOv5 and YOLOv8 models, encompassing data collection, preprocessing, model training, and vegetation type detection.

- Designing efficient data collection routes maximized tree species coverage, utilizing city council datasheets for route planning. This included filtering inaccessible areas, selecting tree species based on frequency, and employing both manual and automated methods to create routes.
- Tree species selection was based on the count of accessible tree species from city datasheets, leading to the final choice of eight tree species due to their prevalence and alignment with the research objectives.
- A trained YOLOv5 model identifies and retains images with urban vegetation, filtering out non-vegetation elements to improve dataset relevance and analysis efficiency.
- Implemented Fast Fourier Transform and Laplacian Variance techniques to identify and remove blurred images from the dataset, a critical step due to camera instability from vehicle movement, ensuring only clear, high-quality images are retained for accurate analysis.
- Three methods were evaluated and employed - the Mean Squared Error technique, the Normalized Cross-Correlation technique, and the use of GPS coordinates - to effectively identify and eliminate repetitive images from the dataset.
- A pre-trained YOLOv5 model for automated detection and blurring of sensitive elements like faces and license plates in images, ensuring privacy protection in urban vegetation imaging has been used.
- Diverse image augmentation techniques such as blurring, brightness and exposure adjustments, cropping, noise injection, horizontal flipping, cutout, rotational changes, and the mosaic method were applied to enhance the training dataset's size and diversity.
- Various annotation tools discussed like ADAMS, LabelImg, Roboflow, and Label Studio for image annotation, highlighting their features and limitations; a customized version of Label Studio is developed, utilizing city council data for precise plant species identification, setting the stage for the final data preparation and model training phase.
- Three models were trained, including a model for detecting urban vegetation, another for detecting and classifying selected species, and a third for identifying and classifying specific trees.
- Key performance indicators like the confusion matrix, precision, recall, and mean average precision (mAP) were used to evaluate the effectiveness of the trained models.

# 4 Results and discussion

## 4.1 Introduction

In this chapter, a quantitative analysis of the research findings is presented, and these results are integrated into the broader context of urban environmental monitoring. The outcomes of this study are juxtaposed with findings from previous research in the field, providing a comparative perspective. Through this comparative analysis, the relevance of the results is underscored, and they are situated within the existing body of knowledge. This approach enriches the understanding of urban vegetation surveillance through deep learning models.

Initially, the focus is on the dataset preprocessing steps crucial in enhancing image quality for model training. These steps included the removal of non-vegetation elements, blur, and repetitive images, along with ensuring privacy protection, all contributing significantly to the data's accuracy and reliability. Then, the preparation and annotation of the dataset, fundamental to the application of deep learning models, are then examined. The selection of an annotation tool and methods was dictated by the specific requirements of urban vegetation detection.

Following this, the detection of urban vegetation, specific species, and individual trees is explored. Here, the performances of various models like YOLOv5 and YOLOv8 are compared and discussed, focusing on the implications of varying bounding box sizes and diverse training approaches on model efficacy.

In summary, this chapter presents a quantitative analysis of the research findings and integrates these results into the broader context of urban environmental monitoring. It aims to bridge technical findings with their real-world implications, thereby contributing to future research in this field.

## 4.2 Dataset Preprocessing

In this research, dataset preprocessing was identified as a critical step, marked by the implementation of several filters to refine data quality for subsequent analysis. Included in these filters were a non-vegetation filter, designed to eliminate images without vegetation cover; a blur image filter, aimed at removing images lacking clarity; a repetitive images filter, intended to reduce dataset redundancy; and a privacy protection filter, ensuring the removal of any personal information from the images. As elaborated in the system design and methodology chapter, these filters were described in detail, with specific techniques employed to enhance their effectiveness. This section will explore the results of applying these filters, with an evaluation of their advantages and limitations. The assessment of these steps is crucial, as the preprocessing phase forms the foundation for the accuracy and reliability of the deep learning models applied later in the study. The systematic analysis of these preprocessing steps is aimed at demonstrating their importance in enhancing the dataset's suitability for urban vegetation surveillance, thereby contributing to the robustness and precision of the research findings.

### 4.2.1 Non-vegetation filter

In addressing the challenge of non-vegetation images in urban imagery datasets, a YOLOv5x model was employed due to its compatibility with existing scripts, unlike the YOLOv8 model which lacked certain required functionalities (because this version has been recently released). This model effectively identified and retained images of urban vegetation while removing non-vegetation images.

The chosen YOLOv5x model analyzed each image captured during the data collection sessions to identify and retain images containing urban vegetation, irrespective of species, while discarding those devoid of vegetation. The criteria for vegetation detection were based on specific parameters and model training details, ensuring precise identification. More information about the output of this model will be detailed in subsection 4.4.

The application of the non-vegetation filter, utilizing the trained YOLOv5x model, led to the exclusion of a significant portion of images from each dataset, identified as lacking relevant vegetation content. These images accounted for approximately 5 to 10% of the total images in the datasets collected from various locations including Auckland, Hamilton, and Cambridge. The datasets, collected on different dates, all underwent this filtering process. The effectiveness of this

filter is demonstrated in Table 4-1, which provides a comparative overview of the number of non-vegetation images excluded across different collection dates and locations.

**Table 4-1:** Analysis of non-vegetation image filtering across different datasets.

| Date | Location | Total images | Non-vegetation images |
|---|---|---|---|
| 2023-01-20 | Hamilton | 2295 | 195 |
| 2023-03-17 | Hamilton | 3630 | 302 |
| 2023-03-23 | Hamilton | 1230 | 73 |
| 2023-03-27 | Hamilton | 1635 | 112 |
| 2023-03-30 | Hamilton | 4720 | 344 |
| 2023-04-07 | Auckland | 12434 | 1075 |
| 2023-05-24 | Hamilton | 6144 | 448 |
| 2023-06-09 | Hamilton | 5517 | 329 |
| 2023-07-26 | Hamilton | 4640 | 417 |
| 2023-09-01 | Hamilton | 2883 | 285 |
| 2023-09-01 | Cambridge | 6341 | 570 |
| 2023-09-22 | Hamilton | 4177 | 390 |
| 2023-09-22 | Cambridge | 3206 | 301 |

An analysis of the images filtered out by the non-vegetation filter indicates that variations in the number of non-vegetation images during data collection are influenced by several key factors. These include the chosen routes, the frequency and duration of stops, and the speed of the vehicle carrying the imaging system. Areas with more urban infrastructure or less vegetation typically had a higher count of non-vegetation images, while routes with more greenery showed fewer such images. The speed of the vehicle also played a role, with slower speeds leading to more images and a higher likelihood of capturing non-vegetation elements, especially in urban areas. These observations highlight the need for careful planning of data collection routes and vehicle speed to optimize image quality and relevance for urban vegetation surveillance. This meticulous approach to dataset refinement was crucial, enhancing the efficiency of the study and ensuring the deep learning models were trained on highly relevant data, thus improving the models' accuracy and real-world applicability.

### 4.2.2 Blur images filter

In addressing the challenge of blurry images within the urban imagery dataset for this study, two distinct methods were evaluated: the Fast Fourier Transform (FFT) technique and the Laplacian Variance technique. Each method employs a unique approach to identify and filter out blurred images, a crucial step for ensuring the clarity and usability of the dataset in training deep learning models. Following an initial comparison of these techniques for identifying and removing blurred images, Table 4-2 presents a detailed analysis of the advantages and disadvantages of each technique.

**Table 4-2:** Comparative analysis of FFT and Laplacian Variance techniques.

| Technique | Pros | Cons |
|---|---|---|
| Fast Fourier Transform (FFT) | - Assesses high-frequency content to identify blur.<br>- Detects both overall and edge blur.<br>- Works well in different lighting and contrasts. | - More complex implementation than spatial domain methods |
| Laplacian Variance | - Straightforward implementation | - Less effective in low-edge or low-contrast images.<br>- Risk of misclassifying smooth-textured images as blurred |

After the initial comparison of the two techniques, their effectiveness was further evaluated by applying them to a dataset. Several images from this dataset, along with the computational outputs of each technique, are exemplarily showcased in Figure 4.1.

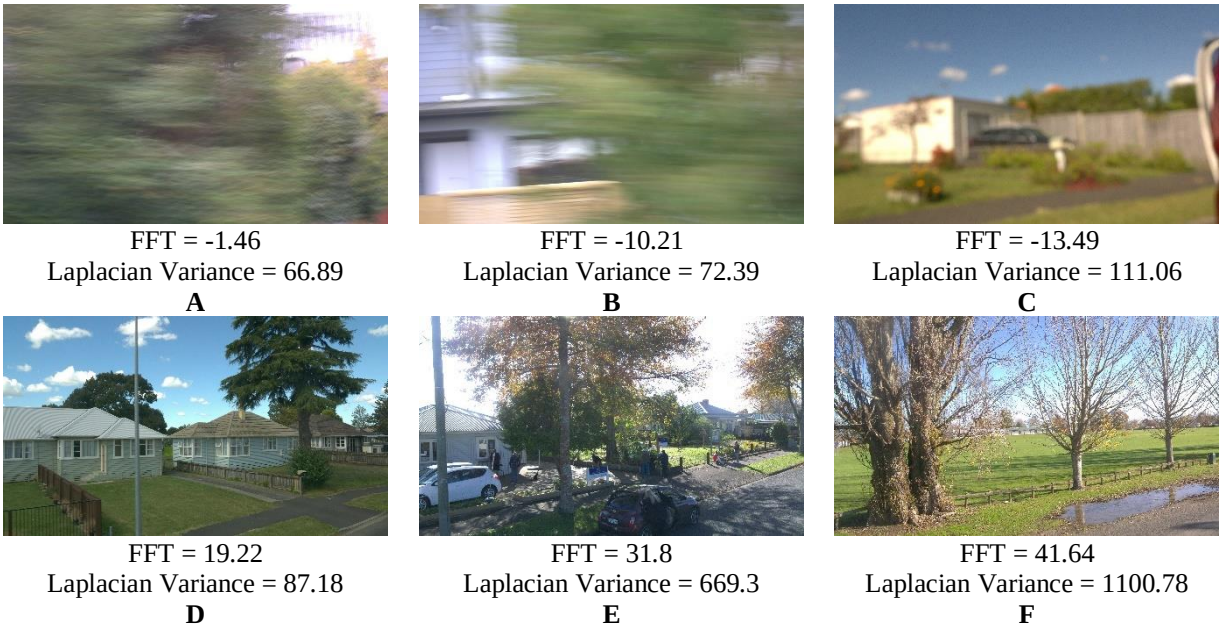



**Figure 4.1:** Comparative analysis of blurred and clear images using different techniques. The top row (A, B, C) are examples of blurred images, while the bottom row (D, E, F) are clear images; each image is annotated with its respective FFT and Laplacian Variance values, illustrating the distinction in numerical readings between blurred and clear images.

As explained in the system design and methodology chapter, both the Fast Fourier Transform (FFT) and Laplacian Variance techniques require defining a threshold to separate blurred images from clear ones. Analyzing the results using the FFT technique, as exemplified with six images in Figure 3, a threshold can be approximately set between -1.45 and 19.20 (the range between Figure 4.1C and Figure 4.1D). This threshold effectively creates a boundary between blurred and clear images, allowing for the removal of blurred images from the dataset. However, selecting a threshold for the Laplacian Variance technique poses a challenge. If a threshold close to 111, the

highest number for blurred images (as shown in Figure 4.1C), is chosen, clear Figure 4.1D gets mistakenly classified as blurred and removed from the dataset. Conversely, setting the threshold based on the lowest number of clear images, slightly less than 87 (Figure 4.1D), would incorrectly retain the blurred Image Figure 4.1C, leading to errors in subsequent stages like model training.

The reason for this error, especially considering the nature of Figure 4.1D, which predominantly features the sky, relates to the basis of the Laplacian technique in identifying and calculating edges. A significant portion of Figure 4.1D, being the sky, lacks edges, resulting in a lower calculated value using the Laplacian technique. Since some of the collected images, due to the camera angle relative to the ground, include sections of the sky, this technique was not utilized for removing blurred images from the dataset.

After applying the non-vegetation filter and removing such images from the datasets, the remaining images in each dataset were then analyzed with the blur images filter to identify and eliminate blurred images. The effectiveness of this filter is demonstrated in Table 4-3, which provides a comparative overview of the number of blurred images removed from each dataset.

**Table 4-3:** Analysis of blur image filtering across different datasets.

| Date | Location | Total images | Blur images |
|---|---|---|---|
| 2023-01-20 | Hamilton | 2100 | 630 |
| 2023-03-17 | Hamilton | 3328 | 798 |
| 2023-03-23 | Hamilton | 1157 | 231 |
| 2023-03-27 | Hamilton | 1523 | 304 |
| 2023-03-30 | Hamilton | 4376 | 350 |
| 2023-04-07 | Auckland | 11359 | 628 |
| 2023-05-24 | Hamilton | 5696 | 227 |
| 2023-06-09 | Hamilton | 5188 | 190 |
| 2023-07-26 | Hamilton | 4223 | 253 |
| 2023-09-01 | Hamilton | 2598 | 165 |
| 2023-09-01 | Cambridge | 5771 | 343 |
| 2023-09-22 | Hamilton | 3787 | 227 |
| 2023-09-22 | Cambridge | 2905 | 170 |

Table 4-3 indicates a decreasing trend in the number of blurred images over time. This reduction can be attributed to the optimization of camera settings and the identification of the most effective configurations for capturing higher-quality images. Additionally, adjustments were made to the camera angle relative to the direction of movement to determine the best possible angle. Furthermore, modifications in the chosen routes were implemented to minimize road irregularities such as speed bumps or potholes, which could cause camera shake. Consequently, in the later data collection sessions, the number of blurred images was reduced to approximately 5 to 6 percent.

### 4.2.3 Repetitive images filter

For finding and removing repetitive images from datasets, three distinct methods were explored: the Mean Squared Error technique (MSE), the Normalized Cross-Correlation technique (NCC), and the utilization of GPS coordinates. Each method offers a unique approach to identify and eliminate duplicate or near-duplicate images, a vital step to ensure the diversity and accuracy of the dataset for deep learning model training. The Mean Squared Error technique focuses on the pixel-by-pixel comparison of images, the Normalized Cross-Correlation technique assesses the similarity between images based on their overall patterns and features, and the GPS coordinates approach uses the geographical location data to identify potential repetitions. An initial comparison of these techniques, assessing their effectiveness in removing repetitive images, is presented in Table 4-4, which includes the advantages and disadvantages of each method.

**Table 4-4:** Comparative analysis of image repetition removal techniques.

| Technique | Pros | Cons |
|---|---|---|
| MSE | - Provides a clear quantitative measure for image similarity.<br>- Simple.<br>- inexpensive to compute.<br>- parameter-free.<br>- Effective for images with consistent lighting and contrast. | - Sensitive to changes in brightness and contrast. |
| NCC | - Effective in measuring image similarity.<br>- insensitive to brightness and contrast variations | - Computationally more intensive than MSE |
| Using GPS Coordinates | - High computational speed<br>- uses geographic data for similarity assessment. | - Limited by the accuracy of GPS data<br>- can mistakenly identify different images as similar on routes with a single-lane.<br>- Cannot detect image similarity based on visual content. |

Considering the disadvantages highlighted for the GPS-based method in Table 4-4, and the errors observed in some data collection periods regarding the accurate recording of GPS coordinates, the GPS coordinates analysis method was not used for identifying and removing repetitive images. Instead, the focus was placed solely on the remaining two methods. Their effectiveness was evaluated by applying them to a dataset. Several images from this dataset, along with the computational outputs of each technique, are exemplarily showcased in Figure 4.2, demonstrating their performance in processing and analysis.

| | | | | MSE | NCC |
|---|---|---|---|---|---|
| **A** |  | And |  | 8.89 | 0.97 |
| **B** |  | And |  | 103.7 | 0.8 |
| **C** |  | And |  | 12.42 | 0.9 |
| **D** |  | And |  | 93.25 | 0.09 |

**Figure 4.2:** Comparative analysis of MSE and NCC techniques in image pair analysis.

Upon analyzing the results obtained from applying these two techniques to images, it was evident that the Mean Squared Error (MSE) technique is sensitive to changes in brightness and contrast. As demonstrated in the pair of images in Figure 4.2, Figure 4.2A and Figure 4.2B were captured in identical conditions and from a fixed angle. The only difference between them is the lighting variations, which are a common occurrence during urban vegetation imagery due to changing light conditions throughout a data collection period. However, the MSE results show significant deviations resulting from these lighting and contrast changes, as mentioned in Table 4-4, which is due to the sensitivity of this method to lighting and contrast variations. On the other hand, the numerical output from the Normalized Cross-Correlation (NCC) technique showed very little change.

When examining Figure 4.2C, they become clear that under consistent lighting and contrast conditions, the MSE technique indicates a low numerical value, suggesting a high similarity. In contrast, image Figure 4.2D, with a change in camera angle, is correctly identified by the MSE as different images, thus recording a higher value.

Given that both techniques require accurately defined thresholds for optimal performance, Figure 4.2 illustrates that selecting a precise threshold for the MSE technique can be challenging due to lighting and contrast variations. In contrast, the NCC method, being less sensitive to these variations, allows for threshold selection with less error margin. Therefore, the NCC technique emerges as a more suitable option for use in the similar image filtering process.

The remaining images from each dataset, after being filtered through the blur image filter, were then analyzed using the Normalized Cross-Correlation (NCC) technique to identify and remove any repetitive images. The effectiveness of this filter is demonstrated in Table 4-5, which provides a comparative overview of the number of duplicate images removed from each dataset.

**Table 4-5:** Analysis of similar images filtering across different datasets.

| Date | Location | Total images | Similar images |
|---|---|---|---|
| 2023-01-20 | Hamilton | 1470 | 238 |
| 2023-03-17 | Hamilton | 2530 | 404 |
| 2023-03-23 | Hamilton | 926 | 103 |
| 2023-03-27 | Hamilton | 1219 | 152 |
| 2023-03-30 | Hamilton | 4026 | 644 |
| 2023-04-07 | Auckland | 10731 | 1931 |
| 2023-05-24 | Hamilton | 5469 | 727 |
| 2023-06-09 | Hamilton | 4998 | 749 |
| 2023-07-26 | Hamilton | 3970 | 537 |
| 2023-09-01 | Hamilton | 2433 | 291 |
| 2023-09-01 | Cambridge | 5428 | 868 |
| 2023-09-22 | Hamilton | 3560 | 427 |
| 2023-09-22 | Cambridge | 2735 | 331 |

An analysis of Table 4-5 revealed that typically between 13 to 17 percent of the total images obtained in each data collection period are repetitive images. As mentioned in the system design and methodology chapter, the primary reason for this is the necessary stops made by the vehicle carrying the imaging system, in compliance with traffic regulations such as stopping at traffic lights or adhering to 'give way' signs. Unlike non-vegetation and blurred images, which can be reduced through specific methods, the number of repetitive images cannot be significantly decreased due to the inherent nature of their occurrence in the datasets.

### 4.2.4 Privacy protection filter

After removing non-vegetation, blurred, and repetitive images, a privacy protection filter was applied. Unlike the previous filters, this filter does not remove images containing personal information. Instead, it blurs specific objects of high privacy importance, such as faces and vehicle

license plates. To demonstrate the results of this filter, which allows for adjustable blurring intensity, an image with varying degrees of blurring is presented in Figure 4.3.

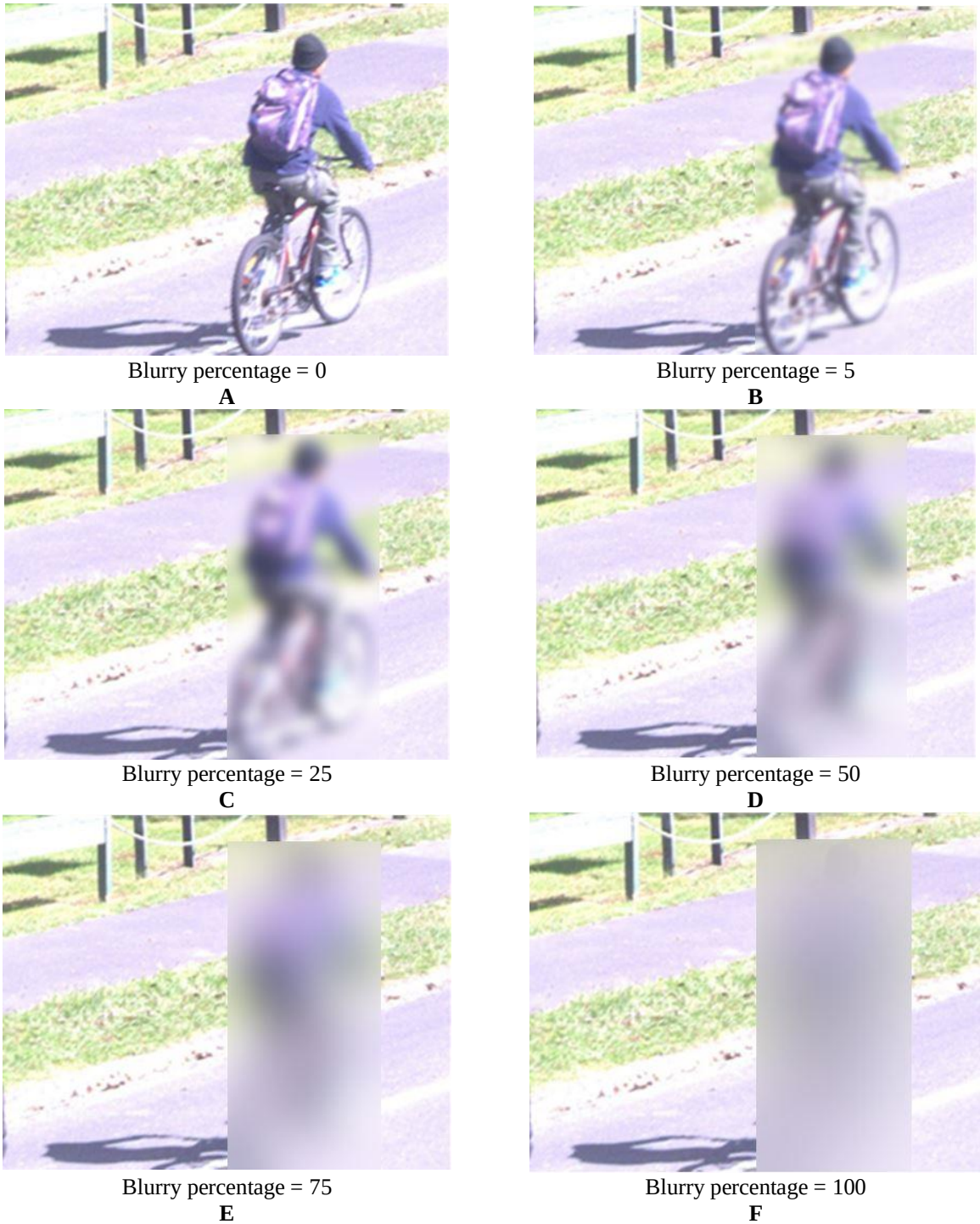



**Figure 4.3:** Examples of privacy protection outputs demonstrating various degrees of blurriness, ranging from 0% to 100%.

Considering that this filter allows for the adjustment of the blurriness intensity at various percentages, a 5% blur level was chosen for this study. The primary goal of blurring personal

information was merely to render it unreadable and unidentifiable. Therefore, the minimum necessary percentage was selected to ensure that blurring items overlapping with the main target of this research, which is vegetation coverage, does not interfere with the process of vegetation identification.

Table 4-6 presents a comprehensive analysis of the urban imagery dataset, comparing the number of images before and after the application of various filters. Initially, the 'Raw Dataset' column reflects the total number of images collected for each dataset. After applying filters for non-vegetation, blurred, and repetitive images, the 'Prepared Dataset' column shows the remaining images, now refined and ready for further analysis. The 'Percentage Removed' column, calculated as the difference between the raw and prepared datasets relative to the raw dataset, quantifies the extent of image reduction. This percentage highlights the efficiency of the applied filters in refining the dataset. For instance, on 2023-01-20 in Hamilton, 46.32% of the images were removed, leaving 1232 images out of the original 2295. This pattern of significant image reduction is consistent across all datasets, demonstrating the effectiveness of the preprocessing steps in optimizing the dataset for subsequent stages of the study.

**Table 4-6:** Comparison of datasets before and after applying Filters.

| Date | Location | Raw dataset | Prepared dataset | Removed images (%) |
|---|---|---|---|---|
| 2023-01-20 | Hamilton | 2295 | 1232 | ≈ 46 |
| 2023-03-17 | Hamilton | 3630 | 2126 | ≈ 41 |
| 2023-03-23 | Hamilton | 1230 | 823 | ≈ 33 |
| 2023-03-27 | Hamilton | 1635 | 1067 | ≈ 35 |
| 2023-03-30 | Hamilton | 4720 | 3382 | ≈ 28 |
| 2023-04-07 | Auckland | 12434 | 8800 | ≈ 29 |
| 2023-05-24 | Hamilton | 6144 | 4742 | ≈ 22 |
| 2023-06-09 | Hamilton | 5517 | 4249 | ≈ 22 |
| 2023-07-26 | Hamilton | 4640 | 3433 | ≈ 26 |
| 2023-09-01 | Hamilton | 2883 | 2142 | ≈ 25 |
| 2023-09-01 | Cambridge | 6341 | 4560 | ≈ 28 |
| 2023-09-22 | Hamilton | 4177 | 3133 | ≈ 25 |
| 2023-09-22 | Cambridge | 3206 | 2404 | ≈ 25 |

## 4.3 Annotation tools

Since five annotation tools were introduced in the system design and methodology chapter, one of these tools needed to be selected based on the project's requirements. Therefore, these tools were examined from various perspectives to determine their suitability for specific needs. The results of this comprehensive analysis are presented in Table 4-7. The tools analyzed include ADAMS, LabelImg, Roboflow.com, Label Studio, and a Customized version of Label Studio.

Each tool is evaluated across criteria such as platform type, user interface, multi-user support, input and output formats, annotation features, compatibility with machine learning models, etc.

**Table 4-7**: Comprehensive comparison of image annotation tools.

| Factors | Image annotation tools | | | | |
|---|---|---|---|---|---|
| | **ADAMS** | **LabelImg** | **Roboflow.com** | **Label Studio** | **Customized Label Studio** |
| Platform | Desktop | Desktop | Web-based | Desktop | Desktop |
| User Interface and Ease of Use | Moderate | Easy | Easy | Easy | Moderate |
| Multi-User Support | No | No | Yes | Yes | Yes |
| Input | Image | Image | Image<br>Video | Text, Images, Videos, Audio, Time Series (CSV, JSON) | Text, Images, Videos, Audio, Time Series (CSV, JSON) |
| Output format | Text file | XML (PASCAL VOC),<br>YOLO format | Coco JSON, VGG, Vott JSON, Marmot XML, YOLO PyTorch, YOLO Darknet TXT, Kaggle CSV | COCO, CSV, JSON, JSON_MIN, PASCAL-VOC-XML, spaCy, TSV, YOLO | COCO, CSV, JSON, JSON_MIN, PASCAL-VOC-XML, spaCy, TSV, YOLO |
| Annotation Features | Polygon, Bounding box | Bounding box | Polygons, Bounding box, Key-points. | Polygons, Bounding box, Key-points. | Polygons, Bounding box, Key-points. |
| Machine Learning Model Compatibility | No | Yes | Yes | Yes | Yes |
| Internet Dependency | No | No | Yes | Optional | Optional |
| Customization and Flexibility | Yes | No | No | Yes | Yes |
| Cost and Subscription Requirements | Free | Free | Free (with limitation)/Paid | Free/Paid | Free/Paid |
| Integration with External Systems | No | No | Yes | Yes | Yes |
| Technical Setup and Infrastructure | Complex | Easy | Easy | Complex | Complex |

The comparative analysis of image annotation tools in Table 5 reveals distinct strengths and limitations for each tool, guiding the selection process based on project requirements. ADAMS and LabelImg, as desktop platforms, offer a degree of stability and control but vary in user interface ease and multi-user support, with ADAMS being more complex and not supporting multiple users. In contrast, Roboflow.com's web-based platform excels in collaborative capabilities, although it

depends on internet connectivity. Label Studio and its customized version provide a balance between robust features and user interface simplicity, with enhanced flexibility in multi-user collaboration.

Input format compatibility is a crucial factor where LabelImg focuses primarily on images, while Roboflow.com and Label Studio offer broader compatibility, including various image and video formats. This makes them more versatile for diverse dataset requirements. In terms of output formats, Roboflow.com and Label Studio stand out with their wide range of options, facilitating integration with different machine-learning models and external systems. This feature is particularly advantageous for complex projects requiring extensive data processing and management.

Annotation features also play a pivotal role. While LabelImg is limited to bounding boxes, ADAMS, Roboflow.com, and Label Studio offer a more comprehensive set of features, including polygons and key-points, enhancing their applicability for detailed annotation tasks. The customization and flexibility of Label Studio, especially in its customized version, offer tailored solutions for specific project needs, although this comes with the trade-off of more complex technical setup and infrastructure requirements.

The cost and subscription requirements present another decision point. ADAMS and LabelImg are free tools, offering cost-effective solutions, whereas Roboflow.com and Label Studio have both free and paid options, potentially adding to the project budget. Finally, the integration with external systems is a notable advantage for Roboflow.com and Label Studio, facilitating seamless data management and enhancing collaborative efforts across distributed teams.

After considering all aspects, the customized version of Label Studio was deemed suitable for subsequent stages of the research due to its ability to read data files created by the city council. However, for this specific phase of the research, which involved identifying a limited number of tree species and individual trees, Roboflow.com was utilized.

## 4.4 Urban vegetation detection

The outcomes of extensive work with YOLOv5 and YOLOv8 sub-models, trained specifically for identifying urban vegetation, have been meticulously examined. This examination focused on optimizing the models' capabilities to accurately recognize vegetation in diverse urban conditions. The comprehensive analysis presented here includes precision, recall, mean Average Precision

(mAP) at various Intersection over Union (IoU) thresholds and accuracy for training and validation phases, along with the time taken for training each model. Table 4-8 is aimed at providing a detailed comparative view of the models' performances, highlighting the key aspects that determine their effectiveness in urban vegetation detection.

**Table 4-8:** Comparative analysis of YOLOv5 and YOLOv8 sub-models.

| Model | Precision | | Recall | | mAP50 | | mAP50-95 | | Training Time (Hour) |
|---|---|---|---|---|---|---|---|---|---|
| | Train | Valid | Train | Valid | Train | Valid | Train | Valid | |
| Yolov5n | 0.93 | 0.87 | 0.90 | 0.85 | 0.93 | 0.90 | 0.68 | 0.59 | 0.94 |
| Yolov5s | 0.96 | 0.90 | 0.88 | 0.89 | 0.97 | 0.94 | 0.77 | 0.67 | 0.935 |
| Yolov5m | 0.96 | 0.92 | 0.92 | 0.89 | 0.97 | 0.94 | 0.80 | 0.70 | 1.026 |
| Yolov5l | 0.96 | 0.92 | 0.92 | 0.90 | 0.97 | 0.94 | 0.83 | 0.72 | 1.15 |
| Yolov5x | 0.96 | 0.93 | 0.90 | 0.90 | 0.97 | 0.95 | 0.84 | 0.74 | 2.05 |
| Yolov8n | 0.94 | 0.86 | 0.90 | 0.82 | 0.96 | 0.89 | 0.76 | 0.62 | 0.995 |
| Yolov8s | 0.96 | 0.89 | 0.91 | 0.87 | 0.96 | 0.92 | 0.81 | 0.69 | 1.031 |
| Yolov8m | 0.92 | 0.87 | 0.91 | 0.84 | 0.96 | 0.91 | 0.78 | 0.65 | 1.297 |
| Yolov8l | 0.96 | 0.87 | 0.87 | 0.84 | 0.96 | 0.90 | 0.80 | 0.65 | 1.52 |
| Yolov8x | 0.96 | 0.88 | 0.90 | 0.85 | 0.98 | 0.91 | 0.83 | 0.67 | 2.14 |

The comprehensive analysis of the YOLOv5 and YOLOv8 sub-models, as shown in Table 4-8, reveals nuanced details about their performance in detecting and classifying urban vegetation. Across the board, the YOLOv5 series exhibits high precision in the training phase, with YOLOv5m, YOLOv5l, and YOLOv5x models all achieving a remarkable precision score of 0.96. This high precision carries through to the validation phase for the YOLOv5x model, suggesting robustness in its predictive accuracy. Recall rates for YOLOv5m and YOLOv5l are equally impressive, standing at 0.92 in training and showing consistent performance in validation.

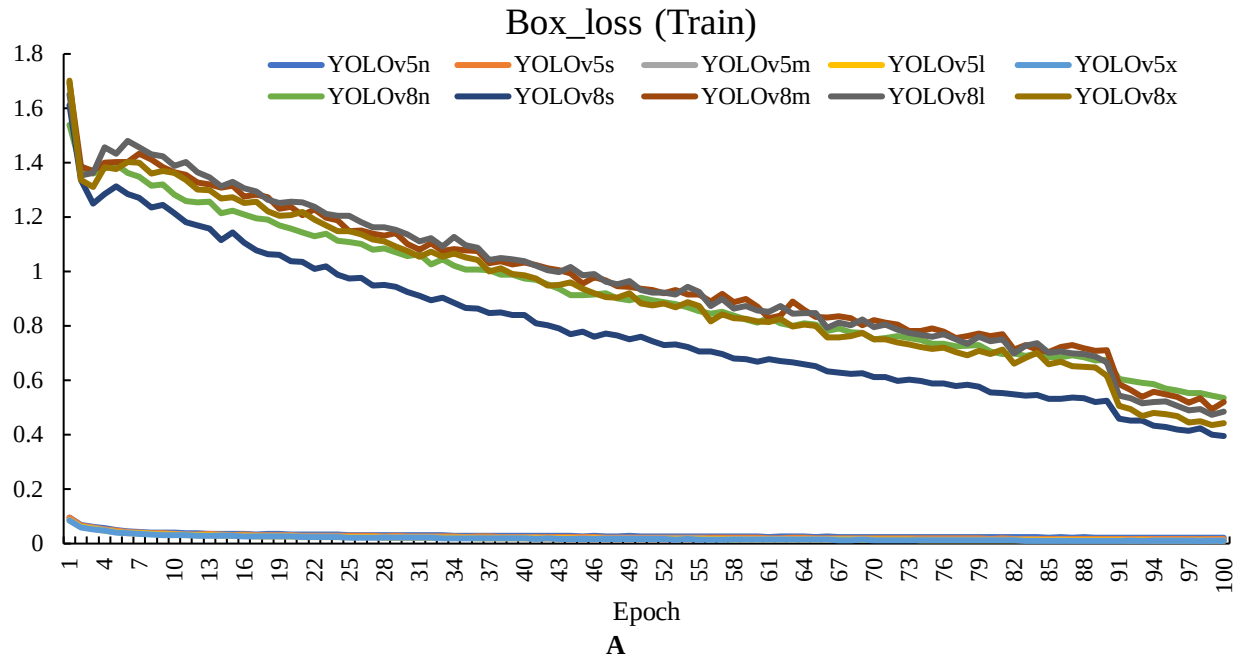
The mean Average Precision (mAP) scores further distinguish the models, with YOLOv5x and YOLOv5m leading in the training phase with a score of 0.97. YOLOv5x, in particular, demonstrates its strength across various Intersections over Union (IoU) thresholds with the highest mAP50-95 score of 0.74 in validation, pointing to its effectiveness in nuanced detection tasks.

When considering training efficiency, YOLOv5n is the most time-efficient model, requiring only 0.94 hours for training. On the other end of the spectrum lies YOLOv5x, which, despite its superior performance, necessitates a significantly longer training time of 2.05 hours. This reflects a clear trade-off between the complexity and performance of a model and the time invested in its training.

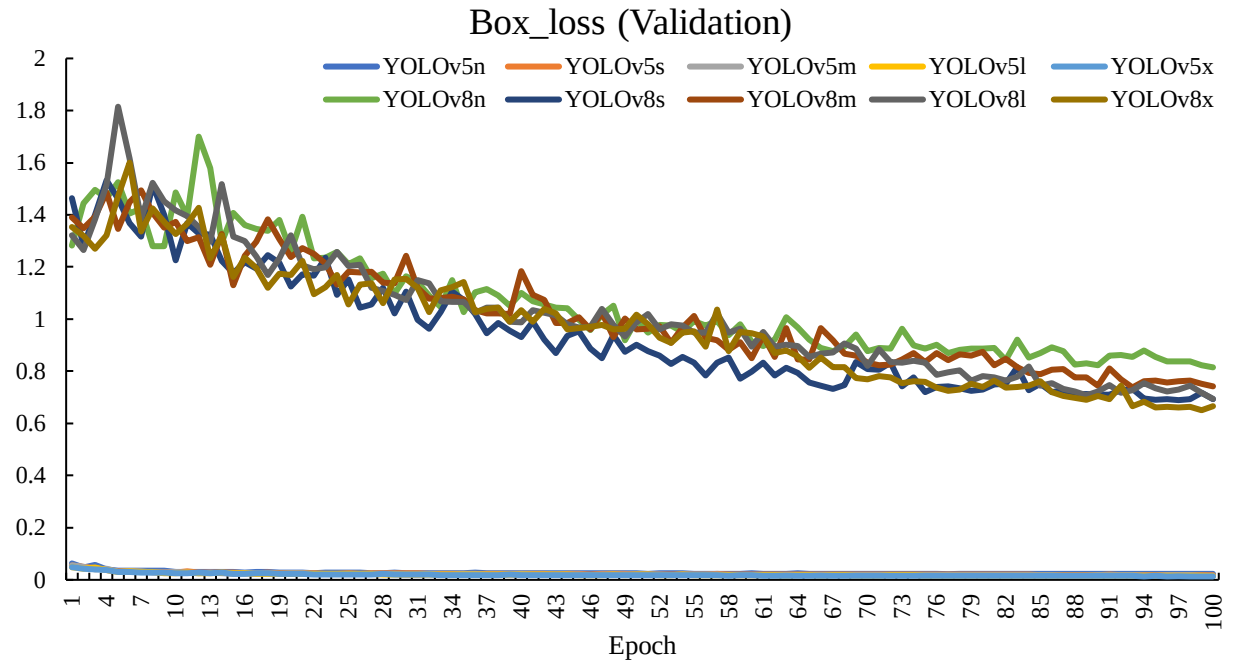
The YOLOv8 series, while demonstrating slightly lower precision and recall in the validation phase—with YOLOv8n's precision dropping from 0.94 in training to 0.86 in validation—still presents competitive metrics. YOLOv8s, in particular, showcases a good balance between performance and training time, with a relatively shorter training period of 1.031 hours compared to the more extensive YOLOv8x, which takes 2.14 hours. In synthesizing these observations, it becomes apparent that the YOLOv5 series, especially the YOLOv5x model, stands out for its detection capabilities in urban vegetation. However, the longer training times indicate a potential cost in terms of efficiency.

Box Loss is a component of the loss function used in object detection models like YOLO models. It quantifies the difference between the predicted bounding boxes and the actual ground truth boxes during model training. This metric is essential for evaluating and optimizing the accuracy of the model in detecting and correctly positioning bounding boxes around objects in an image.

According to Figure 4.4A, during training, a consistent downward trend in Box Loss is observed across epochs for all models, indicative of the models' progressive learning and improvement in bounding box predictions. Notably, models like YOLOv5n and YOLOv5s demonstrate a more rapid decrease in Box Loss, suggesting a higher efficiency in learning compared to other variants. This difference is possibly attributed to variations in model architectures or initial parameter settings. The initial Box Loss values vary significantly, with YOLOv8 models starting at a much higher loss, potentially due to the increased complexity of their tasks or different architectural nuances. As training progresses, a convergence in Box Loss is evident, though the rate and epoch at which this stabilization occurs vary among models, highlighting differences in learning dynamics.




**A**




**B**

**Figure 4.4:** Evolution of Box loss across epochs for various YOLOv5 and YOLOv8 sub-models; A. Train and B. Validation phase.

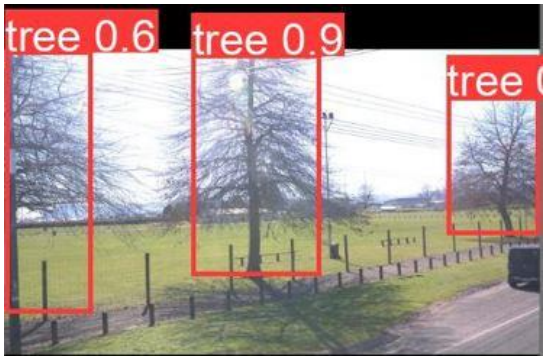
In the validation phase, Figure 4.4B showed a general decrease in Box Loss across epochs for most models, underlining their ability to generalize to new data. YOLOv5 models outperformed YOLOv8 models in terms of a more pronounced reduction in Box Loss, indicating better generalization capabilities. The higher initial Box Loss in YOLOv8 models and their varied convergence behavior suggested different responses to validation data compared to training data,

emphasizing the importance of model selection based on the specific characteristics of the task at hand.

The Box Loss analysis for the training and validation phases explains various YOLO sub-models learning and generalization capabilities. YOLOv5 models, with their rapid decrease in Box Loss and better generalization in the validation phase, might be more suitable for tasks where efficiency and quick adaptation to new data are crucial. In contrast, YOLOv8 models, despite their higher initial Box Loss and varied convergence, could offer advantages in scenarios requiring handling more complex tasks or data.

In support of the detailed analysis presented in this section, Figure 4.5 provides a visual representation of the practical outcomes achieved during the validation phase for each YOLOv5 and YOLOv8 sub-model. This figure includes two sets of images: one set for YOLOv5 sub-models and another for YOLOv8 sub-models, each using the same urban landscape image to ensure consistency in the comparison. These images vividly demonstrate the urban vegetation detection capabilities of each sub-model, with a particular focus on tree detection. It is particularly noteworthy that sub-models YOLOv5n and YOLOv8n, along with YOLOv5m and YOLOv8m, show an enhanced ability to detect a greater number of trees compared to their counterparts. This side-by-side comparison not only visually supplements the quantitative data presented in Table 4-8and Figure 4.4 but also offers intuitive insight into the real-world performance of these models. The superior tree detection abilities of certain sub-models underscore the criticality of model selection based on detection capabilities and efficiency. This comparative visualization plays a key role in highlighting the subtle differences in performance among the sub-models, providing a comprehensive view of their effectiveness in urban vegetation detection tasks.


YOLOv5n




YOLOv8n


**Figure 4.5:** Urban vegetation detection;
comparing YOLOv5 and YOLOv8 sub-models in validation phase

In light of the results from the analysis of YOLOv5 and YOLOv8 sub-models, the next logical step in this section is a comparative review with existing literature. Such a review is crucial to

contextualize our findings within the broader scope of tree detection research using various deep learning models in RGB images. The focus has been placed on studies that are primarily aimed at tree detection, irrespective of species, while most reviewed studies have centered on the identification and classification of different tree species. For more accurate comparisons, several studies have been selectively included based on their year of publication. Through the analysis of these studies, which have employed varied deep learning models, and by focusing on key metrics like precision and recall, the effectiveness of the YOLO-based approach, relative to other strategies, can be assessed. A summary of these comparative studies is set to be presented in Table 4-9, which will provide insights into the alignment or divergence of this research's results with established findings in the field of urban vegetation detection using deep learning techniques.

**Table 4-9:** Comparative analysis of deep learning models for tree detection.

| Model | Object | Precision | Recall | Reference |
|---|---|---|---|---|
| AlexNet | Tree | 0.91 | 0.7 | [339] |
| GoogleNet | | 0.98 | 0.51 | |
| DeepForest | Tree crown | 0.74 | 0.6 | [340] |
| ResNet-34 | Tree crown | 0.89 | 0.5 | [341] |
| ResNet-50 | | 0.86 | 0.91 | |
| ResNet-101 | | 0.89 | 0.82 | |
| RetinaNet | Tree crown | 0.64 | 0.62 | [342] |
| YOLOv5 | Tree trunk | 0.88 | 0.84 | [343] |
| YOLOv5m | Tree | 0.96 | 0.92 | Current |
| YOLOv8m | | 0.92 | 0.91 | research |

The comparative table illuminates the varied performance of several deep learning models in tree detection. AlexNet, with a precision of 0.91, shows high accuracy but a lower recall of 0.7, indicating a tendency to miss true positives. GoogleNet, in stark contrast, achieves a precision of 0.98 but falls short in recall at only 0.51, missing nearly half of the actual trees. DeepForest, targeting tree crown detection, presents a moderate precision of 0.74 and recall of 0.6, reflecting a balanced but not exceptional detection capability. Among the ResNet series, ResNet-34 and ResNet-101 both have a precision of 0.89, but ResNet-50 stands out with the highest recall of 0.91, suggesting its superior ability to identify most tree crowns. However, RetinaNet, used for tree crown detection as well, records lower scores with a precision of 0.64 and a recall of 0.62, indicating less effectiveness in accurate and consistent detection. YOLOv5, specializing in tree trunk detection, demonstrates a balanced high performance with a precision of 0.88 and a recall of 0.84. Most notably, the YOLOv5m and YOLOv8m models from the current research excel in tree detection, exhibiting precision (0.96 and 0.92 respectively) and recall (0.92 and 0.91 respectively). These numbers not only show their superior detection accuracy but also their consistency,

highlighting the advanced capabilities of YOLO models in urban vegetation detection, especially in comparison to other leading deep learning models.

After analyzing the results of the YOLOv5 and YOLOv8 sub-models in the identification of trees without considering species, it is essential to analyze the results related to species detection and classification, which will be addressed in the next section. It is noteworthy that since the results of the YOLOv5 and YOLOv8 models were somewhat similar, one sub-model from each (YOLOv5m and YOLOv8m) has been selected for training and evaluation in species classification. This approach allows for a focused and detailed analysis.

## 4.5 Specific species detection

Having examined the models' ability to identify urban vegetation without specific species consideration, the research then shifted focus to the training and evaluation of selected sub-models from YOLOv5 and YOLOv8, aimed at detecting and classifying eight distinct tree species. This section is dedicated to discussing the performance of the YOLOv5m and YOLOv8m models in this specialized task. It highlights their capacity for accurately identifying and distinguishing among various tree species. The effectiveness of each model in the context of species-specific tree detection, a key element in the wider scope of urban vegetation monitoring and analysis, is thoroughly assessed using the data presented in Table 4-10.

**Table 4-10:** Comparative analysis of YOLOv5m and YOLOv8m models in species detection.

| Model | Precision | | Recall | | mAP50 | | mAP50-95 | | Training Time (Hour) |
|---|---|---|---|---|---|---|---|---|---|
| | Train | Valid | Train | Valid | Train | Valid | Train | Valid | |
| Yolov5m | 0.957 | 0.91 | 0.962 | 0.93 | 0.984 | 0.95 | 0.788 | 0.69 | 2.729 |
| Yolov8m | 0.964 | 0.92 | 0.965 | 0.91 | 0.986 | 0.95 | 0.835 | 0.72 | 3.014 |

The comparative analysis of the YOLOv5m and YOLOv8m models for urban tree species detection offers a nuanced view of their capabilities (Table 4-10). YOLOv5m showcases high precision in both training and validation, with scores of 0.957 and 0.91, respectively, and excellent recall, indicating its effectiveness in identifying true positives. Its mean Average Precision scores are impressive, with a mAP50 of 0.984 during training and 0.95 in validation, signifying a reliable balance between precision and recall across the standard IoU threshold. The model also performs well at more stringent IoU thresholds, as evidenced by its mAP50-95 scores of 0.788 in training and 0.69 in validation. YOLOv5m's training efficiency is highlighted by its training time of 2.729 hours.

Yolov8m, on the other hand, slightly outperforms YOLOv5m in precision and recall, with training scores of 0.964 and 0.965, and validation scores of 0.92 and 0.91, respectively. It achieves a marginally higher mAP50 of 0.986 during training and matches YOLOv5m with a validation score of 0.95. Notably, Yolov8m excels in the mAP50-95 metric, scoring 0.835 in training and 0.72 in validation, suggesting it maintains accuracy across a range of IoU thresholds. However, this increased performance comes with a longer training time of 3.014 hours.

In essence, while both models demonstrate strong performance metrics, Yolov8m marginally surpasses YOLOv5m in most categories, making it a slightly more robust choice for species-specific tree detection. However, this superior detection capability of Yolov8m is balanced against its longer training duration. Deciding between the two models would require weighing the benefits of higher detection accuracy against the practicalities of training time.

As illustrated in Figure 4.6, an evaluation of the box_loss for the YOLOv8m and YOLOv5m models over 100 epochs reveals distinct trends in both training and validation phases. Initially, YOLOv8m registers a significantly higher box loss than YOLOv5m. Yet, as training progresses, both models exhibit a consistent decline in box loss. This pattern indicates an ongoing improvement in their ability to accurately predict bounding boxes, demonstrating the models' capacity for learning and advancement as the epochs advance.

YOLOv5m demonstrates a more pronounced reduction in loss, with its validation loss closely converging with the training loss, suggesting efficient learning and a stable model without overfitting. In comparison, YOLOv8m, despite starting with a higher loss, reduces its validation loss significantly, hinting at an ability to generalize.

By the final epoch, YOLOv5m and YOLOv8m display similar training and validation losses, with both models reaching a point where further training might yield minimal improvements. This close performance towards the end of the training period highlights the potential of both models for accurate object detection, each with distinct learning dynamics and generalization to unseen data.

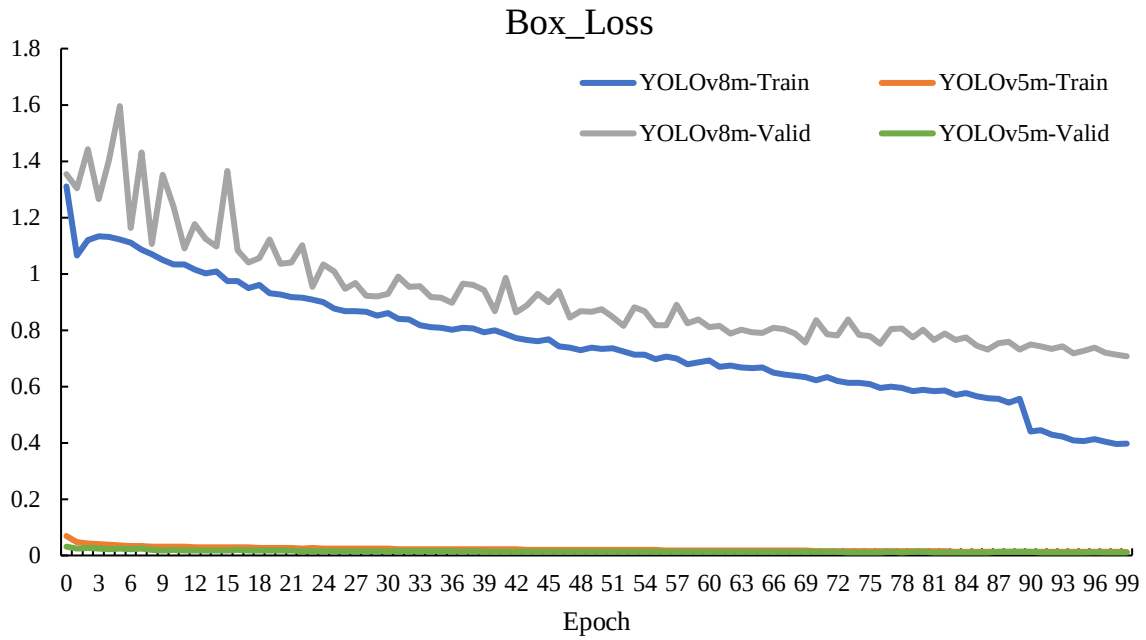
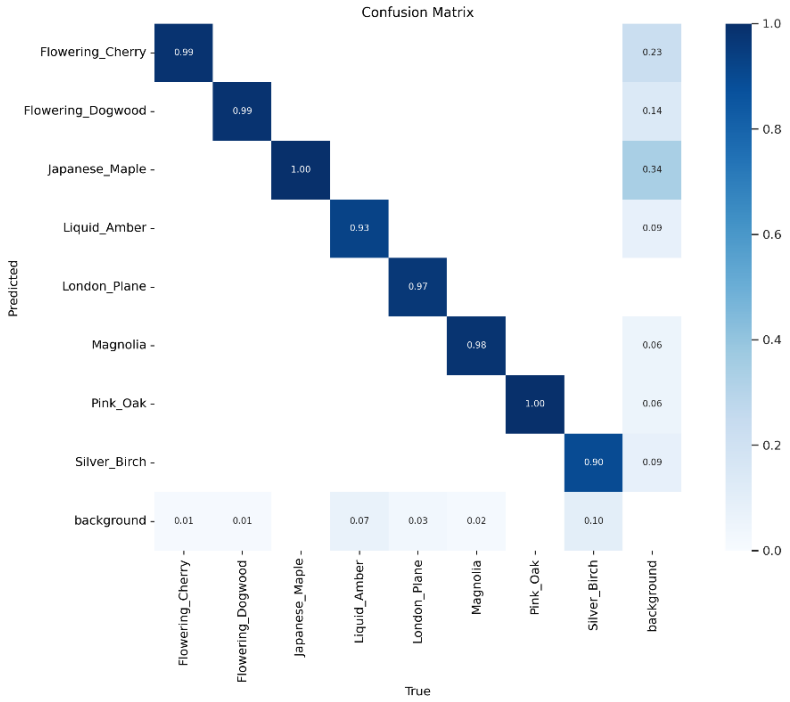


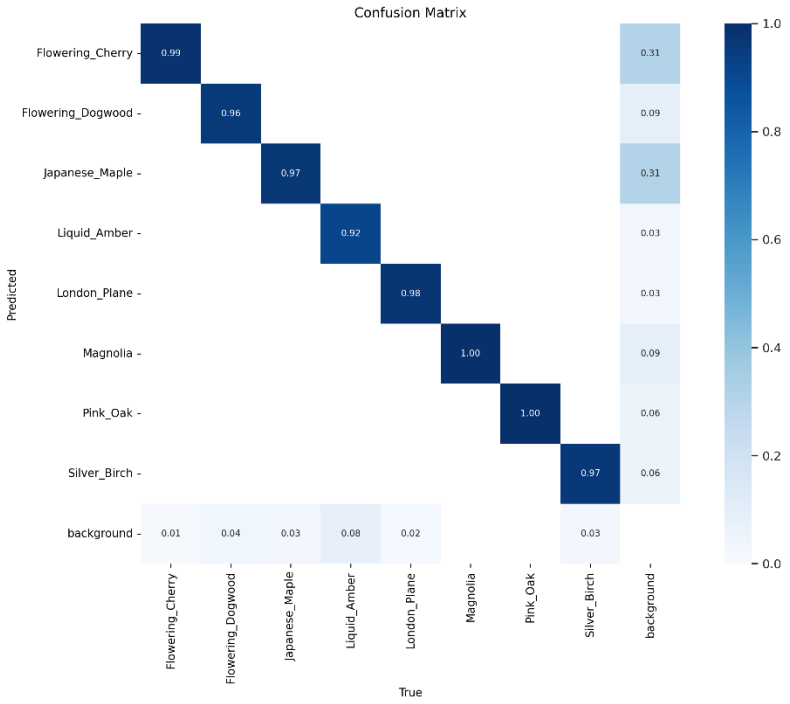

**Figure 4.6:** Evolution of Box loss across epochs for YOLOv5m and YOLOv8m during the train and validation phase.

The confusion matrices for the YOLOv5m and YOLOv8m models have been examined, providing a basis for comparison in the classification of urban tree species (Figure 4.7). In the matrix associated with the YOLOv5m model (Figure 4.7A), a high degree of accuracy is observed across most species' classifications, with the Japanese Maple and Pink Oak categories being identified with nearly perfect precision. It is noted, however, that there is a slight propensity for the model to classify tree species as background, although at a very low frequency, with the highest misclassification observed for Silver Birch at a rate of 0.1.

Conversely, the YOLOv8m model's confusion matrix (Figure 4.7B) also indicates high precision levels, with all species showing precision rates above 0.92 and Magnolia and Pink Oak reaching a perfect score. The rates at which species are misclassified as background are slightly elevated in comparison to YOLOv5m, with Liquid Amber exhibiting the highest rate of 0.08. When these models are compared, both are found to be highly effective in identifying and classifying tree species, with YOLOv5m showing a marginal advantage in reducing misclassifications as background. The preference for one model over the other may be influenced by the specific demands of the application, where YOLOv5m could be favored for its lower false positive rate. Nonetheless, should a negligible increase in false positives be acceptable for marginally improved precision in certain classes, the YOLOv8m model stands out as a viable option.




**A**




**B**

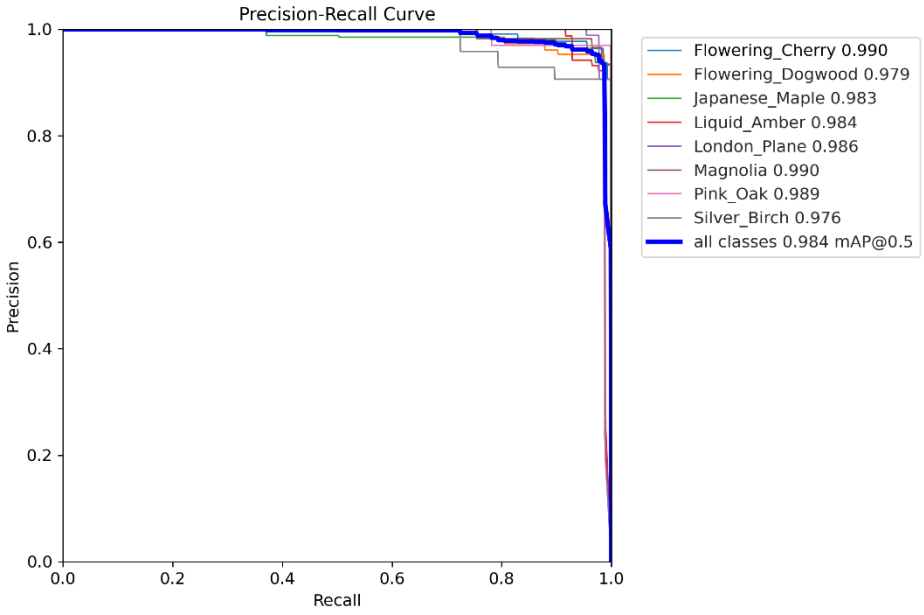
**Figure 4.7:** Comparative confusion matrices for urban tree species classification; A. YOLOv5m and B. YOLOv8m.

The last analysis of the two selected models was based on the precision-recall curves and mean Average Precision (mAP) values at an Intersection over Union (IoU) threshold of 0.5 (Figure 4.8). The YOLOv5m model, while exhibiting a slightly lower overall performance with a mAP of 0.984,

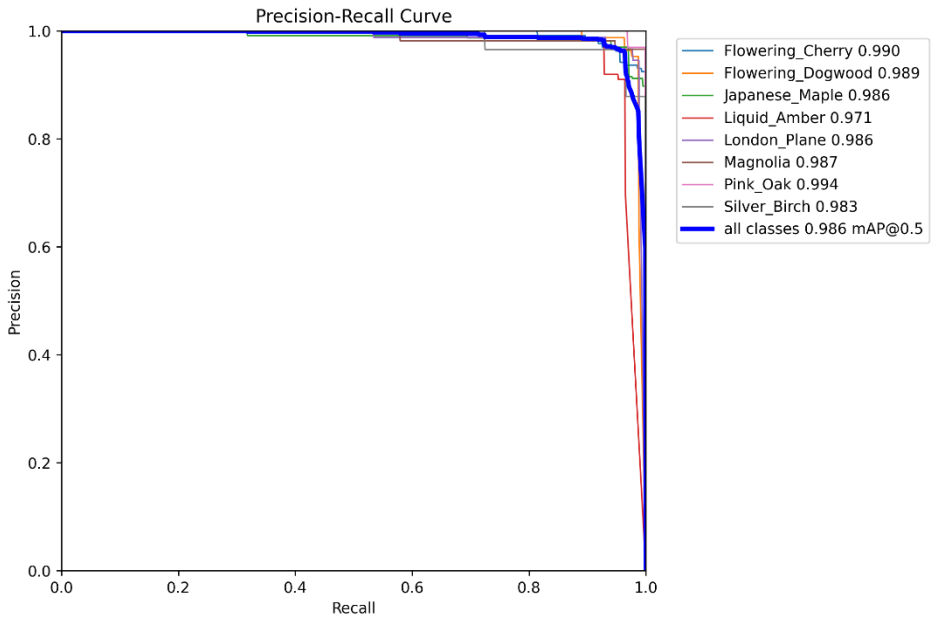
still maintained commendable precision across various classes (Figure 4.8A). Notably, it matched the YOLOv8m in detecting Magnolia and Flowering Cherry with scores of 0.990 but demonstrated marginally reduced precision in classes like Flowering Dogwood (0.979) and Silver Birch (0.976).

In contrast, the YOLOv8m model showcased remarkable precision across all classes, achieving an overall mAP of 0.986 (Figure 4.8B). It attained near-perfect scores in detecting species such as Flowering Cherry (0.990) and Pink Oak (0.994), underscoring its highly accurate identification and localization capabilities. The lowest score for the YOLOv8m model was observed in Liquid Amber, with a score of 0.971, which still signifies a significantly high level of accuracy.

This comparative analysis underscores that both models exhibit a high degree of efficacy in object detection tasks, albeit with minor differences in specific classes. While the YOLOv8m model has a slight edge in generalized performance, the YOLOv5m model's performance is noteworthy, especially given its potential benefits in terms of computational efficiency and resource utilization. These aspects are crucial in practical applications, where the YOLOv5m model could be more suitable, despite the marginal superiority in accuracy of the YOLOv8m model.




**A**




**B**

**Figure 4.8:** Precision-Recall curves;
A. YOLOv5m and B. YOLOv8m.

Figure 4.9 displays several examples of the outputs from the YOLOv5m (Figure 4.9A) and YOLOv8m (Figure 4.9B) models, showcasing their capabilities in identifying various tree species. As evident in the images, both models demonstrate a high degree of accuracy in tree species identification with only marginal differences. Upon closer examination of the images, it can be observed that in some cases, the YOLOv5m model identifies a greater number of trees, particularly those that are partially visible at the edges of the images. This detection is especially noteworthy for trees that are not fully captured in the frame, indicating the model's sensitivity to partial inputs.


**A**

**B**

**Figure 4.9:** Comparative visualization of urban tree species detection during the validation phase by A. YOLOv5m and B. YOLOv8m.

To compare the models trained in this study for the identification and classification of urban tree species, some research was selected. The criterion for their selection was the diversity of the models used in these studies, allowing for a comprehensive comparison with the models utilized in this research. Table 4-11 presents the models used in these studies, along with the number of selected tree species and their respective precision and recall metrics.

**Table 4-11:** Comparative analysis of deep learning models for tree species classification.

| Model | Number of species | Precision | Recall | Reference |
|---|---|---|---|---|
| ResNet-50 | 5 | 0.51 | 0.51 | [218] |
| DenseNet | 5 | 0.82 | 0.8 | [344] |
| YOLOv3 | 6 | 0.77 | 0.26 | [11] |
| AlexNet<br>ResNet-50<br>Decision Tree<br>Random Forest | 5 | 0.72<P<0.96 | 0.48<R<0.94 | [345] |
| YOLOV3 | 5 | 0.76 | 0.73 | [226] |
| Mask R-CNN | 8 | 0.85 | 0.84 | [346] |
| ResNet-101 | 4 | 0.75 | 0.78 | [347] |
| YOLOv5m | 8 | 0.957 | 0.962 | Current |
| YOLOv8m | | 0.964 | 0.965 | research |

Based on Table 4-10, ResNet-50, a model in tree species detection field, showed modest precision and recall at 0.51, focusing on 5 species. DenseNet, another model, also targeted 5 species and improved upon the performance with a precision of 0.82 and a recall of 0.8. In another study where four models – AlexNet, ResNet-50, Decision Tree, and Random Forest – were applied to 5 different tree species, yielding a performance range with precision between 0.72 and 0.96 and recall between 0.48 and 0.94.

Further advancements are seen in models like YOLOv3, which was trained on 6 species and showed a moderate performance. Mask R-CNN, trained on 8 species, achieved higher precision (0.85) and recall (0.84), indicating its effectiveness. ResNet-101, focusing on 4 species, also showed a balanced performance with a precision of 0.75 and a recall of 0.78. However, the most significant progress is evident in the current research employing the YOLOv5m and YOLOv8m models. Trained on 8 species, these models outshine the others with precision values of 0.957 and 0.964 and recall values of 0.962 and 0.965, respectively. This high level of precision and recall underscores a major advancement in the field, demonstrating the capability of these models to accurately and reliably detect and classify a broader range of tree species. This trend signifies the ongoing evolution of deep learning techniques, leading to more sophisticated and capable models in ecological data analysis.

Another approach examined in this research is the use of transfer learning methods and freezing different layers of models, assessing their impact on model performance. The results of these approaches will be analyzed in the following subsection.

### 4.5.1 Trained models with different approaches

In the training process of the YOLOv5m and YOLOv8m models, transfer learning methods were also implemented for both. Additionally, the technique of freezing layers was employed in the YOLOv5m model, which involved freezing the backbone layers (the first 10 layers) and all layers (24 layers). As detailed in the system design and methodology chapter, the YOLOv8 model, released in 2023, does not yet have the capability for layer freezing. The following table provides a comprehensive comparison of these models under various approaches. It outlines their performance metrics, including precision, recall, mean Average Precision (mAP50 and mAP50-95), and training time, both in training and validation phases, offering insights into the effectiveness of each approach in detecting different tree species. It is important to note that the YOLOv5m and YOLOv8m models are included once more in Table 4-12. This repetition is intended to facilitate a more straightforward comparison with other YOLO model training methods.

**Table 4-12:** Comparative analysis of different YOLO model approaches for classifying tree species.

| Model | Precision | | Recall | | mAP50 | | mAP50-95 | | Training Time (Hour) |
|---|---|---|---|---|---|---|---|---|---|
| | Train | Valid | Train | Valid | Train | Valid | Train | Valid | |
| Yolov5m | 0.96 | 0.91 | 0.96 | 0.93 | 0.98 | 0.95 | 0.79 | 0.69 | 2.729 |
| YOLOv5m (Transfer Learning) | 0.96 | 0.96 | 0.96 | 0.95 | 0.98 | 0.98 | 0.81 | 0.81 | 2.072 |
| YOLOv5m (Freeze Backbone) | 0.96 | 0.88 | 0.96 | 0.90 | 0.98 | 0.92 | 0.75 | 0.66 | 1.499 |
| YOLOv5m (Freeze All Layers) | 0.28 | 0.24 | 0.55 | 0.51 | 0.21 | 0.18 | 0.11 | 0.1 | 0.92 |
| Yolov8m | 0.97 | 0.92 | 0.96 | 0.91 | 0.98 | 0.95 | 0.83 | 0.72 | 3.014 |
| YOLOv8m (Transfer Learning) | 0.97 | 0.93 | 0.96 | 0.92 | 0.99 | 0.96 | 0.84 | 0.74 | 2.885 |

As evident in Table 4-12, the comparative analysis of the YOLO models reveals the significant influence of different training approaches, such as transfer learning, freezing the backbone, and freezing all layers, on the models' performance. Interestingly, while transfer learning shows an efficiency in training time, it presents no substantial difference in precision and recall compared to the basic model configurations. For example, both the YOLOv5m and YOLOv8m models with transfer learning achieve similar precision and recall rates as their basic counterparts, yet the training time is notably reduced (2.072 hours for YOLOv5m and 2.885 hours for YOLOv8m) compared to the basic YOLOv8m model (3.014 hours).

In the case of the YOLOv5m model, freezing the backbone layers results in a slight decrease in validation performance but maintains high precision and recall during training. This approach also reduces the training time to 1.499 hours, indicating efficiency in the process. However, a drastic reduction in model performance is observed when all layers are frozen. Precision and recall drop significantly in both training and validation phases, underscoring the detrimental impact of this method on the model's learning capability, despite a substantial decrease in training time to just 0.92 hours.

The mean Average Precision (mAP50 and mAP50-95) metrics align with these observations. Transfer learning models, while not outperforming the basic models in terms of accuracy, offer a time-efficient alternative.

In summary, this analysis underscores that transfer learning is a time-efficient alternative to the basic YOLO model configurations, maintaining similar levels of precision and recall. While freezing the backbone strikes a balance between reduced training time and performance, freezing all layers, despite its efficiency in reducing training duration, severely hampers the model's accuracy. Transfer learning, therefore, stands out as an effective approach in YOLO model training, optimizing the balance between training efficiency and maintaining high accuracy in detection tasks.

After comparing the precision, recall, and mean Average Precision (mAP50 and mAP50-95) of YOLO models employing various approaches, the subsequent section will proceed to analyze the models based on the box_loss recorded throughout the training and validation phases. Figure 4.10 presents the box_loss for both phases.

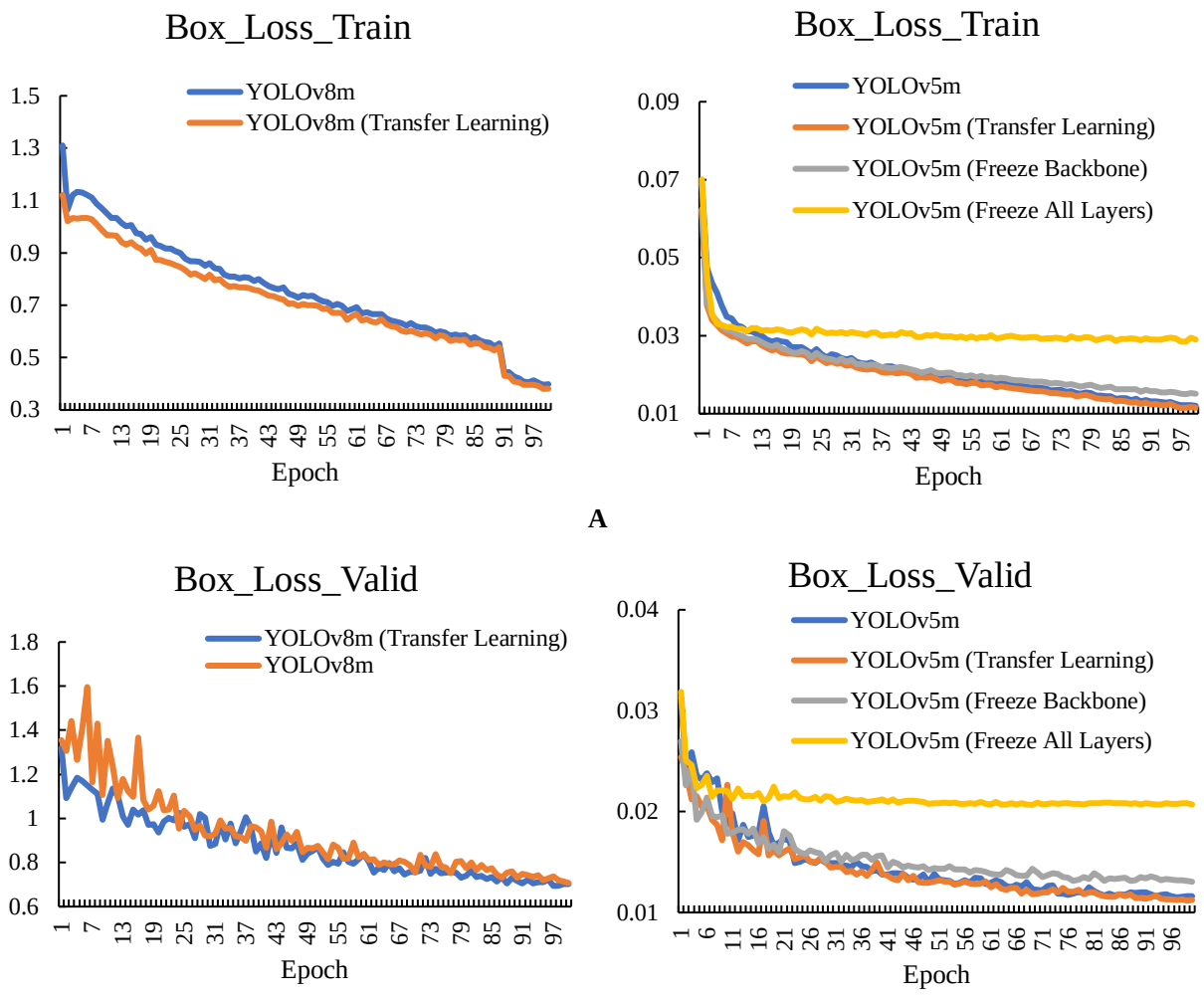



**Figure 4.10:** Evolution of Box loss across epochs for different YOLO models approaches during; A. Train and B. Validation phase.

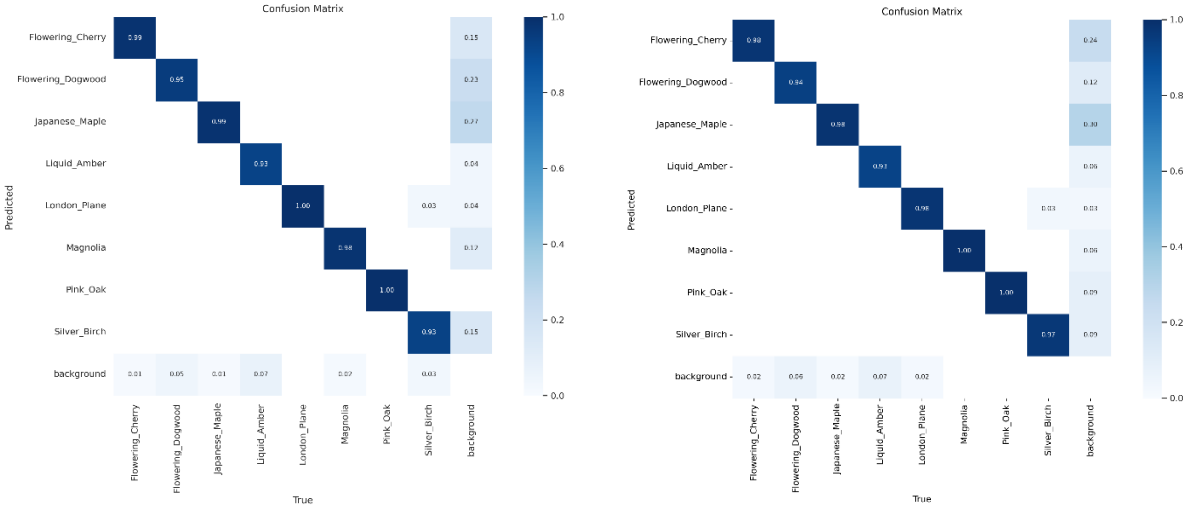
Analyzing the box loss data for various training approaches of the YOLOv5m and YOLOv8m models, it becomes clear that the training approaches significantly influence the models' ability to localize objects accurately (Figure 4.10). Transfer learning configurations for both models show a stable and low box loss throughout the training epochs, indicating an effective utilization of pre-learned features which aid in quicker convergence and potentially better generalization (Figure 4.10A).

When examining the impact of freezing layers, the YOLOv5m model with frozen backbone layers begins with a low box loss, but this does not decrease as substantially over time as with the models employing transfer learning (Figure 4.10A). This suggests that while initially beneficial, freezing the backbone may limit learning as the training advances, potentially leading to a plateau in performance improvements.

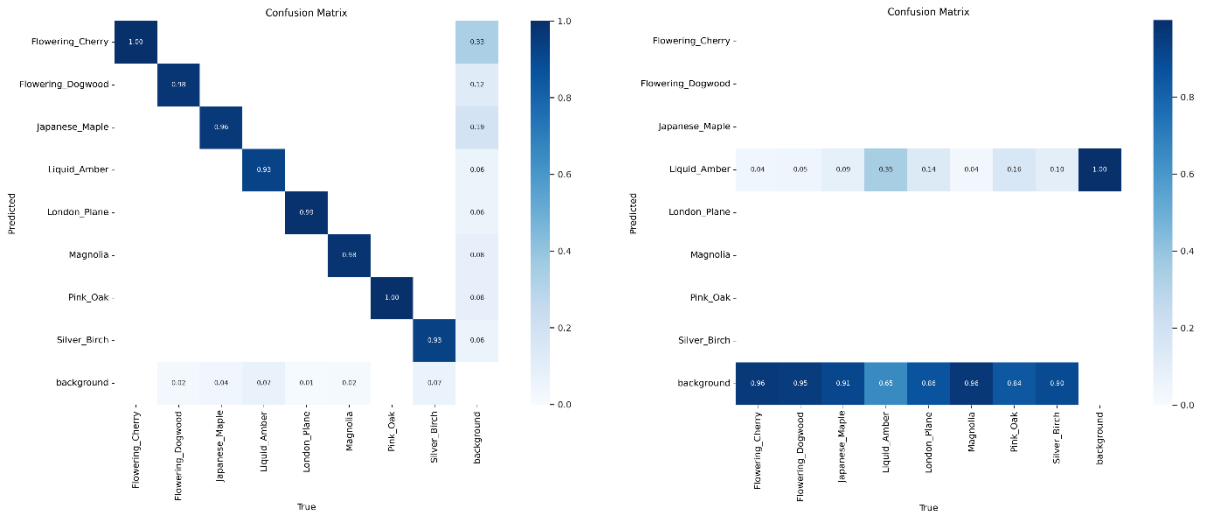
The most pronounced contrast is seen with the YOLOv5m model when all layers are frozen (Figure 4.10A). This approach results in a significantly higher box loss that does not show improvement over the training period, emphasizing the detrimental effect of restricting all layers on the model's capacity to learn. As can be seen in the validation phase charts, the validation phase box loss trends are consistent with the training phase (Figure 4.10B).

In comparison to the standard models, these specialized training configurations reveal that while transfer learning enhances performance, the freezing of layers requires a nuanced approach to prevent hindering the model's learning process. Overall, the training time and box loss trends from this analysis provide a deeper understanding of the trade-offs between rapid convergence and the long-term learning potential of the models, with transfer learning offering a compelling balance between the two.

Comparing the confusion matrix of the YOLOv8m model using transfer learning (Figure 4.11A) with the previously trained model (Figure 4.7B) for the identification of various urban tree species reveals that the YOLOv8m model with transfer learning shows significant precision in classifying tree species. Notably, it achieves perfect scores for London Plane and Pink Oak. The model also demonstrates improved accuracy in identifying Japanese Maple and Silver Birch compared to the standard model. Strikingly balancing performance across species, it effectively distinguishes them from the background, despite a slight increase in misclassification for a few categories. It can be concluded that the YOLOv8m model with transfer learning maintains or improves upon the precision rates of the standard model in many respects.




**A** **B**




**C** **D**

**Figure 4.11:** Comparative confusion matrices for urban tree species classification; A. YOLOv8m (transfer learning), B. YOLOv5m (transfer learning), C. YOLOv5m (freezing_backbone_layers) and D. YOLOv5m (freezing_all_layers).

The confusion matrices of the YOLOv5m model applied with different learning strategies show varied effectiveness in tree species identification. The transfer learning approach (**Figure 4.11**B) demonstrates a robust ability to maintain high precision, particularly for Magnolia and Pink Oak, matching the YOLOv5m model performance (Figure 4.7A). This technique shows a slight decline in precision for Flowering Dogwood and Japanese Maple but compensates with improved accuracy in distinguishing most species from the background, with only a minor increase in misclassification rates for the background. Moreover, freezing backbone layers (**Figure 4.11**C) offer a middle ground, retaining substantial precision across all species with perfect classification for Flowering Cherry and Pink Oak. Although there's a minimal decrease in precision for Japanese Maple and Silver Birch, this approach still yields a good overall performance. The all-layer freezing approach (**Figure 4.11**D), however, significantly underperforms, indicating its unsuitability for tasks requiring detailed feature discrimination.

In the comparative analysis of the precision-recall curve, the YOLOv8m model employing transfer learning (Figure 4.12A) was found to possess enhanced precision in tree species detection. An impressive overall mAP of 0.990 was recorded, with species such as Flowering Cherry and Pink Oak achieving near-perfect precision-recall scores. The model's performance was consistently high, with the lowest precision score for Silver Birch at 0.985 still reflecting a strong capability in accurate detection. The transfer learning approach was shown to marginally outperform the YOLOv8m model (Figure 4.8B), particularly in the precise identification of Liquid

Amber and Flowering Dogwood. This suggests that the transfer learning model is particularly well-suited for applications where slight improvements in detection accuracy are crucial.

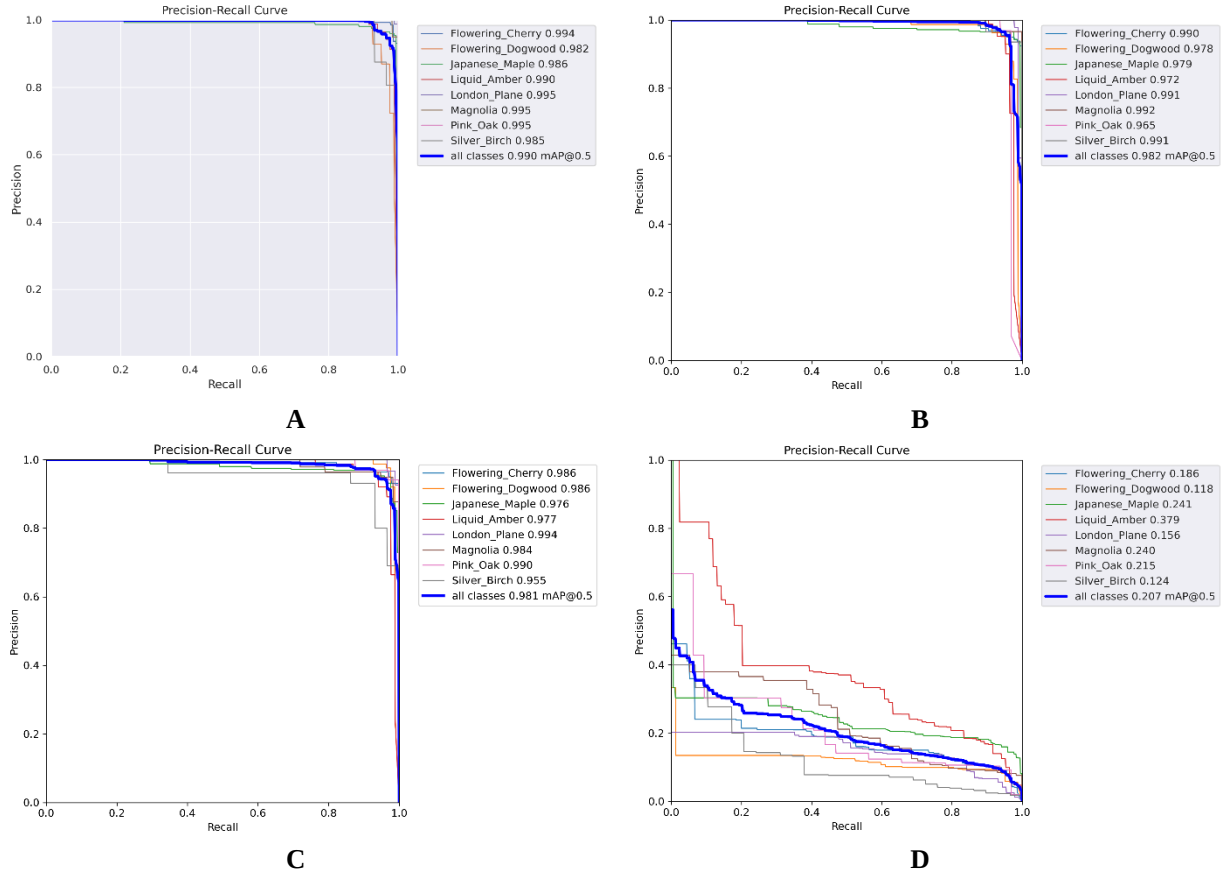



**Figure 4.12:** Precision-Recall curves;
A. YOLOv8m (transfer learning), B. YOLOv5m (transfer learning), C. YOLOv5m (freezing_backbone_layers) and D. YOLOv5m (freezing_all_layers).

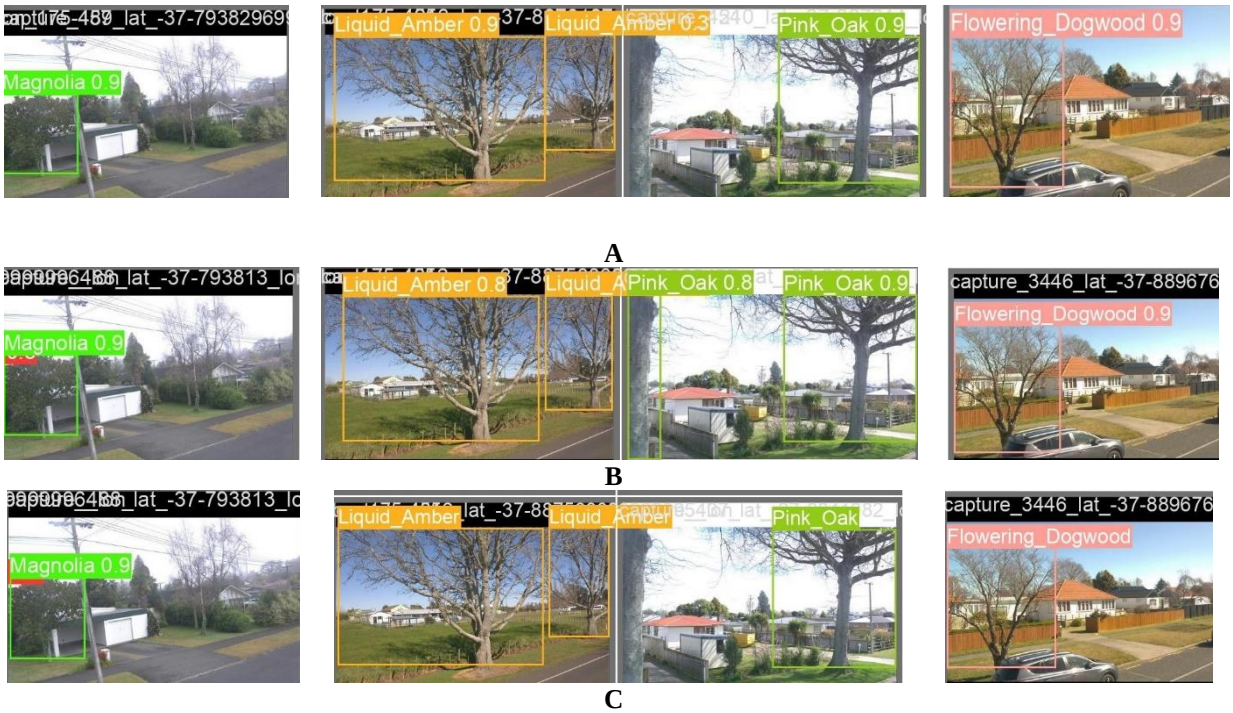
Upon the analysis of the precision-recall curves of the YOLOv5m models with different approaches, it is observed that the model trained using the transfer learning approach (Figure 4.12B) exhibits robust precision, evidenced by an overall mean Average Precision (mAP) of 0.982. A strong detection capability for species such as London Plane and Magnolia is indicated, as these are well-performed by the model. In contrast, the model that employs freezing of backbone layers (Figure 4.12C) presents a competitive performance with an overall mAP of 0.981. It excels in detecting London Plane, surpassing the transfer learning model slightly in this category. A noted decrease in precision is observed for Silver Birch, which indicates that freezing the backbone layers may have restricted the model's ability to learn more complex features specific to this species.

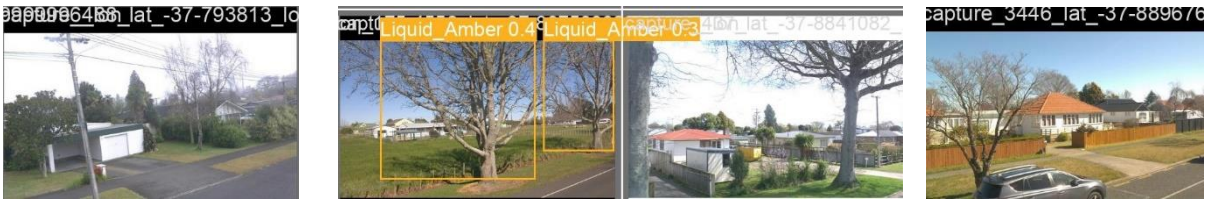
The model that freezes all layers (Figure 4.12D) shows a marked underperformance with precision scores plummeting across all species, culminating in an overall mAP of just 0.207. This stark contrast with the other models underscores the limitation of not allowing any part of the model to learn from the new data, which significantly hampers its detection capabilities.

According to the precision-recall curve, the transfer learning model is close to the yolov5m model (Figure 4.8A), matching or slightly exceeding the yolov5m model's precision in certain classes, thereby highlighting the efficacy of the transfer learning process. It suggests that fine-tuning a pre-trained model on new, specific data can yield a model that is nearly as effective as one trained from scratch.

The analysis suggests that the transfer learning and freezing backbone layers strategies are effective to a similar degree, closely rivaling the YOLOv5m model's performance. However, freezing all layers is decidedly ineffective for precise tree species detection tasks. The advantage of the transfer learning model lies in its ability to leverage existing knowledge and adapt to new data, making it a suitable choice for applications where accuracy is critical in object detection.

To assess the capability of differentially trained models, three images were selected from the validation output (Figure 4.13). The findings from the analysis are as follows:






**D**

**Figure 4.13:** Comparative visualization of urban tree species detection during the validation phase by A. YOLOv8m (transfer learning), B. YOLOv5m (transfer learning), C. YOLOv5m (freezing_backbone_layers) and D. YOLOv5m (freezing_all_layers).

- The YOLOv8m model, when trained using a transfer learning approach (Figure 4.13A), showed identification and classification capabilities for tree species that were similar to the YOLOv8m model (Figure 4.9B) evaluated in the preceding stage. However, it was noted that, despite certain improvements in the model's accuracy, it remains unable to recognize trees that are partially depicted or situated at the edges of the image.
- The performance of the YOLOv5m model with transfer learning (Figure 4.13B) was observed to be similar to that of the YOLOv5m model (Figure 4.9A).
- When the YOLOv5m model with frozen backbone (Figure 4.13C) layers was analyzed, it was found to be deficient in identifying trees at the image periphery. Nevertheless, this model retained a level of accuracy in classifying various tree species comparable to the YOLOv5m model.
- Lastly, the investigation of images processed by the YOLOv5m model with all layers frozen (Figure 4.13D) indicated a general inadequacy in species recognition. This was particularly evident in the model's overall weak performance in detecting various tree species, except for the Liquid Amber trees, which were identified with low confidence.

Following the comprehensive analysis of the training and validation phases, the developed models underwent an additional layer of evaluation using a distinct dataset. This step was crucial to gauge their real-world performance more accurately. The results of this extensive evaluation were meticulously verified through manual inspection, ensuring the reliability of the findings. The detailed outcomes, encompassing a range of key performance indicators such as True Positive (TP), True Negative (TN), False Positive (FP), False Negative (FN), Precision, Recall, and Accuracy, are systematically presented in Table 4-13. This table serves as an insightful resource, offering a nuanced perspective on the effectiveness and practical applicability of each model variant in diverse scenarios.

**Table 4-13**: Comparative analysis of performance metrics for YOLOv5m and YOLOv8m models across various approaches.

| Models | True Positive | True Negative | False Positive | False Negative | Precision | Recall | Accuracy |
|---|---|---|---|---|---|---|---|
| Yolov5m | 1307 | 1801 | 66 | 529 | 0.95 | 0.71 | 0.83 |
| YOLOv5m (Transfer Learning) | 1414 | 1793 | 74 | 428 | 0.95 | 0.77 | 0.86 |
| YOLOv5m (Freeze Backbone) | 1430 | 1593 | 274 | 396 | 0.84 | 0.79 | 0.82 |
| YOLOv5m (Freeze All Layers) | 35 | 1762 | 243 | 1668 | 0.13 | 0.02 | 0.5 |
| Yolov8m | 1085 | 1804 | 73 | 594 | 0.94 | 0.65 | 0.81 |
| YOLOv8m (Transfer Learning) | 1174 | 1783 | 638 | 82 | 0.64 | 0.93 | 0.81 |

In the YOLOv5 series, the models show different levels of performance. The original Yolov5m has a good balance between precision and recall, with a precision of 0.95 and recall of 0.71, leading to an accuracy of 0.83. However, the YOLOv5m with Transfer Learning performs better in this group. This model keeps the same precision but improves recall to 0.77 and increases overall accuracy to 0.86, making it the top choice among the YOLOv5 models. The YOLOv5m with Freeze Backbone has a slightly lower precision of 0.84 but a higher recall of 0.79, though with a small drop in overall accuracy. In contrast, the YOLOv5m with all layers frozen performs much worse, with very low precision and recall, showing it's not as effective.

On the other hand, the YOLOv8 models work differently. The standard Yolov8m achieves a high precision of 0.94 but a lower recall of 0.65, with an overall accuracy similar to the YOLOv5 models. The YOLOv8m with Transfer Learning focuses more on recall, reaching a high rate of 0.93, but with lower precision of 0.64. The accuracy is the same as the standard Yolov8m model. This makes the YOLOv8m with Transfer Learning better for situations where it's important to catch as many true positives as possible, even if it means a lower precision

Similar to previous analyses, by comparing the models developed in this research with other studies in the similar field of tree species identification and classification using comparable methods, a more insightful analysis of the created models can be conducted. Therefore, Table 4-14 references a number of these studies. It should be noted that most research has trained their models using transfer learning approaches, and only a few studies have focused on tree species identification using different layer freezing techniques, as per the evaluations conducted at the time of writing this section of the thesis.

**Table 4-14:** Comparative analysis of deep learning models for tree species detection using different approaches.

| Model | Approach | Number of species | Precision | Recall | Reference |
|---|---|---|---|---|---|
| ResNet-50 | Transfer learning | 7 | 0.80 | 0.80 | [348] |
| YOLOv5 | Transfer learning | 8 | 0.66 | 0.60 | [349] |
| DeepForest | Transfer learning | 1 | 0.86 | 0.87 | [350] |
| YOLOv5 | | | 0.82 | 0.89 | |
| DeepForest | Transfer learning | Tree | 0.66 | 0.5 | [351] |
| RetinaNet | Transfer learning | 1 | 0.67 | 0.78 | [352] |
| VGG-19(first 4 blocks froze) | Freezing layers | 6 | 0.63 | 0.63 | [353] |
| VGG-19(first 5 blocks froze) | | | 0.6 | 0.6 | |
| VGG-19(all blocks froze) | | | 0.58 | 0.59 | |
| YOLOv5m (backbone layers froze) | Freezing layers | 8 | 0.96 | 0.96 | Current research |
| YOLOv5m (all layers froze) | | | 0.28 | 0.55 | |
| YOLOv5m | Transfer learning | 8 | 0.96 | 0.96 | Current research |
| YOLOv8m | | | 0.97 | 0.96 | |

In terms of the transfer learning approach, the ResNet-50 model displays a balanced precision and recall of 0.80, indicating reliable performance for detecting seven species. YOLOv5 and DeepForest models, also utilizing transfer learning, exhibit varying degrees of effectiveness. The YOLOv5 model achieves a precision of 0.66 and recall of 0.60 for eight species, while DeepForest shows higher precision and recall for a single species, suggesting its suitability for more specialized tasks. RetinaNet, using transfer learning, records moderate precision and recall values for one species. In the context of current research, the YOLOv5m and YOLOv8m models demonstrate high precision and recall, highlighting the strength of transfer learning in enhancing model performance.

In contrast, when evaluating the models employing the freezing layers approach, different patterns emerge. The VGG-19 model with various configurations of freezing layers shows a noticeable decrease in precision and recall as more layers are frozen, underlining the importance of layer adaptability in learning complex features. The YOLOv5m model, when its backbone layers are frozen, maintains an impressive precision and recall of 0.96. However, freezing all its layers results in a substantial decline in performance, with precision notably dropping to 0.28.

This comparative analysis provides insights into the efficacy of different training strategies in tree species detection. While transfer learning proves to be a robust approach across various models, the strategy of freezing layers, particularly freezing all layers, seems less effective for this

specific task. The findings underscore the significance of choosing an appropriate training method that aligns with the specific objectives and requirements of the study.

After examining various models with different training approaches for the identification and classification of diverse tree species, the YOLOv5m and YOLOv8m models were utilized, employing transfer learning specifically for the recognition and categorization of specific trees. The subsequent section will delve into the results obtained from this selection, providing a detailed analysis.

## 4.6 Specific trees detection with variable bounding box sizes

Having explored the models' capabilities in identifying various tree species, this study now turns its attention to detecting specific individual trees using the YOLOv5m and YOLOv8m models trained with the transfer learning approach. This section delves into evaluating these models, particularly when using two different bounding box sizes. The first is the standard bounding box size, typically used in object detection tasks, and the second is a larger bounding box size. The findings from this evaluation, detailed in Table 4-15, provide insights into how the size of the bounding box influences the accuracy and reliability of the models in specific tree detection.

**Table 4-15:** Comparative analysis of YOLOv5m and YOLOv8m models with different bounding box sizes.

| Model | Precision | | Recall | | mAP50 | | mAP50-95 | | Training Time (Hour) |
|---|---|---|---|---|---|---|---|---|---|
| | Train | Valid | Train | Valid | Train | Valid | Train | Valid | |
| YOLOV5m_Bigger_size | 0.95 | 0.91 | 0.99 | 0.96 | 0.99 | 0.95 | 0.81 | 0.73 | 4.83 |
| YOLOV5m_Normal_size | 0.96 | 0.91 | 0.98 | 0.95 | 0.99 | 0.95 | 0.80 | 0.72 | 4.14 |
| YOLOV8m_Bigger_size | 0.96 | 0.95 | 0.99 | 0.98 | 0.99 | 0.98 | 0.84 | 0.79 | 5.925 |
| YOLOV8m_Normal_size | 0.96 | 0.94 | 0.98 | 0.97 | 0.99 | 0.98 | 0.81 | 0.77 | 5.308 |

The analysis of the YOLOv5m and YOLOv8m models, as presented in Table 4-15, offers detailed insights into their effectiveness in detecting specific trees with varying bounding box sizes. The YOLOv5m model, both in its bigger and normal size configurations, maintains high precision during the training phase, with scores of approximately 0.95 and 0.96, respectively. This level of precision remains fairly stable in the validation phase for both sizes, suggesting consistent predictive accuracy. The recall rates are notable as well, with the bigger size achieving 0.99 in training and 0.96 in validation, and the normal size slightly lower in both phases.

In terms of mean Average Precision (mAP), the YOLOv5m models perform strongly. The mAP50 scores are close to 0.99 in training, with a small decrease in validation. The mAP50-95 scores are particularly strong for the bigger size model, reaching 0.81 in training and 0.73 in validation, indicating effective detection across various IoU thresholds.

The YOLOv8m models, in both their bigger and normal bounding box sizes, display precision and recall rates that are marginally higher than those of the YOLOv5m models in both training and validation. The bigger size YOLOv8m model shows a training precision of 0.96 and a validation precision of 0.95. Its recall and mAP scores are also high, reinforcing its efficiency in detecting specific trees.

As far as training time is concerned, both YOLOv5m and YOLOv8m models require significant time. This pattern is particularly evident in the YOLOv8m model with a bigger bounding box size, which demands the most extensive training duration, clocking in at approximately 5.925 hours. This observation underscores the inherent trade-off between model complexity and the breadth of classification tasks against the time required for training. Notably, a trend observed across the models discussed in previous sections is that as the number of classes increases in both models, there is a corresponding increase in training time.

Overall, both the YOLOv5m and YOLOv8m series, particularly the models with bigger bounding box sizes, demonstrate effective tree detection capabilities. The YOLOv8m models edge out slightly in terms of precision and recall during validation, suggesting their higher suitability for accuracy-focused tasks. However, their longer training times reflect the extensive computational resources needed for such detailed analysis.

The subsequent analysis, the focus will be shifted to the box loss graphs of the YOLOv5m and YOLOv8m models. These graphs (Figure 4.14) will be evaluated across four specific scenarios: training and validation phases with bigger-size bounding boxes (bigger-size_train and bigger-size_valid) and with normal-size bounding boxes (normal-size_train and normal-size_valid).

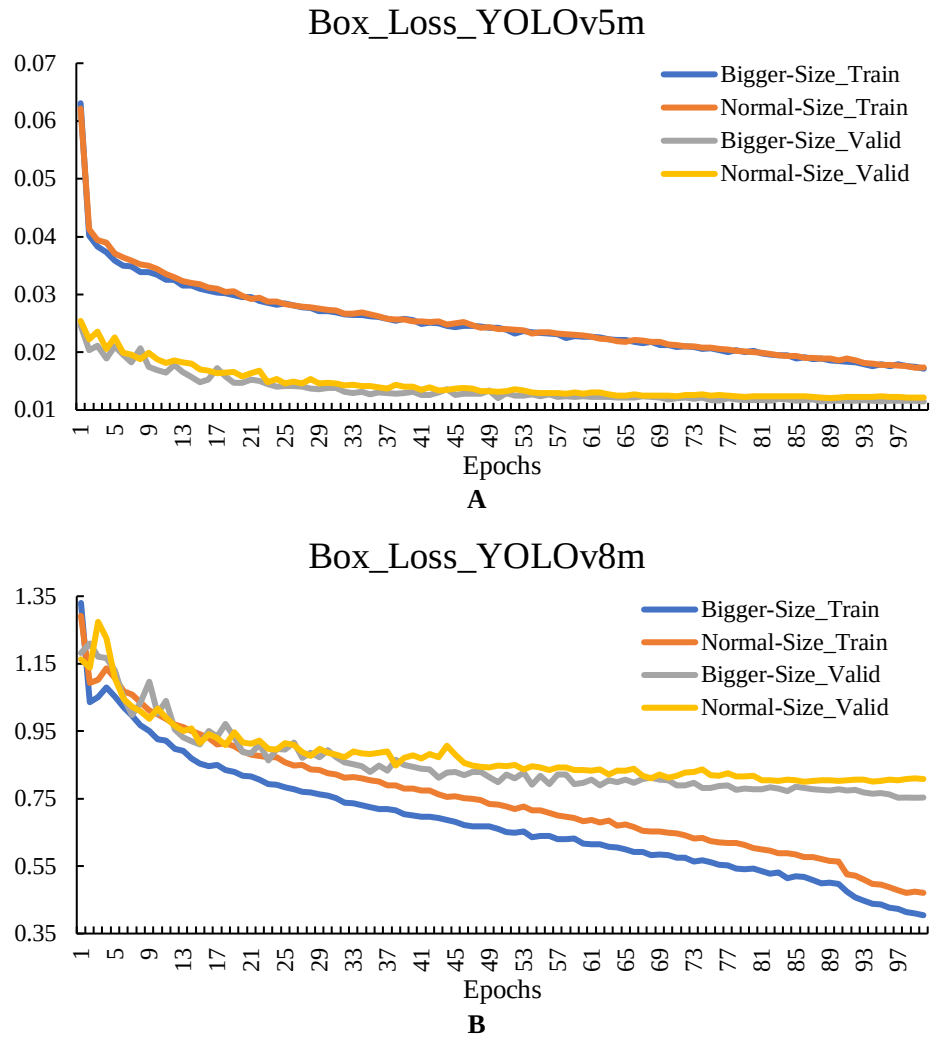


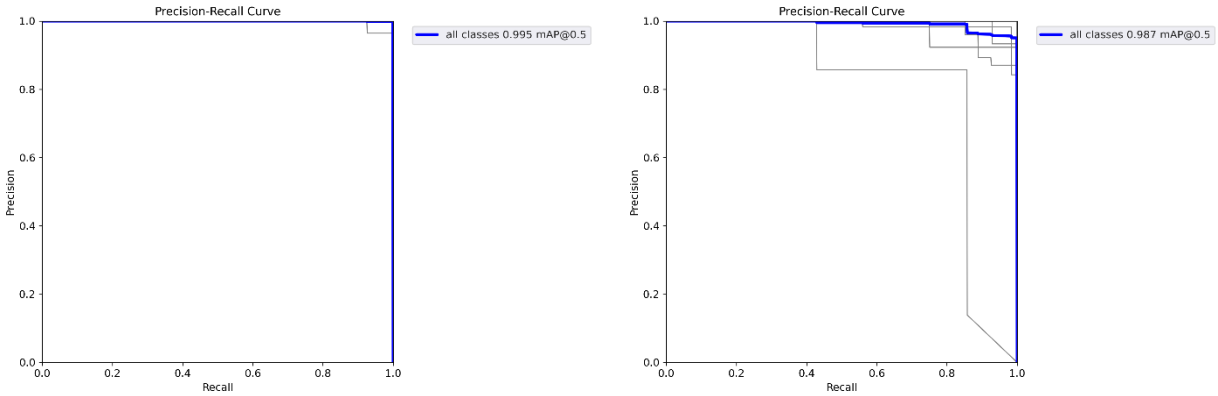

**Figure 4.14:** Evolution of Box loss across epochs for different bounding box sizes during train and validation phase for; A. YOLOv5m and B. YOLOv8m.

In the YOLOv5m trained model with different bounding box sizes (Figure 4.14A), initial observations indicate a higher box loss when using a bigger bounding box size. This suggests a period of adjustment as the model learns to interpret the larger scope of the bounding boxes. However, this discrepancy in box loss diminishes over time, as seen by the converging loss values between the models trained with normal and larger bounding boxes. The convergence evident in Figure 2.10A implies that the YOLOv5m is capable of effectively adapting to various bounding box sizes through the course of its training. In the validation phase, the box loss trends align with those of the training phase. Both the normal-size and bigger-size bounding box configurations display a consistent decline in box loss across epochs, as shown in Figure 2.10B. This consistency suggests that the bounding box size does not significantly influence the model's ability to generalize in the validation phase.

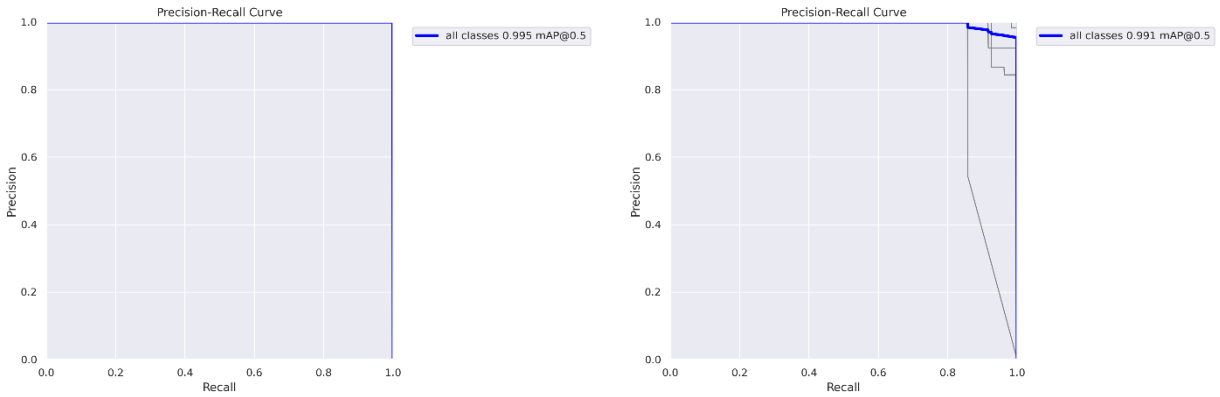
Similarly, the YOLOv8m model (Figure 4.14B) exhibits an initial higher box loss with the larger bounding box size, akin to the YOLOv5m model's initial performance. As training progresses, this loss decreases steadily, reflecting the model's learning and adjustment to the bounding box dimensions. The loss for both the normal and larger bounding box sizes shows a convergence pattern, as seen in the corresponding figure for the YOLOv8m model. During the validation phase, the YOLOv8m also demonstrates that the box loss trends for the larger and normal bounding box sizes are consistent with the training phase, suggesting effective learning and generalization capabilities of the model.

Comparing the box loss data of YOLOv5m and YOLOv8m, both models show an initial adjustment phase with higher box losses for larger bounding box sizes. In subsequent training epochs, the models learn and adjust, resulting in a convergence of box loss between the different bounding box sizes. This pattern holds true across both the training and validation phases, indicating that neither model's generalization ability is significantly impacted by the initial bounding box size. The consistent decline in box loss across epochs for both models underscore the YOLO architecture's robustness and adaptability to bounding box size variations in object localization tasks.

Analyzing the precision and recall curves, as well as the mean Average Precision (mAP) at an Intersection over Union (IoU) threshold of 0.5, provides a deeper understanding of the models' performance across all 32 classes (Figure 4.15).




**A** **B**




**C** **D**

**Figure 4.15:** Precision-Recall curves;
A. YOLOv5m (bigger size), B. YOLOv5m (normal size), C. YOLOv8m (bigger size), and C. YOLOv8m (normal size).

For the YOLOv5m model trained with larger bounding boxes (Figure 4.15A), the mAP@0.5 is 0.995, indicating an exceptionally high level of precision and recall. This high mAP suggests that the model is almost perfectly distinguishing between the specific trees classes and accurately localizing them with the larger boxes. When the YOLOv5m model is trained with normal-sized bounding boxes (Figure 4.15B), there is a slight drop in mAP to 0.987. While this is still a high score, it is marginally lower than the mAP achieved with larger boxes. This could indicate that while the model is very accurate, there might be a slight edge in performance when using larger bounding boxes.

The YOLOv8m model with larger bounding boxes (Figure 4.15C) also achieves a 0.995 mAP@0.5, mirroring the impressive performance of the YOLOv5m model under similar conditions. The consistency in mAP between the two models suggests that the advancements in YOLOv8 have maintained high detection accuracy for larger bounding box sizes. For the YOLOv8m model trained with normal-sized bounding boxes (Figure 4.15D), the mAP@0.5 is 0.991. This is slightly lower than the mAP for larger bounding boxes but still represents a high level of performance. The YOLOv8m model appears to be slightly less sensitive to changes in bounding box size than the YOLOv5m model, as indicated by the smaller difference in mAP between the two bounding box sizes.

So, both the YOLOv5m and YOLOv8m models demonstrate high precision and recall across all 32 classes, with the bigger bounding boxes providing a marginal improvement in mAP@0.5. The YOLOv8m shows a strong performance that is less affected by bounding box size changes, maintaining a high mAP close to that of its larger box counterpart.

After a comprehensive analysis during training and validation phases, the models underwent further evaluation with a unique dataset, aimed at assessing their performance in real-world conditions. The outcomes of this evaluation are meticulously detailed in Table 4-16, offering a clear perspective on the effectiveness and practical applicability of each model across different scenarios.

**Table 4-16**: Comparative analysis of performance metrics for YOLOv5m and YOLOv8m models with varying bounding box sizes.

| Models | True Positive | True Negative | False Positive | False Negative | Precision | Recall | Accuracy |
|---|---|---|---|---|---|---|---|
| YOLOV5m_Bigger_size | 417 | 3432 | 47 | 50 | 0.90 | 0.89 | 0.97 |
| YOLOV5m_Normal_size | 403 | 3467 | 37 | 39 | 0.92 | 0.91 | 0.98 |
| YOLOV8m_Bigger_size | 294 | 3533 | 7 | 112 | 0.97 | 0.72 | 0.97 |
| YOLOV8m_Normal_size | 413 | 3322 | 194 | 17 | 0.68 | 0.96 | 0.95 |

For the YOLOv5m models, both variants (bigger size and normal size) demonstrate strong performance. The YOLOv5m with bigger bounding boxes shows a slightly lower precision (0.8987) compared to the normal-sized variant (0.9159), which suggests that while it is highly accurate in positive predictions, the normal-sized model is slightly more precise. However, the recall rate is almost identical for both, indicating a consistent ability to detect true positives. The overall accuracy of both YOLOv5m models is high, with the normal-sized model slightly outperforming the bigger-sized one (0.9807 vs. 0.9754).

In contrast, the YOLOv8m models exhibit a more varied performance. The bigger bounding box variant of YOLOv8m shows an exceptionally high precision (0.9767) but a lower recall (0.7241), suggesting it is very accurate in its predictions but may miss more positive cases compared to the YOLOv5m models. The normal-sized YOLOv8m, on the other hand, has a significantly lower precision (0.6804) but a very high recall (0.9605), indicating it detects most positive cases but also has a higher rate of false positives.

In Figure 4.16, the outputs from various models are depicted, each tailored with different bounding box sizes. The models include YOLOV5m with a bigger size bounding box (Figure 4.16A), YOLOV5m with a normal size bounding box (Figure 4.16B), YOLOV8m with a bigger size bounding box (Figure 4.16C), and YOLOV8m with a normal size bounding box (Figure 4.16D). These maps systematically illustrate the spatial distribution of True Positives (TP), False Negatives (FN), False Positives (FP), and the correct locations of specific trees. Each map distinctly delineates the regions where the models have successfully identified trees in captured

images (TP), alongside the areas where trees exist yet were not detected (FN), and the locations erroneously classified as trees (FP). This visual approach provides an integral understanding of the practical implications of the computed precision, recall, and accuracy metrics for each model variant, thus offering an expansive perspective on their operational effectiveness in real-world scenarios.


**A**


**B**


**C**


**D**

**Figure 4.16:** Spatial visualization of models' outputs with different bounding box sizes for detecting specific trees in the Hamilton East area, Hamilton City; A. YOLOV5m_Bigger_size, B. YOLOV5m_Normal_size, C. YOLOV8m_Bigger_size, and D. YOLOV8m_Normal_size.

Considering the performance metrics and visual outputs of the YOLOv5m and YOLOv8m models, the selection of the optimal model is contingent on the specific application requirements. For scenarios where precision is paramount and the cost of false positives is significant, the YOLOv8m model with a larger bounding box configuration, exhibiting exceptionally high precision, is the preferable choice. This model ensures high accuracy in positive predictions, as evidenced by its precision metric of 0.9767.

On the other hand, applications that cannot afford to miss true positives would benefit more from the YOLOv5m models. The normal-sized variant of YOLOv5m, in particular, stands out due to its balanced blend of precision and recall, combined with the highest overall accuracy (0.9807). This model's slightly more precise nature (0.9159 precision) and its nearly identical recall rate compared to its bigger-sized counterpart make it an ideal choice for scenarios demanding a well-rounded performance.

The visualization in Figure 4.16 reinforces these findings. It showcases the performance of the YOLOv5m and YOLOv8m models with different bounding box sizes, illustrating the real-world impact of their precision, recall, and accuracy. The maps in this figure provide a clear depiction of each model's effectiveness in identifying specific trees, highlighting the strengths and weaknesses of each configuration in practical applications. This comprehensive analysis, encompassing both quantitative metrics and spatial visualizations, offers a holistic view of the models' capabilities, guiding the selection of the most suitable model based on the intended use case.

Upon examination of the extant literature in this domain, it has been observed that no prior research has specifically concentrated on the identification of individual trees. This observation underscores a distinctive aspect of the current study, which has been dedicated to exploring this particular area. Through meticulous analysis and comprehensive investigation, this research has contributed to a deeper understanding of the complexities involved in tree identification. The insights gained from this study help the existing body of knowledge and provide a clearer perspective on the specific challenges associated with recognizing individual trees.

## 4.7 Summery

- The Fast Fourier Transform (FFT) technique was ultimately selected for filtering blurred images in the dataset, owing to its ability to accurately assess high-frequency content and effectively detect various types of blurs.
- The final selection for removing repetitive images was the NCC technique, due to its robustness against brightness and contrast changes.
- A privacy protection filter was applied to the dataset after removing non-vegetation, blurred, and repetitive images. This filter specifically targets and blurs objects of high privacy importance, such as faces and vehicle license plates, rather than removing the images.
- For selecting an annotation tool, Roboflow.com was chosen for its comprehensive features and collaborative efficiency.
- The selection of YOLOv5m and YOLOv8m models for tree detection is based on their superior accuracy and consistent performance, with precision rates of 0.96 and 0.92, and recall rates of 0.92 and 0.91 respectively, highlighting their reliability in urban vegetation detection and species classification.
- Among the various training strategies evaluated, transfer learning proved to be a highly efficient and accurate approach for both the YOLOv5m and YOLOv8m models. It successfully maintained superior model performance and optimized training duration, distinguishing itself from other methods such as freezing backbone layers or freezing all layers in the YOLOv5m model.

- The YOLOv8m with a bigger bounding box is ideal for precision-focused scenarios in specific tree detection and classification, ensuring high accuracy and minimizing false positives.
- The normal-sized YOLOv5m excels in balancing precision and recall, offering the best overall accuracy for reliably detecting and classifying various specific tree without missing true positives.

# 5 Conclusion

In the culmination of this thesis, attention is turned to the consolidation of the research journey and the insights that have been accrued. The endeavor to develop a sophisticated computer vision system for urban vegetation detection, employing the YOLOv5 and YOLOv8 models, has been marked by both challenges and significant achievements. The subsequent sections, titled "Comprehensive Overview of Research Outcomes" and "Future Research Pathways and Opportunities," are dedicated to encapsulating the essence of this exploration. In "Comprehensive Overview of Research Outcomes," a detailed recounting of the key discoveries and their broader implications is presented. "Future Research Pathways and Opportunities," on the other hand, is focused on outlining prospective avenues for further research, building upon the foundational work established in this study.

## 5.1 Comprehensive overview of research outcomes

In this thesis, the development and deployment of an advanced computer vision system utilizing YOLOv5 and YOLOv8 models was undertaken to address the critical need for effective urban vegetation detection in three different cities in New Zealand: Hamilton, Cambridge, and Auckland. This study, motivated by the environmental challenges posed by modern urbanization, focused specifically on identifying various tree species and individual trees through RGB images.

Key findings of this research include substantial contributions to the field through meticulous image preprocessing techniques, such as Fast Fourier Transform and Normalized Cross-Correlation. These techniques not only removed blur and repetitive images from the dataset, which were big challenges but also significantly enhanced its quality. Additionally, privacy information in images was detected and blurred using a privacy information filter. The YOLOv5m and YOLOv8m models demonstrated their superiority, evidenced by high precision and recall rates, underscoring their reliability in urban vegetation detection and species classification. The adoption

of transfer learning as a training strategy proved efficient, maintaining superior model performance while optimizing training duration. The study found that the YOLOv8m model, with a larger bounding box, is especially effective for precision-focused scenarios in specific tree detection and classification, ensuring high accuracy with minimal false positives. Meanwhile, the normal-sized YOLOv5m model excels in balancing precision and recall, offering overall accuracy for reliable detection and classification of various specific trees, effectively minimizing missed true positives.

This research's significance lies in its innovative approach to overcoming the challenges of traditional methods for monitoring urban green spaces. By leveraging street-level imagery and advanced computer vision models, it successfully addressed the limitations of satellite and UAV-based approaches. This methodology provides a detailed and accurate view of urban vegetation, playing a crucial role in urban planning and environmental conservation. The comprehensive route planning for data collection, strategic selection of tree species, and sophisticated application of image processing and augmentation techniques were pivotal in achieving the study's objectives. The customization and training of the models, enriched by diverse augmentation techniques and advanced annotation tools, were instrumental in refining the system's ability to detect and classify urban vegetation effectively.

In conclusion, this thesis has not only advanced our understanding of urban vegetation monitoring using computer vision but also highlighted the potential of AI in enhancing environmental conservation efforts. The successful application of YOLOv5 and YOLOv8 models marks a significant step forward in the efficient and accurate management of urban vegetation. Additionally, the models are adaptable for use in other New Zealand cities with access to similar urban vegetation datasheets, crucial for evaluating model accuracy and enhancing precision in future models, contributing valuable insights and tools to the field of urban environmental management.

## 5.2 Future research pathways and opportunities

In anticipation of the expansion and progression of this field of study, this part of the conclusion chapter is intended to delineate a spectrum of potential avenues for further investigation. Stemming from the foundational efforts encapsulated in this thesis, a series of suggestions are presented, aimed at augmenting the capabilities and broadening the applications of computer vision systems in urban vegetation monitoring. These suggestions, ranging from

technological integrations and enhancements in model precision to wider-scale applications and considerations of ethical implications, are outlined. It is envisaged that these points will serve as a comprehensive roadmap, guiding future research endeavors to enrich our comprehension and elevate the efficacy of methods in the realm of urban vegetation detection and analysis. These suggestions are as follows:

**Integration with additional imaging technologies**: Future studies could explore the integration of RGB imaging with hyperspectral or multispectral imaging techniques. This could provide more comprehensive data, enabling the detection of a wider range of vegetation characteristics and health indicators.

**Expansion to different urban environments**: Given that a significant portion of urban vegetation cover is located in parks, green spaces, and protected urban areas, and considering that this research was conducted with a system not designed for capturing images in these areas, future research could focus on modifying the design and construction of the imaging system. These changes would enable access to and imaging of these critical urban green spaces, thereby expanding the scope and applicability of the study to a broader range of urban environmental settings.

**Enhancing species identification accuracy**: Further research could focus on refining the models to improve the accuracy of species identification, especially for species with similar appearances. Advanced machine learning techniques could be employed to fine-tune the models.

**Real-time urban vegetation monitoring**: Developing a real-time monitoring system that uses the trained models could significantly benefit urban planning and environmental monitoring. This system could provide instant data on urban green spaces, aiding in timely decision-making and interventions.

**Automated health assessment of vegetation**: Extending the models to not only identify species but also assess the health of individual trees and vegetation could be a valuable area of exploration. This could include detecting signs of disease, drought stress, or pest infestations.

**Combining with urban planning software**: Integrating the computer vision system with urban planning and GIS software could create a powerful tool for urban developers and ecologists, helping in effective planning and management of green spaces.

**Scalability and efficiency improvements**: Research could be directed towards improving the scalability and efficiency of the models, enabling them to process large datasets more rapidly and effectively.

**Developing a user-friendly mobile app**: Future research could focus on creating a mobile application that is user-friendly and accessible to the general public, including urban planners and environmentalists. This app would enable users to detect and classify trees, providing detailed information about each tree directly on their mobile devices, thus facilitating community involvement in environmental monitoring.

**Creating a comprehensive software platform for urban vegetation**: Future work could involve the development of a sophisticated software platform. This platform would be dedicated to processing and analyzing urban vegetation datasets, including features like image preprocessing, analysis with trained models, and visualizing located trees on maps. It would also categorize and store images in distinct folders for each tree species, enhancing data management and analysis efficiency for researchers and urban planners.

# 6 References


[1] K. J. Wallace and B. D. Clarkson, "Urban forest restoration ecology: a review from Hamilton, New Zealand," *Journal of the Royal Society of New Zealand,* vol. 49, no. 3, pp. 347-369, 2019.

[2] I. Jo *et al.*, "Ecological importance of the Myrtaceae in New Zealand's natural forests," *Journal of Vegetation Science,* vol. 33, no. 1, p. e13106, 2022.

[3] J. Jang and S.-Y. Woo, "Native Trees as a Provider of Vital Urban Ecosystem Services in Urbanizing New Zealand: Status Quo, Challenges and Prospects," *Land,* vol. 11, no. 1, p. 92, 2022.

[4] T. Endreny, R. Santagata, A. Perna, C. De Stefano, R. F. Rallo, and S. Ulgiati, "Implementing and managing urban forests: A much needed conservation strategy to increase ecosystem services and urban wellbeing," *Ecological Modelling,* vol. 360, pp. 328-335, 2017.

[5] M. M. Nizamani, A. Harris, X. L. Cheng, Z. X. Zhu, C. Y. Jim, and H. F. Wang, "Positive relationships among aboveground biomass, tree species diversity, and urban greening management in tropical coastal city of Haikou," *Ecology and Evolution,* vol. 11, no. 17, pp. 12204-12219, 2021.

[6] G. Brown, M. F. Schebella, and D. Weber, "Using participatory GIS to measure physical activity and urban park benefits," *Landscape and urban planning,* vol. 121, pp. 34-44, 2014.

[7] P. Liu, M. Liu, T. Xia, Y. Wang, and P. Guo, "The relationship between landscape metrics and facial expressions in 18 urban forest parks of Northern China," *Forests,* vol. 12, no. 12, p. 1619, 2021.

[8] C. G. Threlfall, N. S. Williams, A. K. Hahs, and S. J. Livesley, "Approaches to urban vegetation management and the impacts on urban bird and bat assemblages," *Landscape and Urban Planning,* vol. 153, pp. 28-39, 2016.

[9] K. Berthon, F. Thomas, and S. Bekessy, "The role of 'nativeness' in urban greening to support animal biodiversity," *Landscape and Urban Planning,* vol. 205, p. 103959, 2021.

[10] M. Rahman, D. Armson, and A. Ennos, "Effect of urbanization and climate change in the rooting zone on the growth and physiology of Pyrus calleryana," *Urban Forestry & Urban Greening,* vol. 13, no. 2, pp. 325-335, 2014.

[11] M. Dyrmann, A. K. Mortensen, L. Linneberg, T. T. Høye, and K. Bjerge, "Camera assisted roadside monitoring for invasive alien plant species using deep learning," *Sensors,* vol. 21, no. 18, p. 6126, 2021.

[12] A. Duarte, N. Borralho, P. Cabral, and M. Caetano, "Recent advances in forest insect pests and diseases monitoring using UAV-based data: A systematic review," *Forests,* vol. 13, no. 6, p. 911, 2022.

[13] K. Johansen, M. Sohlbach, B. Sullivan, S. Stringer, D. Peasley, and S. Phinn, "Mapping banana plants from high spatial resolution orthophotos to facilitate plant health assessment," *Remote Sensing,* vol. 6, no. 9, pp. 8261-8286, 2014.

[14] R. Heim, I. J. Wright, A. Allen, I. Geedicke, and J. Oldeland, "Developing a spectral disease index for myrtle rust (Austropuccinia psidii)," *Plant Pathology,* vol. 68, no. 4, pp. 738-745, 2019.

[15] B. Steward, J. Gai, and L. Tang, "The use of agricultural robots in weed management and control," in *Robotics and automation for improving agriculture*: Burleigh Dodds Science Publishing, 2019, pp. 161-186.

[16] R. Mathieu, J. Aryal, and A. K. Chong, "Object-based classification of Ikonos imagery for mapping large-scale vegetation communities in urban areas," *Sensors,* vol. 7, no. 11, pp. 2860-2880, 2007.

[17] A. A. d. Santos *et al.*, "Assessment of CNN-based methods for individual tree detection on images captured by RGB cameras attached to UAVs," *Sensors,* vol. 19, no. 16, p. 3595, 2019.

[18] M. Gašparović, D. Dobrinić, and D. Medak, "Urban vegetation detection based on the land-cover classification of planetscope, rapideye and worldview-2 satellite imagery," in *Proceedings of the 18th International Multidisciplinary Scientific Geo-Conference SGEM2018, Albena, Bulgaria*, 2018, pp. 249-256.

[19] S. Gadal, W. Ouerghemmi, R. Barlatier, and G. Mozgeris, "Critical analysis of urban vegetation mapping by satellite multispectral and airborne hyperspectral imagery," in *5th International Conference on Geographical Information Systems Theory, Applications and Management*, 2019, vol. 1, no. ISBN 978-989-758-371-1, pp. 97-104.

[20] M. Gašparović, D. Medak, I. Pilaš, L. Jurjević, and I. Balenović, "Fusion of sentinel-2 and planetscope imagery for vegetation detection and monitoring," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 42, pp. 155-160, 2018.

[21] H. Pourazar, F. Samadzadegan, and F. Dadrass Javan, "Aerial multispectral imagery for plant disease detection: Radiometric calibration necessity assessment," *European Journal of Remote Sensing*, vol. 52, no. sup3, pp. 17-31, 2019.

[22] M. G. Selvaraj *et al.*, "Detection of banana plants and their major diseases through aerial images and machine learning methods: A case study in DR Congo and Republic of Benin," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 169, pp. 110-124, 2020.

[23] D. Patón, "Normalized difference vegetation index determination in urban areas by full-spectrum photography," *Ecologies*, vol. 1, no. 1, pp. 22-35, 2020.

[24] K. Xia, H. Wang, Y. Yang, X. Du, and H. Feng, "Automatic detection and parameter estimation of Ginkgo biloba in urban environment based on RGB images," *Journal of Sensors*, vol. 2021, pp. 1-12, 2021.

[25] S. Amin and F. Galasso, "Geometric proposals for faster R-CNN," in *2017 14th IEEE International Conference on Advanced Video and Signal Based Surveillance (AVSS)*, 2017, pp. 1-6.

[26] M. Maity, S. Banerjee, and S. S. Chaudhuri, "Faster r-cnn and yolo based vehicle detection: A survey," in *2021 5th international conference on computing methodologies and communication (ICCMC)*, 2021, pp. 1442-1447: IEEE.

[27] X. Cheng and J. Yu, "RetinaNet with difference channel attention and adaptively spatial feature fusion for steel surface defect detection," *IEEE Transactions on Instrumentation and Measurement*, vol. 70, pp. 1-11, 2020.

[28] C.-L. Lin and K.-C. Wu, "Development of revised ResNet-50 for diabetic retinopathy detection," *BMC bioinformatics*, vol. 24, no. 1, pp. 1-18, 2023.

[29] M. Strongina, "Social and Economic Problems of Urbanization (Survey of the Literature)," *Problems in Economics*, vol. 17, no. 6, pp. 23-43, 1974.

[30] K. Wang, T. Wang, and X. Liu, "A review: Individual tree species classification using integrated airborne LiDAR and optical imagery with a focus on the urban environment," *Forests*, vol. 10, no. 1, p. 1, 2018.

[31] Q. Chang, X. Liu, J. Wu, and P. He, "MSPA-based urban green infrastructure planning and management approach for urban sustainability: Case study of Longgang in China," *Journal of Urban Planning and Development*, vol. 141, no. 3, p. A5014006, 2015.

[32] A. R. Shahtahmassebi *et al.*, "Remote sensing of urban green spaces: A review," *Urban Forestry & Urban Greening*, vol. 57, p. 126946, 2021.

[33] R. Neyns and F. Canters, "Mapping of urban vegetation with high-resolution remote sensing: A review," *Remote sensing*, vol. 14, no. 4, p. 1031, 2022.

[34] D. J. Nowak, "Air pollution removal by Chicago's urban forest," *Chicago's urban forest ecosystem: Results of the Chicago urban forest climate project*, pp. 63-81, 1994.

[35] K. P. Beckett, P. Freer-Smith, and G. Taylor, "Urban woodlands: their role in reducing the effects of particulate pollution," *Environmental pollution*, vol. 99, no. 3, pp. 347-360, 1998.

[36] A. Avdagić, B. Balić, D. Hadžidervišagić, and N. Kalača, "Inventory of Urban Greenery Using GIS Applications," in *International Conference "New Technologies, Development and Applications"*, 2023, pp. 179-188: Springer.

[37] J. Wang, W. Zhou, and M. Jiao, "Location matters: planting urban trees in the right places improves cooling," *Frontiers in Ecology and the Environment*, vol. 20, no. 3, pp. 147-151, 2022.

[38] H. Akbari, M. Pomerantz, and H. Taha, "Cool surfaces and shade trees to reduce energy use and improve air quality in urban areas," *Solar energy*, vol. 70, no. 3, pp. 295-310, 2001.

[39] D. J. Nowak and D. E. Crane, "Carbon storage and sequestration by urban trees in the USA," *Environmental pollution*, vol. 116, no. 3, pp. 381-389, 2002.

[40] R. M. De Carvalho and C. F. Szlafsztein, "Urban vegetation loss and ecosystem services: The influence on climate regulation and noise and air pollution," *Environmental Pollution*, vol. 245, pp. 844-852, 2019.

[41] T. Susca, S. R. Gaffin, and G. Dell'Osso, "Positive effects of vegetation: Urban heat island and green roofs," *Environmental pollution*, vol. 159, no. 8-9, pp. 2119-2126, 2011.

[42] M. L. Cadenasso, S. T. Pickett, and M. J. Grove, "Integrative approaches to investigating human-natural systems: the Baltimore ecosystem study," *Natures Sciences Sociétés,* vol. 14, no. 1, pp. 4-14, 2006.
[43] S. T. Pickett *et al.*, "Urban ecological systems: Scientific foundations and a decade of progress," *Journal of environmental management,* vol. 92, no. 3, pp. 331-362, 2011.
[44] A. Ossola and M. E. Hopton, "Measuring urban tree loss dynamics across residential landscapes," *Science of the Total Environment,* vol. 612, pp. 940-949, 2018.
[45] D. Nutsford, A. L. Pearson, and S. Kingham, "An ecological study investigating the association between access to urban green space and mental health," *Public health,* vol. 127, no. 11, pp. 1005-1011, 2013.
[46] D. J. Nowak, *Sustaining America's urban trees and forests* (no. 62). United States Department of Agriculture, Forest Service, Northern Research …, 2010.
[47] G. Johnson, C. Giblin, R. Murphy, E. North, and A. Rendahl, "Boulevard tree failures during wind loading events," *Arboriculture & Urban Forestry,* vol. 45, no. 6, 2019.
[48] B. Zhang and C. L. Brack, "Urban forest responses to climate change: A case study in Canberra," *Urban Forestry & Urban Greening,* vol. 57, p. 126910, 2021.
[49] E. B. Ward *et al.*, "Positive long-term impacts of restoration on soils in an experimental urban forest," *Ecological Applications,* vol. 31, no. 5, p. e02336, 2021.
[50] I. S.-P. P. o. Biodiversity and E. Services, "Global assessment report on biodiversity and ecosystem services of the Intergovernmental Science-Policy Platform on Biodiversity and Ecosystem Services," ed: IPBES secretariat Bonn, Germany, 2019.
[51] S. Oswalt, C. Oswalt, A. Crall, R. Rabaglia, M. K. Schwartz, and B. K. Kerns, "Inventory and monitoring of invasive species," *Invasive species in forests and rangelands of the United States,* p. 231, 2021.
[52] R. Liedtke *et al.*, "Hiking trails as conduits for the spread of non-native species in mountain areas," *Biological Invasions,* vol. 22, pp. 1121-1134, 2020.
[53] J. J. Lembrechts, A. Milbau, and I. Nijs, "Alien roadside species more easily invade alpine than lowland plant communities in a subarctic mountain ecosystem," *PloS one,* vol. 9, no. 2, p. e89664, 2014.
[54] S. Moricca *et al.*, "Biotic factors affecting ecosystem services in urban and peri-urban forests in Italy: The role of introduced and impending pathogens and pests," *Forests,* vol. 9, no. 2, p. 65, 2018.
[55] R. Eschen, J.-C. Grégoire, G. M. Hengeveld, B. M. de Hoop, L. Rigaux, and R. P. Potting, "Trade patterns of the tree nursery industry in Europe and changes following findings of citrus longhorn beetle, Anoplophora chinensis Forster," *NeoBiota,* vol. 26, pp. 1-20, 2015.
[56] R. J. Hauer, I. S. Hanou, and D. Sivyer, "Planning for active management of future invasive pests affecting urban forests: The ecological and economic effects of varying Dutch elm disease management practices for street trees in Milwaukee, WI USA," *Urban Ecosystems,* vol. 23, no. 5, pp. 1005-1022, 2020.
[57] E. J. Hudgins, F. H. Koch, M. J. Ambrose, and B. Leung, "Hotspots of pest-induced US urban tree death, 2020–2050," *Journal of Applied Ecology,* vol. 59, no. 5, pp. 1302-1312, 2022.
[58] R. L. Chazdon, "Beyond deforestation: restoring forests and ecosystem services on degraded lands," *science,* vol. 320, no. 5882, pp. 1458-1460, 2008.
[59] A. Boerema, "Ecosystem services: study of human influences on nature and the effects for society," Universiteit Antwerpen (Belgium), 2016.
[60] D. Yemshanov, J. Biggs, D. W. McKenney, and T. Lempriere, "Effects of permanence requirements on afforestation choices for carbon sequestration for Ontario, Canada," *Forest Policy and Economics,* vol. 14, no. 1, pp. 6-18, 2012.
[61] M. G. Betts *et al.*, "Global forest loss disproportionately erodes biodiversity in intact landscapes," *Nature,* vol. 547, no. 7664, pp. 441-444, 2017.
[62] K. C. Seto, B. Güneralp, and L. R. Hutyra, "Global forecasts of urban expansion to 2030 and direct impacts on biodiversity and carbon pools," *Proceedings of the National Academy of Sciences,* vol. 109, no. 40, pp. 16083-16088, 2012.
[63] J. A. Foley *et al.*, "Global consequences of land use," *science,* vol. 309, no. 5734, pp. 570-574, 2005.
[64] J. Jin, S. E. Gergel, Y. Lu, N. C. Coops, and C. Wang, "Asian cities are greening while some North American cities are browning: long-term greenspace patterns in 16 cities of the Pan-Pacific region," *Ecosystems,* vol. 23, pp. 383-399, 2020.
[65] R. Ahn, T. F. Burke, and A. M. McGahan, "Innovating for healthy urbanization," Springer2015.
[66] S. Pierre, P. M. Groffman, M. E. Killilea, and E. E. Oldfield, "Soil microbial nitrogen cycling and nitrous oxide emissions from urban afforestation in the New York City Afforestation Project," *Urban Forestry & Urban Greening,* vol. 15, pp. 149-154, 2016.

[67] C. D'Almeida, C. J. Vörösmarty, G. C. Hurtt, J. A. Marengo, S. L. Dingman, and B. D. Keim, "The effects of deforestation on the hydrological cycle in Amazonia: a review on scale and resolution," *International Journal of Climatology: A Journal of the Royal Meteorological Society*, vol. 27, no. 5, pp. 633-647, 2007.

[68] C. Rosenzweig *et al.*, "Attributing physical and biological impacts to anthropogenic climate change," *Nature*, vol. 453, no. 7193, pp. 353-357, 2008.

[69] M. Nastran, M. Kobal, and K. Eler, "Urban heat islands in relation to green land use in European cities," *Urban Forestry & Urban Greening*, vol. 37, pp. 33-41, 2019.

[70] S. Chapman, J. E. Watson, A. Salazar, M. Thatcher, and C. A. McAlpine, "The impact of urbanization and climate change on urban temperatures: a systematic review," *Landscape Ecology*, vol. 32, pp. 1921-1935, 2017.

[71] R. Pu and S. Landry, "A comparative analysis of high spatial resolution IKONOS and WorldView-2 imagery for mapping urban tree species," *Remote Sensing of Environment*, vol. 124, pp. 516-533, 2012.

[72] P. Alighaleh, R. Pakdel, N. Ghanei Ghooshkhaneh, S. Einafshar, A. Rohani, and M. H. Saeidirad, "Detection and Classification of Saffron Adulterants by Vis-Nir Imaging, Chemical Analysis, and Soft Computing," *Foods*, vol. 12, no. 11, p. 2192, 2023.

[73] S. Zhang, X. Li, Y. Ba, X. Lyu, M. Zhang, and M. Li, "Banana Fusarium Wilt Disease Detection by Supervised and Unsupervised Methods from UAV-Based Multispectral Imagery," *Remote Sensing*, vol. 14, no. 5, p. 1231, 2022.

[74] R. N. Sahoo, S. Ray, and K. Manjunath, "Hyperspectral remote sensing of agriculture," *Current science*, pp. 848-859, 2015.

[75] A. Khan, A. D. Vibhute, S. Mali, and C. Patil, "A systematic review on hyperspectral imaging technology with a machine and deep learning methodology for agricultural applications," *Ecological Informatics*, vol. 69, p. 101678, 2022.

[76] M. Hemati, M. Hasanlou, M. Mahdianpari, and F. Mohammadimanesh, "A systematic review of landsat data for change detection applications: 50 years of monitoring the earth," *Remote sensing*, vol. 13, no. 15, p. 2869, 2021.

[77] R. C. Sharma, "An Ultra-Resolution Features Extraction Suite for Community-Level Vegetation Differentiation and Mapping at a Sub-Meter Resolution," *Remote Sensing*, vol. 14, no. 13, p. 3145, 2022.

[78] F. A. Al-Wassai and N. Kalyankar, "Major limitations of satellite images," *arXiv preprint arXiv:1307.2434*, 2013.

[79] J. Rogan and D. Chen, "Remote sensing technology for mapping and monitoring land-cover and land-use change," *Progress in planning*, vol. 61, no. 4, pp. 301-325, 2004.

[80] L. Royimani, O. Mutanga, J. Odindi, T. Dube, and T. N. Matongera, "Advancements in satellite remote sensing for mapping and monitoring of alien invasive plant species (AIPs)," *Physics and Chemistry of the Earth, Parts A/B/C*, vol. 112, pp. 237-245, 2019.

[81] A. B. Owusu, "An assessment of urban vegetation abundance in Accra Metropolitan Area, Ghana: a geospatial approach," *Journal of Environmental Geography*, vol. 11, no. 1-2, pp. 37-44, 2018.

[82] Y. Zhang and Z. Shao, "Assessing of urban vegetation biomass in combination with LiDAR and high-resolution remote sensing images," *International Journal of Remote Sensing*, vol. 42, no. 3, pp. 964-985, 2021.

[83] M. Gašparović and D. Dobrinić, "Comparative assessment of machine learning methods for urban vegetation mapping using multitemporal sentinel-1 imagery," *Remote Sensing*, vol. 12, no. 12, p. 1952, 2020.

[84] P. Kowe, O. Mutanga, and T. Dube, "Advancements in the remote sensing of landscape pattern of urban green spaces and vegetation fragmentation," *International Journal of Remote Sensing*, vol. 42, no. 10, pp. 3797-3832, 2021.

[85] H. Hashim, Z. Abd Latif, and N. A. Adnan, "Urban vegetation classification with NDVI threshold value method with very high resolution (VHR) Pleiades imagery," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 42, pp. 237-240, 2019.

[86] M. M. F. Wong, J. C. H. Fung, and P. P. S. Yeung, "High-resolution calculation of the urban vegetation fraction in the Pearl River Delta from the Sentinel-2 NDVI for urban climate model parameterization," *Geoscience Letters*, vol. 6, pp. 1-10, 2019.

[87] S. W. Myint, P. Gober, A. Brazel, S. Grossman-Clarke, and Q. Weng, "Per-pixel vs. object-based classification of urban land cover extraction using high spatial resolution imagery," *Remote sensing of environment*, vol. 115, no. 5, pp. 1145-1161, 2011.

[88] L. Ma, M. Li, X. Ma, L. Cheng, P. Du, and Y. Liu, "A review of supervised object-based land-cover image classification," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 130, pp. 277-293, 2017.

[89] Y. Ban, H. Hu, and I. M. Rangel, "Fusion of Quickbird MS and RADARSAT SAR data for urban land-cover mapping: Object-based and knowledge-based approach," *International Journal of Remote Sensing*, vol. 31, no. 6, pp. 1391-1410, 2010.

[90] D. C. Tsouros, S. Bibi, and P. G. Sarigiannidis, "A review on UAV-based applications for precision agriculture," *Information*, vol. 10, no. 11, p. 349, 2019.

[91] S. A. H. Mohsan, M. A. Khan, F. Noor, I. Ullah, and M. H. Alsharif, "Towards the unmanned aerial vehicles (UAVs): A comprehensive review," *Drones*, vol. 6, no. 6, p. 147, 2022.

[92] T. Alladi, V. Chamola, N. Sahu, and M. Guizani, "Applications of blockchain in unmanned aerial vehicles: A review," *Vehicular Communications*, vol. 23, p. 100249, 2020.

[93] P. Chandhar and E. G. Larsson, "Massive MIMO for connectivity with drones: Case studies and future directions," *IEEE access*, vol. 7, pp. 94676-94691, 2019.

[94] M. Shahbazi, J. Théau, and P. Ménard, "Recent applications of unmanned aerial imagery in natural resource management," *GIScience & Remote Sensing*, vol. 51, no. 4, pp. 339-365, 2014.

[95] Q. Feng, J. Liu, and J. Gong, "UAV remote sensing for urban vegetation mapping using random forest and texture analysis," *Remote sensing*, vol. 7, no. 1, pp. 1074-1094, 2015.

[96] O. Hassaan, A. K. Nasir, H. Roth, and M. F. Khan, "Precision forestry: trees counting in urban areas using visible imagery based on an unmanned aerial vehicle," *IFAC-PapersOnLine*, vol. 49, no. 16, pp. 16-21, 2016.

[97] R. Gini, G. Sona, G. Ronchetti, D. Passoni, and L. Pinto, "Improving tree species classification using UAS multispectral images and texture measures," *ISPRS International Journal of Geo-Information*, vol. 7, no. 8, p. 315, 2018.

[98] S. E. Franklin and O. S. Ahmed, "Deciduous tree species classification using object-based analysis and machine learning with unmanned aerial vehicle multispectral data," *International Journal of Remote Sensing*, vol. 39, no. 15-16, pp. 5236-5245, 2018.

[99] D. Lobo Torres *et al.*, "Applying fully convolutional architectures for semantic segmentation of a single tree species in urban environment on high resolution UAV optical imagery," *Sensors*, vol. 20, no. 2, p. 563, 2020.

[100] C. Zhang, K. Xia, H. Feng, Y. Yang, and X. Du, "Tree species classification using deep learning and RGB optical images obtained by an unmanned aerial vehicle," *Journal of Forestry Research*, vol. 32, no. 5, pp. 1879-1888, 2021.

[101] M. Onishi, S. Watanabe, T. Nakashima, and T. Ise, "Practicality and robustness of tree species identification using UAV RGB image and deep learning in temperate Forest in Japan," *Remote Sensing*, vol. 14, no. 7, p. 1710, 2022.

[102] X. Yu, Y. Her, W. Huo, G. Chen, and W. Qi, "Spatio-temporal monitoring of urban street-side vegetation greenery using Baidu Street View images," *Urban Forestry & Urban Greening*, vol. 73, p. 127617, 2022.

[103] Y. Xia, N. Yabuki, and T. Fukuda, "Development of a system for assessing the quality of urban street-level greenery using street view images and deep learning," *Urban Forestry & Urban Greening*, vol. 59, p. 126995, 2021.

[104] E. Barbierato, I. Bernetti, I. Capecchi, and C. Saragosa, "Integrating remote sensing and street view images to quantify urban forest ecosystem services. Remote Sens. 12 (2), 329," ed, 2020.

[105] X. Li, C. Zhang, W. Li, R. Ricard, Q. Meng, and W. Zhang, "Assessing street-level urban greenery using Google Street View and a modified green view index," *Urban Forestry & Urban Greening*, vol. 14, no. 3, pp. 675-685, 2015.

[106] J. Yang, L. Zhao, J. Mcbride, and P. Gong, "Can you see green? Assessing the visibility of urban forests in cities," *Landscape and Urban Planning*, vol. 91, no. 2, pp. 97-104, 2009.

[107] C. Wang *et al.*, "A review of deep learning used in the hyperspectral image analysis for agriculture," *Artificial Intelligence Review*, vol. 54, no. 7, pp. 5205-5253, 2021.

[108] M. Y. Teng, R. Mehrubeoglu, S. A. King, K. Cammarata, and J. Simons, "Investigation of epifauna coverage on seagrass blades using spatial and spectral analysis of hyperspectral images," in *2013 5th Workshop on Hyperspectral Image and Signal Processing: Evolution in Remote Sensing (WHISPERS)*, 2013, pp. 1-4: IEEE.

[109] B. Lu, P. D. Dao, J. Liu, Y. He, and J. Shang, "Recent advances of hyperspectral imaging technology and applications in agriculture," *Remote Sensing*, vol. 12, no. 16, p. 2659, 2020.

[110] M. Shimoni, R. Haelterman, and C. Perneel, "Hypersectral imaging for military and security applications: Combining myriad processing and sensing techniques," *IEEE Geoscience and Remote Sensing Magazine*, vol. 7, no. 2, pp. 101-117, 2019.

[111] D. Krupnik and S. Khan, "Close-range, ground-based hyperspectral imaging for mining applications at various scales: Review and case studies," *Earth-science reviews*, vol. 198, p. 102952, 2019.

[112] A. ul Rehman and S. A. Qureshi, "A review of the medical hyperspectral imaging systems and unmixing algorithms' in biological tissues," *Photodiagnosis and Photodynamic Therapy,* vol. 33, p. 102165, 2021.
[113] S. Li, W. Song, L. Fang, Y. Chen, P. Ghamisi, and J. A. Benediktsson, "Deep learning for hyperspectral image classification: An overview," *IEEE Transactions on Geoscience and Remote Sensing,* vol. 57, no. 9, pp. 6690-6709, 2019.
[114] W. Lv and X. Wang, "Overview of hyperspectral image classification," *Journal of Sensors,* vol. 2020, 2020.
[115] M. L. Clark, "Comparison of multi-seasonal Landsat 8, Sentinel-2 and hyperspectral images for mapping forest alliances in Northern California," *ISPRS journal of photogrammetry and remote sensing,* vol. 159, pp. 26-40, 2020.
[116] M. P. Ferreira, F. H. Wagner, L. E. Aragão, Y. E. Shimabukuro, and C. R. de Souza Filho, "Tree species classification in tropical forests using visible to shortwave infrared WorldView-3 images and texture analysis," *ISPRS journal of photogrammetry and remote sensing,* vol. 149, pp. 119-131, 2019.
[117] F. Schiefer *et al.*, "Mapping forest tree species in high resolution UAV-based RGB-imagery by means of convolutional neural networks," *ISPRS Journal of Photogrammetry and Remote Sensing,* vol. 170, pp. 205-215, 2020.
[118] S. Wu *et al.*, "Monitoring tree-crown scale autumn leaf phenology in a temperate forest with an integration of PlanetScope and drone remote sensing observations," *ISPRS Journal of Photogrammetry and Remote Sensing,* vol. 171, pp. 36-48, 2021.
[119] W. Y. Yan, K. van Ewijk, P. Treitz, and A. Shaker, "Effects of radiometric correction on cover type and spatial resolution for modeling plot level forest attributes using multispectral airborne LiDAR data," *ISPRS Journal of Photogrammetry and Remote Sensing,* vol. 169, pp. 152-165, 2020.
[120] S. Kothari *et al.*, "Community-wide consequences of variation in photoprotective physiology among prairie plants," *Photosynthetica,* vol. 56, pp. 455-467, 2018.
[121] J. Delegido *et al.*, "Chlorophyll content mapping of urban vegetation in the city of Valencia based on the hyperspectral NAOC index," *Ecological Indicators,* vol. 40, pp. 34-42, 2014.
[122] S. Abbas *et al.*, "Characterizing and classifying urban tree species using bi-monthly terrestrial hyperspectral images in Hong Kong," *ISPRS Journal of Photogrammetry and Remote Sensing,* vol. 177, pp. 204-216, 2021.
[123] M. Liu *et al.*, "The impact of spatial resolution on the classification of vegetation types in highly fragmented planting areas based on unmanned aerial vehicle hyperspectral images," *Remote Sensing,* vol. 12, no. 1, p. 146, 2020.
[124] P. Ghamisi *et al.*, "Advances in hyperspectral image and signal processing: A comprehensive overview of the state of the art," *IEEE Geoscience and Remote Sensing Magazine,* vol. 5, no. 4, pp. 37-78, 2017.
[125] X. Li, Z. Li, H. Qiu, G. Hou, and P. Fan, "An overview of hyperspectral image feature extraction, classification methods and the methods based on small samples," *Applied Spectroscopy Reviews,* vol. 58, no. 6, pp. 367-400, 2023.
[126] W. H. Su and D. W. Sun, "Multispectral imaging for plant food quality analysis and visualization," *Comprehensive reviews in food science and food safety,* vol. 17, no. 1, pp. 220-239, 2018.
[127] L. Duempelmann, B. Gallinet, and L. Novotny, "Multispectral imaging with tunable plasmonic filters," *ACS Photonics,* vol. 4, no. 2, pp. 236-241, 2017.
[128] C.-T. Pan, M. D. Francisco, C.-K. Yen, S.-Y. Wang, and Y.-L. Shiue, "Vein pattern locating technology for cannulation: a review of the low-cost vein finder prototypes utilizing near infrared (NIR) light to improve peripheral subcutaneous vein selection for phlebotomy," *Sensors,* vol. 19, no. 16, p. 3573, 2019.
[129] X. Ai *et al.*, "Multispectral optoacoustic imaging of dynamic redox correlation and pathophysiological progression utilizing upconversion nanoprobes," *Nature communications,* vol. 10, no. 1, p. 1087, 2019.
[130] Y. Chang, Y. Hu, Z. Chen, and D. Xing, "Co-impulse multispectral photoacoustic microscopy and optical coherence tomography system using a single supercontinuum laser," *Optics Letters,* vol. 44, no. 18, pp. 4459-4462, 2019.
[131] M. A. Hossen, P. K. Diwakar, and S. Ragi, "Total nitrogen estimation in agricultural soils via aerial multispectral imaging and LIBS," *Scientific Reports,* vol. 11, no. 1, p. 12693, 2021.
[132] Y. Yang, X. Zhao, M. Huang, X. Wang, and Q. Zhu, "Multispectral image based germination detection of potato by using supervised multiple threshold segmentation model and Canny edge detector," *Computers and Electronics in Agriculture,* vol. 182, p. 106041, 2021.
[133] K.-J. Fan and W.-H. Su, "Applications of fluorescence spectroscopy, RGB-and MultiSpectral imaging for quality determinations of white meat: a review," *Biosensors,* vol. 12, no. 2, p. 76, 2022.

[134] B. Park, S.-C. Yoon, W. R. Windham, K. C. Lawrence, M. S. Kim, and K. Chao, "Line-scan hyperspectral imaging for real-time in-line poultry fecal detection," *Sensing and instrumentation for food quality and safety*, vol. 5, pp. 25-32, 2011.

[135] Q. Guo *et al.*, "Urban tree classification based on object-oriented approach and random forest algorithm using unmanned aerial vehicle (uav) multispectral imagery," *Remote Sensing*, vol. 14, no. 16, p. 3885, 2022.

[136] P. K. Soni, N. Rajpal, R. Mehta, and V. K. Mishra, "Urban land cover and land use classification using multispectral sentinal-2 imagery," *Multimedia Tools and Applications*, pp. 1-15, 2021.

[137] K. Y. Peerbhay, O. Mutanga, and R. Ismail, "Investigating the capability of few strategically placed Worldview-2 multispectral bands to discriminate forest species in KwaZulu-Natal, South Africa," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 7, no. 1, pp. 307-316, 2013.

[138] T. Shin, S. Jeong, and J. Ko, "Development of a Radiometric Calibration Method for Multispectral Images of Croplands Obtained with a Remote-Controlled Aerial System," *Remote Sensing*, vol. 15, no. 5, p. 1408, 2023.

[139] K. Ren, W. Sun, X. Meng, G. Yang, J. Peng, and J. Huang, "A locally optimized model for hyperspectral and multispectral images fusion," *IEEE Transactions on Geoscience and Remote Sensing*, vol. 60, pp. 1-15, 2021.

[140] H. K. Aggarwal and A. Majumdar, "Single-sensor multi-spectral image demosaicing algorithm using learned interpolation weights," in *2014 IEEE Geoscience and Remote Sensing Symposium*, 2014, pp. 2011-2014: IEEE.

[141] S. Nezami, E. Khoramshahi, O. Nevalainen, I. Pölönen, and E. Honkavaara, "Tree species classification of drone hyperspectral and RGB imagery with deep learning convolutional neural networks," *Remote Sensing*, vol. 12, no. 7, p. 1070, 2020.

[142] G. Yan *et al.*, "Improving the estimation of fractional vegetation cover from UAV RGB imagery by colour unmixing," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 158, pp. 23-34, 2019.

[143] S. Sankaran, J. J. Quirós, and P. N. Miklas, "Unmanned aerial system and satellite-based high resolution imagery for high-throughput phenotyping in dry bean," *Computers and Electronics in Agriculture*, vol. 165, p. 104965, 2019.

[144] B. Ayhan and C. Kwan, "Tree, shrub, and grass classification using only RGB images," *Remote Sensing*, vol. 12, no. 8, p. 1333, 2020.

[145] Z. Bitarafan, J. Rasmussen, J. C. Westergaard, and C. Andreasen, "Seed yield and lodging assessment in red fescue (Festuca rubra L.) sprayed with trinexapac-ethyl," *Agronomy*, vol. 9, no. 10, p. 617, 2019.

[146] I. Borra-Serrano *et al.*, "Towards an objective evaluation of persistency of Lolium perenne swards using UAV imagery," *Euphytica*, vol. 214, pp. 1-18, 2018.

[147] J. A. Fernandez-Gallego, S. C. Kefauver, T. Vatter, N. A. Gutiérrez, M. T. Nieto-Taladriz, and J. L. Araus, "Low-cost assessment of grain yield in durum wheat using RGB images," *European Journal of Agronomy*, vol. 105, pp. 146-156, 2019.

[148] S. Postolache, P. Sebastião, V. Viegas, O. Postolache, and F. Cercas, "IoT-Based Systems for Soil Nutrients Assessment in Horticulture," *Sensors*, vol. 23, no. 1, p. 403, 2022.

[149] Z. Shi, C. Chen, Z. Xiong, D. Liu, and F. Wu, "Hscnn+: Advanced cnn-based hyperspectral recovery from rgb images," in *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition Workshops*, 2018, pp. 939-947.

[150] M. Taghizadeh, A. A. Gowen, and C. P. O'Donnell, "Comparison of hyperspectral imaging with conventional RGB imaging for quality evaluation of Agaricus bisporus mushrooms," *Biosystems engineering*, vol. 108, no. 2, pp. 191-194, 2011.

[151] J. Zhao *et al.*, "Deep learning in hyperspectral image reconstruction from single RGB images—A case study on tomato quality parameters," *Remote Sensing*, vol. 12, no. 19, p. 3258, 2020.

[152] B. Zhou, A. Elazab, J. Bort, O. Vergara, M. Serret, and J. L. Araus, "Low-cost assessment of wheat resistance to yellow rust through conventional RGB images," *Computers and Electronics in Agriculture*, vol. 116, pp. 20-29, 2015.

[153] M. Schirrmann, A. Giebel, F. Gleiniger, M. Pflanz, J. Lentschke, and K.-H. Dammer, "Monitoring agronomic parameters of winter wheat crops with low-cost UAV imagery," *Remote Sensing*, vol. 8, no. 9, p. 706, 2016.

[154] U. Hasan, M. Sawut, and S. Chen, "Estimating the leaf area index of winter wheat based on unmanned aerial vehicle RGB-image parameters," *Sustainability*, vol. 11, no. 23, p. 6829, 2019.

[155] W. Liu, Y. Li, J. Liu, and J. Jiang, "Estimation of plant height and aboveground biomass of Toona sinensis under drought stress using RGB-D imaging," *Forests*, vol. 12, no. 12, p. 1747, 2021.

[156] A. Gracia-Romero *et al.*, "Comparative performance of ground vs. aerially assessed RGB and multispectral indices for early-growth evaluation of maize performance under phosphorus fertilization," *Frontiers in Plant Science*, vol. 8, p. 2004, 2017.
[157] Y. Guo *et al.*, "Integrating spectral and textural information for monitoring the growth of pear trees using optical images from the UAV platform," *Remote Sensing*, vol. 13, no. 9, p. 1795, 2021.
[158] J. Su *et al.*, "Machine learning-based crop drought mapping system by UAV remote sensing RGB imagery," *Unmanned systems*, vol. 8, no. 01, pp. 71-83, 2020.
[159] M. N. Megat Mohamed Nazir, R. Terhem, A. R. Norhisham, S. Mohd Razali, and R. Meder, "Early monitoring of health status of plantation-grown eucalyptus pellita at large spatial scale via visible spectrum imaging of canopy foliage using unmanned aerial vehicles," *Forests*, vol. 12, no. 10, p. 1393, 2021.
[160] G. Yang *et al.*, "Unmanned aerial vehicle remote sensing for field-based crop phenotyping: current status and perspectives," *Frontiers in plant science*, vol. 8, p. 1111, 2017.
[161] M. de Araújo Carvalho *et al.*, "A deep learning-based mobile application for tree species mapping in RGB images," *International Journal of Applied Earth Observation and Geoinformation*, vol. 114, p. 103045, 2022.
[162] P. A. P. ZamboniThgeThe, J. Marcato, G. T. Miyoshi, J. de Andrade Silva, J. Martins, and W. N. Gonçalves, "Assessment of CNN-based methods for single tree detection on high-resolution RGB images in urban areas," in *2021 IEEE International Geoscience and Remote Sensing Symposium IGARSS*, 2021, pp. 590-593: IEEE.
[163] X. Wang, Y. Wang, C. Zhou, L. Yin, and X. Feng, "Urban forest monitoring based on multiple features at the single tree scale by UAV," *Urban Forestry & Urban Greening*, vol. 58, p. 126958, 2021.
[164] M. Onishi and T. Ise, "Explainable identification and mapping of trees using UAV RGB image and deep learning," *Scientific reports*, vol. 11, no. 1, p. 903, 2021.
[165] S. Egli and M. Höpke, "CNN-based tree species classification using high resolution RGB image data from automated UAV observations," *Remote Sensing*, vol. 12, no. 23, p. 3892, 2020.
[166] Y. Shao, W. Li, H. Chu, Z. Chang, X. Zhang, and H. Zhan, "A multitask cascading cnn with multiscale infrared optical flow feature fusion-based abnormal crowd behavior monitoring uav," *Sensors*, vol. 20, no. 19, p. 5550, 2020.
[167] T. Peng, Q. Li, and P. Zhu, "Rgb-t crowd counting from drone: A benchmark and mmccn network," in *Proceedings of the Asian conference on computer vision*, 2020.
[168] S. Hartling, V. Sagan, and M. Maimaitijiang, "Urban tree species classification using UAV-based multi-sensor data fusion and machine learning," *GIScience & Remote Sensing*, vol. 58, no. 8, pp. 1250-1275, 2021.
[169] M. Rowe, "An introduction to machine learning for clinicians," *Academic Medicine*, vol. 94, no. 10, pp. 1433-1436, 2019.
[170] J. Zhang, M. Li, Y. Feng, and C. Yang, "Robotic grasp detection based on image processing and random forest," *Multimedia Tools and Applications*, vol. 79, pp. 2427-2446, 2020.
[171] Y. Chen, H. Sun, G. Zhou, and B. Peng, "Fruit classification model based on residual filtering network for smart community robot," *Wireless Communications and Mobile Computing*, vol. 2021, pp. 1-9, 2021.
[172] T. Kasinathan and S. R. Uyyala, "Machine learning ensemble with image processing for pest identification and classification in field crops," *Neural Computing and Applications*, vol. 33, pp. 7491-7504, 2021.
[173] N. Islam *et al.*, "Early weed detection using image processing and machine learning techniques in an Australian chilli farm," *Agriculture*, vol. 11, no. 5, p. 387, 2021.
[174] A. Hage Chehade, N. Abdallah, J.-M. Marion, M. Oueidat, and P. Chauvet, "Lung and colon cancer classification using medical imaging: A feature engineering approach," *Physical and Engineering Sciences in Medicine*, vol. 45, no. 3, pp. 729-746, 2022.
[175] M. Lin *et al.*, "Artificial intelligence in tumor subregion analysis based on medical imaging: A review," *Journal of Applied Clinical Medical Physics*, vol. 22, no. 7, pp. 10-26, 2021.
[176] C. Palaiokostas, "Predicting for disease resistance in aquaculture species using machine learning models," *Aquaculture Reports*, vol. 20, p. 100660, 2021.
[177] A. Setiawan, H. Hadiyanto, and C. E. Widodo, "Shrimp Body Weight Estimation in Aquaculture Ponds Using Morphometric Features Based on Underwater Image Analysis and Machine Learning Approach," *Revue d'Intelligence Artificielle*, vol. 36, no. 6, p. 905, 2022.
[178] P. Sicard *et al.*, "Object-based classification of urban plant species from very high-resolution satellite imagery," *Urban Forestry & Urban Greening*, vol. 81, p. 127866, 2023.
[179] Z. Cetin and N. Yastikli, "The use of machine learning algorithms in urban tree species classification," *ISPRS International Journal of Geo-Information*, vol. 11, no. 4, p. 226, 2022.

[180] R. Pu, S. Landry, and Q. Yu, "Assessing the potential of multi-seasonal high resolution Pléiades satellite imagery for mapping urban tree species," *International Journal of Applied Earth Observation and Geoinformation,* vol. 71, pp. 144-158, 2018.
[181] A. Chahal and P. Gulia, "Machine learning and deep learning," *Int. J. Innov. Technol. Explor. Eng,* vol. 8, no. 12, pp. 4910-4914, 2019.
[182] L. G. Di Maggio, "Intelligent fault diagnosis of industrial bearings using transfer learning and CNNs pre-trained for audio classification," *Sensors,* vol. 23, no. 1, p. 211, 2022.
[183] A. Aamer, L. Eka Yani, and I. Alan Priyatna, "Data analytics in the supply chain management: Review of machine learning applications in demand forecasting," *Operations and Supply Chain Management: An International Journal,* vol. 14, no. 1, pp. 1-13, 2020.
[184] J. Latif, C. Xiao, A. Imran, and S. Tu, "Medical imaging using machine learning and deep learning algorithms: a review," in *2019 2nd International conference on computing, mathematics and engineering technologies (iCoMET)*, 2019, pp. 1-5: IEEE.
[185] S. Zhang, N. Qu, T. Zheng, and C. Hu, "Series arc fault detection based on wavelet compression reconstruction data enhancement and deep residual network," *IEEE Transactions on Instrumentation and Measurement,* vol. 71, pp. 1-9, 2022.
[186] N. G. Paterakis, E. Mocanu, M. Gibescu, B. Stappers, and W. van Alst, "Deep learning versus traditional machine learning methods for aggregated energy demand prediction," in *2017 IEEE PES Innovative Smart Grid Technologies Conference Europe (ISGT-Europe)*, 2017, pp. 1-6: IEEE.
[187] A. L'heureux, K. Grolinger, H. F. Elyamany, and M. A. Capretz, "Machine learning with big data: Challenges and approaches," *Ieee Access,* vol. 5, pp. 7776-7797, 2017.
[188] Y. Diez, S. Kentsch, M. Fukuda, M. L. L. Caceres, K. Moritake, and M. Cabezas, "Deep learning in forestry using uav-acquired rgb data: A practical review," *Remote Sensing,* vol. 13, no. 14, p. 2837, 2021.
[189] S. Ghosal *et al.*, "A weakly supervised deep learning framework for sorghum head detection and counting," *Plant Phenomics,* 2019.
[190] E. Rahimy, "Deep learning applications in ophthalmology," *Current opinion in ophthalmology,* vol. 29, no. 3, pp. 254-260, 2018.
[191] Q. Rao and J. Frtunikj, "Deep learning for self-driving cars: Chances and challenges," in *Proceedings of the 1st international workshop on software engineering for AI in autonomous systems*, 2018, pp. 35-38.
[192] B. Ayhan *et al.*, "Vegetation detection using deep learning and conventional methods," *Remote Sensing,* vol. 12, no. 15, p. 2502, 2020.
[193] N. Audebert, B. Le Saux, and S. Lefèvre, "Deep learning for classification of hyperspectral data: A comparative review," *IEEE geoscience and remote sensing magazine,* vol. 7, no. 2, pp. 159-173, 2019.
[194] P. G. Brodrick, A. B. Davies, and G. P. Asner, "Uncovering ecological patterns with convolutional neural networks," *Trends in ecology & evolution,* vol. 34, no. 8, pp. 734-745, 2019.
[195] T. Kattenborn, J. Leitloff, F. Schiefer, and S. Hinz, "Review on Convolutional Neural Networks (CNN) in vegetation remote sensing," *ISPRS journal of photogrammetry and remote sensing,* vol. 173, pp. 24-49, 2021.
[196] J. Muro *et al.*, "Predicting plant biomass and species richness in temperate grasslands across regions, time, and land management with remote sensing and deep learning," *Remote Sensing of Environment,* vol. 282, p. 113262, 2022.
[197] K. Albarrak, Y. Gulzar, Y. Hamid, A. Mehmood, and A. B. Soomro, "A deep learning-based model for date fruit classification," *Sustainability,* vol. 14, no. 10, p. 6339, 2022.
[198] N. O'Mahony *et al.*, "Deep learning vs. traditional computer vision," in *Advances in Computer Vision: Proceedings of the 2019 Computer Vision Conference (CVC), Volume 1 1*, 2020, pp. 128-144: Springer.
[199] Z. Zhang, P. Cui, and W. Zhu, "Deep learning on graphs: A survey," *IEEE Transactions on Knowledge and Data Engineering,* vol. 34, no. 1, pp. 249-270, 2020.
[200] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *nature,* vol. 521, no. 7553, pp. 436-444, 2015.
[201] A. Shrestha and A. Mahmood, "Review of deep learning algorithms and architectures," *IEEE access,* vol. 7, pp. 53040-53065, 2019.
[202] C. Shorten, T. M. Khoshgoftaar, and B. Furht, "Deep Learning applications for COVID-19," *Journal of big Data,* vol. 8, no. 1, pp. 1-54, 2021.
[203] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "ImageNet classification with deep convolutional neural networks," *Communications of the ACM,* vol. 60, no. 6, pp. 84-90, 2017.
[204] S. Bhowmick, S. Nagarajaiah, and A. Veeraraghavan, "Vision and deep learning-based algorithms to detect and quantify cracks on concrete surfaces from UAV videos," *Sensors,* vol. 20, no. 21, p. 6299, 2020.

[205] R. I. Hasan, S. M. Yusuf, and L. Alzubaidi, "Review of the state of the art of deep learning for plant diseases: A broad analysis and discussion," *Plants*, vol. 9, no. 10, p. 1302, 2020.
[206] W. Yang, X. Zhang, Y. Tian, W. Wang, J.-H. Xue, and Q. Liao, "Deep learning for single image super-resolution: A brief review," *IEEE Transactions on Multimedia*, vol. 21, no. 12, pp. 3106-3121, 2019.
[207] Z.-Q. Zhao, P. Zheng, S.-t. Xu, and X. Wu, "Object detection with deep learning: A review," *IEEE transactions on neural networks and learning systems*, vol. 30, no. 11, pp. 3212-3232, 2019.
[208] J. Tang, S. Li, and P. Liu, "A review of lane detection methods based on deep learning," *Pattern Recognition*, vol. 111, p. 107623, 2021.
[209] 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.
[210] L. Alzubaidi *et al.*, "Review of deep learning: Concepts, CNN architectures, challenges, applications, future directions," *Journal of big Data*, vol. 8, pp. 1-74, 2021.
[211] S. Nevo *et al.*, "ML for flood forecasting at scale," *arXiv preprint arXiv:1901.09583*, 2019.
[212] H. Chen, O. Engkvist, Y. Wang, M. Olivecrona, and T. Blaschke, "The rise of deep learning in drug discovery," *Drug discovery today*, vol. 23, no. 6, pp. 1241-1250, 2018.
[213] K. Nagpal, D. Foote, Y. Liu, P. Chen, and E. Wulczyn, "Stumpe MC Development and validation of a deep learning algorithm for improving Gleason scoring of prostate cancer. Nature Partner Journal Digital," *Medicine*, vol. 2, no. 48, p. 1, 2019.
[214] E. Wulczyn *et al.*, "Deep learning-based survival prediction for multiple cancer types using histopathology images," *PloS one*, vol. 15, no. 6, p. e0233678, 2020.
[215] Y. Benhammou, B. Achchab, F. Herrera, and S. Tabik, "BreakHis based breast cancer automatic diagnosis using deep learning: Taxonomy, survey and insights," *Neurocomputing*, vol. 375, pp. 9-24, 2020.
[216] K. Cao and X. Zhang, "An improved res-unet model for tree species classification using airborne high-resolution images," *Remote Sensing*, vol. 12, no. 7, p. 1128, 2020.
[217] X.-Y. Tong *et al.*, "Land-cover classification with high-resolution remote sensing images using transferable deep models," *Remote Sensing of Environment*, vol. 237, p. 111322, 2020.
[218] S. Natesan, C. Armenakis, and U. Vepakomma, "Resnet-based tree species classification using uav images," *The International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. 42, pp. 475-481, 2019.
[219] Y. Sun, J. Huang, Z. Ao, D. Lao, and Q. Xin, "Deep learning approaches for the mapping of tree species diversity in a tropical wetland using airborne LiDAR and high-spatial-resolution remote sensing images," *Forests*, vol. 10, no. 11, p. 1047, 2019.
[220] W. Li, H. Fu, L. Yu, and A. Cracknell, "Deep learning based oil palm tree detection and counting for high-resolution remote sensing images," *Remote sensing*, vol. 9, no. 1, p. 22, 2016.
[221] N. Flood, F. Watson, and L. Collett, "Using a U-net convolutional neural network to map woody vegetation extent from high resolution satellite imagery across Queensland, Australia," *International Journal of Applied Earth Observation and Geoinformation*, vol. 82, p. 101897, 2019.
[222] S. Hartling, V. Sagan, P. Sidike, M. Maimaitijiang, and J. Carron, "Urban tree species classification using a WorldView-2/3 and LiDAR data fusion approach and deep learning," *Sensors*, vol. 19, no. 6, p. 1284, 2019.
[223] S. Branson, J. D. Wegner, D. Hall, N. Lang, K. Schindler, and P. Perona, "From Google Maps to a fine-grained catalog of street trees," *ISPRS Journal of Photogrammetry and Remote Sensing*, vol. 135, pp. 13-30, 2018.
[224] Z. Wang, C. Fan, and M. Xian, "Application and evaluation of a deep learning architecture to urban tree canopy mapping," *Remote Sensing*, vol. 13, no. 9, p. 1749, 2021.
[225] G. B. Martins *et al.*, "Deep learning-based tree species mapping in a highly diverse tropical urban setting," *Urban Forestry & Urban Greening*, vol. 64, p. 127241, 2021.
[226] K. Choi *et al.*, "An automatic approach for tree species detection and profile estimation of urban street trees using deep learning and Google street view images," *Isprs Journal of Photogrammetry and Remote Sensing*, vol. 190, pp. 165-180, 2022.
[227] W. Hu, Y. Huang, L. Wei, F. Zhang, and H. Li, "Deep convolutional neural networks for hyperspectral image classification," *Journal of Sensors*, vol. 2015, pp. 1-12, 2015.
[228] Ø. D. Trier *et al.*, "Tree species classification in Norway from airborne hyperspectral and airborne laser scanning data," *European Journal of Remote Sensing*, vol. 51, no. 1, pp. 336-351, 2018.
[229] P. Wang, E. Fan, and P. Wang, "Comparative analysis of image classification algorithms based on traditional machine learning and deep learning," *Pattern Recognition Letters*, vol. 141, pp. 61-67, 2021.

[230] Q. Yue and C. Ma, "Deep learning for hyperspectral data classification through exponential momentum deep convolution neural networks," *Journal of Sensors,* vol. 2016, 2016.
[231] L. Mou, P. Ghamisi, and X. X. Zhu, "Deep recurrent neural networks for hyperspectral image classification," *IEEE Transactions on Geoscience and Remote Sensing,* vol. 55, no. 7, pp. 3639-3655, 2017.
[232] K. Simonyan and A. Zisserman, "Very deep convolutional networks for large-scale image recognition," *arXiv preprint arXiv:1409.1556,* 2014.
[233] C. Szegedy *et al.*, "Going deeper with convolutions," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2015, pp. 1-9.
[234] R. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2014, pp. 580-587.
[235] S. Bahrampour, N. Ramakrishnan, L. Schott, and M. Shah, "Comparative study of deep learning software frameworks," *arXiv preprint arXiv:1511.06435,* 2015.
[236] L. Xin, Z. Tang, W. Gai, and H. Liu, "Vision-based autonomous landing for the uav: A review," *Aerospace,* vol. 9, no. 11, p. 634, 2022.
[237] P. McGlone, S. J. Richardson, and G. J. Jordan, "Comparative biogeography of New Zealand trees: species richness, height, leaf traits and range sizes," 2010.
[238] K. De and V. Masilamani, "Image sharpness measure for blurred images in frequency domain," *Procedia Engineering,* vol. 64, pp. 149-158, 2013.
[239] R. Bansal, G. Raj, and T. Choudhury, "Blur image detection using Laplacian operator and Open-CV," in *2016 International Conference System Modeling & Advancement in Research Trends (SMART)*, 2016, pp. 63-67: IEEE.
[240] I. Aizenberg, T. Bregin, C. Butakoff, V. Karnaukhov, N. Merzlyakov, and O. Milukova, "Type of blur and blur parameters identification using neural network and its application to image restoration," in *Artificial Neural Networks—ICANN 2002: International Conference Madrid, Spain, August 28–30, 2002 Proceedings 12*, 2002, pp. 1231-1236: Springer.
[241] J.-F. Cai, H. Ji, C. Liu, and Z. Shen, "Framelet-based blind motion deblurring from a single image," *IEEE Transactions on Image Processing,* vol. 21, no. 2, pp. 562-572, 2011.
[242] I. Abdel-Qader, O. Abudayyeh, and M. E. Kelly, "Analysis of edge-detection techniques for crack identification in bridges," *Journal of Computing in Civil Engineering,* vol. 17, no. 4, pp. 255-263, 2003.
[243] M. Zhao, D. Li, Z. Shi, S. Du, P. Li, and J. Hu, "Blur feature extraction plus automatic KNN matting: a novel two stage blur region detection method for local motion blurred images," *IEEE Access,* vol. 7, pp. 181142-181151, 2019.
[244] V. S. Thakur, K. Thakur, and G. Sinha, "Image Enhancement in Spatial domain by using LOG Operator," *International Journal of Applied Engineering Research,* vol. 3, no. 4, pp. 539-546, 2008.
[245] Z. Wang and A. C. Bovik, "Mean squared error: Love it or leave it? A new look at signal fidelity measures," *IEEE signal processing magazine,* vol. 26, no. 1, pp. 98-117, 2009.
[246] X. Ye, Y. Wang, X.-Y. Tang, H. Ji, B. Wang, and Z. Huang, "On the design of a new simulated inductor using a contactless electrical tomography system as an example," *Sensors,* vol. 19, no. 11, p. 2463, 2019.
[247] A. Banharnsakun, "Feature point matching based on ABC-NCC algorithm," *Evolving Systems,* vol. 9, no. 1, pp. 71-80, 2018.
[248] D.-H. Duong, C.-S. Chen, and L.-C. Chen, "Absolute Depth Measurement Using Multiphase Normalized Cross-Correlation for Precise Optical Profilometry," *Sensors,* vol. 19, no. 21, p. 4683, 2019.
[249] M. Annaby, Y. Fouda, and M. A. Rushdi, "Improved normalized cross-correlation for defect detection in printed-circuit boards," *IEEE Transactions on Semiconductor Manufacturing,* vol. 32, no. 2, pp. 199-211, 2019.
[250] F. Zhao, Q. Huang, and W. Gao, "Image matching by normalized cross-correlation," in *2006 IEEE international conference on acoustics speech and signal processing proceedings*, 2006, vol. 2, pp. II-II: IEEE.
[251] S. Zerr, S. Siersdorfer, J. Hare, and E. Demidova, "Privacy-aware image classification and search," in *Proceedings of the 35th international ACM SIGIR conference on Research and development in information retrieval*, 2012, pp. 35-44.
[252] A. C. Squicciarini, S. Sundareswaran, D. Lin, and J. Wede, "A3p: adaptive policy prediction for shared images over popular content sharing sites," in *Proceedings of the 22nd ACM conference on Hypertext and hypermedia*, 2011, pp. 261-270.

[253] S. Ahern, D. Eckles, N. S. Good, S. King, M. Naaman, and R. Nair, "Over-exposed? Privacy patterns and considerations in online and mobile photo sharing," in *Proceedings of the SIGCHI conference on Human factors in computing systems*, 2007, pp. 357-366.
[254] R. Girshick, "Fast r-cnn. arXiv 2015," *arXiv preprint arXiv:1504.08083*.
[255] S. Ren, K. He, R. Girshick, and J. Sun, "Faster r-cnn: Towards real-time object detection with region proposal networks," *Advances in neural information processing systems*, vol. 28, 2015.
[256] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *arXiv preprint arXiv:2004.10934*, 2020.
[257] J. Redmon and A. Farhadi, "Yolov3: An incremental improvement," *arXiv preprint arXiv:1804.02767*, 2018.
[258] J. Redmon and A. Farhadi, "YOLO9000: better, faster, stronger," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2017, pp. 7263-7271.
[259] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You only look once: Unified, real-time object detection," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 779-788.
[260] T. Ahmad, M. Cavazza, Y. Matsuo, and H. Prendinger, "Detecting human actions in drone images using YOLOv5 and stochastic gradient boosting," *Sensors*, vol. 22, no. 18, p. 7020, 2022.
[261] L. Cao, X. Zheng, and L. Fang, "The Semantic Segmentation of Standing Tree Images Based on the Yolo V7 Deep Learning Algorithm," *Electronics*, vol. 12, no. 4, p. 929, 2023.
[262] I. Colkesen *et al.*, "Deep learning based poplar tree detection and counting using multispectral UAV images," *Advanced Engineering Days (AED)*, vol. 6, pp. 64-67, 2023.
[263] Y. Zhang *et al.*, "CURI-YOLOv7: A Lightweight YOLOv7tiny Target Detector for Citrus Trees from UAV Remote Sensing Imagery Based on Embedded Device," *Remote Sensing*, vol. 15, no. 19, p. 4647, 2023.
[264] D. Q. da Silva, F. N. dos Santos, V. Filipe, A. J. Sousa, and P. M. Oliveira, "Edge AI-based tree trunk detection for forestry monitoring robotics," *Robotics*, vol. 11, no. 6, p. 136, 2022.
[265] P. Hofinger, H.-J. Klemmt, S. Ecke, S. Rogg, and J. Dempewolf, "Application of YOLOv5 for Point Label Based Object Detection of Black Pine Trees with Vitality Losses in UAV Data," *Remote Sensing*, vol. 15, no. 8, p. 1964, 2023.
[266] T. Jintasuttisak, E. Edirisinghe, and A. Elbattay, "Deep neural network based date palm tree detection in drone imagery," *Computers and Electronics in Agriculture*, vol. 192, p. 106560, 2022.
[267] Y. Zheng and G. Wu, "YOLOv4-Lite–Based Urban plantation tree detection and positioning with high-resolution remote sensing imagery," *Frontiers in Environmental Science*, vol. 9, p. 641, 2022.
[268] Y. Zhu, J. Zhou, Y. Yang, L. Liu, F. Liu, and W. Kong, "Rapid Target Detection of Fruit Trees Using UAV Imaging and Improved Light YOLOv4 Algorithm," *Remote Sensing*, vol. 14, no. 17, p. 4324, 2022.
[269] M. Xia, W. Li, H. Fu, L. Yu, R. Dong, and J. Zheng, "Fast and robust detection of oil palm trees using high-resolution remote sensing images," in *Automatic Target Recognition XXIX*, 2019, vol. 10988, pp. 65-73: SPIE.
[270] R. Aburasain, E. Edirisinghe, and A. Albatay, "Palm tree detection in drone images using deep convolutional neural networks: Investigating the effective use of YOLO v3," in *Conference on Multimedia, Interaction, Design and Innovation*, 2020, pp. 21-36: Springer.
[271] A. Safonova, Y. Hamad, A. Alekhina, and D. Kaplun, "Detection of norway spruce trees (Picea Abies) infested by bark beetle in UAV images using YOLOs architectures," *IEEE Access*, vol. 10, pp. 10384-10392, 2022.
[272] K. Itakura and F. Hosoi, "Automatic tree detection from three-dimensional images reconstructed from 360 spherical camera using YOLO v2," *Remote Sensing*, vol. 12, no. 6, p. 988, 2020.
[273] P. N. Chowdhury, P. Shivakumara, L. Nandanwar, F. Samiron, U. Pal, and T. Lu, "Oil palm tree counting in drone images," *Pattern Recognition Letters*, vol. 153, pp. 1-9, 2022.
[274] R. Xu, H. Lin, K. Lu, L. Cao, and Y. Liu, "A forest fire detection system based on ensemble learning," *Forests*, vol. 12, no. 2, p. 217, 2021.
[275] M. Everingham, S. A. Eslami, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes challenge: A retrospective," *International journal of computer vision*, vol. 111, pp. 98-136, 2015.
[276] T.-Y. Lin *et al.*, "Microsoft coco: Common objects in context," in *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, 2014, pp. 740-755: Springer.

[277] C.-Y. Wang, H.-Y. M. Liao, Y.-H. Wu, P.-Y. Chen, J.-W. Hsieh, and I.-H. Yeh, "CSPNet: A new backbone that can enhance learning capability of CNN," in *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition workshops*, 2020, pp. 390-391.

[278] K. Wang, J. H. Liew, Y. Zou, D. Zhou, and J. Feng, "Panet: Few-shot image semantic segmentation with prototype alignment," in *proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 9197-9206.

[279] G. Wang, Y. Chen, P. An, H. Hong, J. Hu, and T. Huang, "UAV-YOLOv8: A Small-Object-Detection Model Based on Improved YOLOv8 for UAV Aerial Photography Scenarios," *Sensors*, vol. 23, no. 16, p. 7190, 2023.

[280] J. Terven and D. Cordova-Esparza, "A comprehensive review of YOLO: From YOLOv1 to YOLOv8 and beyond," *arXiv preprint arXiv:2304.00501*, 2023.

[281] S. Liu, L. Qi, H. Qin, J. Shi, and J. Jia, "Path aggregation network for instance segmentation," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2018, pp. 8759-8768.

[282] X. Li *et al.*, "Generalized focal loss: Learning qualified and distributed bounding boxes for dense object detection," *Advances in Neural Information Processing Systems*, vol. 33, pp. 21002-21012, 2020.

[283] Z. Zheng, P. Wang, W. Liu, J. Li, R. Ye, and D. Ren, "Distance-IoU loss: Faster and better learning for bounding box regression," in *Proceedings of the AAAI conference on artificial intelligence*, 2020, vol. 34, no. 07, pp. 12993-13000.

[284] C. Feng, Y. Zhong, Y. Gao, M. R. Scott, and W. Huang, "Tood: Task-aligned one-stage object detection," in *2021 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2021, pp. 3490-3499: IEEE Computer Society.

[285] M. Xu, S. Yoon, A. Fuentes, and D. S. Park, "A comprehensive survey of image augmentation techniques for deep learning," *Pattern Recognition*, p. 109347, 2023.

[286] M. Xu, S. Yoon, A. Fuentes, J. Yang, and D. S. Park, "Style-consistent image translation: A novel data augmentation paradigm to improve plant disease recognition," *Frontiers in Plant Science*, vol. 12, p. 3361, 2022.

[287] L. Taylor and G. Nitschke, "Improving deep learning with generic data augmentation," in *2018 IEEE symposium series on computational intelligence (SSCI)*, 2018, pp. 1542-1547: IEEE.

[288] S. C. Wong, A. Gatt, V. Stamatescu, and M. D. McDonnell, "Understanding data augmentation for classification: when to warp?," in *2016 international conference on digital image computing: techniques and applications (DICTA)*, 2016, pp. 1-6: IEEE.

[289] C. Shorten and T. M. Khoshgoftaar, "A survey on image data augmentation for deep learning," *Journal of big data*, vol. 6, no. 1, pp. 1-48, 2019.

[290] J. Kaur and W. Singh, "Tools, techniques, datasets and application areas for object detection in an image: a review," *Multimedia Tools and Applications*, vol. 81, no. 27, pp. 38297-38351, 2022.

[291] B. C. Russell, A. Torralba, K. P. Murphy, and W. T. Freeman, "LabelMe: a database and web-based tool for image annotation," *International journal of computer vision*, vol. 77, pp. 157-173, 2008.

[292] N. Manikandan and K. Ganesan, "Deep learning based automatic video annotation tool for self-driving car," *arXiv preprint arXiv:1904.12618*, 2019.

[293] S. Shao *et al.*, "Objects365: A large-scale, high-quality dataset for object detection," in *Proceedings of the IEEE/CVF international conference on computer vision*, 2019, pp. 8430-8439.

[294] M. Dobrovolny, J. Benes, O. Krejcar, and A. Selamat, "Sperm-cell Detection Using YOLOv5 Architecture," in *International Work-Conference on Bioinformatics and Biomedical Engineering*, 2022, pp. 319-330: Springer.

[295] U. o. Waikato. (2021). *ADAMS*. Available: https://adams.cms.waikato.ac.nz/

[296] A. Betti, B. Michelozzi, A. Bracci, and A. Masini, "Real-Time target detection in maritime scenarios based on YOLOv3 model. ," *arXiv preprint arXiv:2003.00800*, 2020.

[297] S. Hahn, C. S. Morris, D. J. Bertges, and S. Wshah, "Deep learning for recognition of endoleak after endovascular abdominal aortic aneurysm repair," in *2019 IEEE 16th International Symposium on Biomedical Imaging (ISBI 2019)*, 2019, pp. 759-763: IEEE.

[298] R. Nakasi, E. Mwebaze, A. Zawedde, J. Tusubira, B. Akera, and G. Maiga, "A new approach for microscopic diagnosis of malaria parasites in thick blood smears using pre-trained deep learning models," *SN Applied Sciences*, vol. 2, pp. 1-7, 2020.

[299] A. Roihan, M. Hasanudin, and E. Sunandar, "Evaluation methods of bird repellent devices in optimizing crop production in agriculture," in *Journal of Physics: Conference Series*, 2020, vol. 1477, no. 3, p. 032012: IOP Publishing.

[300] S. Tabassum, S. Ullah, N. H. Al-Nur, and S. Shatabda, "Poribohon-BD: Bangladeshi local vehicle image dataset with annotation for classification," *Data in brief*, vol. 33, 2020.
[301] C.-W. Yu, Y.-L. Chen, K.-F. Lee, C.-H. Chen, and C.-Y. Hsiao, "Efficient intelligent automatic image annotation method based on machine learning techniques," in *2019 IEEE International Conference on Consumer Electronics-Taiwan (ICCE-TW)*, 2019, pp. 1-2: IEEE.
[302] M. P. Schilling, S. Schmelzer, L. Klinger, and M. Reischl, "KaIDA: a modular tool for assisting image annotation in deep learning," *Journal of Integrative Bioinformatics*, vol. 19, no. 4, p. 20220018, 2022.
[303] M. Aljabri, M. AlAmir, M. AlGhamdi, M. Abdel-Mottaleb, and F. Collado-Mesa, "Towards a better understanding of annotation tools for medical imaging: a survey," *Multimedia tools and applications*, vol. 81, no. 18, pp. 25877-25911, 2022.
[304] Roboflow. (2023). Available: https://roboflow.com/
[305] S. K. Shandilya, A. Srivastav, K. Yemets, A. Datta, and A. K. Nagar, "YOLO-based segmented dataset for drone vs. bird detection for deep and machine learning algorithms," *Data in Brief*, vol. 50, p. 109355, 2023.
[306] M. Tkachenko, M. Malyuk, N. Shevchenko, A. Holmanyuk, and N. Liubimov, "Label studio: Data labeling software," *Open source software available from https://github. com/heartexlabs/label-studio*, 2020.
[307] B. Di Martino, F. Marulli, M. Graziano, and P. Lupi, "PrettyTags: an open-source tool for easy and customizable textual multilevel semantic annotations," in *Complex, Intelligent and Software Intensive Systems: Proceedings of the 15th International Conference on Complex, Intelligent and Software Intensive Systems (CISIS-2021)*, 2021, pp. 636-645: Springer.
[308] A. Iosifidis and A. Tefas, *Deep learning for robot perception and cognition*. Academic Press, 2022.
[309] I. S. Isa, M. S. A. Rosli, U. K. Yusof, M. I. F. Maruzuki, and S. N. Sulaiman, "Optimizing the hyperparameter tuning of YOLOv5 for underwater detection," *IEEE Access*, vol. 10, pp. 52818-52831, 2022.
[310] S. Ioffe and C. Szegedy, "Batch normalization: Accelerating deep network training by reducing internal covariate shift," in *International conference on machine learning*, 2015, pp. 448-456: pmlr.
[311] I. Kandel and M. Castelli, "The effect of batch size on the generalizability of the convolutional neural networks on a histopathology dataset," *ICT express*, vol. 6, no. 4, pp. 312-315, 2020.
[312] E. M. Dogo, O. J. Afolabi, and B. Twala, "On the Relative Impact of Optimizers on Convolutional Neural Networks with Varying Depth and Width for Image Classification," *Applied Sciences*, vol. 12, no. 23, p. 11976, 2022.
[313] A. Kumar, S. Sarkar, and C. Pradhan, "Malaria disease detection using cnn technique with sgd, rmsprop and adam optimizers," *Deep learning techniques for biomedical and health informatics*, pp. 211-230, 2020.
[314] R. O. Ogundokun, R. Maskeliunas, S. Misra, and R. Damaševičius, "Improved CNN based on batch normalization and adam optimizer," in *International Conference on Computational Science and Its Applications*, 2022, pp. 593-604: Springer.
[315] Y. Tian, D. Su, S. Lauria, and X. Liu, "Recent advances on loss functions in deep learning for computer vision," *Neurocomputing*, vol. 497, pp. 129-158, 2022.
[316] S. Abdel-Salam and A. Rafea, "Performance study on extractive text summarization using BERT models," *Information*, vol. 13, no. 2, p. 67, 2022.
[317] A. N. Yumang, M. C. M. S. Juana, and R. L. C. Diloy, "Detection and classification of defective fresh excelsa beans using mask r-CNN algorithm," in *2022 14th International Conference on Computer and Automation Engineering (ICCAE)*, 2022, pp. 97-102: IEEE.
[318] G. Jocher *et al.*, "ultralytics/yolov5: v6. 2-yolov5 classification models, apple m1, reproducibility, clearml and deci. ai integrations," *Zenodo*, 2022.
[319] H. Tabrizchi, S. Parvizpour, and J. Razmara, "An improved VGG model for skin cancer detection," *Neural Processing Letters*, vol. 55, no. 4, pp. 3715-3732, 2023.
[320] P. Kanani and M. Padole, "Deep learning to detect skin cancer using google colab," *International Journal of Engineering and Advanced Technology Regular Issue*, vol. 8, no. 6, pp. 2176-2183, 2019.
[321] L. Shao, F. Zhu, and X. Li, "Transfer learning for visual categorization: A survey," *IEEE transactions on neural networks and learning systems*, vol. 26, no. 5, pp. 1019-1034, 2014.
[322] N. Al-Qubaydhi *et al.*, "Detection of unauthorized unmanned aerial vehicles using YOLOv5 and transfer learning," *Electronics*, vol. 11, no. 17, p. 2669, 2022.
[323] L. A. J. Abel, T. C. N. Oconer, and J. C. D. Cruz, "Realtime object detection of pantry objects using yolov5 transfer learning in varying lighting and orientation," in *2022 2nd International Conference on Innovative Research in Applied Science, Engineering and Technology (IRASET)*, 2022, pp. 1-7: IEEE.
[324] N. Kwak and D. Kim, "A study on Detecting the Safety helmet wearing using YOLOv5-S model and transfer learning," *International Journal of Advanced Culture Technology*, vol. 10, no. 1, pp. 302-309, 2022.

[325] J. Wen-ping and J. Zhen-cun, "Research on early fire detection of Yolo V5 based on multiple transfer learning," *Fire Science and Technology*, vol. 40, no. 1, p. 109, 2021.
[326] W. Liu, K. Quijano, and M. M. Crawford, "YOLOv5-Tassel: Detecting tassels in RGB UAV imagery with improved YOLOv5 based on transfer learning," *IEEE Journal of Selected Topics in Applied Earth Observations and Remote Sensing*, vol. 15, pp. 8085-8094, 2022.
[327] Q. Huang, Y. Zhou, T. Yang, K. Yang, L. Cao, and Y. Xia, "A Lightweight Transfer Learning Model with Pruned and Distilled YOLOv5s to Identify Arc Magnet Surface Defects," *Applied Sciences*, vol. 13, no. 4, p. 2078, 2023.
[328] H. Song and Y. Hu, "Detecting emerging frost crystals on leaf surfaces using microscopic imaging," *Computers and Electronics in Agriculture*, vol. 210, p. 107931, 2023.
[329] W. Li, X. Li, W. Liu, Z. Liu, J. Jia, and J. Li, "X-ray Prohibited Items Recognition Based on Improved YOLOv5," in *International Conference on Web Information Systems and Applications*, 2023, pp. 26-38: Springer.
[330] A. Cardellicchio *et al.*, "Detection of tomato plant phenotyping traits using YOLOv5-based single stage detectors," *Computers and Electronics in Agriculture*, vol. 207, p. 107757, 2023.
[331] D. Shao, Z. He, H. Fan, and K. Sun, "Detection of Cattle Key Parts Based on the Improved Yolov5 Algorithm," *Agriculture*, vol. 13, no. 6, p. 1110, 2023.
[332] A. Efraimsson and E. Lemón, "Railway Fastener Fault Detection using YOLOv5," ed, 2022.
[333] C. Liu, K. Wang, H. Lu, Z. Cao, and Z. Zhang, "Robust Object Detection with Inaccurate Bounding Boxes," in *European Conference on Computer Vision*, 2022, pp. 53-69: Springer.
[334] C. Cao, D. Chicco, and M. M. Hoffman, "The MCC-F1 curve: a performance evaluation technique for binary classification," *arXiv preprint arXiv:2006.11278*, 2020.
[335] R. Padilla, W. L. Passos, T. L. Dias, S. L. Netto, and E. A. Da Silva, "A comparative analysis of object detection metrics with a companion open-source toolkit," *Electronics*, vol. 10, no. 3, p. 279, 2021.
[336] D. A. Q. Palma, "Enhancing Indoor Human Detection: A Comprehensive Study of YOLOv5 Algorithm with Thermal Imagery," 2023.
[337] J. Terven and D. Cordova-Esparza, "A comprehensive review of YOLO: From YOLOv1 and beyond. arXiv 2023," *arXiv preprint arXiv:2304.00501*.
[338] Z. Zou, K. Chen, Z. Shi, Y. Guo, and J. Ye, "Object detection in 20 years: A survey," *Proceedings of the IEEE*, 2023.
[339] H. Tao *et al.*, "Deep learning-based dead pine tree detection from unmanned aerial vehicle images," *International Journal of Remote Sensing*, vol. 41, no. 21, pp. 8238-8255, 2020.
[340] B. G. Weinstein, S. Marconi, M. Aubry-Kientz, G. Vincent, H. Senyondo, and E. P. White, "DeepForest: A Python package for RGB deep learning tree crown delineation," *Methods in Ecology and Evolution*, vol. 11, no. 12, pp. 1743-1751, 2020.
[341] Z. Hao, C. J. Post, E. A. Mikhailova, L. Lin, J. Liu, and K. Yu, "How does sample labeling and distribution affect the accuracy and efficiency of a deep learning model for individual tree-crown detection and delineation," *Remote Sensing*, vol. 14, no. 7, p. 1561, 2022.
[342] Z. Roslan, Z. Awang, M. N. Husen, R. Ismail, and R. Hamzah, "Deep learning for tree crown detection in tropical forest," in *2020 14th International Conference on Ubiquitous Information Management and Communication (IMCOM)*, 2020, pp. 1-7: IEEE.
[343] C. Badgujar, D. Flippo, S. Gunturu, and C. Baldwin, "Tree trunk detection of eastern red cedar in rangeland environment with deep learning technique," *Croatian Journal of Forest Engineering: Journal for Theory and Application of Forestry Engineering*, vol. 44, no. 2, pp. 357-368, 2023.
[344] S. Natesan, C. Armenakis, and U. Vepakomma, "Individual tree species identification using Dense Convolutional Network (DenseNet) on multitemporal RGB images from UAV," *Journal of Unmanned Vehicle Systems*, vol. 8, no. 4, pp. 310-333, 2020.
[345] L. Wang *et al.*, "Early detection of encroaching woody juniperus virginiana and its classification in multi-species forest using UAS imagery and semantic segmentation algorithms," *Remote Sensing*, vol. 13, no. 10, p. 1975, 2021.
[346] C. Zhang *et al.*, "Multi-species individual tree segmentation and identification based on improved mask R-CNN and UAV imagery in mixed forests," *Remote Sensing*, vol. 14, no. 4, p. 874, 2022.
[347] M. Beloiu, L. Heinzmann, N. Rehush, A. Gessler, and V. C. Griess, "Individual Tree-Crown Detection and Species Identification in Heterogeneous Forests Using Aerial RGB Imagery and Deep Learning," *Remote Sensing*, vol. 15, no. 5, p. 1463, 2023.
[348] E. Trieschman, "Transfer learning models for tree species detection in agroforestry."

[349] D. Liu, Y. Jiang, R. Wang, and Y. Lu, "Establishing a citywide street tree inventory with street view images and computer vision techniques," *Computers, Environment and Urban Systems*, vol. 100, p. 101924, 2023.
[350] H. Jemaa, W. Bouachir, B. Leblon, A. LaRocque, A. Haddadi, and N. Bouguila, "UAV-Based Computer Vision System for Orchard Apple Tree Detection and Health Assessment," *Remote Sensing*, vol. 15, no. 14, p. 3558, 2023.
[351] Y. Gan, Q. Wang, and A. Iio, "Tree Crown Detection and Delineation in a Temperate Deciduous Forest from UAV RGB Imagery Using Deep Learning Approaches: Effects of Spatial Resolution and Species Characteristics," *Remote Sensing*, vol. 15, no. 3, p. 778, 2023.
[352] M. Culman, S. Delalieux, and K. Van Tricht, "Individual palm tree detection using deep learning on RGB imagery to support tree inventory," *Remote Sensing*, vol. 12, no. 21, p. 3476, 2020.
[353] M. M. D. Oghaz, L. B. Saheer, and J. Zarrin, "Urban Tree Detection and Species Classification Using Aerial Imagery," in *Science and Information Conference*, 2022, pp. 469-483: Springer.