1290

UNIVERSIDADE Ð
COIMBRA

Dinis Duarte Costa

Intelligent Pest Management Framework

Dissertation in the context of the Master in Informatics Engineering, specialisation in Intelligent Systems, advised by Professor Catarina Helena Branco Simões da Silva, Professor Joana Madeira Martins Costa and Professor Bernardete Martins Ribeiro and presented to the Department of Informatics Engineering of the Faculty of Sciences and Technology of the University of Coimbra.

September 2023

DEPARTAMENTO DE
ENGENHARIA INFORMÁTICA
FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE Ð
COIMBRA

Dinis Duarte Costa

Intelligent Pest Management Framework

Dissertation in the context of the Master in Informatics Engineering, specialisation in Intelligent Systems, advised by Professor Catarina Helena Branco Simões da Silva, Professor Joana Madeira Martins Costa and Professor Bernardete Martins Ribeiro and presented to the Department of Informatics Engineering of the Faculty of Sciences and Technology of the University of Coimbra.

September 2023

DEPARTAMENTO DE
ENGENHARIA INFORMÁTICA
FACULDADE DE
CIÊNCIAS E TECNOLOGIA
UNIVERSIDADE D
COIMBRA

Dinis Duarte Costa

Sistema Inteligente para a Gestão de Pragas em Plantas

Dissertação no âmbito do Mestrado em Engenharia Informática, especialização em Sistemas Inteligentes, orientada pela Professora Doutora Catarina Helena Branco Simões da Silva, Professora Doutora Joana Madeira Martins Costa e Professora Doutora Bernardete Martins Ribeiro e apresentada ao Departamento de Engenharia Informática da Faculdade de Ciências e Tecnologia da Universidade de Coimbra.

Setembro 2023

Acknowledgements

This work was supported by project PEGADA 4.0 (PRR-C05-i03-000099), financed by the PPR - Plano de Recuperação e Resiliência and by national funds through FCT, within the scope of the project CISUC (UID/CEC/00326/2020).

I would like to express my gratitude to my advisors, Professors Catarina, Joana, and Bernardete, for guiding and motivating me to delve into this research area.

I am truly grateful to my family, friends, rowing partners and coaches for helping me accurately balance my studies and rowing duties.

A special acknowledgement to Afonso, my brother and my primary rowing partner. His camaraderie, both on and off the water, has been invaluable.

I express a deep appreciation for Inês, who has been a constant pillar of support and a fundamental partner in every step of my journey.

A heartfelt thank you to my mother, for always being there, for always making me feel special, and because I know she is proud.

Abstract

Pest detection has become increasingly critical. As the global population grows and the demand for food rises, farming efficiency is a top priority for researchers in the field. Pests significantly affect crop yields, making their early detection crucial for optimising agricultural production. Traditional human-driven pest detection methods are error prone and modern intelligent techniques have shown superior results. However, these modern techniques are predominantly dependent on traps and their locations within plantations, indicating room for further refinement.

In this dissertation, a novel method is proposed for detecting whiteflies in their natural habitat, using object detection models. Such a method will enable the construction of a reliable tool for farmers, where they can trust and make their judgements on when to act regarding the presence of this pest.

Recognising the data deficiency in training models for such tasks, this dissertation presents a unique dataset. This dataset comprises images taken in an ideal greenhouse environment during the final stages of tomato plant cultivation when the whitefly infestation was at its peak.

Building these models involves using advanced machine learning techniques to enhance training efficiency. The application of Active Learning (AL), designed to choose the most relevant training data, demonstrated that the models do not need the entirety of the available data to achieve their best performance. The results show that the models trained with AL used 10% less data compared to those trained with randomly selected data.

In situations with limited data, providing models with the best quality data becomes critical. This study also examined the importance of data quality, which is closely related to annotation quality, on the performance of object detection models. By training a model using data of lesser annotation quality and then using it to produce better annotations, additional manpower was not necessary. The outcomes indicate that using improved annotations can increase the Mean Average Precision (mAP) score by 1.1 points.

When faced with a scarcity of data for training, Transfer Learning (TL), viewed as a non-meta-learning Few-Shot Learning (FSL) technique, offers a significant advantage to new models by using knowledge from pre-trained models. This research also explored the benefits of TL. The results suggest that making use of it can significantly improve the training process, increasing the mAP performance by an average of 24% and reducing the training time by 10%.

Keywords

Pest Detection, Object Detection, Active Learning, Transfer Learning, Data-Centric Approaches

Resumo

A deteção de pragas tem vindo a ganhar cada vez mais destaque. Com o crescimento da população mundial e o aumento da procura por alimentos, a eficiência agrícola é cada vez mais relevante. As pragas afetam significativamente os rendimentos das colheitas, tornando a sua deteção precoce essencial para otimizar as produções. Os métodos tradicionais de deteção de pragas são conduzidos por humanos, tornando-os propensos a erros. Já as técnicas inteligentes têm mostrado resultados superiores. Ainda assim, as técnicas modernas dependem predominantemente de armadilhas e das suas posições nas plantações, deixando espaço para melhorias.

Nesta dissertação, é proposto um novo método para detetar moscas brancas no seu habitat natural usando modelos de deteção de objetos. Este método permitirá construir uma ferramenta essencial para os agricultores, na qual podem confiar e usar para decidir quando agir perante a presença desta praga.

Devido à falta de dados para treinar modelos capazes de realizar tais tarefas, esta dissertação apresenta um novo e diferenciado conjunto de dados composto por imagens capturadas num ambiente de estufa ideal, durante as fases finais da cultura de tomate, no auge da infestação de mosca branca.

Construir estes modelos envolve a utilização de técnicas avançadas de aprendizagem computacional para melhorar a eficiência do treino. O uso de técnicas de aprendizagem ativa, desenhadas para escolher os dados de treino mais relevantes, demonstrou que os modelos não precisam da totalidade dos dados disponíveis para alcançar o seu melhor desempenho. Uma vez que com a utilização desta técnica os modelos usaram 10% menos dados em comparação com outros treinados usando dados selecionados aleatoriamente.

Em situações de escassez de dados, é ainda mais importante fornecer aos modelos informação de melhor qualidade possível. Este estudo também examinou a importância da qualidade dos dados, que está regularmente relacionada com a qualidade de anotação. Ao treinar um modelo usando dados de qualidade de anotação inferior e depois usar as suas previsões para produzir melhores anotações, não foram necessários recursos humanos adicionais. Os resultados indicam que o uso de anotações melhoradas pode aumentar a pontuação mAP em 1.1 pontos.

Perante a escassez de dados, as técnicas de transferência de conhecimento oferecem uma vantagem significativa aos novos modelos ao usar o conhecimento de modelos pré-treinados. Nesta dissertação também foram explorados os benefícios destas técnicas. Os resultados sugerem que a sua utilização pode melhorar significativamente o processo de treino, aumentando a performance mAP em média 24% e reduzindo o tempo de treino em 10%.

Palavras-Chave

Deteção de Pragas, Deteção de Objectos, Aprendizagem Ativa, Transferência de Conhecimento, Abordagens Centradas nos Dados

The contributions of this dissertation resulted in the following publications in international peer-reviewed conferences:

Conference papers:

- D. Costa, C. Silva, J. Costa and B. Ribeiro, *Smart Pest Detection in the Wild*, Experiment International Conference (exp.at'23), June 2023.
- D. Costa, C. Silva, J. Costa and B. Ribeiro, *Optimizing Object Detection Models via Active Learning*, Iberian Conference on Pattern Recognition and Image Analysis (IbPRIA 2023), pp. 82-93, June 2023.
- D. Costa, C. Silva, J. Costa and B. Ribeiro, *Enhancing Pest Detection Models through Improved Annotations*, Portuguese Conference on Artificial Intelligence (EPIA 2023), to be present in September 2023

Workshop papers:

- D. Costa, B. Cardoso, C. Silva, J. Costa and B. Ribeiro *Intelligent System for Automatic Whitefly Detection in Tomato Greenhouses*, Experiment International Conference (exp.at'23), June 2023 - **Best Demo Award**

Additionally, another research paper was submitted to a well reputed international conference:

- D. Costa, C. Silva, J. Costa and B. Ribeiro, *Improving Pest Detection via Transfer Learning*, Iberoamerican Congress on Pattern Recognition (CIARP 2023), submitted

Contents

Acronyms

AI Artificial Intelligence.

AL Active Learning.

AP Average Precision.

AUC Area Under the Curve.

CNN Convolutional Neural Network.

CSPNet Cross Stage Partial Network.

CV Computer Vision.

DL Deep Learning.

DNN Deep Neural Network.

Fast R-CNN Fast Region-based Convolutional Neural Network.

Faster RCNN Faster Region-based Convolutional Neural Network.

FSL Few-Shot Learning.

IoU Intersection over Union.

mAP Mean Average Precision.

ML Machine Learning.

MSE Mean Squared Error.

PANet Path Aggregation Network.

RCNN Region-based Convolutional Neural Network.

RMSE Root Mean Squared Error.

RoI Region of Interest.

RPCO Ratio of the Predicted Counted Objects.

RPN Region Proposal Network.

SPP Spatial Pyramid Pooling.

TL Transfer Learning.

YOLO You Only Look Once.

List of Figures

List of Tables

Chapter 1

Introduction

The main purpose of this work is to create a system that provides farmers with a reliable tool for pest detection, considering the significant impact that pests have on crop yields. Given the scarcity of data in this domain, the research also investigates cutting-edge methods that aim to improve the efficiency of the Machine Learning (ML) algorithms.

1.1 Motivation

Agriculture was the key to the beginning of the human civilisation process. It marked the beginning of a transition from a nomadic lifestyle to a settled one, driven by the need to live close to the lands they cultivated [Harari, 2013]. Today, agriculture not only provides the majority of the world's food, but also significantly contributes to the global economy. Furthermore, with the increasing demand for food due to population growth, the topic of agriculture remains highly relevant.

As the demand for food increases, the evolution of agricultural practises has led to the emergence of Smart Farming, often termed Agriculture 4.0. This progressive strategy seeks to revolutionise conventional agricultural methods by incorporating advanced technologies. The goal is to increase the productivity and efficiency of both crop cultivation and livestock management [Moysiadis et al., 2021].

In the context of plant production, various factors such as light exposure, soil quality, water availability, extreme weather events, diseases, and pests can influence yields. ML offers innovative solutions to tackle these challenges. By analysing data related to weather patterns and soil conditions, ML can help farmers determine optimal planting schedules, forecast crop outputs, and tailor irrigation strategies. A particularly significant application of ML is in pest detection, offering an upgrade from traditional methods that rely on specialists to manually identify and count pests within crops.

ML algorithms can be broadly categorised into supervised and unsupervised learning. Supervised learning algorithms require labelled data in advance. The labelling process can be time-consuming and labour-intensive, especially depending on the type of data to be labelled. This is particularly true for object detection models, which aim to identify and categorise objects within images, as they necessitate prior image annotations. Furthermore, Computer Vision (CV) algorithms, by using images for training, demand a considerable volume of data, translating to extensive computational power and time requirements.

To address the need for large amounts of data when training CV models, AL is employed. This approach seeks to enhance the efficiency of ML by minimising the data necessary for training, specifically choosing the most relevant data for initial training. This strategy not only improves the efficiency of ML but also reduces the effort involved in annotating dataset images.

However, AL techniques might be insufficient in cases of data scarcity. In such situations, it remains crucial to explore more efficient methods. Few-Shot Learning (FSL) represents a domain within ML that focuses on training models with a minimal number of samples, generalising the model's knowledge to perform new, unseen tasks. TL is categorised as FSL technique. It uses the insights from pre-trained models to boost its training process, providing new models without prior knowledge a head start in the learning journey.

Many researchers focus on enhancing model designs to achieve better performance. However, at times, improved performance can be achieved by providing models with higher-quality data, enabling a more effective learning process. This is the essence of a data-centric approach, which emphasises improving data quality over tweaking model designs.

In this dissertation, the objective is to construct a system capable of detecting pests through images in natural settings, such as identifying pests directly on plant leaves. Providing farmers with a reliable tool for pest management. Specifically, the focus is on the detection of the whitefly pest, which is cited as one of the most damaging pests in greenhouses [Nieuwenhuizen et al., 2018], on tomato leaves. To establish this system, a robust dataset, composed of real-world mirroring setups was developed. This motivated the exploration for efficient ML techniques for training, such as, AL, TL and measures to enhance the data quality and explore its effect on models' performance.

1.2 Research Goals

In this dissertation, the emphasis is placed on the development of an advanced system tailored for the agricultural community, especially farmers. This system aims to equip them with data-driven insights, offering guidance on the ideal times to address potential pest infestations. Beyond offering a pragmatic solution for agricultural pest management, there is an intrinsic motivation to explore the frontiers of cutting-edge technologies in Artificial Intelligence (AI), guaranteeing that the implemented system is both precise and resource-efficient.

To achieve these goals, the following objectives are outlined:

- Construct a dataset: Recognising the scarcity of data in the agricultural sector concerning pests, the main goal is to compile a dataset centred on whitefly-infested tomato leaves. Furthermore, there is a commitment to make this dataset available to the larger research community.
- Investigate the effectiveness of AL: Armed with the dataset, the next step is to evaluate the utility of AL through a structured set of comparative experiments.
- Analyse the influence of data quality: The objective in this phase is to explore how the quality of annotations can enhance the performance metrics of object detection algorithms.
- Explore the advantages of FSL: A significant goal is to study the benefits of FSL, especially in scenarios where TL plays a crucial role in the training of object detection models.

In addition, the models conceptualised in this research are designed with practical application in mind. Farmers, with a camera or a smartphone in hand, can take photos or shoot videos of their crops. Using the constructed models, not only can pests be identified but their counts can also be easily obtained. This functionality empowers farmers with immediate actionable data, allowing timely interventions in the agricultural field.

1.2.1 First Semesters

During the first semester, an emphasis was placed on understanding the broader landscape of smart farming through a detailed review of the literature, especially in the area of pest detection. Simultaneously, a comprehensive study of notable object detection models, particularly the You Only Look Once (YOLO) and Region-based Convolutional Neural Network (RCNN) families, was carried out. This phase was also marked by practical efforts: sourcing and collecting a wide range of images to establish a robust dataset that would serve as the base for subsequent investigations. As this dataset began to take shape, the intricate task of image annotation was initiated. Preliminary experimentation, leveraging the capabilities of the YOLOv5 model, provided initial insights into the potential pathways the research could take.

1.2.2 Second Semester

In the second semester, the emphasis was shifted towards experimental work, which required a deeper exploration of cutting-edge techniques in efficient AI ensured that the research was aligned with current advances. This time was marked by detailed experimentation, specifically targeting areas such as: AL which resulted in a contribution to an already published conference paper; the crucial role of data quality enhancement, which also resulted in a contribution to a conference paper, and the advantages offered by TL, which resulted in another submitted conference paper.

A comprehensive overview of activities across both semesters is illustrated in the Gantt chart depicted in Figure 1.1.

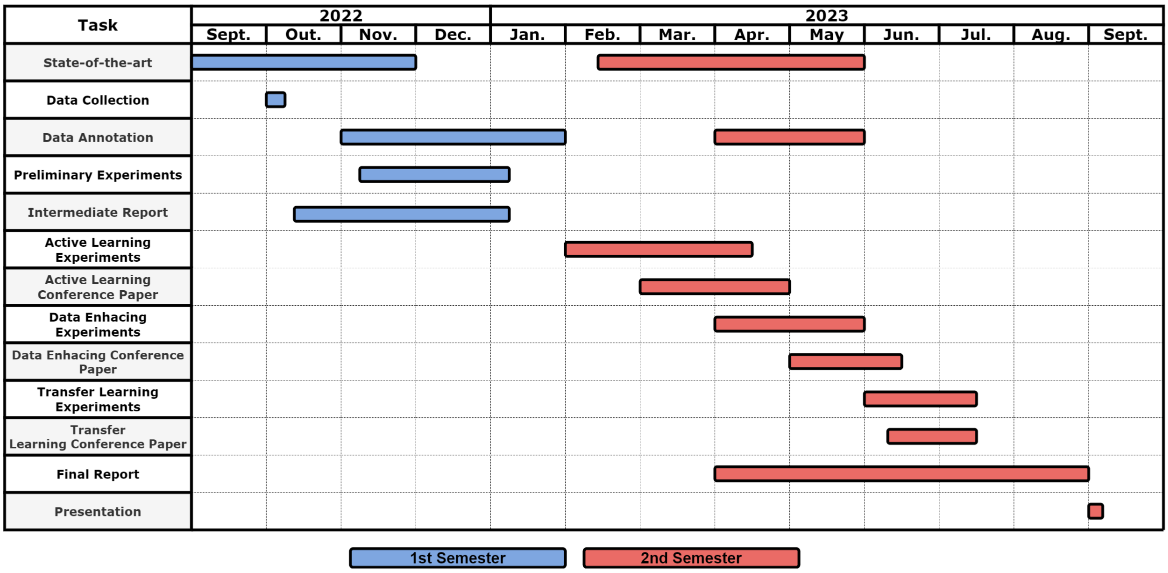


Figure 1.1: Gantt Diagram detailing the work plan for this dissertation, differentiating between both semesters.

1.3 Document Structure

This document is organised into six distinct chapters.

Chapter 1 serves as the introduction, presenting the motivation, laying out the objectives, and elucidating the overarching approach of this dissertation.

Chapter 2 provides a foundation by offering insight into pertinent model architectures. The emphasis is on the areas of object detection and classification, offering a thorough understanding of their operational mechanisms. This chapter also explores advanced techniques such as AL and FSL, and data-centric approaches, underscoring their role in enhancing AI efficiency.

Chapter 3 is dedicated to the detection of pests in plants, highlighting both traditional and intelligent methods.

Chapter 4 introduces the research methodology employed in this study. It meticulously describes the experiments conducted, the rationale behind them, and the performance metrics chosen for the evaluations, ensuring a clear understanding of the investigative process.

Chapter 5 provides an in-depth analysis of the results achieved. It not only examines the results, but also engages in a discussion of their implications, strengths, and potential areas of improvement.

The final chapter draws the conclusions, summarising the significant findings and contributions of this dissertation. It also looks ahead, suggesting potential avenues for future research based on the work already undertaken.

Chapter 2

Background

This chapter offers insights into pertinent model architectures in the field of object detection and the algorithms that support their construction.

Additionally, the chapter introduces cutting-edge techniques designed to enhance model accuracy. These methods focus on strategies that increase the efficiency of the training process for Machine Learning (ML) models.

2.1 Object Detection and Classification

Object detection and classification models play a crucial role in contemporary technology. These visual recognition systems have achieved exceptional results in their performance, as highlighted in [Zhou et al., 2017]. The primary goal of these models is to predict the locations of objects within an image and classify each object according to a specific class. The predicted location is represented by a bounding box, defined by the centre coordinates of the object, along with its width and height. Alongside the predicted bounding box, the models also output the predicted class and a confidence score representing the prediction probability.

In this section, the architectures of Convolutional Neural Network (CNN) are examined, as they currently serve as the foundation for object detection and classification.

The construction of object detection models began in the 1990s due to the absence of effective means for image representation [Zou et al., 2023]. The 2000s saw the proposal of more robust models [Dalal and Triggs, 2005; Felzenszwalb et al., 2008; Viola and Jones, 2001, 2004]. By 2014, the first models based on CNN were introduced. Object detection models are now typically categorised into one- or two-stage models. This section also outlines the significant model architectures in the field of object detection, emphasising Faster Region-based Convolutional Neural Network (Faster RCNN) and You Only Look Once (YOLO) version 5.

2.1.1 Object Annotation

Object annotation, often referred to as labelling, consists of attaching labels to specify the class, dimensions, and position of objects within an image or video. This procedure can be labour intensive, since it is performed by a knowledgeable human expert. Annotations enables ML models to recognise and classify objects. This meticulous labelling guides the model in data interpretation, allowing it to address real-world problems.

For the purpose of annotating objects in images, bounding boxes are predominantly used. These rectangles surround the object, with their detailed composition differing according to the selected format. For architectures like YOLO, annotations in the YOLO Darknet format are essential during training. Conversely, the Region-based Convolutional Neural Network (RCNN) lineage employs the PASCAL VOC[1] (Visual Object Classes) XML format.

2.1.2 Data Pre-Processing

In object detection models, data pre-processing plays a helpful role in enhancing model performance. A primary step in this process involves maintaining uniformity in the image dimensions by resizing them to specific standards. Several image pre-processing techniques exist, and [Pal and Sudeep, 2016] provides an in-depth look at methods specifically designed for CNN training. By integrating these techniques, models benefit from training on more consistent and information-rich datasets, leading to improved performance during the inference phase.

Pre-processing of data is a pivotal step to enhance model performance. Ensuring uniformity in image dimensions, by resizing them to standardised values, is essential. There exist several image pre-processing techniques, and in [Pal and Sudeep, 2016], the authors collate methods tailored for CNN training:

- **Mean Normalisation:** This approach aims at pixel-wise normalisation of the image. The normalisation is realised as:

  $$X' = X - \mu \tag{2.1}$$

  Here, X' represents the normalised data, X stands for the original data, and μ is the average value spanning all pixels of X.

- **Standardisation:** The data undergoes mean normalisation initially. Subsequently, standard deviation, computed pixel-wise, is used to divide the data:

  $$X' = \frac{X - \mu}{\sigma} \tag{2.2}$$

  In this case, σ signifies the standard deviation computed across every pixel of X.

[1] http://host.robots.ox.ac.uk/pascal/VOC/

- **Zero Component Analysis (ZCA):** This technique accentuates object edges, which the convolutional layers leverage to discern features via feature maps. Initially, data normalisation is performed using feature scaling:

  $$X' = \frac{X - \mu}{255} \tag{2.3}$$

  The data then undergoes mean normalisation as described in (2.1). Thereafter, the singular value decomposition (SVD) of the covariance matrix corresponding to the mean normalised data is computed. The final step is whitening, articulated as:

  $$X_{ZCA} = \mathbb{U} \times \operatorname{diag}\left(\frac{1}{\sqrt{\operatorname{diag}(S) + \epsilon}}\right) \times \mathbb{U}^T \times X' \tag{2.4}$$

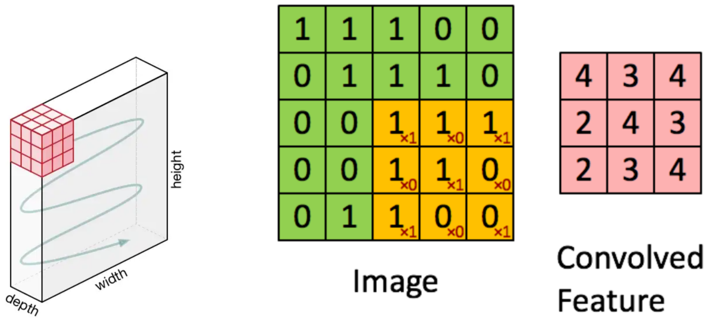
  Here, $\operatorname{diag}(a)$ represents the diagonal matrix of the given matrix a, $\mathbb{U}$ is the matrix of eigenvectors, and S is the matrix of eigenvalues resulting from the SVD of the covariance matrix. $\mathbb{U}^T$ is the transposed version of the eigenvector matrix $\mathbb{U}$, and ϵ is a whitening coefficient introduced to avoid numerical instability.

2.1.3 Convolutional Neural Networks

ML is a technology used by machines to make predictions, match content with specific users, power autonomous vehicles, or detect objects in images. ML traditional techniques are often limited in their ability to process natural data in their raw form [LeCun et al., 2015]. It can be challenging to design a feature extractor that is capable of producing suitable features that a model can detect or predict from the input data. Deep Learning (DL), on the other hand, includes a set of methods that can automatically discover the representations needed for detection or classification. In this dissertation, the emphasis is on Deep Neural Network (DNN), specifically CNNs for object detection.

CNNs are a type of artificial neural network commonly used to process data that have a grid-like topology, such as images. They can learn relevant features from the input data without requiring much pre-processing. Their first layers are the convolutional layers, which aim to reduce the data into a form that is easier to process. To do this, filters, also known as kernels, are applied to the image to extract its relevant features. These filters are small numerical matrices that during the convolution operation are slid through the input data according to stride length, multiplying their values at each position and summing the results. Resulting in a new feature map which has been filtered to highlight certain features from the input. Figure 2.1 represents an example of the convolution operation.

After the convolutional layers, there are typically pooling layers which are similar to the previous layers but with the aim of reducing the spatial size of the convolved feature maps, in order to decrease the computational power required to process the data. Figure 2.2 represents an example of the pooling operation. There are two types of pooling:



(a) Kernel Shift

(b) Kernel operation with $stride = 1$

Figure 2.1: Illustration of the convolutional operation, where a kernel (or filter) is applied to the input matrix to create a feature map. This localised processing uncovers spatial hierarchies and helps in identifying patterns, such as edges or textures, in the underlying data [Saha, 2018].

- **Max pooling**: returns the maximum value of the respective portion of the data.
- **Average pooling**: returns the average of the values of the respective portion of the data.

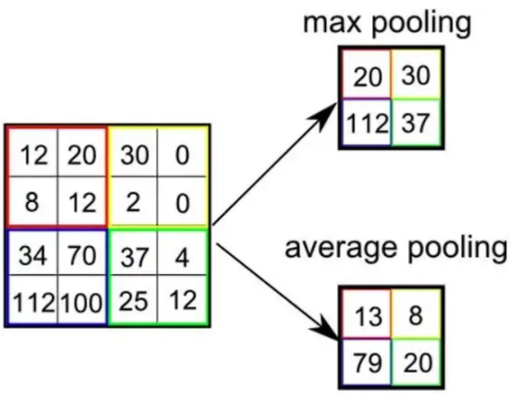


Figure 2.2: Illustration of both Max and Average Pooling operations within a CNN [Saha, 2018].

At the end of this process, the model is able to extract information from the features. Usually, one or more fully connected layers follow, which are layers in which each output of every node is connected to the input of every node in the next layer, as can be seen in Figure 2.3. These layers aim to learn non-linear combinations from feature maps [Saha, 2018]. The final layer uses the softmax technique, which consists of converting the raw predicted class scores into a probability distribution over the classes by generalising the logistic function:

$$\sigma(\vec{z})_i = \frac{e^{z_i}}{\sum_{j=1}^{K} e^{z_j}} \text{ , for } i = 1, ..., K, \tag{2.5}$$

where σ is the softmax function and z is an input vector with K real numbers.

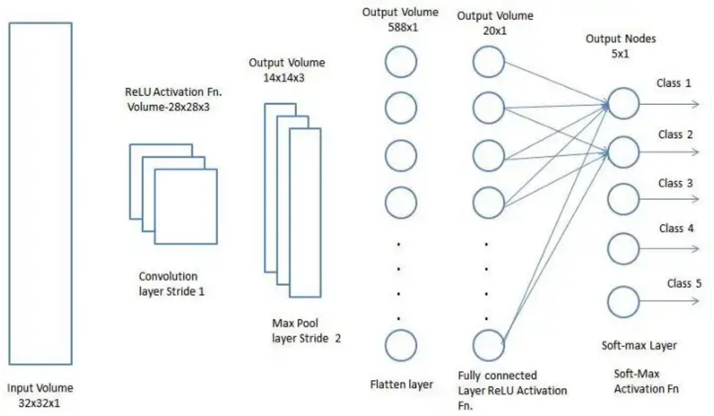


Figure 2.3: Diagram of a CNN, highlighting the convolutional layers for detecting patterns, the pooling layers for reducing data size, and the fully connected layers for making predictions [Saha, 2018].

2.1.4 Two Phase Models

Two-stage models process the images in two main parts: identifying regions where objects could be, and then classifying them. These types of models were first introduced in [Girshick et al., 2014], which presented the RCNN. This approach mainly works by identifying regions in an image where objects might be located. Then, each of the regions is processed by a CNN to extract the relevant features of the proposed regions. Afterward, the regions, which are assumed to be objects, are classified according to a specific class. Finally, along with the classification, the Bounding Box Regressor fine-tunes the locations of the objects.

Later, Fast Region-based Convolutional Neural Network (Fast R-CNN) was proposed in [Girshick, 2015], which, as the name indicates, is a faster version of RCNN and increased its Mean Average Precision (mAP) performance by 11.5% on the VOC07 dataset. Shortly after, Faster RCNN was proposed in [Ren et al., 2015] and is considered the state-of-the-art two-stage model. It can be divided into its two main parts [Abbas and Singh, 2018]:

- **Region Proposal Network (RPN)**: the RPN uses the features map generated by a CNN and is responsible for predicting the Region of Interest (RoI) which are the regions that have objects and their coordinates. Using a rectangular frame, a sliding window is passed across the feature map. For each window, a number of potential region proposals are created. Each proposal is characterised by a reference box, called an anchor box, which has two parameters: scale and aspect ratio. Figure 2.4 represents this process. Each anchor is given a positive or negative objectness score based on the Intersection over Union (IoU), given by:

$$Objectness_{score}(IoU) = \begin{cases} 1, & if\ IoU > 0.7 \\ -1, & if\ IoU < 0.3 \\ 0, & \text{otherwise} \end{cases} \tag{2.6}$$

  if the objectness score is positive, it means that the anchor likely contains an object. If it is negative, it means that the anchor does not contain an object. If it is 0, it means that the anchor is not relevant for training.

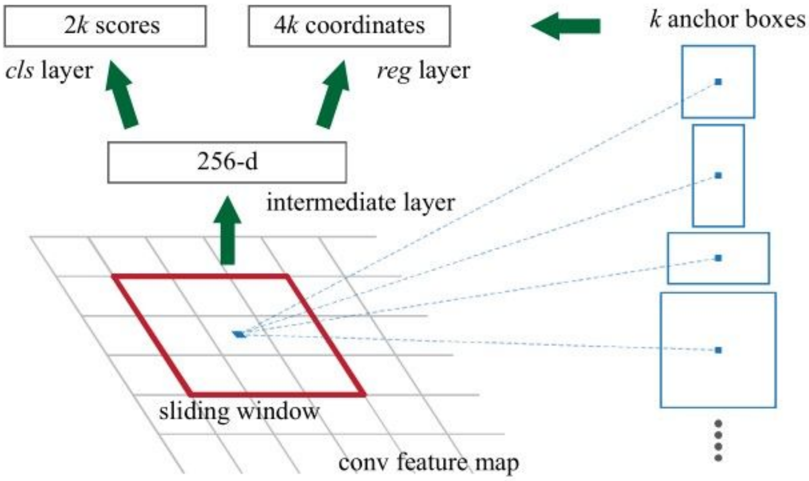


Figure 2.4: Illustrating the process used to identify Regions of Interest in an image: a sliding window is passed across the feature map [Ren et al., 2015].

- **Classification**: for classification, the first step is to resize all the regions to the same size. This is done using a max pooling RoI pooling layer, followed by two fully connected layers. The output of the last fully connected layer is split into two branches: one uses the softmax function to predict the object class, and the other uses another fully connected layer to predict the bounding box of the detected object, as shown in Figure 2.5.

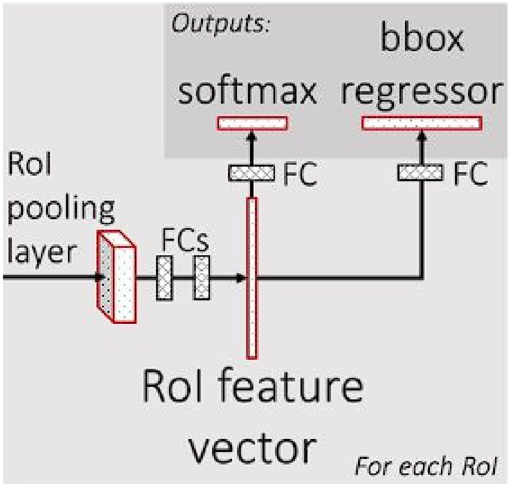


Figure 2.5: Illustration of the classification process in a Faster RCNN: the regions are resised through max pooling, features are extracted with fully connected layers, and final predictions are made for object class and bounding box [Girshick, 2015].

2.1.5 One-Stage Models

The inception of one-stage models began with the introduction of the YOLO family in [Redmon et al., 2016]. Over time, the YOLO approach has undergone significant evolution, with a series of advancements as evident in subsequent works [Bochkovskiy et al., 2020; Jocher et al., 2022, 2023; Li et al., 2022; Redmon and Farhadi, 2017, 2018; Wang et al., 2021, 2023]. One of the defining characteristics that distinguish these models from their two-stage counterparts is their ability to process images in a singular pass. This attribute, coupled with their design, enables them to deliver real-time detection capabilities. They are often celebrated for their lightweight architecture, rapid processing, and ease of deployment.

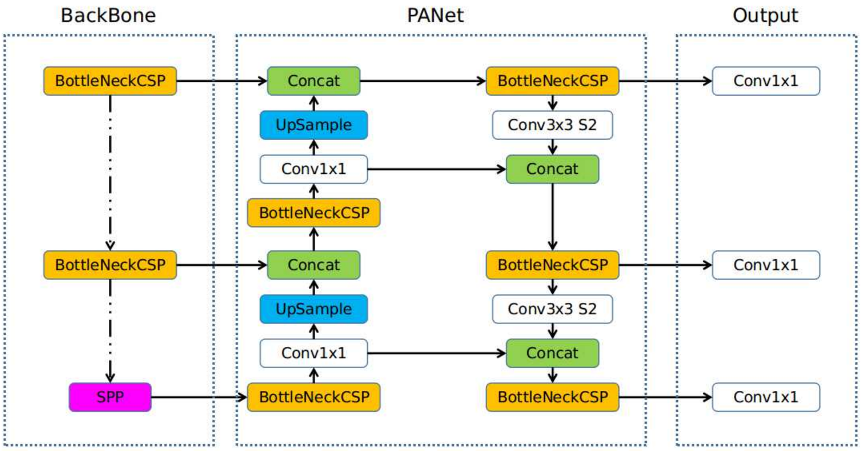


Figure 2.6: Overview of YOLOv5 architecture: The model's Backbone extracts relevant features from the input; the Neck processes the extracted features, providing insight to the next component; and the Head is responsible for making the predictions [Jocher et al., 2022].

In the context of this dissertation, the spotlight is on the YOLOv5 architecture, primarily due to the foundational work previously established by the research group. Introduced by Ultralytics in [Jocher et al., 2022], the YOLOv5 enjoyed a period as the state-of-the-art before the advent of its successor models. Structurally, the YOLOv5 architecture is organised into three distinct segments, as highlighted in Figure 2.6:

- **Backbone**: Composed of a Cross Stage Partial Network (CSPNet), the backbone is based on a CNN architecture where each layer is responsible for extracting features from the input. This network was pre-trained on the ImageNet dataset. The feature map of the base layer is divided into two parts by the CSPNet and then the two parts are merged through a cross-stage hierarchy, starting with low-level features such as edges and corners and gradually building up to higher-level features such as patterns and shapes. The backbone also includes a Spatial Pyramid Pooling (SPP) which is based on the idea of dividing the input data into a grid of cells and then applying pooling operations within each cell.
- **Neck**: uses a Path Aggregation Network (PANet) that processes the feature maps from the backbone network by passing them through multiple

paths, including a series of convolutional layers. The PANet is responsible for aggregating features from different scales and providing context for the detection head.

- **Head**: the model Head uses as input the features map outputted by the previous layers. It is composed from three convolution layers that predicts the location of the bounding boxes, the scores and the objects classes. YOLOv5 uses the SiLU function in the hidden layers of the convolution operations, while the sigmoid function is used in the output layers.

2.2 Deep Learning Approches

In this section, the focus is on cutting-edge ML techniques that aim to boost the models performance, by contributing to the effectiveness of the training process.

One such technique is Active Learning (AL), which strategically selects the most relevant data for training, reducing the large volume of data typically required for DL models. Instead of constantly refining model designs, an effective strategy focuses on improving data annotations. Moreover, Few-Shot Learning (FSL) aims at training models with limited examples. Within the scope of FSL, Transfer Learning (TL) stands out, offering new models a significant advantage by leveraging insights from previously well-trained models.

2.2.1 Active Learning

Active Learning is a learning technique that aims to train models with a small amount of labelled data by actively selecting the next set of data to be labelled [Haussmann et al., 2017]. The selection of appropriate data is critical to the success of this process.

Labelling can be a hard and time-consuming process [Costa et al., 2023] that may require expert knowledge. This often encourages the use of AL techniques.

Active Learning can be categorised as stream-based or pool-based [Hsu and Lin, 2015]. In a stream-based approach, data instances are continuously and sequentially presented to the learning model from a specific distribution. For each individual instance, the learner must make an immediate decision about whether or not to request its label. In a pool-based approach, the learning model is initially given two distinct sets of data: one that consists of unlabelled examples, denoted $D_u = \{x_1, x_2, ..., x_n\}$, and another that contains labelled examples, denoted $D_l = \{(x'_1, y'_1), (x'_2, y'_2), ..., (x'_m, y'_m)\}$, where y'_i is the label of x'_i. A learner trains a classifier f_0 using D_l and then, during iterations $t = 1, 2, ..., T$, the learner selects, based on f_{t-1}, an instance $x_j \in D_u$ and queries its label y_j. The pair of instances labelled (x_j, y_j) is then added to D_l, allowing the learner to train a new classifier f_t using the updated D_l. The goal is to maximise the average test accuracy across f_t where the test set is a separated pool of D_l.

The key to this process is selecting the appropriate data for training. Once the first classifier is trained, it is used to classify the unlabelled data, which provides information such as the probability of a classification given by the model to belong to a certain class. This information is fed into a score function, which is the key factor for classifying the unlabelled data from most relevant to least relevant, with the most relevant being the data that the model is most uncertain about, and therefore should be the first data to be queried for label. A batch of the most relevant data is then selected, labelled and added to the training set, where the model will be trained again and gain more knowledge. This process is repeated until a desirable performance is achieved. Figure 2.7 summarises the AL cycle.

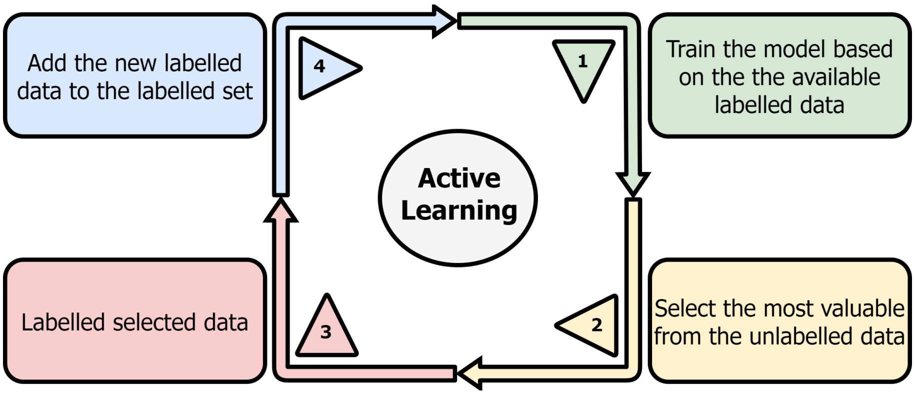

Add the new labelled data to the labelled set
4
1
Train the model based on the the available labelled data
Active Learning
Labelled selected data
3
2
Select the most valuable from the unlabelled data

Figure 2.7: Illustration of Active Learning cycle.

AL can improve the efficiency of ML by reducing the amount of data needed for training, resulting in a model with satisfactory performance. This process can save time in both labelling and training tasks and also requires less effort from experts who may be needed to label the data. The Computer Vision (CV) field, more precisely, object detection and classification models, require images where the objects to detect must be prior annotated.

A recent study by [Haussmann et al., 2020] aimed to compare the effectiveness of different scoring functions to select relevant data to be added to the training set of a model. The study compared four scoring functions against random selection, using six different models. The dataset used in the study comprised more than 800k images, each annotated with up to five classes: car, pedestrian, bicycle, traffic sign, and traffic light. The authors assumed that the model outputs a 2D map of probabilities, where each class is represented. Each position on the map corresponds to a specific patch of pixels within the input image. The probability value at each position indicates the likelihood that an object of the corresponding class has its bounding box centred at that particular location. They compared the following scoring functions:

- Entropy: the entropy of the Bernoulli random variable at each position in the probability map of a specific class. This entropy is computed as follows:

$$\mathcal{H}(p_c) = p_c \log p_c + (1 - p_c) \log(1 - p_c) \tag{2.7}$$

  where p_c represents the probability at position p for class c.

- Mutual Information ($\mathcal{MI}$): this approach uses an ensemble E of models to measure disagreement, which encourages uncertain samples with high disagreement among the ensemble models. First, for each position p and class c, the average probability between all members of the ensemble is calculated as:

$$\overline{p_c} = \frac{1}{|E|} \sum_{e \in E} p_c^{(e)} \tag{2.8}$$

where $|E|$ is the cardinality of E. Then, the mutual information is computed as:

$$\mathcal{MI}(p_c) = \mathcal{H}(\overline{p_c}) - \frac{1}{|E|} \sum_{e \in E} \mathcal{H}({p_c}^{(e)}) \tag{2.9}$$

- Gradient of the output layer: this approach measures the uncertainty of the model based on the magnitude of "hallucinated" gradients [Ash et al., 2019].

- Bounding boxes with confidence: this approach uses the predicted bounding boxes by the model and its confidence to measure the uncertainty of the prediction.

After computing the above values for each position or bounding box, it is necessary to compute a single score for each image. This can be done by aggregating all the measures from an image in the following ways:

- Get the maximum or minimum value of all the predictions within an image: in the case of $\mathcal{MI}$ or entropy, getting the maximum values of each image can be seen as scoring each image with its most uncertain prediction. On the other hand, in the case of the bounding box with confidence, it is necessary to get its minimum value, since the probability of a bounding box is seen as the confidence.

- Average all the values: This approach computes the average of all predictions within an image, reflecting an overall measure of confidence or uncertainty.

In the study, the researchers set a score function ($\mathcal{MI}$) and trained the set of models with an initial random selected set of 100k images. They then iterated the process three times, adding 200k images to the training set in each iteration. During each iteration, they compared three different methods for selecting the data to be added: 200k randomly selected images from the unlabelled set (X_u), 200k images selected based on a scoring function of the X_u set, and 200k images selected based on a scoring function of the X_u set combined with the set of images already used for training (X_l). In this last method, there can be repeated images, but can lead to better results since they are being selected for being considered the most relevant for the model to learn.

The results achieved by [Haussmann et al., 2020] show that the model can benefit from selecting the most relevant data for training, since the model achieved better results (73.2% weight mAP) using 700k sample images selected using AL techniques over random selection, where the model achieved 69.2% weight mAP using the same amount of data. With the full data set (850k images), the model achieved 69.0% weight mAP.

This research was based on a large dataset, and its results cannot be generalised to all the models trained for all sized datasets. As a result, exploring AL methods using a more limited dataset offers an intriguing direction for additional research.

2.2.2 Label Quality Enhancement

Many research efforts focus on improving the design of the model to achieve optimal performance and reduce the resources and time required for training [Terven and Cordova-Esparza, 2023; Zha et al., 2023]. In contrast, the Data-Centric AI approach prioritises improving data quality rather than improving model design.

Several studies have highlighted the issue of label noise in object detection and classification models, which can result from incorrect or missing labels. For instance, in [Ma et al., 2022], the authors re-annotated the labels of 80k images from the Microsoft Common Object in Context (MS COCO) dataset [Lin et al., 2014] and 5k images from the Google Open Images dataset [Kuznetsova et al., 2020]. To minimise ambiguity, they established guidelines for the re-annotation process. Subsequently, they trained a set of models to investigate how the quality of the annotations in each dataset split affects the model's performance. Specifically, they trained models using all possible combinations of the splits, including the original dataset and the re-annotated dataset. The results showed that the new annotation negatively affected mAP for the MS COCO dataset. On the other hand, the re-annotation process improved the mAP for the OpenImages dataset.

Alternatively, other studies focused on studying the label location and the impact on the performance of the model [Bernhard and Schubert, 2021; Wang et al., 2022]. In [Bernhard and Schubert, 2021] a method was used to correct the location of the label on aerial and satellite images, which achieved substantial improvements. In contrast, in [Wang et al., 2022] noise was added to the label locations, resulting in a substantial decrease in performance.

The errors in the test sets are numerous and widespread [Northcutt et al., 2021]. Quality is of great importance in the test sets, since most of the ML models are evaluated based on the performance achieved on those sets. A better model is one that outperforms another in the test set. If the test set has errors in the labels or in the location of the labels (in the case of object detection), the final judgment of a model could be mistaken. For this reason, the importance of the annotation quality of the test set is highly relevant, since it is assumed to reflect the real world.

These studies demonstrate the importance of labelling quality. Noise in the labels or in their location can significantly affect a model's performance and may lead to incorrect conclusions. Therefore, it is crucial that the labelling process is carefully designed to avoid any ambiguities.

2.2.3 Few-Shot Learning

DL requires a large amount of data to learn, so it can be a poor fit when only a few data are available. FSL was created to fix this problem by allowing DL models to learn from just a few examples. This method is inspired by the way that humans can quickly understand new ideas with little information, with the aim of transforming ML more like human learning [Parnami and Lee, 2022]. Unlike traditional methods, which might require extensive demonstrations, FSL takes advantage of previous learning experiences to inform promising strategies. Using only a few examples, learning becomes more efficient and faster. Based on the principles of inductive reasoning, this approach has motivated the creation of FSL methods [Fei-Fei et al., 2006; Wang et al., 2020]. These methods can generally be classified into two categories: meta-learning and non-meta-learning algorithms. A meta-learning algorithm aims to learn new representations across few-shot tasks to predict a new set of test tasks with limited available data [Finn et al., 2017]. A meta-learner iteratively updates model parameters and generalises to new experimental tasks from a limited amount of labelled data.

Transfer Learning, is categorised as a non-meta-learning approach. This technique involves leveraging the knowledge that a model has gained from performing tasks that are similar, yet not identical. Figure 2.8 provides a schematic representation of the TL process in a CNN. Such approach conserves time and resources, contributing for ML efficiency.

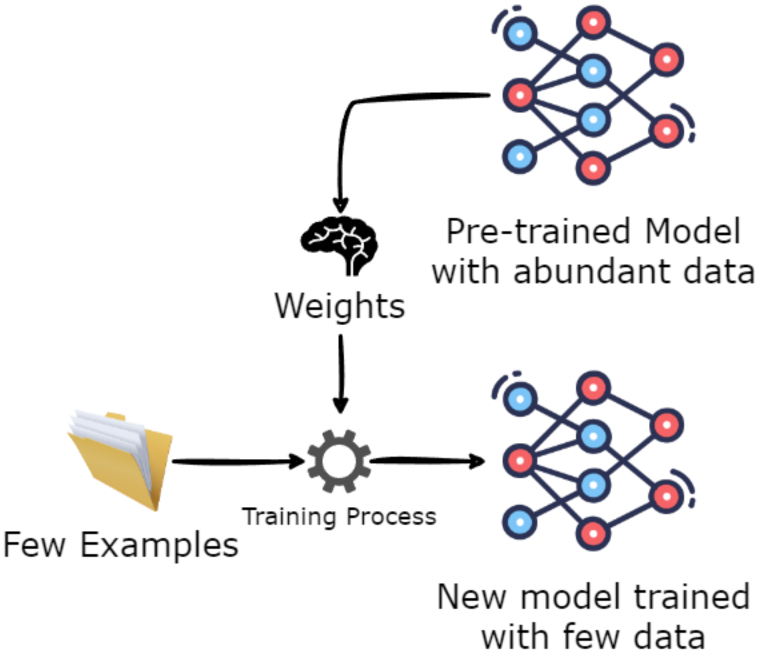


Figure 2.8: An Overview of TL in CNNs: A new model is initiated with the weights of a pre-trained model, facilitating its training process

As defined in [Pan and Yang, 2010], a domain is denoted as $\mathcal{D} = \{\mathcal{X}, P(X)\}$, where $\mathcal{X}$ represents the feature space and $P(X)$ is the marginal probability distribution. A task $\mathcal{T}$ consists of a label space Y and a predictive function $f(.)$.

Given a source domain $\mathcal{D}_s$ and a source task $\mathcal{T}_s$, a target domain $\mathcal{D}_t$ and a target task $\mathcal{T}_t$, TL aims to help improve the learning of the target predictive function $f_t(.)$ in $\mathcal{D}_t$ using the knowledge in $\mathcal{D}_s$ and $\mathcal{T}_s$, where $\mathcal{D}_s \neq \mathcal{D}_t$, or $\mathcal{T}_s \neq \mathcal{T}_t$. In the context of object detection model training, TL is commonly implemented by using the weights of a pre-existing model as a starting point. This means that the training process does not start from scratch as the model already possesses some knowledge about performing a similar task.

2.3 Conclusion

Object detection traces its origins to the 1990s. Nevertheless, the significant adoption of CNN architectures in object detection models began around 2014 [Zou et al., 2023]. This transition highlights the crucial role of CNNs in modern CV.

Object detection tasks belong to supervised learning. This means the objects targeted for detection need pre-labelling in the images designated for training. The annotation process is complex and time-intensive, typically requiring the skills of a human expert. Quality data are often closely linked to quality annotations, emphasising the need for careful attention during this stage.

Although two-stage models offer accuracy, they come with longer training and detection times [Li et al., 2017]. Their slower performance contrasts with the more agile one-stage models. Additionally, they can require significant computational resources in terms of processing time and memory. This makes them less favourable in situations where rapid results or limited resources are factors.

In contrast, one-stage detectors, such as the YOLOv5 model, are more suitable for systems that emphasise efficiency. Their lightweight structure, fast detection rate, and ease of deployment position them as the top choices for real-time operations and environments with limited resources.

The abundant data requirements for building DL models, present challenges in domains where data is scarce, such as pest detection. In such fields, state-of-the-art ML techniques like TL become invaluable. TL leverages the expertise of pre-trained models, offering a significant head start during the training process.

When resources and time are restricted, AL techniques emerge as a viable solution, especially when data annotation is involved. These techniques make use of model insights to selectively choose the most informative data, considerably alleviating the annotation process. This is because, often, not all data is required for a model to train effectively and achieve optimal results.

Alongside these techniques, ensuring high-quality data is fundamental. Inaccurate data can misguide the training process. Often, data quality is linked to annotation precision, and erroneous annotations can lead models to learn incorrectly, resulting in incorrect task outcomes.

Chapter 3

Pest Detection

Smart farming, commonly known as Agriculture 4.0, integrates technology with traditional agriculture practises. It incorporates elements such as unmanned aerial vehicles, unmanned ground vehicles, image processing, Machine Learning (ML), big data, cloud computing, and wireless sensor networks [Moysiadis et al., 2021].

Most advances in this domain have been documented since 2015 [Kamilaris and Prenafeta Boldú, 2018]. This work includes techniques for weed, seed and pest detection, plant recognition, fruit counting, yield prediction, and crop type classification.

To improve the consistency of growth in plants, some methods advocate special trimming, resulting in a uniform yield of fruits. Additionally, by using data related to weather and soil conditions, farmers can make informed decisions about planting times and anticipate the quantity of their produce.

A notable avenue in Smart Farming is pest detection, which has substantial potential. Detecting pests in their early stages can notably increase crop yields while also reducing the need for pesticides. Of the various pests that affect crops, vulnerabilities can vary according to plant type, location, growth conditions, and season.

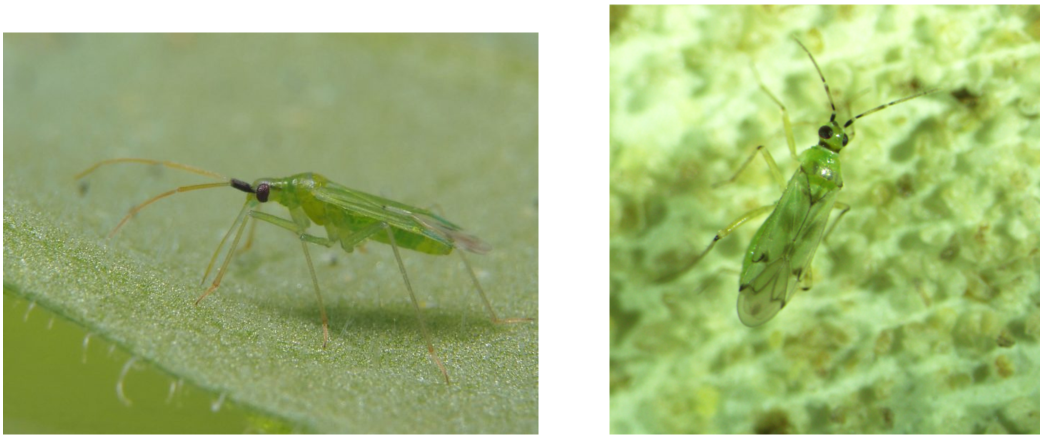
Despite significant advances in Smart Farming, opportunities for innovation continue to emerge. This research aims to introduce a new system in this domain that can detect the whitefly pest in natural settings, providing invaluable insights for farmers to take proactive control measures.

Pest control is central to agriculture and involves the management of pests that impact human activities. There are several methods of pest control, including:

- **Chemical Control**: The use of pesticides is an effective way to control crop pests, but they can be harmful and may not always be suitable for a particular plant. In some cases, there may not be a legal pesticide available to combat the pest. Therefore, it is important to minimise the use of pesticides due to their impact on the environment.

- **Biological control**: This technique uses living organisms, such as predators, to control pests. In vegetable crop greenhouses, it is common to use *Macrolophus* and *Nesidiocoris* insects, which act as predators and kill pests present in the field [Nieuwenhuizen et al., 2018]. Figure 3.1 shows both species used in biological control.

- **Cultural control**: This method involves using techniques that aim to make it more difficult for pests to thrive, such as crop rotation and pruning diseased parts of the plant, to interrupt the life cycle of pests and prevent the spread of diseases.
- **Physical control**: This includes methods such as hand-picking, traps, and physical barriers to prevent pests from reaching plants. Nets are commonly used on plantations, but may not be sufficient to stop some pest due to their small size.



(a)

(b)

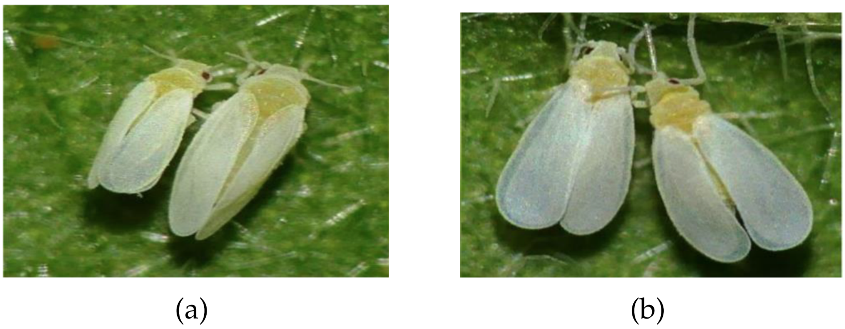
Figure 3.1: Insects used in biological control in greenhouses: (a) *Macrolophus [GulfAgriculture, 2022]* and (b) *Nesidiocoris* [Adeleye and Seal, 2021].

It is common to use a combination of these methods to increase their effectiveness. Physical control is often used as a means of detecting pests, allowing farmers to use chemical and biological control methods to kill pests.

Aphids are one of the most common pests, but in this dissertation the focus is on the whitefly pest, which is listed among the top 10 pests in greenhouse vegetable crops [Nieuwenhuizen et al., 2018].

The whitefly, also called *Alaerodidae* due to its family name, belongs to the order of *Hemiptera*, which is a group of insects known as true bugs.

The detection of whiteflies is based on its presence in greenhouses, which is where this insect has a relevant presence and has a significant impact on economic matter. In Figure 3.2 is shown *Bemisia tabaci* and *Trialeurodes vaporariorum* which are the species of whiteflies that cause the most damage to tomato crops. For the purpose of this research, it is important to identify whiteflies as a group, rather than focusing on specific species. Typically, these insects feed on the undersides of plant leaves, sucking their sap, and causing damage in the process. They are also known to transmit plant diseases, so it is crucial to control this pest.



(a)

(b)

Figure 3.2: Species of whiteflies that cause the most damage to tomato crops: (a) *Bemisia tabaci* and (b) *Trialeurodes vaporariorum* [Patel et al., 2022].

3.1 Traditional Methods

The detection of whitefly in tomato greenhouses is usually done by a technician directly inspecting the leaves of tomato plants or using yellow sticky traps (Figure 3.3). To detect the pest in its early stages of life, identification requires the use of a magnifying glass, which is a time-consuming task.

Since greenhouses can be very large, it is practically impossible to count all the whiteflies in a plantation. As a result, the number of whiteflies must be estimated by extrapolation. A technician counts the number of whiteflies in a specific region (which can be a trap) of the greenhouse and multiplies that number by the total number of regions. However, this method is prone to error as not all regions have the same concentration of whiteflies, and it is also time-consuming and requires a technician to perform. As a result, it cannot be done regularly. In some cases, extrapolation cannot be done due to the high concentration of pests in the traps, which leads the plantation to be labelled as having “too many whiteflies”. Figure 3.3 shows a trap where counting whiteflies is clearly a very difficult and practically impossible task.

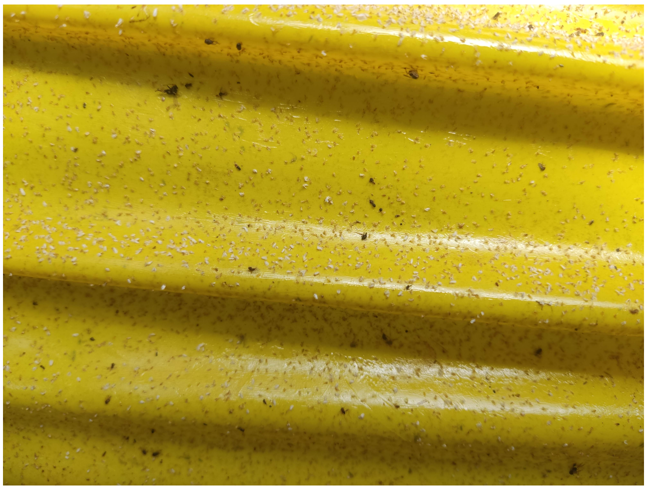


Figure 3.3: Yellow sticky trap from a tomato greenhouse used to catch insects.

For these reasons, traditional methods are:

- **Costly**: one reason that traditional pest detection methods can be expensive is because they often require the labour of trained technicians or experts, who need to be compensated for their time and expertise. This can contribute significantly to the cost of pest detection.

- **Time-consuming**: These methods can be time-consuming because they often involve manually inspecting plants or other areas for pests, identifying the type of pest present, estimating pest numbers by extrapolation, and conducting repeated inspections. All of these processes can contribute to the time required for traditional pest detection.

- **Inaccurate**: this pest detection method may be less accurate due to the small size or poor visibility of some pests, the difficulty of accurately identifying certain pests, and the potential for error when estimating pest numbers through extrapolation. All these factors can impact the accuracy of traditional pest detection methods.

- **Ineffective**: this method can become ineffective if pests are not detected early on, either through insufficient inspections or by only detecting them when their population is already high. In addition, traditional pest detection methods that are based on visual identification can be prone to errors, resulting in the implementation of ineffective pest control measures.

3.2 Smart Methods

Recently, object detection models have been employed to detect and count insects in yellow sticky traps [Cardoso et al., 2022; Domingues et al., 2022; Nieuwenhuizen et al., 2018]. In the study by [Nieuwenhuizen et al., 2018], high resolution images were captured from a Scoutbox (Figure 3.4) and a smartphone to create a dataset with annotations for whiteflies, *Macrolophus*, and *Nesidiocoris*. The Fast Region-based Convolutional Neural Network (Fast R-CNN) model was then used to detect whiteflies. Meanwhile, [Cardoso et al., 2022] used the same data set to train a model and create a structured trap with an internal camera connected to the cloud. This system was designed to capture images and count the number of detected whiteflies (Figure 3.5). In their study, the authors compared the performance of the Fast R-CNN and YOLOv5 models in detecting whiteflies in the traps. The YOLOv5X model achieved a Mean Average Precision (mAP) of 89%, while Fast R-CNN achieved a mAP of 76.15%

Both of these works involve using yellow sticky traps and address some of the limitations of traditional pest detection methods:

- **Cost-effective**: these detection methods rely on automation, including cameras, ML algorithms, and sensors. This automation reduces the need for labour-intensive manual pest detection, making it a more cost-effective option

- **Instantaneous**: smart methods, such as those using cameras and sensors, have the ability to detect pests in real-time, allowing for an immediate response to the presence of pests as soon as they are detected.
- **More effective**: these methods are able to conduct high-frequency inspections, which allows for early detection of pests, and rapid response can be taken to control pests and minimise damage.

However, there is still room for improvement, as current detection methods rely on yellow sticky traps and may not cover all regions of a greenhouse.

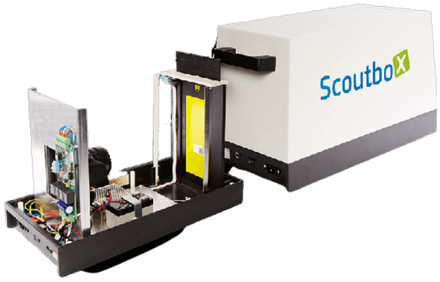


Figure 3.4: Scoutbox from Agrocares used to record images of yellow sticky traps [Agrocares, 2020]

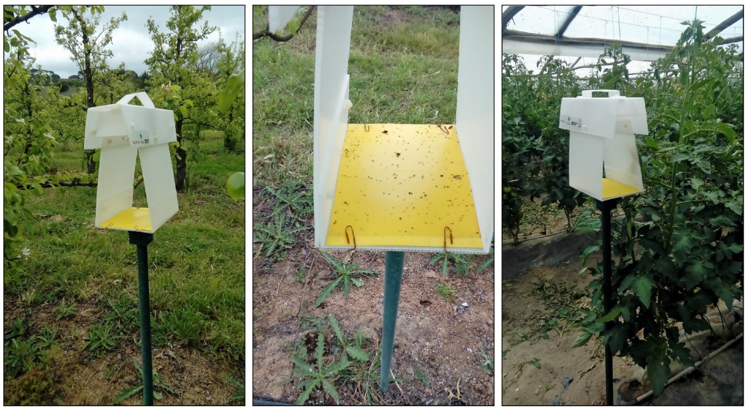


Figure 3.5: Yellow sticky trap from [Cardoso et al., 2022] study

3.3 Conclusion

Smart farming has emerged and many researches have followed, aiming to integrate technology with agriculture, striving for enhanced efficiency in response to increasing population demands.

Numerous techniques have been introduced, with pest detection being a key area where ML can greatly boost agricultural practices. Certain pests, if left unchecked, can drastically reduce crop yields.

Timely pest detection is vital, allowing farmers to act quickly and manage pest infestations before they infest an entire plantation.

Traditional methods, which largely involve manual inspections, are fraught with errors. Modern innovations address some of these shortcomings. However, most still rely on traps placed within the fields, potentially missing comprehensive coverage across extensive farming areas.

Chapter 4

Methodology

In this dissertation, the primary objective is to develop an intelligent system for pest detection, emphasising the exploration of Machine Learning (ML) techniques that enhance the training process efficiency. This chapter introduces the problem that anchors the dissertation's workflow, details the specifics of the experiments conducted, and provides an in-depth explanation of the metrics used to evaluate object detection models. Moreover, a new metric is introduced, aiming to offer insights specifically for scenarios where the objective is to count the objects detected. Three distinct experiments are explored: one focusing on the effectiveness of Active Learning (AL); another examining the influence of data quality on model performance; and the last utilising Transfer Learning (TL) to determine the potential benefits of this approach.

4.1 Problem Definition

Pest detection is crucial for enhancing crop yields. The whitefly pest, common in vegetable greenhouses such as tomato facilities, can pose significant economic challenges. In [Cardoso et al., 2022], a novel method was proposed for detecting this pest on yellow sticky traps within controlled environments, like regulated light exposure and high-resolution images. This solution addressed several limitations of traditional techniques that often rely on human expertise, making them expensive, time-intensive, and susceptible to errors. However, the innovation was bound by the specific positioning of the traps within the plantation.

The goal of this research is to design a system capable of identifying this pest in natural environments, for instance, by detecting pests directly on the plant leaves. Implementing such a model would allow farmers to capture images or videos across various plantation areas, where the system could then detect and count the whitefly pests present.

In Figure 4.1, the methodology designed for this research is illustrated, providing a comprehensive overview of each of the primary steps involved in the process. Following this representation, there are specific sections in this chapter dedicated to elaborating on each of these steps, offering a more in-depth understanding of their significance and implementation.

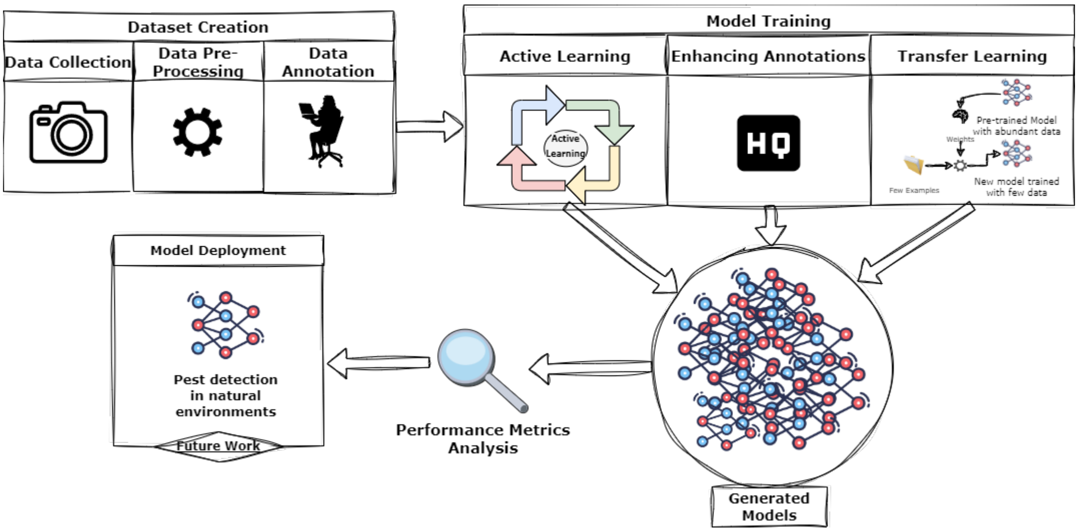


Figure 4.1: Methodology design for the research.

The success of implementing such a system depends on the obtaining of high-quality data, which is essential for model development. Acquiring data to design pest detection systems is challenging and demands effort.

Once the data is collected, the annotation task follows. This process can be particularly time-consuming and costly, especially when it requires expert input.

Using AL techniques can simplify the annotation process. These techniques focus on providing models with the most crucial data for initial training, ensuring that only the most relevant data is annotated.

However, in situations where data is limited, AL might not be enough. Models might need all available data to achieve the optimal performance. In these scenarios, enhancing the data, rather than searching for better model designs, becomes essential to offer an optimal knowledge base for model training.

An impactful strategy that can significantly affect the training of ML models is TL. This approach uses insights from existing models to improve the training of new ones. Relying on well-established models can offer considerable benefits, not just because new models start their learning process from a well-informed base, but also because foundational models might offer a deeper understanding of features not available to the new models during training.

4.2 Dataset Creation

Due to the lack of data in the domain of pest detection, particularly concerning whiteflies and, more specifically, whiteflies in natural environments, there was a need to develop a new dataset. This dataset aims to offer the proposed system an optimal foundation for learning.

4.2.1 Data Collection

The foundational stage of this research was marked by an extensive data collection process conducted at the beginning of October in a tomato greenhouse located in Coimbra, Portugal. This timing was crucial, as the greenhouse, which measures 200x100 meters, was in its advanced phases of cultivation, offering a diverse environment for detailed observation.

Tomatoes were systematically arranged in rows throughout the extensive area of the greenhouse. For a comprehensive and unbiased representation, images were captured from three distinct rows, each showing different light exposures. These rows, chosen at random, ensured a balanced representation of the varied environmental conditions that might be encountered during that time of year.

The use of two smartphones for image capturing was a deliberate decision, aiming to create a realistic dataset. These devices, prevalent in everyday scenarios, were essential in representing the natural variability of conditions. They naturally account for factors like hand movements, focus inconsistencies, and auto-exposure adjustments. Such variations highlight the realistic challenges faced when using such universally accessible tools for data collection, making the findings more pertinent to real-world scenarios.

One smartphone was responsible for capturing 300 images, each with a resolution of 4000x3000 pixels, while the other smartphone took 200 images, each having a resolution of 3000x4000 pixels. A visual representation of this data collection technique can be seen in Figure 4.2.

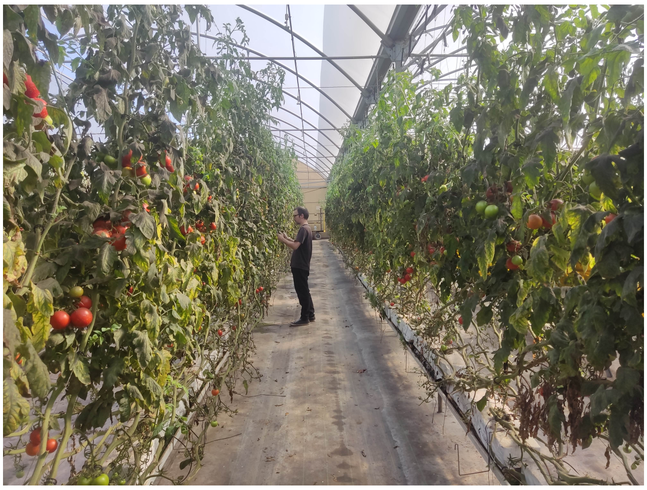


Figure 4.2: Image recorded on the process of data collection in a tomato greenhouse.

4.2.2 Data Pre-Processing

To guarantee uniformity across the dataset and facilitate model training, all images were resized to a consistent resolution of 3000x3000 pixels, as shown in Figure 4.3. Resizing is essential to accommodate the needs of various object detection models.

You Only Look Once (YOLO) models, for instance, are designed to handle images of various sizes during training. Their flexibility allows them to effectively manage a wide range of image resolutions without necessitating uniformity. On the other hand, models based on the Region-based Convolutional Neural Network (RCNN) architecture demand more specific input conditions. They require training data to maintain a consistent image size for optimal functionality.

By standardising the image resolution, the dataset becomes more versatile, making it usable for several object detection models.

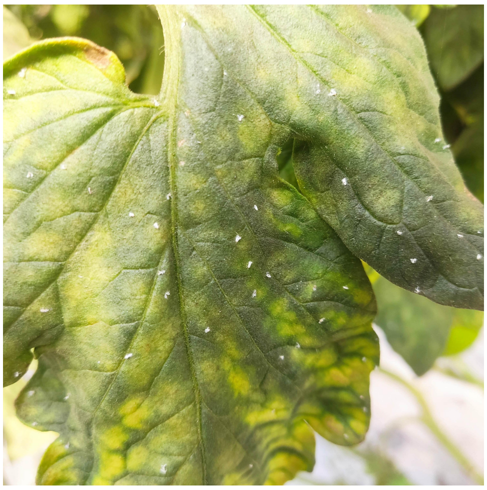


Figure 4.3: Image of tomato leaf infested with whiteflies collect in a greenhouse and resized to a resolution of 3000x3000 pixels.

4.2.3 Object Annotation

From the smartphone capturing images with a resolution of 4000x3000, 200 images were chosen at random and marked for whiteflies using the open-source online tool, *labelImg*[1]. This user-friendly tool supports the creation of bounding boxes around detected whiteflies with the aid of a computer mouse. It is compatible with both YOLO and PASCAL VOC XML formats, which are necessary for the training of YOLO and Faster Region-based Convolutional Neural Network (Faster RCNN) models, respectively.

For the purpose of this research, emphasis was placed on the YOLO annotation format, structured as:

```
<object-class> <x> <y> <width> <height>
```

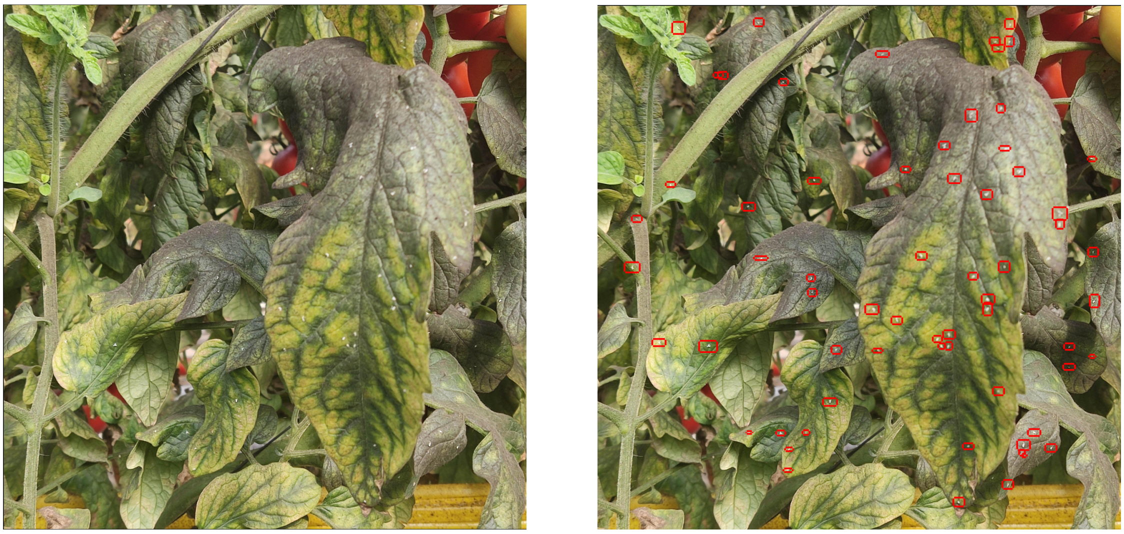
Breaking down the components:

- <object-class> refers to the class of the object being identified (e.g. whitefly);

[1] https://github.com/heartexlabs/labelImg

- <x> and <y> pinpoint the centre of the bounding box that encompasses the object, with normalization within [0, 1];
- <width> and <height> detail the dimensions of the bounding box, which are also normalised between [0, 1].

Illustration 4.4 contrasts an image before and after the annotation process.



(a) Dataset image pre-annotation

(b) Dataset image post-annotation

Figure 4.4: Annotation Procedure

In summary, a total of 10747 whiteflies were marked across the 200 images. This is equivalent to about 54 whiteflies per image. The annotation process was done with a focus on efficiency rather than meticulous precision, due to the time and effort allocated to this activity.

4.3 Model Architecture Selection

In this research, three different approaches are studied that improve the performance of object detection models. Therefore, to reduce the number of experiments to be conducted, the focus is only on exploring one model architecture. In [Cardoso et al., 2022], the YOLOv5 was employed and performed satisfactorily on the whitefly detection task. However, YOLOv8 is considered the state-of-the-art model for object detection. So, to choose the right architecture to explore, a benchmark experiment was conducted for comparing YOLOv5 sub-architectures and YOLOv8 sub-architectures with our dataset. The following models were trained and compared:

- YOLOv5:
  - Nano;
  - Small;
  - Medium;
  - Large;
  - XLarge.

- YOLOv8:
  - Nano;
  - Small;
  - Medium;
  - Large;
  - XLarge.

The nano, small, medium, large, and XLarge architectures of both models vary in the size of the layers, with the first ones representing the smaller sizes and the last ones representing the larger sizes. These configurations present a balance between computational efficiency and detection accuracy.

The benchmark experiment trained each model using the same dataset split, allocating 20% of the images for testing, 20% for validation, and 60% for training.

The YOLOv5 Small (YOLOv5s) provided the most favourable balance between processing time and performance, registering a 74.7% Mean Average Precision (mAP) in 1143 seconds. In contrast, the most effective YOLOv8 architecture was the Medium version, which recorded a 68.5% mAP in 2160 seconds.

While ideally the experiment should have been conducted multiple times with varied configurations to enhance diversity and generate more representative results, the benchmark was carried out just once. Given that it was only a benchmark and the primary focus of this dissertation is not on model architectures, and considering the discernible differences in both performance and training time in a single run, there was no compelling need to perform additional runs since YOLOv5 evidently surpassed YOLOv8 in performance. Subsequent experiments were therefore carried out using YOLOv5s. Additionally, the YOLOv5 architecture facilitated the application of TL based on the research presented in [Cardoso et al., 2022], positioning YOLOv5 as an optimal choice for continued research.

4.4 Experiments Design

As previously mentioned, in this research, three different techniques are explored to improve model performance while keeping the training process as efficient as possible. These techniques include the use of AL when dealing with small datasets like the one built in this dissertation, improving data quality and using TL based on the work developed in [Cardoso et al., 2022].

4.4.1 Active Learning

To investigate the efficacy of using AL with small datasets, an experiment was designed using the dataset comprising 200 images. The intent was to understand the potential advantages of incremental training with increasing subsets of this dataset, beginning with a base of 60 images and progressively adding more until all available images were utilised.

The dataset was randomly divided into 160 images for training and validation, and 40 images for testing, as detailed in Table 4.1. And the detailed configuration

Table 4.1: Dataset specification for AL experiment: highlighting the number of images and whitefly annotations on each set.

Total: 200 images (10747 whiteflies)	
Train and Validation	**Test**
160 images (8268 whiteflies)	40 images (2479 whiteflies)

of each setup is outlined in Table 4.2.

Table 4.2: Summary of experimental setups for AL experiment, detailing the total number of training and validation images and the percentage of the dataset allocated to each setup.

Setup S_t	Total number of Images for Training and Validation	Dataset splits (%)	
		Train	Validation
S_0	20	5	5
S_1	35	10	7.5
S_2	50	15	10
S_3	65	20	12.5
S_4	80	25	15
S_5	95	30	17.5
S_6	105	35	20
S_7	110	40	20
S_8	130	45	20
S_9	140	50	20
S_{10}	150	55	20
S_{11}	160	60	20

In the initial setup, specifically with 20 images denoted as S_0, a subset of 20 images is randomly selected from the available dataset of 160 images, represented as D_a, and then removed from it. These selected images are randomly divided following the training and validation distributions, which are labelled as D_{tr} and D_v, respectively. A model is then trained using D_{tr} and D_v, initialising the weights randomly.

In subsequent setups, the model derived from the previous setup, S_{t-1}, is utilised to make predictions on the images in D_a. This prediction aids in the computation of the bounding box confidence score. This score subsequently allows for the ranking of images based on their average confidence in predictions. Accordingly, these images are sorted in ascending order based on their confidence levels, from the least to the most confident.

Once the images in D_a are sorted, for each setup S_t, the difference in the number of images between S_t and S_{t-1} is removed from D_a. These images are again randomly partitioned according to the configurations D_{tr} and D_v for S_t.

The process continues until setup S_{11} is achieved.

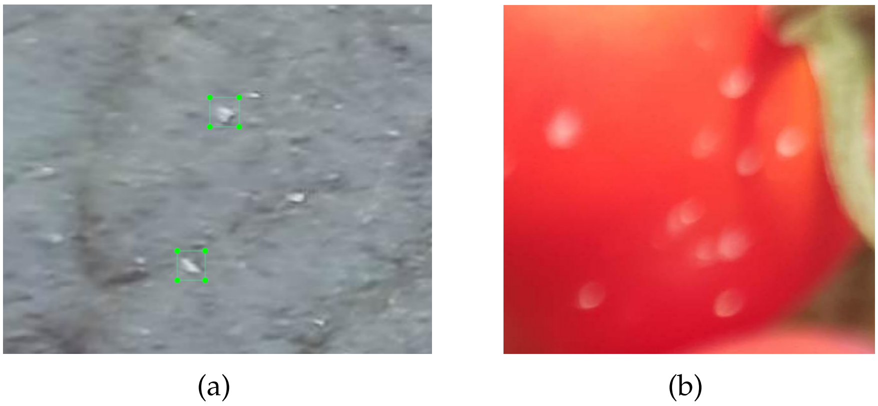
Parallel to this experiment, using the same D_{tr} and D_v from S_0, an analogous experiment is executed. The difference here lies in the method of selecting images from D_a, which is performed randomly.

This parallel experiment provides a basis for a comparative analysis on the genuine efficacy of training with AL versus the random selection of images for training. While AL strategically chooses the most informative data points, enhancing the learning efficiency of the model, random selection adopts an unbiased approach, potentially introducing a wider variety of data. The comparison between these methods will highlight whether a directed and information-rich selection surpasses the unpredictability and diversity of a random choice.

4.4.2 Enhancing Data Quality

The images in the dataset, taken under varying light conditions, present a challenge due to the diminutive size of whiteflies, which measure between 1 and 3 mm in length [Sani et al., 2020]. Such conditions make the labelling process complex, as pests become difficult to spot.

Given that the initial labelling effort was directed for efficiency rather than meticulous precision, certain ambiguities arose. The clarity with which whiteflies are visible can fluctuate, depending on their size, orientation, and position in the frame. Moreover, lens focus can introduce additional complexity, especially for those insects near the image edges, which can appear less distinct or even out of focus. The primary phase of labelling was characterised by a method that annotated all the white spots in the images, assuming that they were whiteflies. However, a secondary review by a different individual indicated that some of these annotated points were not easily identifiable as whiteflies. To add to the challenge, certain whiteflies could be mistaken for water spots. Figure 4.5 shows this situation. Given these inconsistencies, a meticulous reassessment of the annotations became imperative. The dataset characterised by these initial annotations has since been named the *Original Dataset*.



(a)

(b)

Figure 4.5: Images from *Original Dataset* highlighting annotation errors: (a) White dots wrongly assumed to be whiteflies; and (b) Whiteflies that could be mistaken as water drops.

Preliminary evaluations demonstrated that the YOLOv5s model effectively identified the location of whiteflies within the images. This capability enhanced the precision of bounding boxes, especially when compared to those of the *Original Dataset*, as depicted in Figure 4.6. Leveraging this model, predictions were made for the positions of all detected objects and, based on these predictions, the *Improved Dataset* was created.

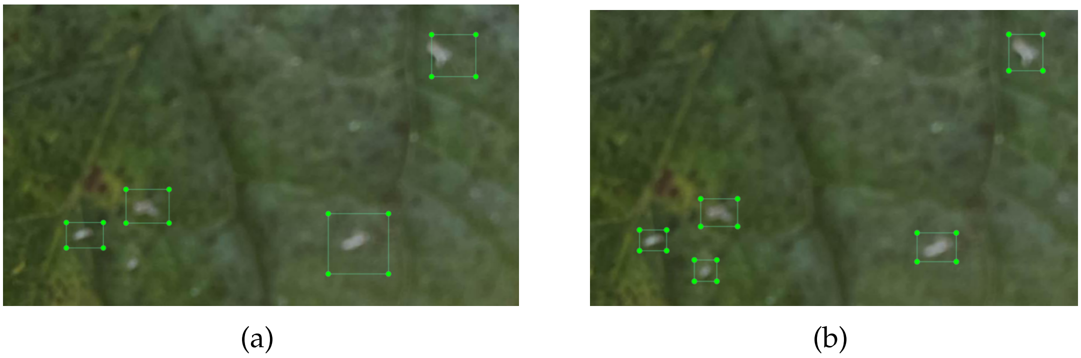


Figure 4.6: Comparison of object annotation using: (a) Whiteflies with noisy bounding box locations from *Original Dataset*; and (b) Whiteflies annotations from *Improved Dataset* with bounding box locations predicted by an YOLOv5s model trained with the *Original Dataset*.

The primary objective of developing an object detection model is its application in real-world tasks. It is commonly believed that a model showing better performance on a test set will also perform effectively in practical scenarios. Therefore, the importance of a test set that mirrors real-world conditions is crucial. However, the labelling process is both detailed and time-intensive, with demands rising in line with the need for finer details. The concept of Human-In-The-Loop focuses on combining human expertise with machine capabilities to improve ML precision [Wu et al., 2022].

In this experiment, this synergistic approach is embraced by realigning annotations on 40 images, randomly selected from the *Improved Dataset*. It is crucial to note that the *Improved Dataset* is fundamentally constructed upon predictions made by a ML model. Thus, by refining the annotations of these 40 images, it is a representation of a collaborative effort where humans and machines work together. This collaboration not only minimises the intensive demands of the labelling process, but also ensures that the model is evaluated using data that reflect both machine accuracy and human expertise.

To ensure clarity and consistency during the adjustment of annotations for the 40 images, specific guidelines were rigorously followed. The outlined guidelines are as follows:

- “V" shape: Objects with a "V" shape are annotated as whiteflies, because of the shape of their wings.
- Triangular shape: Whiteflies can also have triangular shapes because of their wings, and as a result, objects with a triangular shape are annotated as whiteflies.
- Focused sharp white forms: Objects with clear and defined white shapes are annotated as whiteflies.

- Shaky or unfocused white forms: Objects with white forms that are shaky or not sharply defined are not annotated as whiteflies due to the uncertainty of the image quality.

Following the reannotation process, the composition of the *Improved Dataset* is detailed in Table 4.3. The total number of annotations in the *Original Dataset* stands at 10,115. It is important to mention that the annotation process underwent several changes over time due to various adjustments to the dataset, explaining the discrepancy in numbers presented in Section 4.2.3. Meanwhile, the *Improved Dataset* contains 11,517 annotations. This indicates an increase in the number of whiteflies that are considered compared to the original annotations. While it is acknowledged that some might not represent actual whiteflies, it is crucial to emphasise that the accuracy of the bounding boxes has significantly improved, as can be clearly observed in Figure 4.6.

Table 4.3: *Improved Dataset* specification: highlighting the number of images and whitefly annotations on each set. The train and validation annotations were predicted by a YOLOv5s model.

Total: 200 images (11517 whiteflies)	
Train and Validation	**Test**
160 images (9457 whiteflies)	40 images (2060 whiteflies)

The purpose of the experience is to examine the influence of varying annotation qualities on a model performance by comparing outcomes from two distinct datasets: the *Original Dataset* and the *Improved Dataset*. Investigating this distinction is essential because the quality of annotations can significantly affect the training of a model, and consequently its prediction accuracy. By contrasting the results obtained from both datasets, the research seeks to highlight the importance of precise annotations in achieving optimal model performance and to provide insights into potential strategies for data enhancement in future projects.

4.4.3 Transfer Learning

The exploration into the effectiveness of TL techniques, particularly in the domain of pest detection, leads to a model described by [Cardoso et al., 2022]. This model was trained to detect whiteflies using high-resolution images (5184 × 3456 pixels) taken in a controlled environment. This means consistent lighting and a lack of common agricultural disturbances, as illustrated in Figure 4.7.

In comparison, images from the greenhouse dataset, due to the nature of the greenhouse environment, were captured in an uncontrolled setting, which led to variable light exposures and other photographic inconsistencies, such as occasional blurring. What adds to the complexity is the rich visual content of these images, which are complex and rich in content, displaying a range of colours and

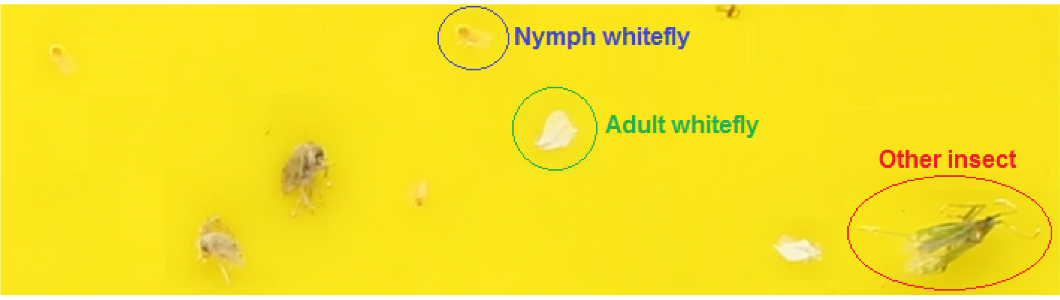


Figure 4.7: Image of the dataset used in [Cardoso et al., 2022], which was taken under regulated light exposure and absence of noise.

elements, including the green of the leaves, the red of the tomato fruits, and the white of the whiteflies, among various other features in the background.

This difference in image environments—controlled versus uncontrolled—poses intriguing questions about how TL techniques might adapt and perform in such varied scenarios.

To thoroughly evaluate the potential advantages of TL for whitefly detection, an experimental approach was outlined that encompassed training various models across a range of scenarios, each differing by the number of training images used. This began with a simple set of just two images, gradually escalating to utilise the entire collection of available examples. For each image set, two distinct models were to be trained:

- The first relied on the pre-trained weights of the YOLOv5 small model, as detailed in [Cardoso et al., 2022], as its foundation;
- The second, in clear contrast, began its training process with randomly initialised weights, refereed as "Scratch" model, signifying that its training starts from the very beginning without any prior knowledge.

Each training scenario was carefully crafted by randomly drawing images for training and validation from the 160 images that make up the training and validation set from the *Improved Dataset*. It should be noted that the test set remained consistent with that used in previous experiments, as it is assumed to represent the highest quality possible. The details and structure of this experiment are detailed in Table 4.4.

This methodology not only provides a comparative assessment between the two training approaches but also offers insight into how the quantity of training data might influence detection performance.

Table 4.4: Overview of the experiment for TL: The first column indicates the number of images used in the training process, while the subsequent columns represent the distribution of images used for training and validation in each scenario, respectively. Note: The exploration of 1-shot learning, i.e., training with only one example, is not feasible since YOLOv5 requires at least one image for the validation.

Number of images	Training Images	Validation Images
0	-	-
2	1	1
4	2	2
7	5	2
15	10	5
20	15	5
30	20	10
50	40	10
160	120	40

4.5 Performance Metrics

The evaluation of object detection and classification models is typically carried out using the metric mAP, which provides valuable information on the effectiveness of the model in detecting annotated objects.

In some specific cases, such as fruit counting or pest detection, where the primary goal is the number of objects detected, other metrics such as Mean Squared Error (MSE) and Root Mean Squared Error (RMSE) may be used.

Following from this, mAP is explained in more detail, and a new metric is proposed. This new metric offers a more detailed view of the detected counted objects compared to MSE or RMSE.

4.5.1 Mean Average Precision

To better understand mAP, it is essential to understand several other metrics that contribute to its calculation:

- Confusion Matrix: A confusion matrix is a table that summarises the predictions made by a ML model. It consists of:
  - True positives (TPs): The number of times the model correctly predicted the positive class;
  - False positives (FPs): The number of times the model incorrectly predicted the positive class;
  - True negatives (TNs): The number of times the model correctly predicted the negative class;

- False negatives (FNs): The number of times the model incorrectly predicted the negative class.

With a computed confusion matrix, it is possible to calculate relevant metrics for ML model's evaluation:

- Precision: This quantifies the number of correct positive predictions made out of all the positive predictions. It is calculated as follows:

$$Precision = \frac{TP}{TP + FP} \tag{4.1}$$

- Recall: This quantifies the number of correct positive predictions made out of all positive predictions that could have been made. It is calculated as follows:

$$Recall = \frac{TP}{TP + FN} \tag{4.2}$$

To calculate precision and recall, object detection models rely on the Intersection over Union (IoU), which is a metric that measures the overlap between the bounding box predicted by the model and the ground truth bounding box (the annotated one). The IoU is calculated as follows:

$$\text{Intersection over Union (IoU)} = \frac{A \cap B}{A \cup B} \tag{4.3}$$

where A is the ground truth bounding box and B is the predicted bounding box obtained from the detection model. The numerator represents the overlap area, i.e., the shared region between the predicted and ground truth bounding boxes, and the denominator represents the union area, i.e., the combined area of both these boxes. The ground truth corresponds to the manually annotated object location in the image. In Figure 4.8 a visual representation of this metric is presented.

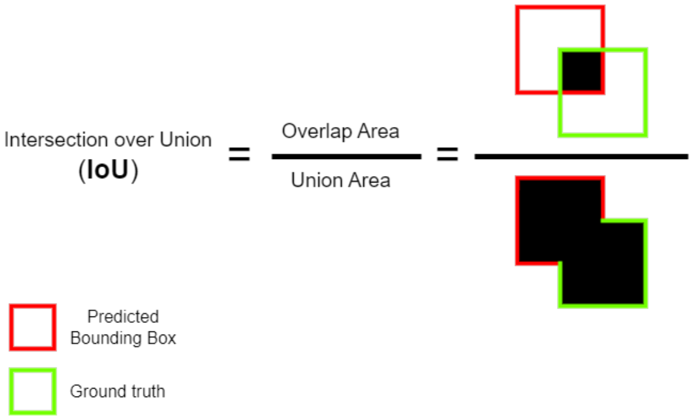


Figure 4.8: Visual representation of the IoU metric, showing the overlap area and the union area between the predicted bounding box and the ground truth.

In mAP, an object is considered a true positive (TP) if $IoU \geq 0.5$, and a false positive (FP) if $IoU < 0.5$. This applies if a threshold of $IoU = 0.5$ is considered.

The Average Precision (AP) is calculated by finding the Area Under the Curve (AUC) of the precision-recall curve, which plots the precision of the model at different levels of recall for various confidence thresholds assigned to a bounding box by the model. These thresholds indicate how confident the model is that a particular bounding box contains an object. The mAP averages the AP across all classes. Figure 4.9 illustrates the mAP computation for a scenario where there is only one class.

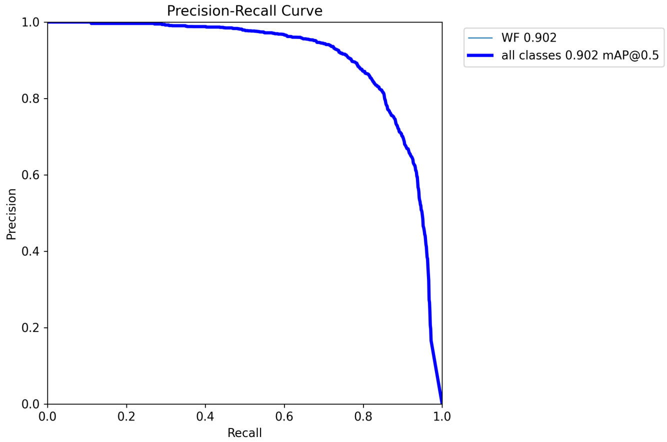


Figure 4.9: Precision-Recall curve computed for the scenario of one class (WF). In this scenario, since there is only one class, the mAP is equal to the AP, where AP is the AUC for each class.

4.5.2 Ratio of the Predicted Counted Objects

Both MSE and RMSE are commonly used to measure the performance of regression ML models. When the goal is to count the number of detections made by a model, these metrics provide valuable insight into the error between the actual number of objects and the count predicted by the model. MSE is the average squared error between the actual and predicted values and is calculated as follows:

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

where, in the case of counting objects with object detection models, y_i is the actual count, and $\hat{y}_i$ is the predicted count. The RMSE is the square root of the mean squared error and is computed as follows:

$$RMSE = \sqrt{MSE} \tag{4.5}$$

Both metrics measure the average error, but MSE gives more emphasis to larger errors, since it averages the squared errors. RMSE gives a better insight into the real error since it represents the square root of the mean squared error, providing a more intuitive understanding of the magnitude of the error. However, both metrics fail to provide insight into whether the error is more positive or negative. Knowing whether a model predicts more or less than it should be is essential information. To fill this gap, in this research, it is proposed to use Ratio of the Predicted Counted Objects (RPCO) as a complementary metric for evaluation. For this purpose, the following calculation is used:

$$RPCO = \frac{Predicted_{count}}{Actual_{count}} \tag{4.6}$$

where $Predicted_{count}$ represents the total number of objects detected by the model in a collection of images and can be calculated as:

$$Predicted_{count} = TP + FP$$

while $Actual_{count}$ is the total number of objects annotated in the same collection of images, given by:

$$Actual_{count} = TP + FN$$

RPCO can then be derived. If $RPCO > 1$, the model detects more objects than those annotated. On the other hand, if $RPCO < 1$, the model detects fewer objects than those annotated in the dataset.

4.6 Conclusion

Pest detection has advanced over the years, moving towards automated methods. However, many current techniques still rely on traps. A more direct method would be to detect pests where they live, such as on plant leaves.

To address the data gap for training such detection models, this dissertation introduces a dataset of images from a tomato greenhouse. Here, the whiteflies were abundant during the final stages of cultivation. The images were resized, and a subset was annotated to identify whiteflies. This task was challenging due to the high resolution of the images and some focus inconsistencies. This annotated dataset is crucial for building models to detect and count whiteflies in images or videos, aiding farmers in timely pest management decisions.

Given the annotation challenges and limited data, three experiments were designed. They seek to assess the efficiency of object detection models in situations with limited and low-quality data, using techniques like AL, data enhancement techniques and TL.

To evaluate such models, mAP is a mandatory metric to use, as it is the metric employed in the training process. However, when the task involves counting detected objects, metrics like MSE and RMSE may be more suitable, although both lack the ability to provide information about whether the model is detecting more or fewer objects than it should be. To fill this gap, RPCO is a metric that offers better insight into this situation.

Chapter 5

Experiments and Results

This chapter presents and discusses the results of the designed experiments.

Three experiments were carefully structured to evaluate the effectiveness of the proposed methods in specific contexts.

The initial section presents the results derived from the use of Active Learning (AL).

The subsequent sections turn attention to the significance of enhancing data quality. The impact of improved annotations and image quality on model performance is analysed.

Lastly, a discussion of the results of Transfer Learning (TL) is conducted. This technique uses knowledge from an existing task to help another. The influence of using pre-trained models on the acceleration and enhancement of results is evaluated.

5.1 Active Learning

In the experiment examining the utility of AL with limited datasets, there was a clear difference in performance depending on the amount of data used for training and validation. Each scenario underwent training for 160 epochs, and the resolution of the model's images was 1280. Table 5.1 displays the results obtained from the test set.

Initially, 5% of the available data was used for training, with an equal percentage used for validation. Predictably, this configuration yielded the lowest performance metrics. As the volume of data allocated for training and validation increased, a corresponding improvement in detection accuracy and reliability was observed. This reinforces the idea that the volume of data plays a pivotal role in model proficiency, especially in the field of object detection.

Upon reviewing the Mean Average Precision (mAP) results from the second split, it becomes clear that the performance derived from random selection surpasses

Table 5.1: Results Table for the AL experiment: Comparison of mAP (%), training time (using an NVIDIA GeForce RTX 4080), and Ratio of the Predicted Counted Objects (RPCO) when the model is trained on various quantities of images selected randomly or with Active Learning (AL)

Training and Validation	**mAP (%)**		**Training time (s)**		**RPCO**	
Images	**AL**	**Random**	**AL**	**Random**	**AL**	**Random**
20	5.9	5.9	289.2	310.2	0.35	0.35
35	49.5	51.5	386.4	375.5	1.58	1.40
50	60.4	59.4	463.3	462.9	1.74	1.47
65	79.4	76.5	472.2	461.1	1.32	1.37
80	85.8	85.5	548.2	534.5	1.53	1.34
95	86.6	86.1	612.6	602.4	1.42	1.36
110	89.5	87.6	672.9	658.9	1.39	1.43
120	89.4	87	731.6	731.5	1.43	1.43
130	90.7	90.1	764.9	769.2	1.33	**1.24**
140	**92.6**	91.7	835.3	828.8	**1.27**	1.34
150	92	91	890.2	890.7	1.44	1.30
160	92.6	**92.9**	932.1	936.5	1.30	1.26

that of the AL techniques by about 2 mAP percentage points. A potential reason for this could be the broader variety of data introduced by the random selection, offering a more diverse set for training. Additionally, the variation could be the result of the model's sub optimal learning from the previous split, in which only 10% of the data served for training and validation. As a result, the images selected for the second split relied on the predictions of an under performing model.

From the third split up to the penultimate one, AL consistently exceeded the results from the random data selection. The findings indicate that the AL approach reached its peak performance when utilising 70% (140 images) of the data for training and validation. In contrast, the random selection method peaked in its performance in the final setup, using 80% (160 images) of the data for the same purpose. Although the top performance from the random experiment exceeded that of AL by 0.3 mAP percentage points, the efficiency of using lesser data with AL cannot be overlooked. Specifically, the AL experiment achieved 92.6% of mAP by leveraging only 1.51 images per mAP point, while the random approach required 1.72 images to achieve a similar result. This underlines the idea that the additional effort and data required for a mere 0.3% increase in mAP might not be justifiable.

The process of annotating each whitefly in the images demands a considerable amount of time and effort. Taking this into account, the slight performance improvement achieved by incorporating 20 more images might not justify the added workload. Instead, focusing on enhancing annotation quality or tapping into alternative data sources might yield better results in boosting the model's performance.

In relation to RPCO, except for the initial setup, the values for the subsequent setups were consistently above one. This suggests that the model identified a higher number of objects than the one that was annotated. Such results can be interpreted in two ways: The model might recognise more whiteflies than initially annotated, or it might exhibit a higher accuracy in whitefly detection than the human annotator. Given the dataset's nature of variable-quality images taken under diverse lighting conditions, dismissing the latter possibility would be premature. Such challenging conditions inherently complicate the task of pest detection. Moreover, the model's ability to identify pests even in their nascent stages, when they are minuscule and especially hard to spot, could also contribute to the discrepancy in detection counts.

Analysing Figure 5.1, it is possible to observe an expected trend: a significant increase in mAP in the first setups, followed by a stabilisation of the curve. When we superimpose the training time curve onto the mAP curve, we see that the two curves converge, suggesting that further performance improvements come at the cost of an increased training time.

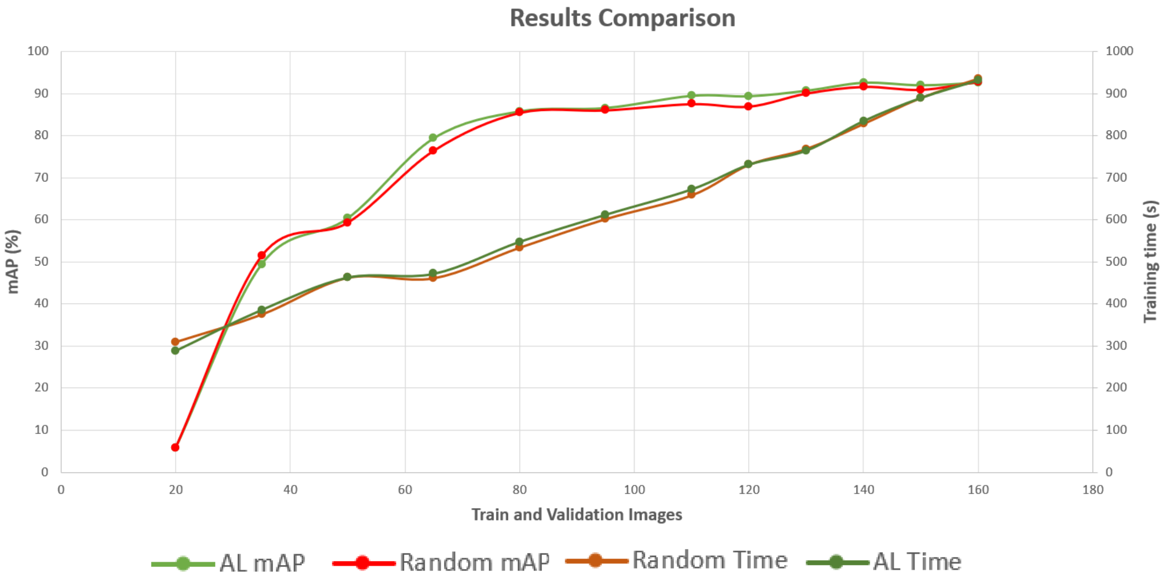


Figure 5.1: Comparison Chart: mAP achieved with incremental number of images used in training and validation, selected via Active Learning (AL) vs Random Selection. Training time comparison included.

5.2 Enhancing Data Quality

Regarding the goal of understanding the effect of varying annotation qualities on a model's performance, the assessment was carried out using the same test set from the *Improved Dataset*, which is presumed to have perfect annotations. This investigation involved 30 iterations, where in each, two models were trained for 200 epochs and a model's image resolution of 1280, using identical images but distinct annotations sourced from the two datasets: *Original Dataset* and *Improved Dataset*. Adopting this method ensures a varied data composition in each iteration since the training and validation sets undergo random partitioning for each iteration.

Table 5.2 presents the results of the experiment. When using the *Improved Dataset*, the model performance, measured by the average mAP, is higher by 1.1 mAP points compared to using other datasets. This confirms the initial expectation that better annotations lead to better performance. On the other hand, the training time is slightly faster with the *Original Dataset* by about 2.49 seconds. The reason for this is the larger count of annotations present in the *Improved Dataset*. In essence, quality annotations contribute to better accuracy, but they can also require a bit more time in the training process. It is a balance between quality and speed.

Table 5.2: Results obtained on the data quality experiment: the average **mAP (%)** and **Training Time (s)** are presented with standard deviation in parentheses over 30 runs.

Dataset	**mAP (%)**	**Training Time (s)**	**Best mAP (%)**	**Fastest Time (s)**
Original Dataset	90.44 (0.70)	**758.20 (5.76)**	91.7	**736.28**
Improved Dataset	**91.54 (0.66)**	760.69(3.62)	**93.1**	753.88

For the purpose of validation, the normality of the mAP and time results was assessed across the 30 paired values. The *Shapiro-Wilk* test, a widely used method, was chosen to evaluate the distribution of the data. Interestingly, only the training time data from the *Original Dataset* did not reject the null hypothesis with a $p-value$ of 0.05, implying that this specific set of data is normally distributed. On the other hand, the rest of the sets appear to not follow a normal distribution.

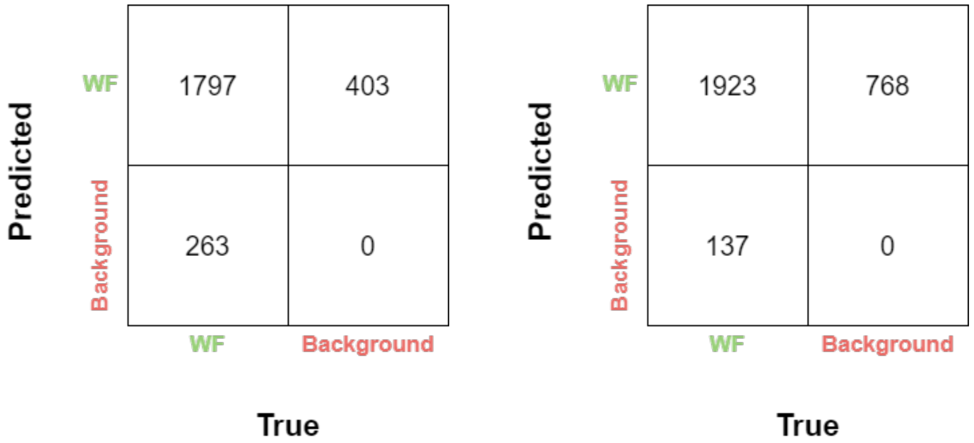
In the results presented, the *Wilcoxon* test was used to determine the statistical significance of the observed differences. The null hypothesis of the *Wilcoxon* test posits that the differences between the paired observations are symmetrically distributed around zero.

Considering the mAP pairings, the test did not reject the null hypothesis at $p-value$ of 0.05. This indicates that the observed differences in mAP between the values are not statistically significant, suggesting that the use of the *Improved Dataset* might not have led to a notable improvement in mAP results.

On the other hand, when evaluating training time, the test rejected the null hypothesis at the same $p-value$. This points to a significant difference in training times between the two datasets. As observed, training with the *Original Dataset* proved to be quicker.

The aforementioned results were surprising given the initial expectations. However, it is crucial to note that the statistical analysis took into account all 30 runs. When focusing on the results of the top performing models trained using each dataset, there seems to be a potential benefit in using the *Improved Dataset*.

By examining the confusion matrices for the top performing models of each dataset (as shown in Figure 5.2), a distinction emerges. The model trained using the *Improved Dataset* outperforms in accurately identifying whiteflies, evident from its reduced count of false negatives (FN). On the contrary, this model registers a higher count of false positives (FP), indicating an inclination to detect whiteflies even when absent. Given the difficulties of this task, coupled with the challenges of the dataset's varying light conditions, differing focal points, and the rigorous annotation guidelines for the test set, it is conceivable that some of these perceived FPs might indeed be true positives (TP). This is particularly plausible considering that younger whiteflies in their early life stages or those in less stable image conditions can be challenging to spot and accurately annotate.



		True	
		WF	Background
Predicted	WF	1797	403
	Background	263	0

(a) Confusion Matrix from the best model trained with the *Original Dataset*

		True	
		WF	Background
Predicted	WF	1923	768
	Background	137	0

(b) Confusion Matrix from the best model trained with the *Improved Dataset*

Figure 5.2: Comparison of the confusion matrices for the best model trained with different dataset annotations: *Original Dataset* Annotations and *Improved Dataset* Annotations.

5.3 Transfer Learning

For the TL experiment, due to inherent randomness in splitting the data into training and validation sets, multiple runs were conducted essential to validate the outcomes. Each scenario was repeated five times, each being trained over 200 epochs and the model's image resolution of 1280. It is important to note that the test set remained consistent with the one used in the previous experiment. The aggregated results, reflecting the average mAP and the training time over all these iterations for each scenario, are documented in Table 5.3.

Examining the experimental results summarises the considerable gains from employing TL. In all the scenarios tested, the models that took advantage of TL consistently exceeded those in which the Convolutional Neural Network (CNN) weights were randomly initialised. This superiority manifested itself both in terms of mAP performance and efficiency of the training process. To put it into perspective, the introduction of TL amplified the mAP by a significant average of 24%. Furthermore, this approach decreased the training time by 10%.

This phenomenon is visually represented in Figure 5.3, where the progression

Table 5.3: Comparison of models trained with the same image set but different weight initialization methods on the context of TL experiment: "TL" (Transfer Learning with pre-trained model weights) and "Scratch" (randomly initialized weights). The number of images used, mAP results, and training times are outlined for each approach. Bold figures denote superior performance. The models were trained using an NVIDIA GeForce RTX 4080.

Number of images	Average mAP		Average Training Time (s)	
for Training and Validation	TL	Scratch	TL	Scratch
0	**0.02**	0.00	-	-
2	**0.16**	0.00	**225**	248
4	**0.18**	0.00	**210**	241
7	**0.19**	0.00	**227**	253
15	**0.35**	0.00	**245**	287
20	**0.42**	0.04	**298**	346
30	**0.56**	0.23	**300**	373
50	**0.62**	0.34	**405**	443
160	**0.77**	0.68	**917**	962

of mAP throughout the training period is plotted, specifically for the scenario where the entire available dataset was used for training purposes. This side-by-side comparison distinctly showcases the advantages of using TL. By the time the training process reached epoch 40, the TL integrated model had already reached a notable performance. In contrast, the model relying on the random initialization of weights only matched the performance of its TL integrated model by the time it reached epoch 150.

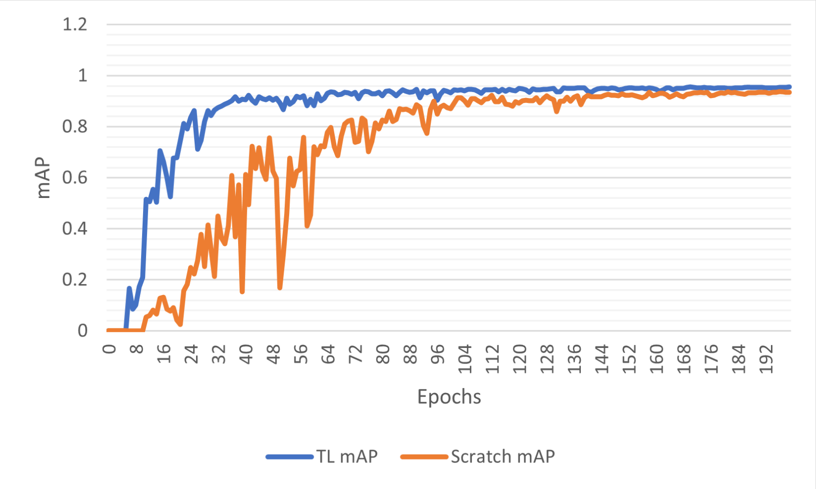


Figure 5.3: mAP evolution during training for a scenario with 160 images. The graphic depicts two training processes: one using Transfer Learning (TL mAP) and the other with randomly initialized weights (Scratch mAP).

However, it is important to note that no model performed satisfactorily when trained with only a few examples. Exclusively in the final scenario, where the full dataset was used, the model achieved an acceptable performance with an mAP of 0.78. This event might be attributable to the differences in the tasks of each model. The pre-trained model was trained to detect whiteflies in a controlled environment with high-resolution images, while our model was trained to detect the same pest but in a completely different environment and with significantly lower resolution. The prior knowledge of the pre-trained model provided a significant advantage in the training process but was definitely insufficient to detect the same pest in an entirely different environment.

Furthermore, the fact that the models were trained for 200 epochs may not have been enough for them to learn all the features necessary for optimal performance. In contrast, the model was achieving higher mAP values during the training process on the validation set, reaching up to 0.96 mAP, indicative of overfitting. Therefore, finding a suitable balance between the number of epochs, the number of examples, and the distribution of examples in the training and validation sets during the training process is a challenging task.

To further investigate the impact of epochs on the training, an experiment was carried out using a 20-shot scenario (i.e., using 20 images), which were divided into 15 for training and 5 for validation. The epochs were varied from 50 to 700, in steps of 50. For every epoch count, two models were trained on the same randomly chosen image set. One used TL and the other had its weights randomly initialized. The results are shown in Figure 5.4. The findings consistently indicated that the TL-assisted models surpassed those with random weight initialisation. However, increasing the epoch count did not guarantee excellent performance using TL; the peak mAP reached was 0.58 at 600 epochs. However, models with TL saw an average mAP boost of around 30% compared to those with random weights.

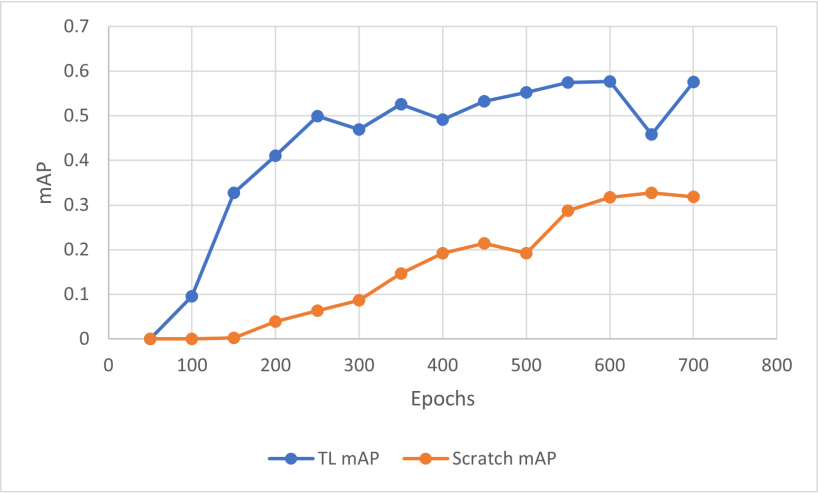


Figure 5.4: mAP performance difference between models utilizing Transfer Learning (TL mAP) and those with randomly initialized weights (Scratch mAP), across various numbers of epochs in training.

5.4 Conclusion

The results from the first experiment indicated that, when using AL techniques, models for object detection can achieve their best performance without requiring all the available data. When selections are made randomly, models necessitate access to the entire dataset. There is a clear compromise to be made: while it is possible to increase performance by adding more data for training, the extra time spent on training and annotation may not be worth it.

By improving the annotations of the data, the model's performance was significantly improved, resulting in an average mAP increase of 1.1 points. Notably, the best model produced using the dataset with enhanced annotations outperformed the model trained on the original dataset in detecting whiteflies. Moreover, the enhanced annotations model identified more whiteflies than those annotated. This observation implies that some detections classified as false positives may actually be accurate detections, possibly due to strict guidelines or image quality issues. These findings underscore the potential of the approach to generate high-quality annotations, even when working with limited data and time constraints.

Furthermore, experiments in the context of TL demonstrated the marked benefits of using TL in the training process. An average increase of 24% in mAP performance was observed coupled with a reduction of 10% in training time. However, when training involved only a handful of examples, these advantages did not always translate into tangible outcomes.

Chapter 6

Conclusion and Future Work

Within smart farming, intelligent pest detection has surpassed traditional methods. However, there is still potential for enhancement. This dissertation proposed a system to detect whitefly pests in natural environments.

Detecting whiteflies in their predominant habitats, such as vegetable greenhouses, offers farmers precise information. This informs them in a timely manner about when to take preventive actions against these pests.

For constructing this system, object detection models, which currently offers state-of-the-art results across various tasks, are employed. Upon reviewing different model designs, the You Only Look Once (YOLO) architecture emerged as the ideal fit due to its speed, efficiency, and ease of deployment.

However, developing this system posed challenges, primarily due to the limited availability of data in the field. This limitation led to the collection of images featuring whitefly-infested tomato leaves. Subsequently, the data annotation phase proved to be one of the most demanding aspects of this work. Given the challenges and time that this step required, exploring techniques to alleviate the process became a priority.

Active Learning (AL) stood out as a potential solution to simplify the annotation process. Its core function is to prioritise the most valuable data for labelling. Although 200 images were already annotated, AL was used in a research context rather than a practical data-selection tool. Results indicated that object detection models could perform optimally without leveraging all available data. However, the minimal performance gains achieved after adding substantial data might not justify the increased training time and annotation efforts.

Following experiments with AL, the focus shifted to exploring other methods that could reduce data needs for training Deep Learning (DL) models. Previous experiments indicated that the trained models were capable of detecting whiteflies in images effectively. Additionally, the predictions were of high quality in terms of location accuracy. This insight led to a deeper interest in improving annotation techniques. Using the predictions of a Machine Learning (ML) model, a new dataset was assembled. From this collection, a subset of images was carefully re-annotated by a human, following strict guidelines. The refined subset was then used as a test set for subsequent experiments. When an experiment was conducted using this dataset, there was a significant improvement, averaging 1.1 mAP points. Importantly, this progress was achieved without additional human effort, as the data used in the training process was annotated using predictions from an object detection model.

The final objective of this research was to explore the benefits derived from leveraging TL techniques. Prior work developed within the research group yielded a model designed to detect whiteflies in controlled environments, such as traps. Subsequently, this model was utilised to provide preliminary knowledge for the training process. The results clearly showcased substantial improvements in performance when adopting these techniques. This not only led to improved accuracy, but also reduced training time.

Future work will focus on researching the AL techniques, especially exploring various scoring functions. Few-Shot Learning (FSL) meta-learning is also emerging as a promising area for future developments, as this research has established a strong basis for the recognition of not only various pests but also other bugs. Given that there are images collected that remain to be annotated, the techniques investigated in this study could facilitate the annotation of these remaining images.

This study did not particularly emphasise the improvements of the model designs, leaving room for future research to explore the optimisation of the model designs.

Having produced multiple object detection models during this research, there is potential for deploying a practical model. The developed models could be integrated into mobile applications, enabling farmers to capture videos or photos on their smartphones to count pests in their crops.

Beyond mobile applications, envisioning a comprehensive greenhouse system that covers all regions is compelling. Imagining a system composed of rollers with an attached camera that travels within the plantation, recording videos and pinpointing whiteflies, suggests a powerful and innovative prototype worth pursuing in the future.

References

Syed Mazhar Abbas and Dr. Shailendra Narayan Singh. Region-based object detection and classification using faster r-cnn. In *2018 4th International Conference on Computational Intelligence & Communication Technology*, pages 1–6, 2018. doi: 10.1109/CIACT.2018.8480413.

Victoria Oluwaseun Adeleye and Dakshina R Seal. Tomato bug, tobacco leaf bug, tomato mirid, green tobacco capsid nesidiocoris tenuis reuter (insecta: Hemiptera: Miridae): Eeny-766/in1323, 7/2021. *EDIS*, 2021(4):5–5, 2021.

Agrocares. Agrocares scoutbox, 2020. URL `https://www.agrocares.com/products/scoutbox/`. (visited on January 2023).

Jordan T Ash, Chicheng Zhang, Akshay Krishnamurthy, John Langford, and Alekh Agarwal. Deep batch active learning by diverse, uncertain gradient lower bounds. *arXiv preprint arXiv:1906.03671*, 2019. doi: 10.48550/arXiv.1906.03671.

Maximilian Bernhard and Matthias Schubert. Correcting imprecise object locations for training object detectors in remote sensing applications. *Remote Sensing*, 13(24), 2021. ISSN 2072-4292. doi: 10.3390/rs13244962.

Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. *arXiv preprint arXiv:2004.10934*, 2020. doi: 10.48550/arXiv.2004.10934.

Bruno Cardoso, Catarina Silva, Joana Costa, and Bernardete Ribeiro. Internet of things meets computer vision to make an intelligent pest monitoring network. *Applied Sciences*, 12(18):9397, 2022. doi: 10.3390/app12189397.

Dinis Costa, Catarina Silva, Joana Costa, and Bernardete Ribeiro. Enhancing pest detection models through improved annotations. In *Progress in Artificial Intelligence*. Springer International Publishing, 2023.

N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *2005 IEEE Computer Society Conference on Computer Vision and Pattern Recognition (CVPR'05)*, volume 1, pages 886–893 vol. 1, 2005. doi: 10.1109/CVPR.2005.177.

Tiago Domingues, Tomás Brandão, Ricardo Ribeiro, and João C Ferreira. Insect detection in sticky trap images of tomato crops using machine learning. *Agriculture*, 12(11):1967, 2022. doi: 10.3390/agriculture12111967.

Li Fei-Fei, R. Fergus, and P. Perona. One-shot learning of object categories. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 28(4):594–611, 2006. doi: 10.1109/TPAMI.2006.79.

Pedro Felzenszwalb, David McAllester, and Deva Ramanan. A discriminatively trained, multiscale, deformable part model. In *2008 IEEE Conference on Computer Vision and Pattern Recognition*, pages 1–8, 2008. doi: 10.1109/CVPR.2008.4587597.

Chelsea Finn, Pieter Abbeel, and Sergey Levine. Model-agnostic meta-learning for fast adaptation of deep networks. In Doina Precup and Yee Whye Teh, editors, *Proceedings of the 34th International Conference on Machine Learning*, volume 70 of *Proceedings of Machine Learning Research*, pages 1126–1135. PMLR, 06–11 Aug 2017. doi: 10.48550/arXiv.1703.03400.

Ross Girshick. Fast r-cnn. In *Proceedings of the IEEE international conference on computer vision*, pages 1440–1448, 2015. doi: https://doi.org/10.48550/arXiv.1504.08083.

Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 580–587, 2014. doi: 10.48550/arXiv.1311.2524.

GulfAgriculture. Feed macrolophus-system for best start in tomatoes, 2022. URL `https://www.gulfagriculture.com/feed-macrolophus-system-for-best-start-in-tomatoes/`. (visited on January 2023).

Yuval Noah Harari. *Sapiens: História breve da humanidade*. Elsinore, 2013.

Elmar Haussmann, Michele Fenzi, Kashyap Chitta, Jan Ivanecky, Hanson Xu, Donna Roy, Akshita Mittel, Nicolas Koumchatzky, Clement Farabet, and Jose M. Alvarez. Active learning techniques and impacts. In *2020 IEEE Intelligent Vehicles Symposium*, page 1430–1435, 2017. doi: 10.1109/IV47402.2020.9304793.

Elmar Haussmann, Michele Fenzi, Kashyap Chitta, Jan Ivanecky, Hanson Xu, Donna Roy, Akshita Mittel, Nicolas Koumchatzky, Clement Farabet, and Jose M. Alvarez. Scalable active learning for object detection. In *2020 IEEE Intelligent Vehicles Symposium*, pages 1430–1435, 2020. doi: 10.1109/IV47402.2020.9304793.

Wei-Ning Hsu and Hsuan-Tien Lin. Active learning by learning. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 29, 2015. doi: 10.1609/aaai.v29i1.9597.

Glenn Jocher, Ayush Chaurasia, Alex Stoken, Jirka Borovec, NanoCode012, Yonghye Kwon, Kalen Michael, TaoXie, Jiacong Fang, imyhxy, Lorna, Zeng Yifu, Colin Wong, Abhiram V, Diego Montes, Zhiqiang Wang, Cristi Fati, Jebastin Nadar, Laughing, UnglvKitDe, Victor Sonck, tkianai, yxNONG, Piotr

Skalski, Adam Hogan, Dhruv Nair, Max Strobel, and Mrinal Jain. ultralytics/yolov5: v7.0 - YOLOv5 SOTA Realtime Instance Segmentation, November 2022.

Glenn Jocher, Ayush Chaurasia, and Jing Qiu. Yolo by ultralytics, January 2023. URL `https://github.com/ultralytics/ultralytics`.

Andreas Kamilaris and Francesc Prenafeta Boldú. Deep learning in agriculture: A survey. *Computers and Electronics in Agriculture*, 147, 04 2018. doi: 10.1016/j.compag.2018.02.016.

Alina Kuznetsova, Hassan Rom, Neil Alldrin, Jasper Uijlings, Ivan Krasin, Jordi Pont-Tuset, Shahab Kamali, Stefan Popov, Matteo Malloci, Alexander Kolesnikov, Tom Duerig, and Vittorio Ferrari. The open images dataset v4. *International Journal of Computer Vision*, 128(7):1956–1981, mar 2020. doi: 10.1007/s11263-020-01316-z.

Yann LeCun, Yoshua Bengio, and Geoffrey Hinton. Deep learning. *nature*, 521 (7553):436–444, 2015. doi: 10.1038/nature14539.

Chuyi Li, Lulu Li, Hongliang Jiang, Kaiheng Weng, Yifei Geng, Liang Li, Zaidan Ke, Qingyuan Li, Meng Cheng, Weiqiang Nie, et al. Yolov6: A single-stage object detection framework for industrial applications. *arXiv preprint arXiv:2209.02976*, 2022. doi: 10.48550/arXiv.2209.02976.

Zeming Li, Chao Peng, Gang Yu, Xiangyu Zhang, Yangdong Deng, and Jian Sun. Light-head r-cnn: In defense of two-stage object detector. *arXiv preprint arXiv:1711.07264*, 2017. doi: 10.48550/arXiv.1711.07264.

Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In David Fleet, Tomas Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, *Computer Vision – ECCV 2014*, pages 740–755, Cham, 2014. Springer International Publishing. ISBN 978-3-319-10602-1. doi: 10.48550/arXiv.1405.0312.

Jiaxin Ma, Yoshitaka Ushiku, and Miori Sagara. The effect of improving annotation quality on object detection datasets: A preliminary study. In *2022 IEEE/CVF Conference on Computer Vision and Pattern Recognition Workshops (CVPRW)*, pages 4849–4858, 2022. doi: 10.1109/CVPRW56347.2022.00532.

Vasileios Moysiadis, Panagiotis Sarigiannidis, Vasileios Vitsas, and Adel Khelifi. Smart farming in europe. *Computer Science Review*, 39:100345, 2021. ISSN 1574-0137. doi: 10.1016/j.cosrev.2020.100345.

A. T. Nieuwenhuizen, Jochen Hemming, and Hyun K. Suh. Detection and classification of insects on stick-traps in a tomato crop using faster r-cnn. In *The Netherlands Conference on Computer Vision*, 2018. URL `https://api.semanticscholar.org/CorpusID:69451220`.

Curtis G Northcutt, Anish Athalye, and Jonas Mueller. Pervasive label errors in test sets destabilize machine learning benchmarks. *arXiv preprint arXiv:2103.14749*, 2021. doi: 10.48550/arXiv.2103.14749.

Kuntal Kumar Pal and K. S. Sudeep. Preprocessing for image classification by convolutional neural networks. In *2016 IEEE International Conference on Recent Trends in Electronics, Information & Communication Technology*, pages 1778–1781, 2016. doi: 10.1109/RTEICT.2016.7808140.

Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010. doi: 10.1109/TKDE.2009.191.

Archit Parnami and Minwoo Lee. Learning from few examples: A summary of approaches to few-shot learning, 2022.

Chenesh Patel, Ravi Srivastava, and Jeevakani Samraj. Comparative study of morphology and developmental biology of two agriculturally important whitefly species bemisia tabaci (asia ii 5) and trialeurodes vaporariorum from north-western himalayan region of india. *Brazilian Archives of Biology and Technology*, 65, 04 2022. doi: 10.1590/1678-4324-2022210034.

Joseph Redmon and Ali Farhadi. Yolo9000: better, faster, stronger. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 7263–7271, 2017. doi: 10.48550/arXiv.1612.08242.

Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018. doi: https://doi.org/10.48550/arXiv.1804.02767.

Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 779–788, 2016. doi: 10.48550/arXiv.1506.02640.

Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. *Advances in neural information processing systems*, 28, 2015. doi: 10.48550/arXiv.1506.01497.

Sumit Saha. A comprehensive guide to convolutional neural networks—the eli5 way. *Towards data science*, 15:15, 2018.

Ibrahim Sani, Siti Izera Ismail, Sumaiyah Abdullah, Johari Jalinas, Syari Jamian, and Norsazilawati Saad. A review of the biology and control of whitefly, bemisia tabaci (hemiptera: Aleyrodidae), with special reference to biological control using entomopathogenic fungi. *Insects*, 11(9), 2020. ISSN 2075-4450. doi: 10.3390/insects11090619.

Juan Terven and Diana-Margarita Cordova-Esparza. A comprehensive review of yolo: From yolov1 to yolov8 and beyond, 04 2023.

P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *Proceedings of the 2001 IEEE Computer Society Conference on Computer Vision and Pattern Recognition. CVPR 2001*, volume 1, pages I–I, 2001. doi: 10.1109/CVPR.2001.990517.

Paul Viola and Michael J Jones. Robust real-time face detection. *International journal of computer vision*, 57:137–154, 2004. doi: 10.1023/B:VISI.0000013087.49260.fb.

Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. Scaled-yolov4: Scaling cross stage partial network. In *Proceedings of the IEEE/cvf conference on computer vision and pattern recognition*, pages 13029–13038, 2021. doi: 10.48550/arXiv.2011.08036.

Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7464–7475, 2023. doi: 10.48550/arXiv.2207.02696.

Shaoru Wang, Jin Gao, Bing Li, and Weiming Hu. Narrowing the gap: Improved detector training with noisy location annotations. *Trans. Img. Proc.*, 31: 6369–6380, jan 2022. ISSN 1057-7149. doi: 10.1109/TIP.2022.3211468.

Yaqing Wang, Quanming Yao, James T. Kwok, and Lionel M. Ni. Generalizing from a few examples: A survey on few-shot learning. *ACM Comput. Surv.*, 53 (3), jun 2020. ISSN 0360-0300. doi: 10.1145/3386252.

Xingjiao Wu, Luwei Xiao, Yixuan Sun, Junhang Zhang, Tianlong Ma, and Liang He. A survey of human-in-the-loop for machine learning. *Future Generation Computer Systems*, 135:364–381, 2022. ISSN 0167-739X. doi: 10.1016/j.future.2022.05.014.

Daochen Zha, Zaid Pervaiz Bhat, Kwei-Herng Lai, Fan Yang, Zhimeng Jiang, Shaochen Zhong, and Xia Hu. Data-centric artificial intelligence: A survey. *arXiv preprint arXiv:2303.10158*, 2023. doi: https://doi.org/10.48550/arXiv.2303.10158.

Xinyi Zhou, Wei Gong, WenLong Fu, and Fengtong Du. Application of deep learning in object detection. In *2017 IEEE/ACIS 16th International Conference on Computer and Information Science*, pages 631–634, 2017. doi: 10.1109/ICIS.2017.7960069.

Zhengxia Zou, Keyan Chen, Zhenwei Shi, Yuhong Guo, and Jieping Ye. Object detection in 20 years: A survey. *Proceedings of the IEEE*, 2023. doi: 10.48550/arXiv.1905.05055.