

FACULDADE DE CIÊNCIAS DA UNIVERSIDADE DO PORTO
FACULDADE DE ENGENHARIA DA UNIVERSIDADE DO PORTO

AI Assistant for Pest Detection: Vision-Language Interface for Olive Fruit Fly Monitoring

Bernardo Gil Alves Salgado



Mestrado em Inteligência Artificial

Supervisor: Prof. Hélder Oliveira

Second Supervisor: Margarida Victoriano

July 9, 2026

AI Assistant for Pest Detection: Vision-Language Interface for Olive Fruit Fly Monitoring

Bernardo Gil Alves Salgado

Mestrado em Inteligência Artificial

Approved in oral examination by the committee:

President: Prof. Henrique Cardoso

Referee: Prof. Inês Domingues

July 9, 2026

Resumo

A mosca-da-azeitona (*Bactrocera oleae*) é uma das pragas mais prejudiciais para a olivicultura, causando perdas significativas de produção e degradação da qualidade do azeite. Uma gestão eficaz desta praga depende da monitorização atempada das suas populações. No entanto, os métodos tradicionais de monitorização baseiam-se maioritariamente na inspeção manual de armadilhas, um processo demorado e com capacidade limitada para apoiar a tomada de decisão.

Esta dissertação investiga a integração de Modelos de Visão-Linguagem em sistemas de monitorização da mosca-da-azeitona, com o objetivo de melhorar a interpretabilidade dos resultados e a interação com os utilizadores. Foi desenvolvida uma pipeline de deteção visual baseada em Deep Learning (DL), recorrendo a arquiteturas You Only Look Once (YOLO) treinadas e avaliadas num conjunto de dados obtido pela combinação do dataset DIRT com um dataset recolhido em olivais portugueses. Foram comparadas diferentes composições de dados, estratégias de pré-processamento e arquiteturas YOLO. Os resultados experimentais demonstraram que o YOLOv7 apresentou o melhor desempenho global entre os modelos avaliados.

Para complementar a deteção visual, foi desenvolvido um Assistente de Visão-Linguagem baseado no modelo Large Language and Vision Assistant (LLaVA) 1.5, ajustado ao domínio através da técnica Low-Rank Adaptation (LoRA) e de um conjunto de dados específico para monitorização da mosca-da-azeitona. O sistema resultante combina os resultados da deteção automática com capacidades de raciocínio multimodal, permitindo aos utilizadores consultar os resultados através de linguagem natural e obter explicações sobre o número de insetos detetados, níveis de infestação e distribuição espacial das deteções.

A solução proposta integra com sucesso a deteção automática da mosca-da-azeitona com interação baseada em linguagem natural, demonstrando o potencial dos sistemas multimodais de inteligência artificial para melhorar a interpretabilidade, acessibilidade e utilidade de ferramentas de monitorização agrícola e apoio à decisão.

Abstract

The olive fruit fly (*Bactrocera oleae*) is one of the most damaging pests affecting olive cultivation, causing significant yield losses and reductions in olive oil quality. Effective pest management depends on timely and accurate monitoring of pest populations. However, traditional monitoring methods rely heavily on manual trap inspection and provide limited support for decision-making.

This dissertation investigates the integration of Vision-Language Models (VLMs) into olive fruit fly monitoring systems to improve interpretability and user interaction. A visual detection pipeline was developed using Deep Learning (DL) based object detection models trained and evaluated on a dataset created by combining the DIRT dataset with a Portuguese olive grove dataset. Different dataset compositions, preprocessing strategies and You Only Look Once (YOLO) architectures were evaluated. Experimental results showed that YOLOv7 achieved the best overall performance among the evaluated models.

To complement visual detection, a Vision-Language Assistant based on Large Language and Vision Assistant (LLaVA) 1.5 was fine-tuned using Low-Rank Adaptation (LoRA) and a domain-specific dataset composed of olive fruit fly monitoring images and instruction-answer pairs. The resulting system combines object detection outputs with multimodal reasoning, enabling users to query monitoring results through natural language and receive explanations regarding insect counts, infestation levels and spatial distribution.

The proposed framework successfully integrates automated olive fruit fly detection with language-based interaction, demonstrating the potential of multimodal Artificial Intelligence (AI) systems to improve the interpretability, accessibility and usability of agricultural monitoring tools and decision-support systems.

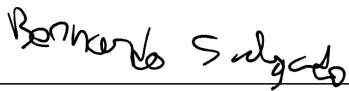
Declaration of honour

I declare that this dissertation is my original work and has not previously been submitted or assessed in any other course or curricular unit, whether at this or any other institution.

References to other authors (statements, ideas, thoughts), as well as the use of published works fully comply with the rules of attribution and are appropriately cited in the text and bibliography, in accordance with reference standards. I am aware that plagiarism and self-plagiarism constitute academic misconduct.

I further declare that I have used Artificial Intelligence tools as assistive tools during the preparation of this dissertation, namely to improve text clarity and assist with code-related questions. Their use and results were carefully reviewed and validated before incorporation. All scientific decisions, analyses, interpretations and conclusions remain entirely my own responsibility.

Bernardo Gil Alves Salgado



Acknowledgements

I would like to express my sincere gratitude to my supervisor, Professor Hélder Oliveira, for his guidance, continuous support and valuable feedback throughout the development of this thesis. His expertise, encouragement and confidence in this work were fundamental to its successful completion.

I am also deeply grateful to my co-supervisor, Professor Margarida Victoriano, for her availability, insightful advice and constructive discussions throughout this research. Her support and recommendations greatly contributed to improving the quality of this dissertation.

I would like to thank INESC TEC for providing the resources, infrastructure and research environment that made this work possible. The opportunity to carry out this research within such a stimulating environment has greatly contributed to both my academic and personal growth.

I am also sincerely grateful to my friends for their support, encouragement and understanding throughout this journey. Knowing that I could always count on them made even the most demanding moments easier to overcome.

Finally, I would like to express my deepest gratitude to my family for their unconditional love, patience, and encouragement throughout this journey. Their constant support has been my greatest source of motivation and strength, especially during the most challenging moments. Without their trust and belief in me, the completion of this dissertation would not have been possible.

Bernardo Gil Alves Salgado

Contents

List of Acronyms	x
1 Introduction	1
1.1 Motivation	1
1.2 Objectives	2
1.3 Contributions	2
1.4 Document Structure	2
2 Literature Review	3
2.1 Background	3
2.1.1 Olive Cultivation: Economic, Agronomical and Biological Context . . .	3
2.1.2 Impact of Olive Fruit Fly Infestation on Yield and Quality	4
2.1.3 Limitations of Conventional Pest Management and Monitoring	6
2.1.4 Digital Pest Monitoring, Decision Support and Adoption Challenges . . .	6
2.1.5 Summary	6
2.2 Datasets	7
2.2.1 DIRT Dataset	7
2.2.2 Portuguese Olive Grove Dataset	8
2.2.3 Large-Scale Insect Classification Dataset	8
2.2.4 ROI-Based Chromotropic Trap Dataset	9
2.2.5 E-Trap Dataset	11
2.2.6 Pepper Diseases and Pests Dataset	11
2.2.7 IDADP Dataset	11
2.2.8 WPIT9K and IP102 Datasets	12
2.3 Vision Based Methods for Olive Fly Detection	13
2.3.1 Edge based Smart Traps for Olive Fly Monitoring	16
2.4 Multimodal Approaches	18
2.5 VLM Approaches	21
2.6 Summary	24
3 Methodology	27
3.1 Visual Detection Pipeline	27
3.1.1 Dataset Compositions	28
3.1.2 Dataset Splitting Strategy	30
3.1.3 YOLO Architectures	30
3.1.4 Experimental Configuration	32
3.2 Vision-Language Model Development	33
3.2.1 Fine-Tuning Dataset Preparation	34

3.2.2	LoRA Fine-Tuning	35
3.2.3	Fine-Tuning Experiments	35
3.2.4	Final VLM Configuration	36
3.2.5	System Integration	36
3.2.6	Web Application Architecture	38
4	Experimental Results	39
4.1	Experimental Setup Overview	39
4.2	Overall Comparison of YOLO Architectures	39
4.3	Cross-Validation Results and Generalization	41
4.3.1	Olive Only Dataset	41
4.3.2	Olive Only Cropped Dataset	41
4.3.3	Force Cropped Dataset	42
4.4	Impact of Dataset Compositions	43
4.5	Impact of Class Balancing Strategies	43
4.6	Qualitative Detection Results	44
4.6.1	Training Behaviour	44
4.6.2	Precision-Recall Analysis	45
4.6.3	Confusion Matrix	46
4.6.4	Detection Examples	46
4.6.5	Qualitative Comparison of Alternative YOLO Architectures	47
4.6.6	Hyperparameter Fine-Tuning	51
4.6.7	Fine-Tuning Discussion	60
4.7	Object Detection Summary	60
4.8	Vision-Language Assistant Evaluation	60
4.8.1	Vision-Language Interaction Examples	60
4.8.2	Limitations and Error Analysis	62
4.9	Summary	63
5	Conclusions and Future Work	64
5.1	Discussion	64
5.2	Conclusions	65
5.3	Future Work	65
	References	67
A	Multilingual Interaction Capability	71

List of Figures

2.1	Damage caused by <i>Bactrocera oleae</i>	5
2.2	Global distribution of <i>Bactrocera oleae</i>	5
2.3	DIRT dataset samples	7
2.4	DIRT annotations	8
2.5	Portuguese dataset samples	8
2.6	Dataset sample images	9
2.7	ROI segmentation process	10
2.8	ROI patch examples	10
2.9	E-trap sample targets	11
2.10	Pepper Diseases and Pests Dataset	12
2.11	WPIT9K image-text correspondence	12
2.12	Yellow mean normalization	13
2.13	Data augmentation techniques	14
2.14	YOLOv7 detection examples	15
2.15	E-trap components	17
2.16	Inference pipeline	18
2.17	PepperDet architecture	19
2.18	MMAE architecture	19
2.19	Dual-(PAL) ^G architecture	20
2.20	MPCD qualitative results	21
2.21	AgriDetectVL architecture	22
2.22	Agri-LLaVA interaction examples	23
2.23	Multimodal VLM architecture	23
2.24	LLMI-CDP architecture	24
2.25	Cognitive task comparison	25
3.1	Example image from the original merged dataset	28
3.2	Example image after the first cropping procedure	29
3.3	Example image using the second cropping preprocessing strategy	30
3.4	YOLOv5 architecture	31
3.5	YOLOv7 architecture	31
3.6	YOLOv8 architecture	32
3.7	LLaVA 1.5 architecture	34
3.8	Overall architecture of the proposed monitoring system	37
3.9	Web application architecture	38
4.1	Final YOLOv7 training and validation curves	45
4.2	Final YOLOv7 Precision-Recall curve	45

4.3	YOLOv7 normalized confusion matrix	46
4.4	Final YOLOv7 detections on unseen test images	47
4.5	YOLOv5s training and evaluation results	48
4.6	YOLOv5m training and evaluation results	49
4.7	YOLOv8 training and evaluation results	50
4.8	Best YOLOv5s fine-tuning evaluation results	53
4.9	Best YOLOv5m fine-tuning evaluation results	55
4.10	Best YOLOv7 fine-tuning evaluation results	57
4.11	Best YOLOv8 fine-tuning evaluation results	59
4.12	Example interaction with the proposed YOLOv7-LLaVA system	61
4.13	Example of an inconsistent VLM response	62

List of Tables

3.1	Summary of the dataset compositions	28
3.2	Initial training configuration used across YOLO architectures	33
3.3	Final LLaVA-LoRA configuration used in the integrated system	36
4.1	Best mAP@0.5:0.95 across dataset configurations	39
4.2	Best individual result obtained for each YOLO architecture	40
4.3	Cross-validation results obtained using the Olive Only dataset	41
4.4	Cross-validation results obtained using the Olive Only Cropped dataset	42
4.5	Cross-validation results obtained using the Force Cropped dataset	42
4.6	YOLOv7 performance obtained for the evaluated preprocessing strategies	43
4.7	Impact of balancing strategies on YOLOv7 performance	43
4.8	YOLOv5s hyperparameter fine-tuning results	52
4.9	YOLOv5m hyperparameter fine-tuning results	54
4.10	YOLOv7 hyperparameter fine-tuning results	56
4.11	YOLOv8 hyperparameter fine-tuning results	58

List of Acronyms

AI	Artificial Intelligence
CNN	Convolutional Neural Network
CV	Computer Vision
DL	Deep Learning
DPMS	Digital Pest Management System
DSS	Decision Support System
HOG	Histogram of Oriented Gradients
LLaVA	Large Language and Vision Assistant
LMM	Large Multimodal Model
LoRA	Low-Rank Adaptation
MAE	Masked Autoencoder
mAP	mean Average Precision
MMAE	Multimodal Masked Autoencoder
MPCD	Multimodal Prior and Conditional Detection
NLP	Natural Language Processing
OD	Object Detection
PCA	Principal Component Analysis
PEFT	Parameter-Efficient Fine-Tuning
RGB	Red, Green and Blue
ROI	Region of Interest
SVM	Support Vector Machine
VLM	Vision-Language Model
YOLO	You Only Look Once

Chapter 1

Introduction

Olive cultivation plays a central role in Mediterranean agriculture, both economically and culturally [37]. However olive production faces persistent challenges related to pests, environmental stress and increasing pressure to improve efficiency and sustainability [37]. Among these challenges, the olive fruit fly (*Bactrocera oleae*) stands out as one of the most damaging biological threats [26, 31], capable of causing severe yield losses and significant degradation of olive oil quality [26].

Effective pest management relies heavily on accurate monitoring of pest populations [11, 30]. In practice, olive fruit fly monitoring is still largely based on manual inspection of insect traps, a process that is slow [30]. As a result, management decisions are often made too late, after damage has already been done [11].

1.1 Motivation

Recent advances in Artificial Intelligence (AI) have enabled automated analysis of trap images using Computer Vision (CV) techniques [28, 30, 38], enabling continuous monitoring [11, 30]. However, most existing systems focus exclusively on visual detection, producing technical outputs such as bounding boxes or confidence scores that are not always intuitive for end users [27]. This creates a gap between automated detection capabilities and practical decision support in real-world agricultural settings [11, 27].

While vision-based pest detection systems have demonstrated strong performance under both controlled and real field conditions, their outputs remain limited to low-level visual information. Farmers must still interpret these outputs, assess risk levels and decide on appropriate interventions, often without clear explanations or contextual guidance [28, 30].

At the same time, recent developments in large multimodal and Vision-Language Models (VLMs) have introduced new possibilities for combining visual perception with Natural Language Processing (NLP). These models enable systems that, not only detect objects in images but also explain visual content in a way that is accessible to human users [13, 22, 39, 41, 44].

In the context of agricultural pest monitoring, this creates an opportunity to move beyond purely technical detection toward interactive and interpretable Decision Support Systems (DSSs).

1.2 Objectives

The main objective of this dissertation is to investigate the use of VLMs for olive fruit fly monitoring based on insect trap images. To achieve this goal, the following objectives are defined:

- Study the applicability of VLMs to agricultural pest monitoring scenarios.
- Explore the integration of visual pest detection and multimodal reasoning within a unified monitoring framework.
- Assess the potential of VLMs to improve the interpretability of pest monitoring results and facilitate user interaction.
- Evaluate the usefulness of multimodal AI systems as decision-support tools for agricultural monitoring applications.

1.3 Contributions

This dissertation contributes to the field of agricultural AI through the following contributions:

- The proposal and implementation of a monitoring framework that combines olive fruit fly detection with Vision-Language interaction capabilities.
- The creation of a domain-specific multimodal pipeline integrating a YOLOv7 object detector and a fine-tuned LLaVA-based assistant.
- An experimental evaluation of different object detection architectures for olive fruit fly monitoring using combined public and field-collected datasets.
- An assessment of the potential of Vision-Language Models to provide more interpretable and user-oriented monitoring outputs for agricultural applications.

1.4 Document Structure

The remainder of this document is organized as follows. The *Literature Review* chapter presents the background context of olive cultivation, pest management challenges, digital agriculture, and reviews existing vision-based, multimodal and VLM approaches for pest detection and monitoring. The *Methodology* chapter describes the proposed approach, including the visual detection pipeline, the VLM assistant and the evaluation plan. The *Experimental Results* chapter presents the experimental results and their analysis. Finally, the *Conclusions and Future Work* chapter summarizes the main findings of this dissertation and discusses directions for future work.

Chapter 2

Literature Review

This chapter reviews the state of the art in automated olive fruit fly monitoring. It first presents the background necessary to contextualize the problem, including the importance of olive cultivation, the impact of the olive fruit fly and the limitations of conventional pest monitoring approaches. A dedicated Datasets section then consolidates the data sources recurrently used across the literature, providing a common reference and avoiding repetition when discussing individual works. The rest of the chapter is organized according to the level of semantic and multimodal integration in the analyzed approaches:

- (i) *vision-based methods*, which operate solely on images and typically output detections or class decisions;
- (ii) *multimodal approaches*, which combine visual inputs with textual descriptions through task-specific combination mechanisms;
- (iii) *VLM approaches*, which leverage large pretrained models to enable visual understanding joined with natural language interaction.

This organization reflects a progression from accurate but task-specific visual pipelines toward more flexible and explainable systems that incorporate language-based reasoning.

2.1 Background

This section provides the agricultural and technological context for the proposed work. It first presents the economic and agronomical relevance of olive cultivation and discusses the impact of the olive fruit fly on crop yield and olive oil quality. It then reviews conventional pest monitoring approaches, their limitations and recent developments in digital pest management and DSSs, highlighting the challenges that motivate the integration of VLMs into agricultural monitoring applications.

2.1.1 Olive Cultivation: Economic, Agronomical and Biological Context

Olive cultivation is a major agricultural activity in the Mediterranean region and an important source of income and employment for rural communities. In Italy, olive farming is particularly

relevant in southern regions, where it contributes significantly to local economies and rural development. The sector therefore plays an important role in the agricultural landscape of Mediterranean countries and remains closely linked to their economic and social sustainability [37].

Despite its importance, olive production systems are often affected by structural and economic constraints that threaten their long-term sustainability. Many olive farms are small to medium-sized and are characterized by traditional orchard structures with low productivity and limited mechanization, which contribute to high production costs [37]. In addition, the economic performance of olive farms is strongly influenced by management practices and investment decisions throughout the orchard life cycle. Labor-intensive operations, particularly pruning and harvesting, represent some of the main production costs and have a significant impact on farm profitability [37]. These challenges highlight the need for more efficient management strategies and innovative solutions capable of reducing operational burdens, improving productivity and supporting better decision-making in olive farming [30, 37].

Olive cultivation in Mediterranean environments is strongly influenced by a combination of abiotic and biotic stress factors, including extreme temperatures, drought, salinity, pollution, pests and pathogens. These pressures can affect tree growth, fruit development and olive oil quality, despite the adaptive traits that allow olive trees to thrive under semi-arid conditions. Their interaction with cultivation practices and orchard management decisions contributes to the complexity of olive production systems and reinforces the importance of effective monitoring and informed decision-making [2].

The main stages of infestation and the resulting physical damage caused by the olive fruit fly are illustrated in Figure 2.1.

Bactrocera oleae, commonly known as the olive fruit fly, is considered the most important pest affecting olive cultivation in Mediterranean regions. Infestations can cause significant yield losses and reduce olive oil quality, leading to important economic impacts for olive producers [31]. The species has multiple overlapping generations per year, and its life cycle is closely associated with olive fruit development, which contributes to the complexity of its management [31]. Female flies lay eggs inside olive fruits, where larvae develop by feeding on the pulp, causing direct damage to the crop [31]. For these reasons, monitoring *Bactrocera oleae* populations is an important component of effective pest management strategies [31].

The geographical distribution of *Bactrocera oleae* closely follows the regions where olive cultivation is established. The worldwide distribution of the pest is shown in Figure 2.2.

2.1.2 Impact of Olive Fruit Fly Infestation on Yield and Quality

In highly susceptible table olive cultivars, the lack of effective control measures can result in severe production losses, in some cases approaching total crop failure [26]. Infestation has also been linked to increased free acidity, enhanced oxidative degradation and a reduction in phenolic compounds, negatively affecting the chemical and sensory properties of olive oil [26]. As infestation levels increase, these quality changes may lead to the downgrading of olive oil from extra virgin to lower commercial categories [26].

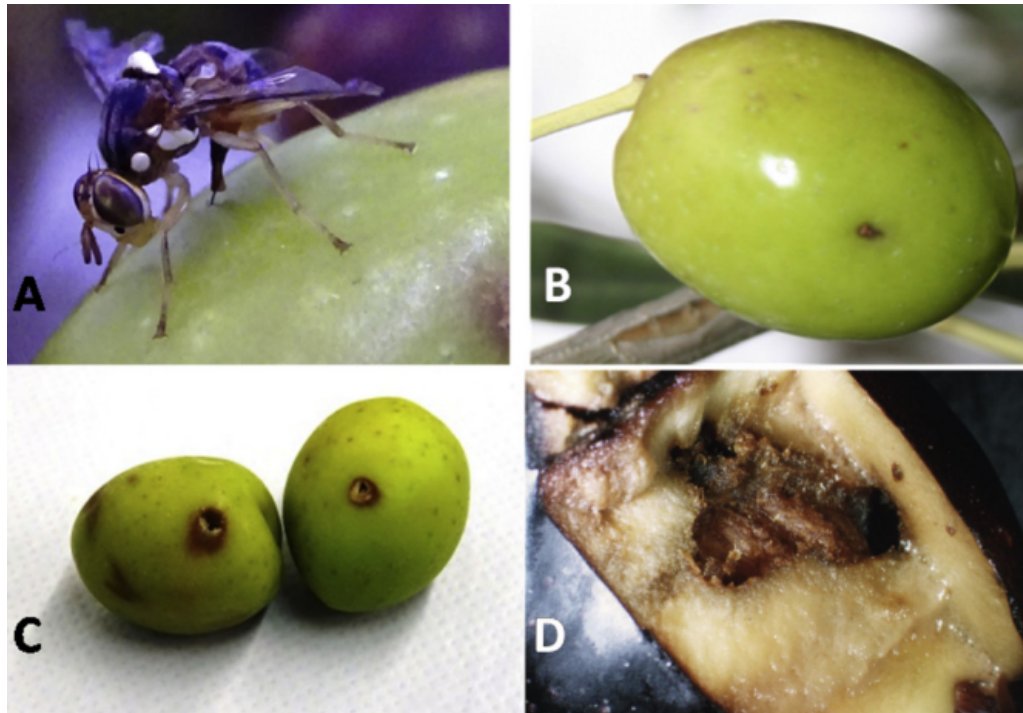


Figure 2.1: Damage caused by *Bactrocera oleae* during different stages of infestation: (A) olive fly ovipositing on an olive fruit; (B) oviposition site visible on the olive surface; (C) exit holes produced by olive fly larvae; and (D) internal damage caused by larval feeding within the olive pulp. Adapted from [26].

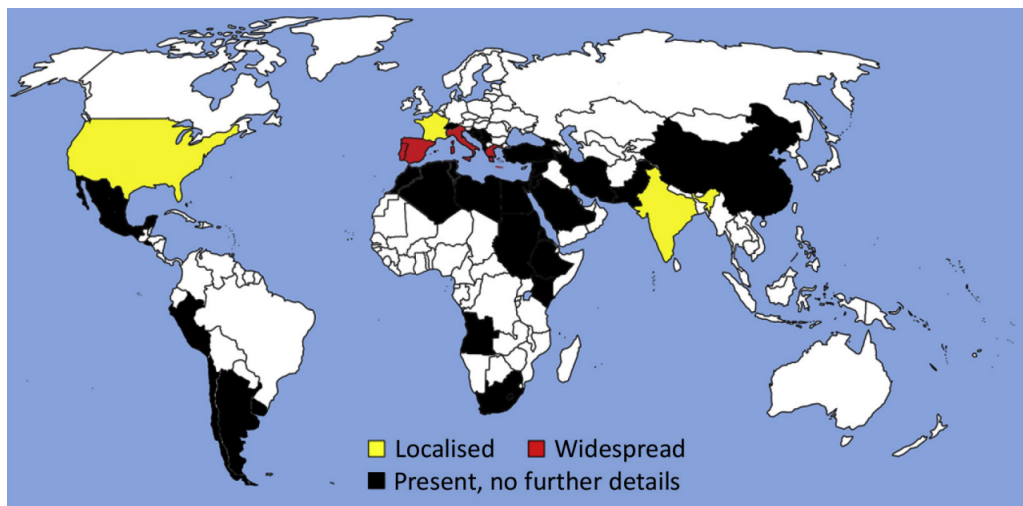


Figure 2.2: Global distribution of *Bactrocera oleae* across major olive-growing regions. Adapted from [26].

2.1.3 Limitations of Conventional Pest Management and Monitoring

The biology of *Bactrocera oleae* presents challenges for pest management approaches that rely exclusively on monitoring and interventions applied only after infestation levels exceed predefined thresholds. Since larval development occurs within the fruit, economically relevant damage can occur before infestations are easily detected. So, studies on fruit fly management have shown that pest populations may increase substantially before action thresholds are reached, limiting the effectiveness of reactive control measures [25]. These constraints highlight the value of DSS tools that support earlier risk assessment and more well timed management decisions [25].

Conventional trap-based monitoring often relies on periodic manual inspections, with trapped insects being collected, identified and counted at regular intervals [30]. Because trap observations are not continuously available, increases in pest populations may only be detected after a delay, limiting the ability to respond rapidly to emerging infestations [30]. These limitations motivate the adoption of automated monitoring systems capable of providing more timely information and supporting more precise pest management decisions [30].

2.1.4 Digital Pest Monitoring, Decision Support and Adoption Challenges

Recent years have seen a growing digital transformation of agriculture, driven by the adoption of precision agriculture principles and data driven management practices. Digital tools such as automated traps, remote sensing platforms and interconnected sensor networks are increasingly integrated into pest management strategies to enable rapid and continuous monitoring [11, 28, 30]. These technologies provide real-time information on pest dynamics, crop status and environmental conditions, supporting earlier and more localized interventions [11, 30, 38]. However, the increasing availability of data does not automatically translate into effective decisions, as Digital Pest Management Systems (DPMSs) still face challenges related to data interpretation and integration with agronomical knowledge [11]. This persistent gap between data generation and human decision-making remains a key limitation in the practical adoption of DPMSs [11, 27].

The adoption of digital technologies in agriculture is strongly influenced by farmers' trust, perceived usefulness and understanding of how these systems operate. Limited access to information, technical assistance and user education, together with low levels of transparency and interpretability, can represent important barriers to technology adoption in smart farming contexts. Improving explainability and user understanding is therefore essential for enabling farmers to assess risks, understand system behaviour and develop confidence in digital DSSs, supporting their effective integration into agricultural practice. [27]

2.1.5 Summary

The analysis highlight the severe economic and quality losses associated with olive fruit fly infestation and the cons of reactive pest management strategies based on late intervention.

Traditional monitoring practices were shown to be insufficient for supporting well timed decision making, while recent advances in digital and precision agriculture offer new opportunities

for automated and data driven pest monitoring. However, challenges related to data interpretation, trust and explainability continue to limit the practical adoption of **DPMSs**. These factors motivate the need for **DSSs** that combine automated monitoring with interpretable and user oriented outputs.

2.2 Datasets

This section consolidates the datasets used across the reviewed literature, which are recurrently employed by multiple vision-based, multimodal and **VLM** approaches for agricultural pest detection.

Field-collected trap imagery typically presents several challenges, including strong illumination variability, shadows, insect debris, visually similar non-target species and inconsistent insect orientations. To cope with these issues, studies often explore different input configurations and select the combinations that provide the best trade-off between robustness and computational efficiency [29].

2.2.1 DIRT Dataset

The Dacus Image Recognition Toolkit (DIRT) [16] is one of the most widely used datasets for olive fruit fly detection. It consists of 848 Red, Green and Blue (**RGB**) images captured using McPhail pheromone traps, where each image contains at least one instance of *Bactrocera oleae*, as in Figure 2.3. The insects are annotated with bounding boxes by entomology experts using LabelImg [10], as in Figure 2.4.

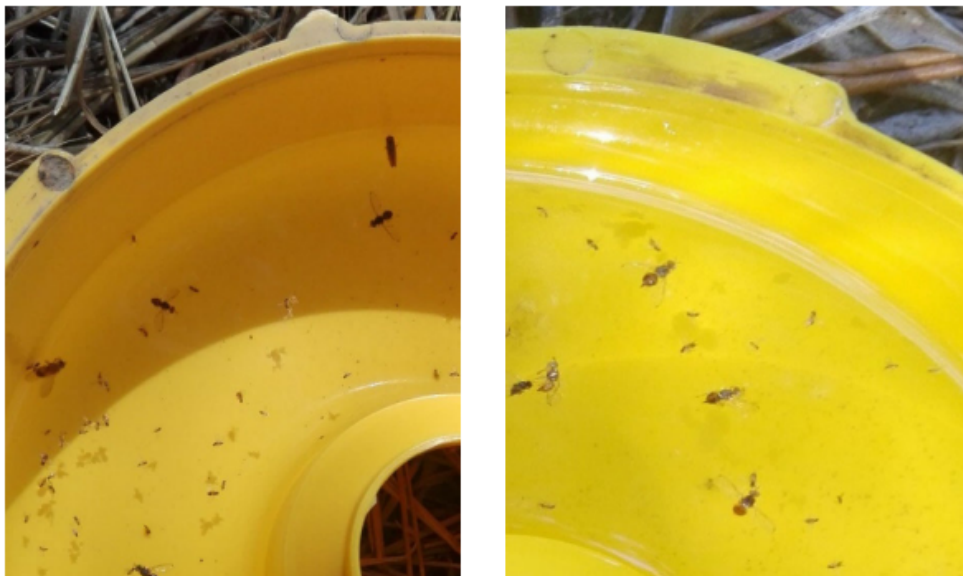


Figure 2.3: Samples from the DIRT Dataset [28]

DIRT includes single-class bounding-box annotations with slicing-based augmentation.

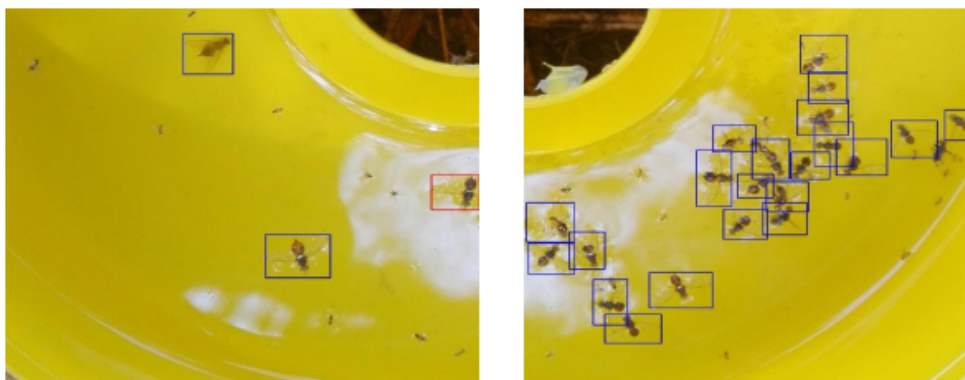


Figure 2.4: Annotated samples from the DIRT Dataset [28]

It is used as the primary training dataset in the You Only Look Once (YOLO) based framework proposed by [28] and is also employed by [38] as the main training source for their YOLO v7-based detector.

2.2.2 Portuguese Olive Grove Dataset

In addition to DIRT, the Portuguese Olive Grove Dataset consists of images collected from yellow sticky traps deployed in a Portuguese olive orchard [38]. The images were acquired under real field conditions and include variations in illumination, insect density and background clutter, as shown in Figure 2.5.



Figure 2.5: Example images from the Portuguese olive grove dataset captured using yellow sticky traps, illustrating different acquisition conditions and insect densities [38].

2.2.3 Large-Scale Insect Classification Dataset

A large-scale insect image dataset comprising more than 75,000 images across 102 insect categories [33]. The dataset contains cropped images of individual insects, including *Bactrocera oleae* and visually similar non-target species, as illustrated in Figure 2.6.

(a) *Dacus* insect

(b) Other insect

Figure 2.6: Sample images from the insect dataset used for binary classification, illustrating examples of *Bactrocera oleae* and non-target insect species, adapted from [33].

This dataset is suitable for benchmarking different Convolutional Neural Network (**CNN**) architectures. However, it does not include full trap images or localization annotations, limiting its applicability to real-world detection scenarios.

2.2.4 ROI-Based Chromotropic Trap Dataset

The ROI-Based Chromotropic Trap Dataset contains 3043 image patches extracted from chromotropic trap images acquired under real field conditions, as in Figure 2.7 [29]. The dataset includes 358 positive samples containing *Bactrocera oleae* and 2685 negative samples correspond-

ing to non-target insects or other objects. All samples were annotated by specialists and extracted as Regions of Interest (ROIs) from full trap images.

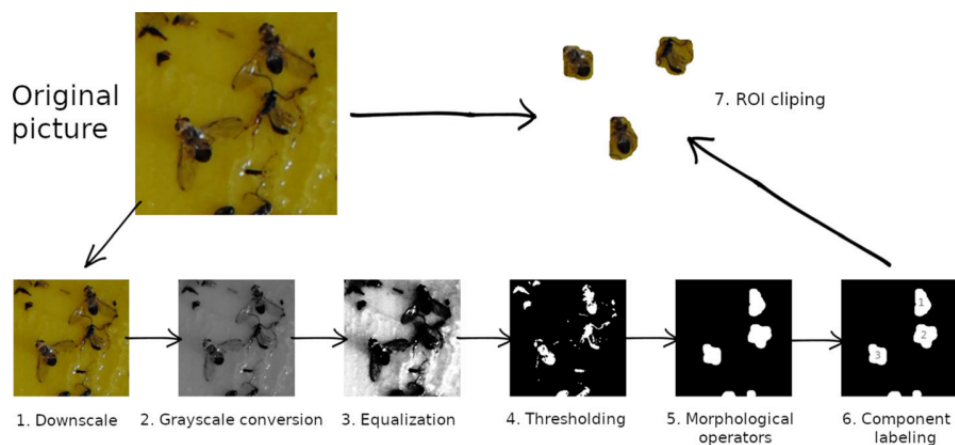


Figure 2.7: Process of segmentation and extraction of ROIs [29]

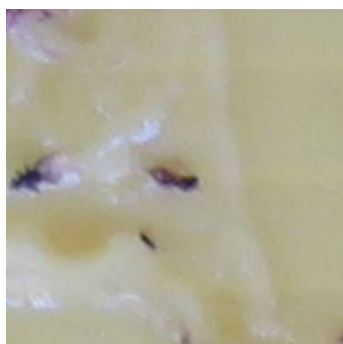
Unlike datasets built around bounding box annotations, this collection consists of classification-ready image patches obtained through a segmentation and ROI extraction process [29]. Some examples are provided in Figure 2.8.



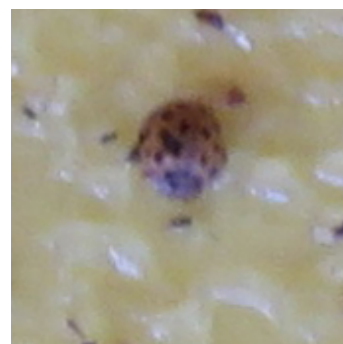
(a) Positive (olive fly)



(b) Negative (common fly)



(c) Negative (small insect)



(d) Negative (lady bug)

Figure 2.8: Example dataset image patches, positive and negative samples [29]

The dataset isolates the impact of data representation choices such as **RGB** versus grayscale images, background inclusion and class balance.

2.2.5 E-Trap Dataset

The electronic trap (e-trap) dataset [30] consists of **RGB** images captured by two e-traps deployed in olive fields in Mallorca, Spain, as shown in Figure 2.9.

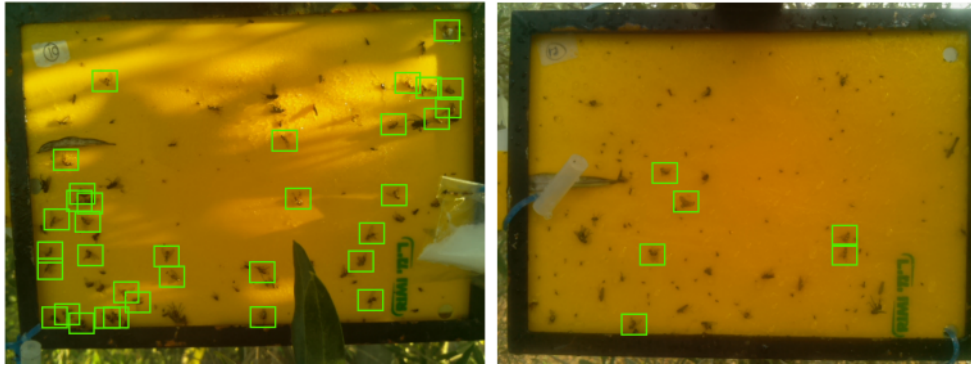


Figure 2.9: Example targets from the two sticky traps [30]

From these images, 32x32 pixel patches are extracted corresponding to visual elements on the sticky trap surface, forming a binary classification dataset that tell apart olive fruit flies from non-target elements such as other insect species, shadows and trap artifacts.

2.2.6 Pepper Diseases and Pests Dataset

The Pepper Diseases and Pests Dataset (PDD) [22], is a multimodal dataset composed of **RGB** images collected from greenhouse environments and online sources. Each image is paired with a natural language description written by an expert and annotated with bounding boxes aligned to the textual content. A few examples of the images in Figure 2.10.

Although not focused on olive fruit fly monitoring, PDD enables language-guided Object Detection (**OD**) and is representative of datasets designed to support multimodal pest and disease analysis.

2.2.7 IDADP Dataset

The IDADP dataset [45], contains 1293 images covering 29 pest species. Each image is paired with a textual description created through manual annotation. This dataset is specifically designed to support few-shot pest recognition under data-scarce conditions and facilitates multimodal learning through image-text alignment.

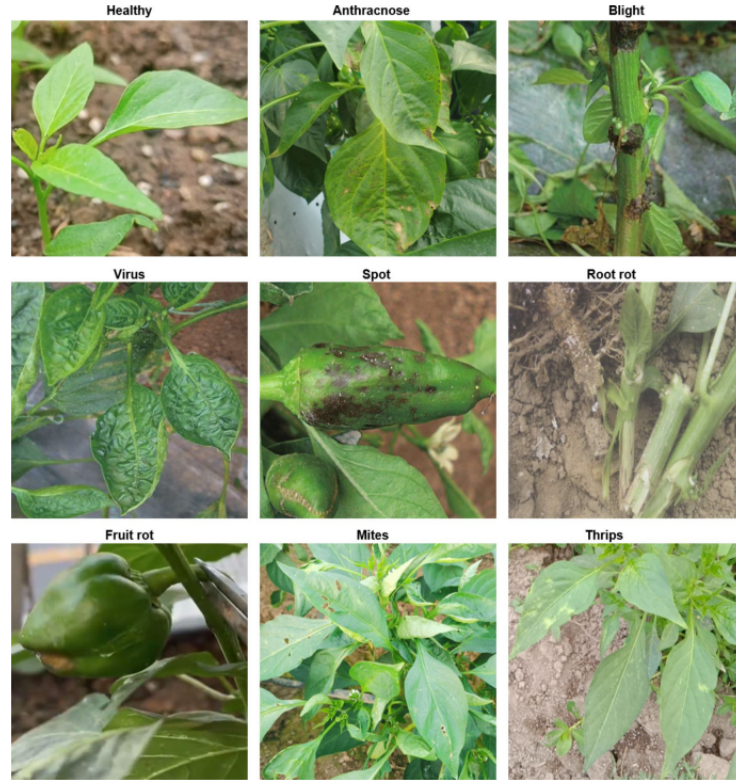


Figure 2.10: Examples from the Pepper Diseases and Pests Dataset [22]

2.2.8 WPIT9K and IP102 Datasets

WPIT9K is a multimodal pest dataset containing paired **RGB** images and expert-written textual descriptions for 17 pest species, as illustrated in Figure 2.11. It is used to evaluate parameter-efficient fine-tuning strategies for large pretrained **VLMs**, enabling explicit image-text alignment [21].

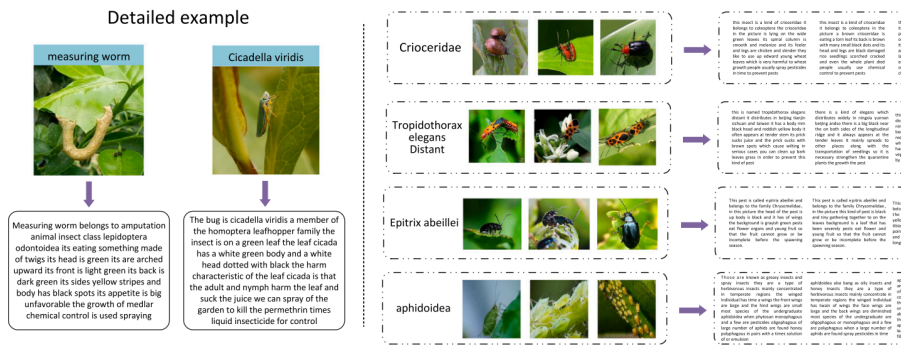


Figure 2.11: Examples of paired image-text samples from the WPIT9K dataset, illustrating the alignment between pest images and expert-written descriptions [21].

The IP102 dataset [43] is a large-scale fine-grained pest recognition benchmark made up of over 75,000 **RGB** images spanning 102 insect categories and multiple life stages. It is employed by

several works, including [21] and [23], to evaluate both recognition performance and multimodal semantic alignment. IP102 provides broad taxonomic coverage and serves as a standard benchmark for agricultural pest recognition, despite lacking bounding-box annotations or trap-specific imagery.

2.3 Vision Based Methods for Olive Fly Detection

Recent advances in computer vision have led to several deep learning methods for automated insect detection in trapping systems.

A YOLO-based detector is trained on the DIRT dataset for olive fruit fly detection [16, 28].

Only clearly visible floating or distinguishable flies are labeled, clustered insects are not detected, male and female specimens are merged into a single class and the original images were sliced, producing additional samples with truncated flies as in Figure 2.4.

Their main challenges include the small size of the target objects, strong variation in lighting and shadows and the presence of visually similar non-target insects [28]. To address these issues, the authors apply yellow mean color normalization to standardize the trap background and reduce illumination variability, as shown in Figure 2.12, enrich the training set with negative samples from the Leeds butterfly dataset to help the detector disregard non-target insects and apply data augmentation to mitigate the limited size of the dataset.

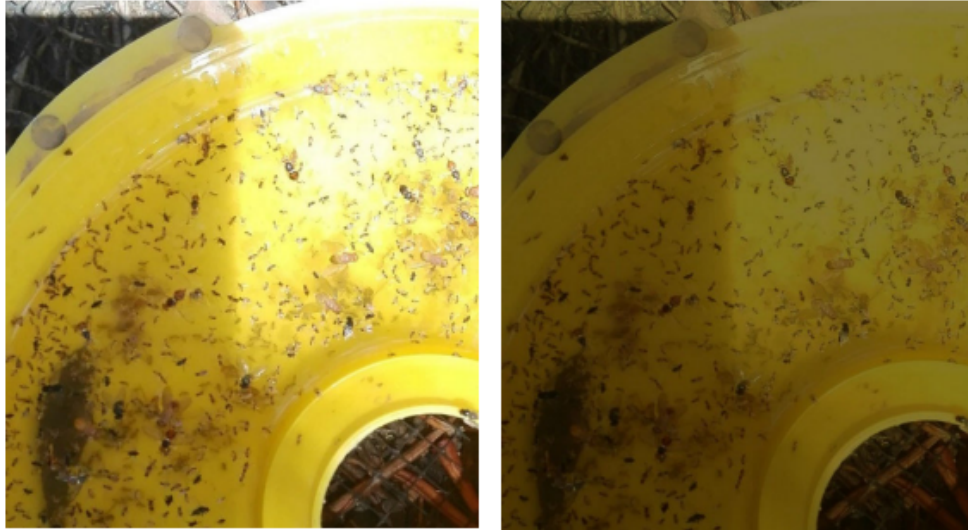


Figure 2.12: Before and after yellow mean normalization [28]

The final model performs end-to-end detection and counting of olive fruit flies directly from trap images and achieves a precision of 0.84, recall of 0.97, F1 score of 0.90 and mean Average Precision (mAP) of 96.68%.

A YOLOv7-based detector is evaluated on both the DIRT dataset and an independent Portuguese olive grove dataset [38].

To overcome the strong visual variability typical of outdoor agricultural environments, including uncontrolled illumination, shadows, background clutter and diverse insect orientations, the authors apply an extensive set of data augmentation techniques during training. These include geometric transformations, brightness and contrast adjustments, blurring, noise injection, cutout and jpeg compression, as summarized in Figure 2.13. This strategy significantly improves robustness to real-world acquisition artifacts.

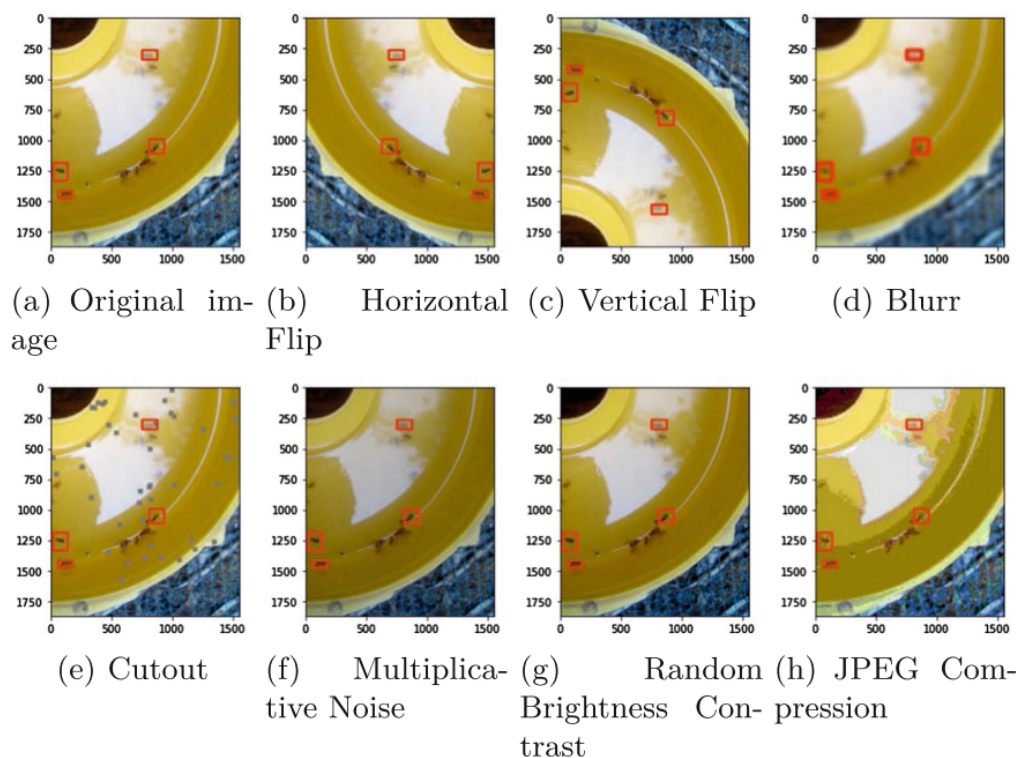


Figure 2.13: Overview of the data augmentation techniques applied during training, including geometric transformations, blurring, brightness and contrast adjustment, noise injection, cutout and JPEG compression [38].

Detection results on both datasets are illustrated in Figure 2.14, highlighting the model's ability to localize olive fruit flies in visually cluttered trap images.

Experimental results demonstrate high detection performance for the target class, with **mAP** @0.5 values approaching 0.99 for *Bactrocera oleae*, confirming the effectiveness of state-of-the-art single-stage detectors for olive fly monitoring. However, despite its strong performance, the approach remains limited to a vision-only pattern, producing bounding boxes and confidence scores without offering semantic explanations or user interaction.

A complementary vision-based approach is [33], who addresses olive fruit fly recognition as a binary image classification problem rather than full-image **OD**. They evaluate a custom **CNN** for binary recognition using a large-scale insect dataset.

Their model is benchmarked against well-known architectures such as AlexNet [17], ResNet50



(a) **YOLO** v7 detection results on the DIRT dataset, illustrating correct localization of *Bactrocera oleae* instances as well as discrimination between target (dacus) and non-target insects under visually cluttered trap conditions.



(b) **YOLO** v7 detection results on the Portuguese yellow sticky trap dataset, demonstrating model generalization to unseen acquisition settings, varying illumination and high insect density scenarios.

Figure 2.14: **YOLO** v7 detection examples on olive fruit fly trap images from the DIRT and Portuguese datasets, highlighting robust localization performance under real field conditions. Adapted from [38].

[5], VGG16 [36] and MobileNetV2 [34], achieving competitive accuracy while maintaining a significantly smaller model size, making it suitable for resource-constrained environments. However, since the approach operates on cropped insect images and does not address localization or detection in full trap images, it does not directly tackle the challenges associated with real-world trap deployment and cluttered visual scenes.

Beyond deep learning-based pipelines, classical machine learning and lightweight vision-based approaches have also been explored for olive fruit fly monitoring under computational and energy constraints. For example, Molina-Rotger et al. [30] evaluate Random Forest and Support Vector Machine (SVM) classifiers on an electronic trap platform, while Mira et al. [29] benchmark several classical and lightweight methods on Region of Interest (ROI) based trap patches, analysing the trade-off between detection accuracy, computational efficiency and energy consumption.

The ROI extraction process used to generate the dataset samples is illustrated in Figure 2.7.

In the preprocessing part the study evaluates variations in the input format, such as RGB versus grayscale images, addition or removal of trap backgrounds and different levels of class balance, rather than applying data augmentation or specialized normalization procedures. This design enables them to isolate how data representation influences accuracy and computational cost across the evaluated models.

Three models are benchmarked:

1. Histogram of Oriented Gradients (HOG) features coupled with a SVM classifier;
2. A lightweight CNN based on the LeNet-5 architecture;
3. A hybrid approach using Principal Component Analysis (PCA) for dimensionality reduction followed by a Random Forest classifier.

Through extensive testing on a Raspberry Pi 4, the PCA + Random Forest [1] pipeline emerges as the best performing method, achieving up to 99% accuracy, while being 6x faster than the CNN and 48x faster than the HOG+SVM, with corresponding energy consumption reductions of 10x and 44x, respectively [29].

Overall, this study demonstrates that classical machine learning techniques, when carefully optimized, can deliver high detection performance while meeting the strict energy and latency constraints of low power edge devices. However, it remains limited to classification of small image patches and does not address full image detection or multi species scenarios.

2.3.1 Edge based Smart Traps for Olive Fly Monitoring

The proposed electronic trap (e-trap) system integrates image acquisition, environmental sensing and on device machine learning for the detection of *Bactrocera oleae* under real field conditions [30]. An overview of its main components is shown in Figure 2.15. The system is designed for long-term deployment in olive orchards, operating under strict computational and energy constraints.



Figure 2.15: Electronic components of the e-trap system (A) E-trap with solar panel, Stevenson screen to protect the temperature and relative humidity sensor, battery and electronics. (B) Camera placed in front of a Rimi® trap. (C) Battery and electronics. [30]

The system is evaluated on an e-trap dataset collected in Mallorca, with patch-based binary classification. The experimental protocol uses 45 images acquired from one trap for training and validation, while 15 images from a second trap are reserved for testing, enabling evaluation under previously unseen acquisition conditions.

Given the limited amount of available data, both organic and synthetic data augmentation strategies were applied. Organic augmentation results from repeated imaging of the same trapped insect under varying illumination across multiple days, while synthetic augmentation includes flipping, rotation and brightness and contrast adjustments. These procedures expand the training set from 1332 to 8069 samples.

Key challenges were identified such as illumination variability, visually similar non-target insects and limited processing capabilities of the target hardware, a Raspberry Pi B+. To address these constraints, they adopt a two stage pipeline based on trap segmentation and candidate object detection, exploiting the contrast between the yellow trap background and darker foreground elements.

Random Forest [1] and SVM [3] models are evaluated, along with a combined decision strategy in which a detection is validated only if both classifiers agree, just like in Figure 2.16.

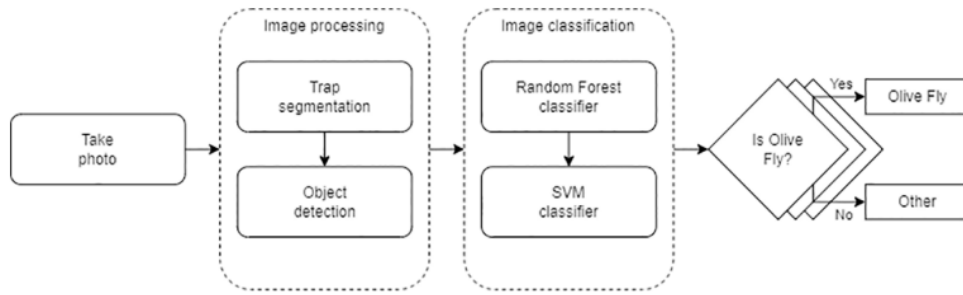


Figure 2.16: Inference pipeline [30]

While several **CNN** architectures are also tested, classical machine learning approaches outperform deep learning models in this low data situation. The combined random forest and **SVM** approach achieves an accuracy of 89.1% for olive fly detection, while accuracies of 91.9% for Random Forest and 94.5% for **SVM** are reported when classifying flies versus non fly elements. It demonstrates that efficiency on device classification is attainable for olive fly monitoring, although the system relies on pre-segmented candidate regions and provides only binary classification outputs.

2.4 Multimodal Approaches

Recent research has investigated multimodal learning as a way to improve pest detection in visually complex agricultural environments, where small target size and background clutter reduce the effectiveness of image only methods [21, 22, 41, 45].

A representative example is the multimodal framework proposed by [22], which addresses pepper disease and pest detection using paired image and natural language data. Although not focused on olive fruit fly monitoring, this work is included because it demonstrates how auxiliary semantic information can enhance detection robustness under challenging field conditions.

They use the PDD dataset [22] with paired image text descriptions for language-guided detection.

Image preprocessing includes resizing to 640x640 pixels and standard data augmentation techniques such as flipping and brightness adjustment, while text descriptions are truncated or padded to a fixed length. The main challenges identified are strong visual similarity between pest categories, background occlusion and small object size. These are addressed by introducing attribute level multimodal combination, allowing language descriptions to guide visual attention.

The proposed PepperDet model combines a Swin Transformer [24] visual backbone with a Bi-LSTM [8] language encoder and an attribute aware multimodal fusion module, followed by a Sparse DETR detection head, as Figure 2.17 shows.

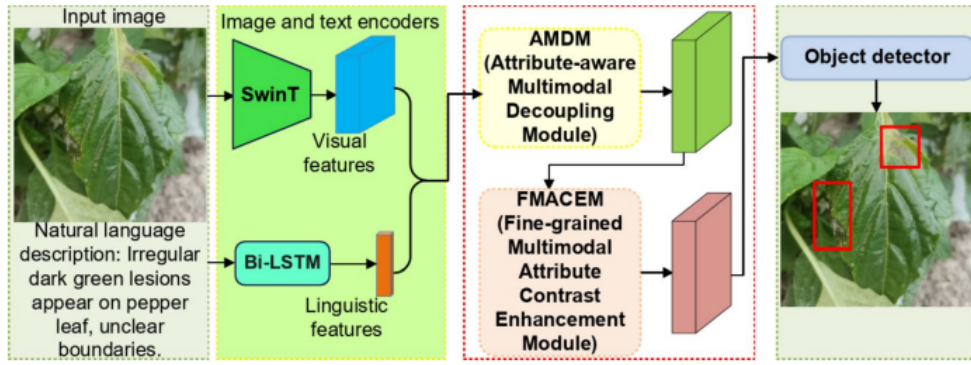


Figure 2.17: Overview of the PepperDet multimodal detection framework [22]

Experimental results show that incorporating natural language improves detection performance from 83.21% to 91.93% mAP @0.5 compared to a vision only baseline. While this approach is not directly applicable to edge based olive fly traps due to its reliance on detailed textual annotations, it highlights the potential of multimodal cues to compensate for visual ambiguity, motivating future extensions beyond purely vision based pest monitoring systems.

Another multimodal approach to pest detection is proposed by [45], who introduce a few-shot recognition model combining image features and textual descriptions using a Multimodal Masked Autoencoder (MMAE) [45], while integrating textual information to improve pest recognition under few-shot conditions.

Training is performed on a paired image-text pest dataset. Images are processed using a Masked Autoencoder (MAE) [6] for feature extraction, while textual descriptions are encoded using the ALBERT model [18]. The resulting visual and textual representations are combined to improve recognition performance under limited data conditions, as illustrated in Figure 2.18.

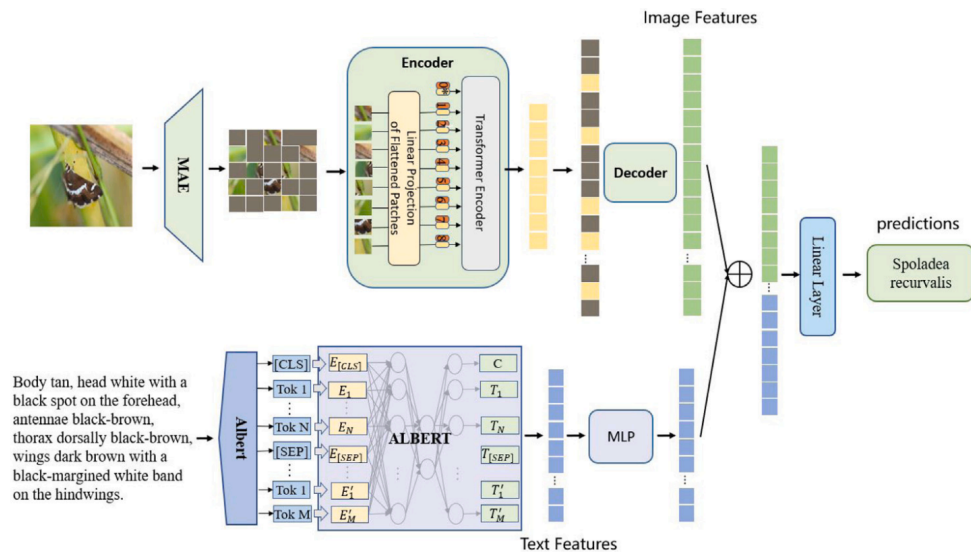


Figure 2.18: MMAE model architecture [45]

The MMAE model outperforms the MAE baseline, achieving a recognition accuracy of 98.12%,

compared to 96.51% for **MAE**. This shows that textual descriptions can significantly improve fine-grained pest recognition in scenarios with limited data.

While this paper does not directly address olive fruit flies, it is relevant for low data pest detection where textual information can support visual data in environments with a few samples.

A more recent multimodal approach based on large pretrained **VLMs** is proposed by Liu et al. [21], who address agricultural pest recognition through a cross-modal framework built on CLIP and Parameter-Efficient Fine-Tuning (**PEFT**). The method is evaluated on the WPIT9K and IP102 datasets [43].

Image preprocessing is limited to resizing and standard data augmentation techniques such as random cropping, flipping, rotation and brightness adjustment, with no task-specific background normalization or insect segmentation applied. The main challenges addressed are limited labeled data, strong visual similarity between pest species and the high computational cost of fully fine-tuning Large Multimodal Models (**LMMs**).

To overcome these limitations, the authors propose Dual-(PAL)^G, which freezes the CLIP backbone [32] and introduces learnable prompts, adapters and Low-Rank Adaptation (**LoRA**) modules on both the image and text branches, coordinated by a gating mechanism that dynamically balances their contribution during training, as shown in Figure 2.19. This design enables efficient cross-modal feature adaptation with a minimal increase in trainable parameters.

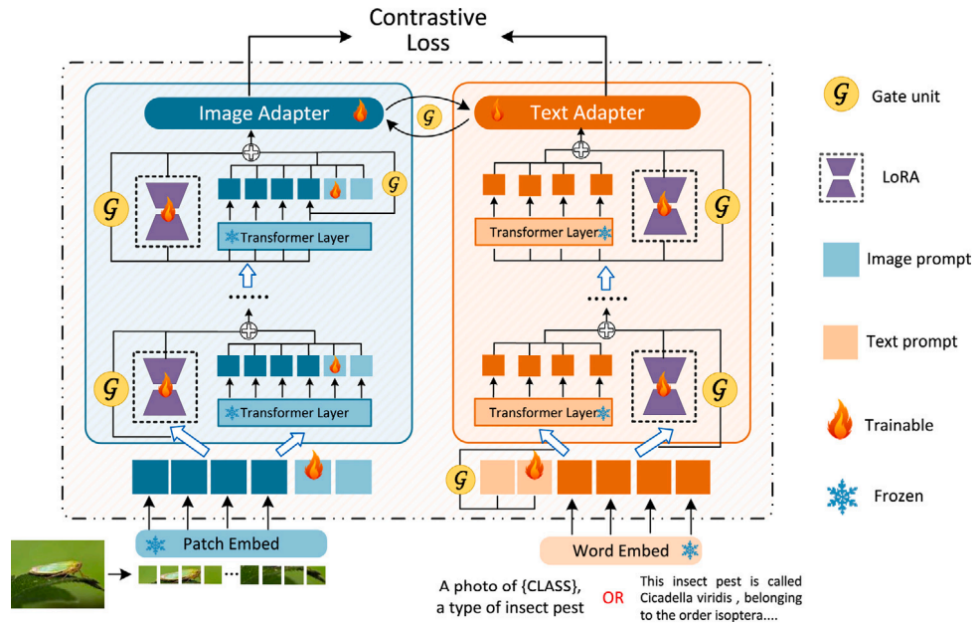


Figure 2.19: Overview of the Dual-(PAL)^G cross-modal architecture based on CLIP and parameter-efficient fine-tuning, highlighting the integration of prompts, adapters, **LoRA** modules and the gating mechanism [21].

Experimental results show strong performance under few-shot settings, achieving approximately 98% accuracy on WPIT9K using only 15% of the training data and outperforming vision-only and prior multimodal baselines on the fine-grained IP102 dataset by [43]. While the approach

does not address **OD** or interactive **VLM** reasoning, it constitutes a strong cross-modal recognition baseline bridging vision-only pest classification and more advanced **VLM** approaches.

Another recent multimodal approach bridging vision-only pest analysis and **VLMs** is proposed by Liu and Song [23], who introduce the Multimodal Prior and Conditional Detection (**MPCD**) framework for pest image generation and semantic alignment. The method is evaluated on IP102 dataset [43]. Representative qualitative results are shown in Figure 2.20.

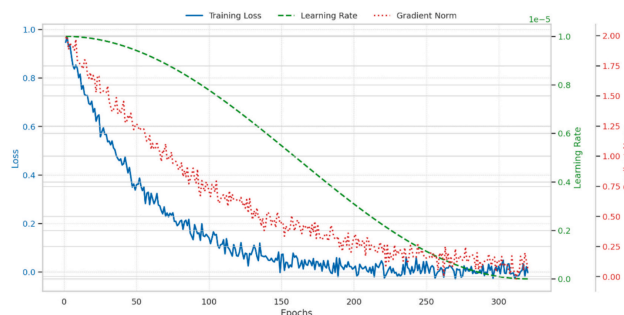


Figure 2.20: Qualitative pest image generation and component-aware attention results produced by the MPCD framework on the IP102 dataset, illustrating improved semantic alignment between visual features and textual descriptions, adapted from Figure 6 of [23].

Image preprocessing is limited to resizing and normalization, with no explicit data augmentation applied. The main challenges addressed include small pest size, cluttered backgrounds, fine-grained inter-class similarity and semantic misalignment between visual and textual cues. To address these issues, **MPCD** integrates biological priors through an object component prediction module and fuses visual, contour and textual features using a gated multimodal Transformer-based architecture.

Experimental results show that **MPCD** outperforms state-of-the-art generative baselines on IP102 [43], achieving a Fréchet Inception Distance [7] of 20.11 and a CLIP score [32] of 0.3276, alongside improved SSIM (Structural Similarity Index Measure) [42] and PSNR (Peak Signal-to-Noise Ratio) values [4]. Although focused on representation enhancement rather than detection, this work demonstrates how multimodal semantic priors can significantly improve pest understanding, making it a suitable precursor to **VLMs** in agricultural monitoring.

2.5 VLM Approaches

Recent advances in **VLMs** and **LMMs** have enabled a new class of intelligent systems capable of reasoning over visual inputs and natural language. In the context of agricultural monitoring, these models open the possibility of moving beyond pure visual detection towards interactive, explainable and user-centered pest analysis tools.

A representative example of this paradigm is AgriDetectVL [39], which introduces a **VLM** assistant specifically designed for agricultural pest and disease analysis. The system integrates visual perception with language-based interaction, enabling users to query images, receive natural

language explanations and obtain semantically rich descriptions of detected agricultural events. An overview of the AgriDetectVL framework and its interaction workflow is shown in Figure 2.21.

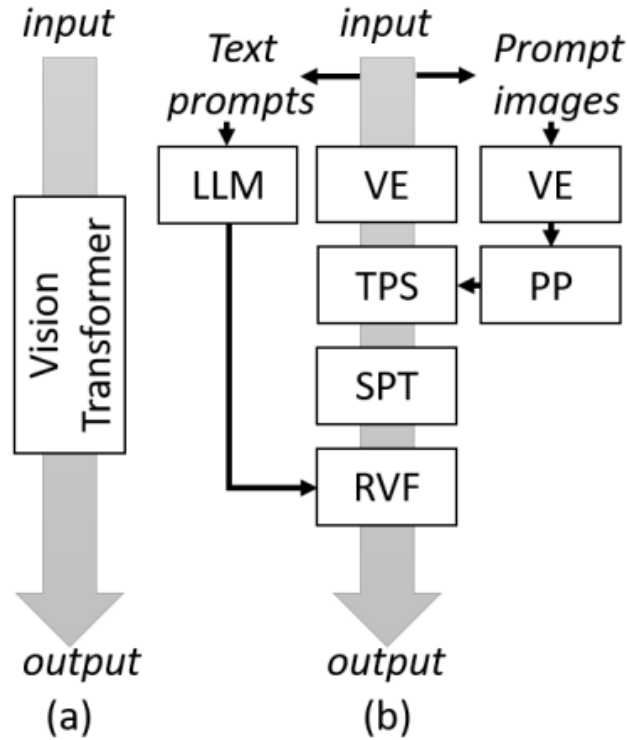


Figure 2.21: Comparison between a standard Vision Transformer pipeline and the proposed AgriDetectVL architecture, highlighting the integration of visual encoding and large language models through prompt-based interaction, adapted from [39].

Unlike traditional computer vision pipelines that output only bounding boxes or class labels, AgriDetectVL emphasizes human-AI interaction by allowing users to ask questions about visual content and receive contextualized responses grounded in the input image. Although the system is not tailored specifically to olive fruit fly monitoring, it closely aligns with the objectives of this dissertation by demonstrating how **VLM** integration can enhance usability, explainability and practical deployment of pest monitoring systems.

Another closely related contribution within the **VLM** standard is Agri-LLaVA [13]. This work extends large **VLMs** to the agricultural domain by incorporating domain-specific knowledge and visual-textual supervision for pest and disease understanding. The framework enables visual question answering and natural language interaction grounded on agricultural images, allowing users to query visual content and obtain context-aware explanations. Example interactions and model capabilities are illustrated in Figure 2.22.

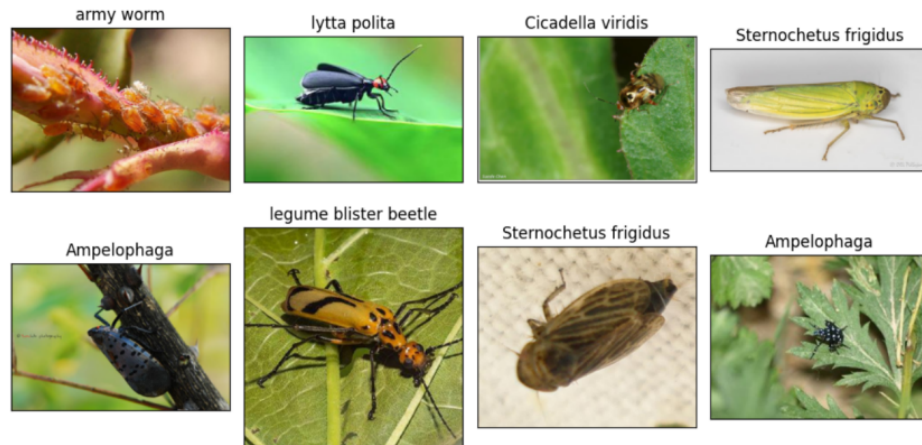


Figure 2.22: Example visual question answering and interaction results produced by Agri-LLaVA, illustrating language-guided reasoning over agricultural images, adapted from [13].

The work shows that large **VLMs** can be adapted to agricultural tasks by combining visual analysis with natural language understanding, enabling more interactive and interpretable decision-support systems for pest and disease management.

Another relevant contribution in the context of **VLMs** and multimodal learning for pest analysis proposes a domain-specific multimodal large language model for forestry pest and disease recognition and reasoning [44]. The approach integrates visual encoders with a large language model to jointly process images and textual prompts, enabling semantic interpretation, question answering and explanatory responses for pest-related scenarios. Unlike conventional vision-based pipelines, the model emphasizes multimodal reasoning and interactive analysis rather than pure visual detection. An overview of the proposed multimodal architecture and its inference workflow is illustrated in Figure 2.23.

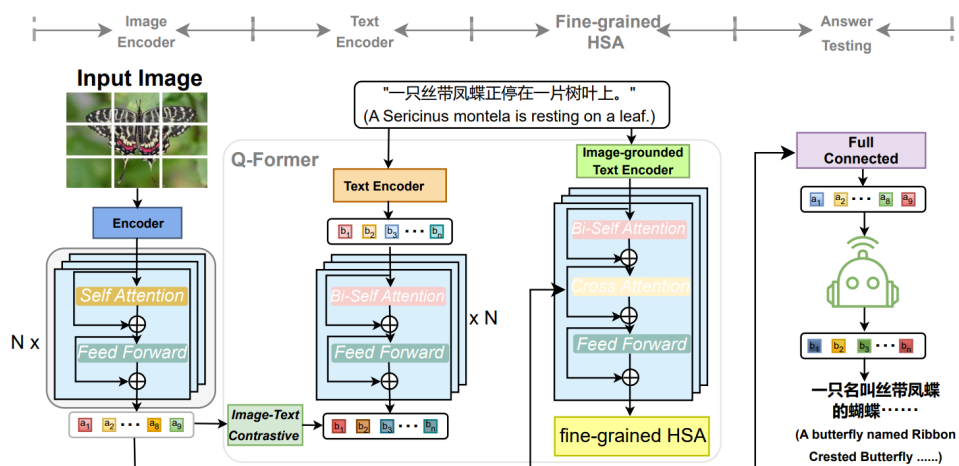


Figure 2.23: Overview of the proposed domain-specific multimodal **VLM** architecture for insect pest recognition and reasoning, illustrating the integration of visual encoders and a large language model, adapted from [44].

Although the system is primarily designed for forestry pest monitoring and is not directly applied to olive fruit fly traps, it is highly relevant to this dissertation as it demonstrates how domain-adapted **VLMs** can support fine-grained pest understanding, user interaction and explainability. This work further motivates the adoption of **VLM** paradigms for olive fly monitoring, where semantic context and natural language interaction can complement visual detection outputs.

The work proposes a multimodal vision-language framework for the identification of crop diseases and pests, combining visual perception with knowledge-grounded question answering in agricultural monitoring scenarios [41].

The proposed architecture follows a **VLM** pattern in which a visual encoder is aligned with a language model through a Q-Former-based cross-modal alignment mechanism [19, 41], enabling effective fusion of visual features and textual reasoning, as illustrated in Figure 2.24.

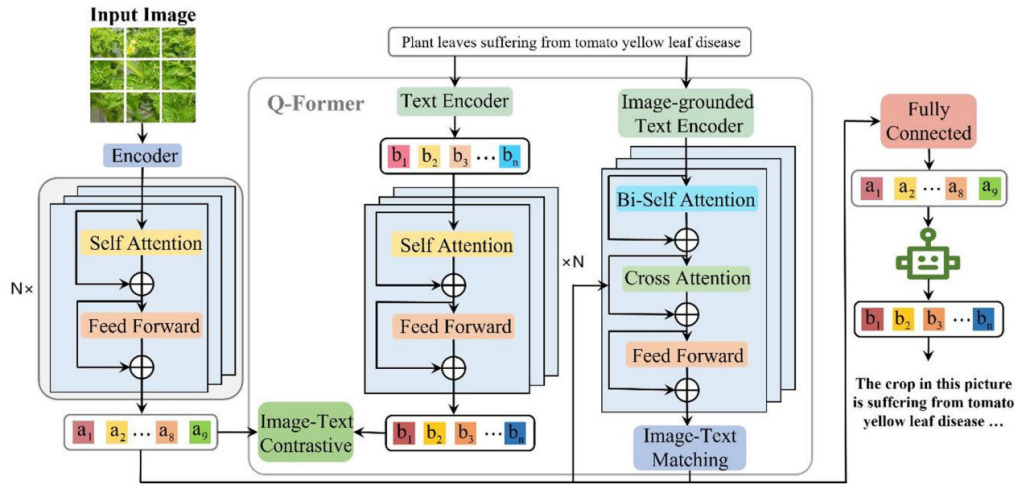


Figure 2.24: Architecture of the LLMI-CDP **VLM** model showing visual encoding, Q-Former-based cross-modal alignment and language-based reasoning, adapted from [41].

Figure 2.25 shows examples illustrating the model's ability to identify crop diseases and pests and to provide knowledge-based diagnostic and management responses.

The reviewed **VLMs** approaches illustrate a shift from task-specific detection pipelines towards interactive and semantically grounded pest analysis systems. Although these models are not yet tailored to olive fruit fly trap monitoring, they provide the conceptual and technical foundations for integrating visual perception with natural language reasoning.

2.6 Summary

The reviewed literature shows a clear progression from vision-only detection methods to multimodal and vision-language approaches. Vision-based systems, especially YOLO-based detectors, achieve strong performance for olive fruit fly detection in both controlled datasets and real field conditions. Classical machine learning methods also remain relevant for edge-based smart traps, where energy consumption and computational efficiency are important constraints.



What disease does the crop in this picture suffer from?

Ours
The tomato crops in this picture are afflicted with late blight disease. The disease manifests as symptoms on the affected leaves, such as brown spots on the underside, irregular-shaped lesions on the surface, and later progressing into lesions with outward-rolled edges.

VisualGLM
The plant in this photo may have tomato leaves that are infected with diseases such as anthracnose, whitefly, etc. These diseases cause black spots on the leaves and make the leaves brittle and prone to breakage. If these problems are not detected in time and measures are not taken to prevent and treat them, they may result in serious economic losses and even affect output and quality.

Qwen-VL
Based on the description, the crops in this picture may be affected by tomato wilt disease. Tomato wilt is a common tomato ailment that can cause the leaves to wither and turn yellow, and in severe cases, it can lead to the death of the tomato plants.

VisCPM
The plants in the picture are afflicted with rust disease, which is a condition where a rust layer forms on the surface of the plants. This could have adverse effects on the health of the plants, potentially leading to their death or a decline in quality.

MiniGPT4
This image shows a plant with brown spots on its leaves. The plant appears to be a tomato plant, which is commonly affected by various plant diseases, including powdery mildew. Powdery mildew is a fungal disease that affects many types of plants, including tomatoes, cucumbers, and peppers. It causes white or gray powdery spots on the leaves and stems of the plant, which can reduce the plant's yield and quality.

Ziya-Visual
The crops in the image may be affected by leaf spot disease. Leaf spot is a common plant ailment characterized by yellow or brown spots and markings on the leaves. These changes are typically associated with fungal infections and are more likely to occur in warm, humid environments.

(a) Comparison of agricultural crop disease recognition experiments



What pest is this picture of?

Ours
The pest in this picture is the rice water weevil. The rice water weevil damages rice as adults and larvae, with the larvae causing the most serious damage. The adults eat the mesophyll or seedling leaf sheaths along the veins of rice leaves, forming long white spots on the leaves. Young larvae feed inside the rice roots, causing holes in the roots and causing the plants to turn yellow and wither.

VisualGLM
This picture shows some kind of pest, such as slugs, beetles, etc., feeding on the sap of plants. Slugs are small insects that usually search for food on plants and suck their juices. Due to their small size and unique appearance, these insects are often considered a pest species. However, for certain types of plants they can be beneficial because they can disperse seeds or control other insect threats to the plant.

Ziya-Visual
The pest in the image is a small insect or beetle. It is crawling on the green plants. This insect may belong to certain categories such as ladybugs, thrips, locusts, etc. But the specific insect species could not be identified due to a lack of further details.

MiniGPT4
This image shows a small beetle on a leaf. The beetle has black and white markings on its body and antennae. It appears to be perched on the edge of a leaf, with its legs extended in front of it. The background is green and there are some small insects crawling on the leaf. The overall appearance of the image is naturalistic.

Qwen-VL
According to the description, this picture shows a pest called the "pink shoulder beetle", whose scientific name is "Chrysolina graminis". This pest often causes damage to agricultural plants.

VisCPM
The pest pictured here is a small beetle sitting on a leaf.

(b) Comparison of agricultural crop pest identification experiments

Figure 2.25: Comparison of cognitive task experimental results from [41].

However, these approaches are mainly limited to visual pattern recognition. They usually provide bounding boxes, confidence scores or binary decisions, without semantic explanation, contextual interpretation or user interaction. Their robustness can also be affected by cluttered trap images, visually similar insects, illumination changes and domain shifts across acquisition conditions.

Multimodal methods address part of these limitations by combining visual features with textual information, improving pest recognition in low-data, fine-grained or visually ambiguous scenarios. Nevertheless, many of these methods depend on task-specific architectures and carefully prepared image-text annotations, which can limit their scalability and practical deployment.

Recent **VLMs** show a further shift towards systems that combine visual understanding with natural language reasoning and interaction. Although existing **VLMs** are not yet specifically designed for olive fruit fly trap monitoring, they demonstrate the potential to enrich visual detections with explanations, semantic descriptions and question-answering capabilities.

Chapter 3

Methodology

The visual detection pipeline was developed through a process involving multiple compositions of the dataset, several **YOLO** architectures (YOLOv5s [14], YOLOv5m [14], YOLOv7 [40] and YOLOv8 [15]) and different dataset splitting strategies. The objective was to evaluate the impact of dataset composition, image preprocessing, model architecture and cross validation procedures on the capability of olive fruit fly detection.

Initially a unified dataset was created by merging the Portuguese Olive Grove Dataset with the DIRT Dataset [16]. After the merge, all images and annotations were carefully validated to ensure label consistency and annotation quality.

Several dataset compositions were then created and evaluated, including cropped variants designed to reduce background noise, improving object focus. Some **YOLO** models (YOLOv5s [14], YOLOv5m [14], YOLOv7 [40] and YOLOv8 [15]) were trained and compared across all dataset configurations.

3.1 Visual Detection Pipeline

The dataset used in this work was constructed from the integration of the Portuguese Olive Grove Dataset and the DIRT Dataset. The combination of both datasets introduced greater variability in image composition, illumination conditions and background characteristics. While the DIRT Dataset mainly contains close-range trap images, the Portuguese Olive Grove Dataset includes larger trap scenes with increased visual clutter and more complex image structures. The resulting merged dataset contained 1,189 annotated images and served as the foundation for all subsequent **OD** experiments.

This integration allowed the creation of a more diverse and representative dataset, supporting the evaluation of the models under different visual conditions and improving their generalisation capability.

3.1.1 Dataset Compositions

Several dataset compositions were created throughout the experimental process in order to evaluate the impact of dataset composition and image preprocessing on model performance and generalization. Table 3.1 summarizes the main characteristics of each dataset composition.

Table 3.1: Summary of the dataset compositions used throughout the experimental process.

Composition	Main modification	Purpose
1	Original merged dataset	Baseline dataset obtained by merging the Portuguese Olive Grove Dataset and the DIRT Dataset.
2	Addition of <i>non_olive_fly</i> class	Evaluate whether introducing visually similar non-target insects improves discrimination between olive fruit flies and other insects. Includes standard, weighted and equalized variants.
3	Yellow trap cropping	Reduce background noise by automatically cropping the yellow trap region using HSV segmentation.
4	Perspective-based cropping	Apply a more aggressive crop with perspective transformation to further center the trap and minimize background.

The first dataset version corresponded to the original merged dataset obtained by integrating the Portuguese Olive Grove Dataset and the DIRT Dataset, as illustrated in Figure 3.1.



Figure 3.1: Example image from the original merged dataset.

A second dataset composition was created using the same original images from Composition 1, but with the introduction of an additional non-olivefly class through manual annotation using LabelImg [10]. The objective was to evaluate whether the inclusion of visually similar non-target

insects could improve the model’s discrimination capability between olive fruit flies and other insects. Initial experiments were performed using a two-class setup containing both *olive_fly* and *non_olive_fly* annotations. However, only a small number of *non_olive_fly* instances were initially annotated, resulting in a highly imbalanced dataset (Standard configuration). To investigate the effect of class weighting, additional experiments were conducted using the same annotations while assigning a weight of 5 to the minority *non_olive_fly* class (Standard + Weighted configuration). In a later stage, the *non_olive_fly* annotations were expanded to create a more balanced dataset (Equalized configuration), and an additional weighted version using the same class weight of 5 was also evaluated (Equalized + Weighted configuration).

The third dataset version introduced a cropping procedure designed to reduce background noise and focus the image on the yellow trap region. This process was performed through HSV-based yellow region segmentation followed by automatic cropping and corresponding adjustment of **YOLO** bounding box annotations. An example of the resulting images is shown in Figure 3.2.



Figure 3.2: Example image after the first cropping procedure.

A fourth dataset version was later developed using a more aggressive cropping strategy. In addition to reducing the amount of background surrounding the trap, a perspective transformation was applied to modify the trap viewpoint and obtain images more centered on the trap region. **YOLO** bounding box annotations were subsequently updated to remain consistent with the transformed images, as illustrated in Figure 3.3.



Figure 3.3: Example image using the second cropping preprocessing strategy.

3.1.2 Dataset Splitting Strategy

Different dataset splitting strategies were used during different stages of the experimental process. Early experiments employed a 90/10 train-validation split, while later experiments used an 80/10/10 train-validation-test split. In addition, 5-fold cross-validation was performed to assess the robustness and generalization of the models across different train-validation splits.

3.1.3 YOLO Architectures

Several **YOLO** architectures were evaluated throughout the experimental process, including YOLOv5s [14], YOLOv5m [14], YOLOv7 [40], and YOLOv8 [15].

YOLOv5s [14] is a single-stage object detector composed of a CSPDarknet backbone, a PANet neck and a detection head. The backbone extracts hierarchical visual features from the input image, while the PANet neck combines information from multiple feature scales to improve the detection of objects of different sizes. The detection head then predicts object classes, confidence scores and bounding box coordinates. Due to its lightweight architecture and reduced number of parameters, YOLOv5s provides fast training and inference times, making it suitable for rapid experimentation during the early stages of this work.

YOLOv5m [14] follows the same architectural design as YOLOv5s, sharing the CSPDarknet backbone, PANet neck and detection head. However, it employs a larger number of layers, feature channels and trainable parameters, resulting in a higher model capacity. This enables the extraction of richer feature representations and can improve the detection of small or visually challenging insects, albeit with increased computational requirements.

Figure 3.4 presents an overview of the YOLOv5 architecture, highlighting the CSPDarknet backbone, the PANet feature aggregation network and the multi-scale detection heads used by both YOLOv5s and YOLOv5m.

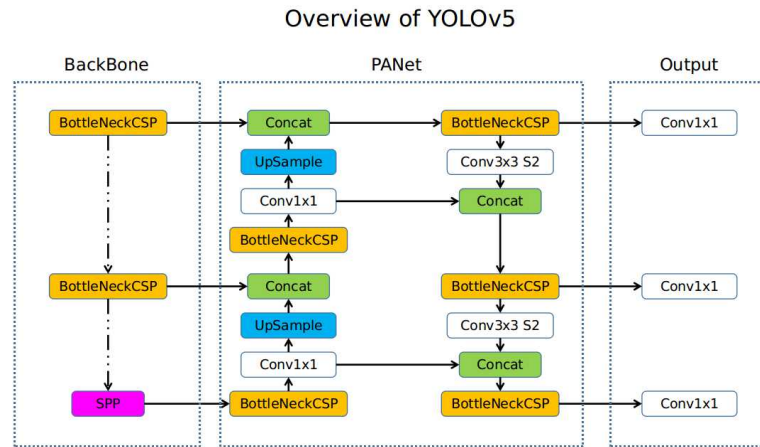


Figure 3.4: Overview of the YOLOv5 architecture, composed of a CSPDarknet backbone, PANet neck and multi-scale detection heads. Source: [35].

YOLOv7 [40] extends the YOLO family through the introduction of the Extended Efficient Layer Aggregation Network (E-ELAN) architecture. E-ELAN improves feature learning by enhancing gradient propagation and promoting more effective aggregation of information across network layers. The architecture also incorporates several training optimizations, including model scaling strategies and trainable bag-of-freebies techniques, which improve detection accuracy while maintaining real-time inference performance.

An overview of the YOLOv7 architecture is shown in Figure 3.5, illustrating the backbone, neck and detection heads responsible for multi-scale object detection.

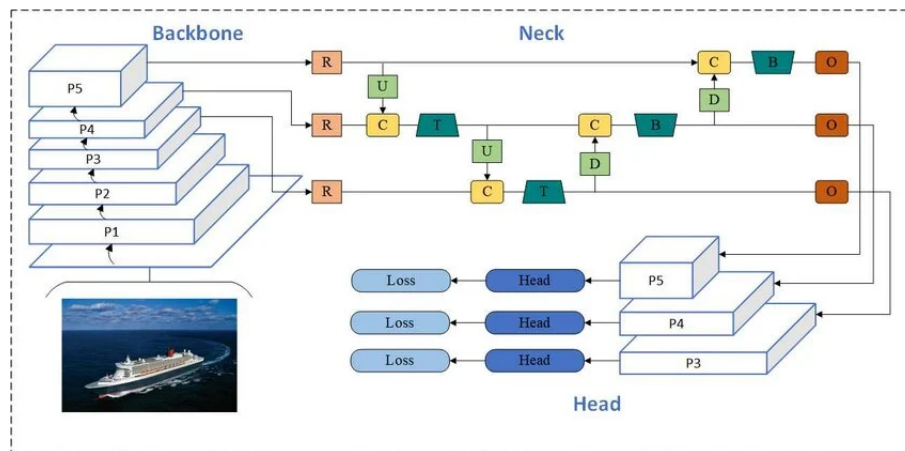


Figure 3.5: Overview of the YOLOv7 architecture, showing the backbone, neck and detection heads used for multi-scale object detection. Source: [46].

YOLOv8 [15] is a more recent YOLO architecture that introduces an anchor-free detection head, eliminating the need for predefined anchor boxes during training. This design simplifies the detection pipeline and can improve both training efficiency and localization accuracy. In addition, YOLOv8 incorporates architectural refinements in the backbone and neck components, enabling more effective feature extraction and feature fusion across multiple scales.

Figure 3.6 presents the YOLOv8 architecture, highlighting the backbone, neck and anchor-free detection heads used for object localization and classification.

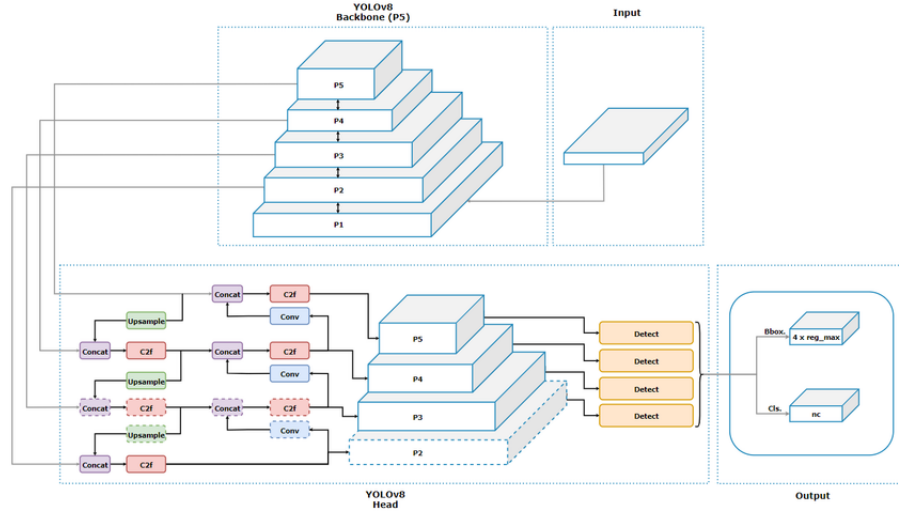


Figure 3.6: Overview of the YOLOv8 architecture, showing the backbone, neck and anchor-free detection heads. Source: [12].

3.1.4 Experimental Configuration

All **YOLO** architectures considered in this work were trained across every dataset version in order to ensure a consistent comparison between preprocessing strategies and model architectures. Experiments were conducted both using fixed dataset splits and 5-fold cross-validation. In the cross-validation setup, the original training and validation subsets were merged and redistributed across folds, while the test subset, when available, remained fixed for final evaluation. This approach allowed a more robust assessment by reducing the dependency on a single train-validation split.

Data augmentation techniques were applied during training using the default augmentation pipeline provided by each **YOLO** framework. These augmentations included HSV augmentation, translation, scaling, horizontal flipping and mosaic augmentation, which were performed dynamically during training rather than being stored as separate images in the dataset.

During the initial experimental phase, the same training hyperparameters were maintained across all **YOLO** architectures whenever supported by the corresponding framework. This decision was made to ensure a fair comparison between the different dataset compositions and model architectures, minimizing the influence of hyperparameter selection on the observed performance.

differences. The selected values were based on the default training configuration provided by each framework.

After identifying the most promising combination of dataset composition and **YOLO** architecture, additional experiments were conducted to evaluate the impact of hyperparameter adjustments on the selected configuration.

Table 3.2 summarizes the main training parameters used during the initial comparison stage.

Table 3.2: Initial training configuration used across **YOLO** architectures.

Parameter	Value
Models	YOLOv5s, YOLOv5m, YOLOv7, YOLOv8
Image size	1280×1280
Batch size	8
Initial learning rate	0.01
Momentum	0.937
Weight decay	0.0005
Patience (early stopping)	100 epochs
Optimizer (YOLOv5/YOLOv7)	SGD
Optimizer (YOLOv8)	Auto
Cross-validation folds	5

3.2 Vision-Language Model Development

The development of the **VLM** component began with an exploratory evaluation of a pretrained Large Multimodal Model. Before performing any domain-specific fine-tuning, the objective was to assess whether a general-purpose VLM could interpret olive fruit fly trap images and generate meaningful agronomic descriptions when provided with visual information and structured detection data.

Large Language and Vision Assistant (**LLaVA**) is a multimodal architecture that combines a pretrained vision encoder with a Large Language Model through a projection layer that maps visual features into the language model embedding space [20]. This design enables the model to process images and natural language jointly, supporting tasks such as image understanding, visual question answering and multimodal reasoning. The vision encoder, based on CLIP ViT-L/336px, extracts high-level visual representations from the input image. These visual features are then projected into the language model embedding space through a Multilayer Perceptron (MLP) connector, allowing the language model to jointly reason over visual and textual information. The resulting architecture enables the generation of natural language descriptions conditioned on both image content and user prompts.

The **LLaVA** 1.5 7B variant was selected due to its strong performance on a wide range of multimodal benchmarks, extensive documentation and widespread adoption within the research community, making it a suitable foundation for domain-specific adaptation experiments.

Figure 3.7 illustrates the high-level architecture of LLaVA 1.5, highlighting the interaction between the vision encoder, the multimodal projection layer and the language model.

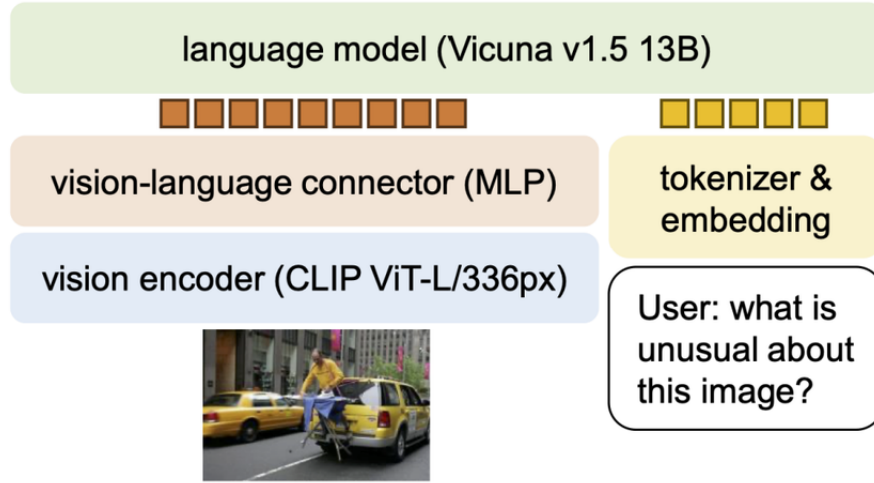


Figure 3.7: High-level architecture of LLaVA 1.5. Visual features extracted by the CLIP ViT-L/336px vision encoder are projected through a multimodal MLP connector and combined with textual embeddings before being processed by the language model. Adapted from [20].

This preliminary experiment served as a proof of concept, demonstrating the feasibility of combining visual trap imagery with structured detection outputs within a multimodal framework. It also provided an initial assessment of the model's strengths and limitations before the development of a domain-specific fine-tuning pipeline.

3.2.1 Fine-Tuning Dataset Preparation

Following the baseline evaluation of the pretrained LLaVA model, a domain-specific dataset was created to support VLM fine-tuning. Although the baseline model was capable of performing several tasks correctly, including object counting and general image description, it frequently failed to identify the yellow sticky monitoring trap, often describing it as a fruit or a generic yellow surface. The objective of the fine-tuning process was therefore to adapt the model to the olive fruit fly monitoring domain and improve its understanding of domain-specific concepts.

The initial fine-tuning dataset consisted of 200 images and used a single fixed question-answer template. Although the template covered the same domain-specific concepts later included in the expanded dataset, the wording of both the questions and answers remained the same across all samples. Only the number of detected olive fruit flies varied between examples. This limited textual diversity likely reduced the effectiveness of the fine-tuning process, as the model may have learned to reproduce repetitive response patterns rather than developing a more robust understanding of the visual concepts and tasks involved.

To increase task diversity and improve domain adaptation, the dataset was later expanded to 600 images. Multiple question-answer pairs were generated for each image, covering tasks such

as olive fruit fly counting, infestation level assessment, trap identification and determination of the trap region containing the highest insect concentration.

The infestation level categories were defined based on the distribution of olive fruit fly counts observed in the dataset. Images containing up to 10 detected flies were classified as low infestation, images with 11 to 30 flies as medium infestation and images with more than 30 flies as high infestation. These thresholds were selected to provide a more interpretable assessment of infestation severity while accounting for the presence of highly zoomed-in trap images in the dataset.

These thresholds were not derived from weekly monitoring guidelines commonly used in practice, as only a subset of the available images corresponded to weekly observations. Instead, they were defined empirically using the distribution of olive fruit fly counts across the dataset.

3.2.2 LoRA Fine-Tuning

Following the preparation of the fine-tuning dataset, parameter-efficient adaptation of the **VLM** was performed using **LoRA** [9]. This approach was selected because it enables domain-specific fine-tuning of large models while requiring substantially fewer trainable parameters and computational resources than full model retraining.

The experiments were conducted using **LLaVA** 1.5 7B [20] as the base **VLM**. To further reduce memory requirements, the model was loaded using 4-bit quantization, allowing training to be performed within the available hardware constraints.

The fine-tuning process followed an instruction-based learning approach. During training, the model received both the image and the associated textual instruction and was optimized to generate the corresponding answer.

LoRA adapters were applied to the transformer attention layers, allowing the model to learn task-specific behaviour while keeping the original pretrained weights frozen.

3.2.3 Fine-Tuning Experiments

Different training configurations were investigated, including learning rates of 2×10^{-4} , 5×10^{-5} and 1×10^{-5} . Initial experiments applied **LoRA** adaptation only to the transformer attention projection layers (*q_proj*, *k_proj*, *v_proj* and *o_proj*). Later experiments extended **LoRA** adaptation to additional feed-forward transformer components (*up_proj*, *down_proj* and *gate_proj*), allowing a larger portion of the model's internal representations to be adjusted while maintaining parameter-efficient training.

A chat masking strategy was introduced in the final configuration. Under this approach, the loss was computed only for the assistant responses, while the user instructions were excluded from the optimization process. This configuration encouraged the model to focus on generating accurate responses conditioned on the provided image and prompt, rather than learning to reproduce the input instructions.

The different configurations were evaluated qualitatively through interactive inference experiments, leading to the selection of the final fine-tuned model used in the integrated system.

3.2.4 Final VLM Configuration

The final **VLM** configuration was selected based on the qualitative evaluation of the different fine-tuning experiments. The selected model was based on **LLaVA** 1.5 7B and employed **LoRA** fine-tuning using the expanded training dataset, chat masking strategy and the lowest learning rate.

This resulted in more consistent responses during hands-on evaluation and improved the model's ability to answer domain-specific questions related to olive fruit fly monitoring.

It was subsequently integrated into the application layer, enabling users to interact with the system through natural language queries regarding olive fruit fly detections and infestation conditions.

The final **VLM** configuration is summarized in Table 3.3.

Table 3.3: Final LLaVA-LoRA configuration used in the integrated system.

Parameter	Value
Base model	LLaVA 1.5 7B
LoRA rank	16
LoRA alpha	32
Learning rate	1×10^{-5}
Epochs	5
Quantization	4-bit

3.2.5 System Integration

As shown in Figure 3.8, the uploaded image is first processed by YOLOv7 [40]. The resulting detections are exported as labels, converted into a JSON representation and used to generate a structured detection summary. Simultaneously, an annotated version of the image is produced. Both the annotated image and the generated detection summary are then provided to the fine-tuned **VLM**, which generates a natural language response that is presented through the web interface.

So, when an image is uploaded, it is first processed by the YOLOv7 model, which detects olive fruit flies and generates the corresponding detection outputs. These detections are subsequently converted into a structured JSON representation containing information such as the number of detected olive fruit flies and the average detection confidence.

The **VLM** then receives the YOLO-annotated image together with a structured textual summary containing detection statistics, confidence scores and spatial information. Based on these inputs, the model generates natural language responses to user queries regarding infestation conditions and other image-related observations.

A web-based interface and API layer were developed to support user interaction with the system. This architecture enables the complete processing pipeline, from image upload and **OD** to multimodal reasoning and conversational interaction, within a single integrated application.

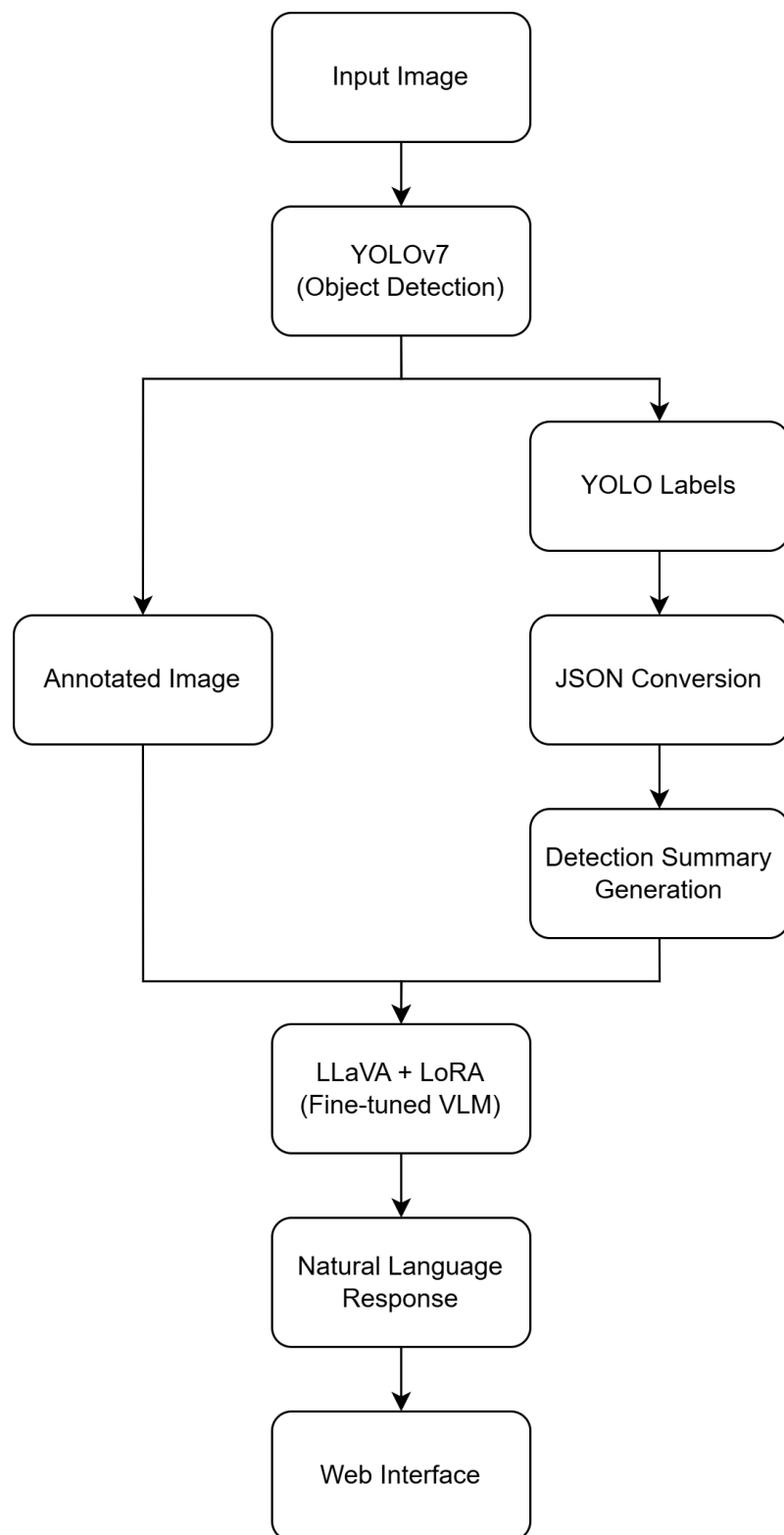


Figure 3.8: Overall architecture of the proposed monitoring system integrating YOLOv7 **OD** and the fine-tuned LLaVA-based **VLM**.

3.2.6 Web Application Architecture

In addition to the machine learning components described previously, a dedicated application layer was developed to provide a practical interface for system interaction. The application follows a client-server architecture composed of a frontend interface and a backend service responsible for coordinating user requests.

The frontend was designed to support image upload and conversational interaction, allowing users to submit trap images and query the system using natural language. The backend exposes the application functionality through a REST-based API and serves as the communication layer between the user-facing interface and the underlying machine learning components.

Figure 3.9 presents the high-level architecture of the developed application.

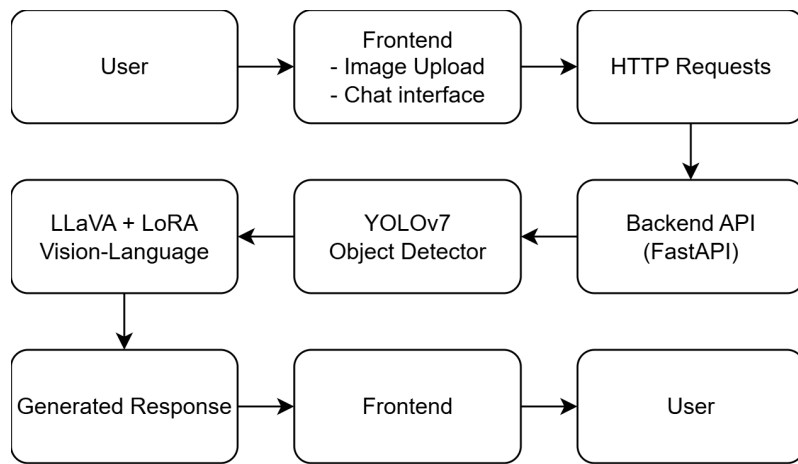


Figure 3.9: High-level architecture of the developed web application, showing the interaction between the user, frontend interface, backend service and the machine learning components.

All user interactions are initiated through the frontend interface and transmitted to the backend through HTTP requests. The backend manages the execution of the required processing tasks and returns the generated results to the frontend, which is responsible for presenting them to the user. This separation of responsibilities simplifies system maintenance and enables the machine learning components to remain independent from the presentation layer.

The complete application was deployed on an Ubuntu-based server equipped with an NVIDIA GeForce RTX 3090 GPU, providing the computational resources required for OD and multimodal inference.

Chapter 4

Experimental Results

4.1 Experimental Setup Overview

The first part of this chapter presents the experimental results obtained from the visual detection pipeline described in Chapter 3. The main objective of these experiments was to evaluate how different **YOLO** architectures and dataset compositions affect the detection of olive fruit flies in trap images.

The evaluated architectures included YOLOv5s, YOLOv5m, YOLOv7 and YOLOv8. These models were trained on several dataset compositions created during the experimental process, including the original merged dataset, multi-class variants with non-olivefly annotations and cropped versions designed to reduce background noise and focus the model on the trap region.

The performance of each model was evaluated using standard object detection metrics, namely precision, recall, mAP@0.5 and mAP@0.5:0.95.

4.2 Overall Comparison of YOLO Architectures

Table 4.1 summarizes the best mAP@0.5:0.95 result obtained by each **YOLO** architecture across the different dataset configurations evaluated during this work. These results correspond to the *olive_fly* class and allow the influence of dataset composition and preprocessing strategies to be analyzed independently from the final architecture comparison.

Table 4.1: Best mAP@0.5:0.95 obtained for the *olive_fly* class across different dataset configurations.

Model	Standard	Olive Only	Cropped	Force Cropped
YOLOv5s	0.401	0.4477	0.4478	0.4438
YOLOv5m	0.437	0.4571	0.4571	0.4478
YOLOv7	0.435	0.4700	0.4698	0.4664
YOLOv8	0.430	0.4609	0.4682	0.4594

The results show that the Olive Only and Cropped datasets consistently produced the strongest performance across all evaluated architectures. In contrast, the Standard dataset generally yielded lower scores, suggesting that the presence of additional non-target insect annotations did not improve olive fruit fly detection performance. Also, the Force Cropped dataset did not provide a consistent advantage over the other preprocessing strategies. These observations support the decision to focus the subsequent analysis on the best-performing configurations of each architecture.

Table 4.2 summarizes the best individual result obtained for each YOLO architecture across all evaluated dataset configurations. Since the objective of this comparison is to evaluate the detection capability of each architecture, only the best-performing configuration of each model is considered.

Table 4.2: Best individual result obtained for each YOLO architecture.

Model	Dataset Version	mAP@0.5:0.95	mAP@0.5	Precision	Recall
YOLOv5s	Olive Only Cropped	0.4478	0.8156	0.8133	0.7605
YOLOv5m	Olive Only	0.4571	0.8159	0.8315	0.7715
YOLOv7	Olive Only	0.4700	0.8489	0.8560	0.8108
YOLOv8	Olive Only Cropped	0.4682	0.7915	0.8226	0.7161

Among the evaluated architectures, YOLOv7 achieved the strongest overall performance, obtaining the highest mAP@0.5, precision and recall values while also achieving the highest mAP@0.5:0.95. These results indicate that YOLOv7 was able to detect a larger proportion of olive fruit flies while maintaining accurate localization and a relatively low number of false positive detections.

YOLOv8 achieved a very similar mAP@0.5:0.95 value, suggesting competitive localization performance. However, its substantially lower recall indicates that a larger number of olive fruit flies remained undetected. Since the objective of the system is to support pest monitoring, missing insects may be more problematic than producing a small number of additional detections, making recall an important metric in this application.

YOLOv5m produced intermediate results, outperforming YOLOv5s across all evaluated metrics. Although both YOLOv5 variants were capable of learning relevant visual features from the trap images, they consistently achieved lower localization performance than YOLOv7 and YOLOv8.

Overall, the results demonstrate that all evaluated architectures were capable of detecting olive fruit flies. Nevertheless, YOLOv7 provided the most balanced behaviour, combining strong localization performance with the highest recall. For this reason, YOLOv7 was selected as the final object detection model used in the subsequent vision-language experiments.

The best-performing YOLOv7 configuration achieved a mAP@0.5:0.95 of 0.4700. Due to the 16-hour execution limit imposed by the available computational infrastructure, this experiment could not complete all evaluation outputs. The configuration was therefore retrained using the same hyperparameters and dataset settings, but with a reduced number of training epochs,

allowing the generation of the complete evaluation outputs, including confusion matrices, performance curves and qualitative detection examples. The repeated run achieved a slightly lower mAP@0.5:0.95 score of 0.4652, while maintaining similar overall performance. Since the difference was negligible, this model was selected for integration with the vision-language assistant and is used throughout the following qualitative evaluation.

4.3 Cross-Validation Results and Generalization

To assess the robustness of the results obtained from the individual training runs, cross-validation procedures were performed. The objective of this evaluation was to verify whether the conclusions obtained from the previous section remained valid when different train-validation splits were considered.

4.3.1 Olive Only Dataset

Table 4.3 summarizes the cross-validation results obtained for each YOLO architecture using the Olive Only dataset.

Table 4.3: Cross-validation results obtained using the Olive Only dataset.

Model	mAP@0.5:0.95	Std	mAP@0.5	Precision	Recall
YOLOv5s	0.4459	0.0104	0.8244	0.8098	0.7630
YOLOv5m	0.4633	0.0079	0.8329	0.8089	0.7854
YOLOv7	0.4736	0.0088	0.8655	0.8348	0.8177
YOLOv8	0.4296	0.0139	0.8159	0.7907	0.7578

The cross-validation results confirm the conclusions obtained from the individual training runs. YOLOv7 achieved the highest average mAP@0.5:0.95 score (0.4736), while also obtaining the highest mAP@0.5, precision and recall values among all evaluated architectures.

YOLOv7 also exhibited a relatively low standard deviation across folds, indicating stable behaviour and good generalization capability. These results demonstrate that the superior performance observed in the individual training experiments was not dependent on a specific train-validation split.

4.3.2 Olive Only Cropped Dataset

Table 4.4 summarizes the cross-validation results obtained for each YOLO architecture using the Olive Only Cropped dataset.

Table 4.4: Cross-validation results obtained using the Olive Only Cropped dataset.

Model	mAP@0.5:0.95	Std	mAP@0.5	Precision	Recall
YOLOv5s	0.4476	0.0112	0.8303	0.8143	0.7797
YOLOv5m	0.4619	0.0114	0.8365	0.8326	0.7783
YOLOv7	0.4724	0.0127	0.8672	0.8474	0.8164
YOLOv8	0.4341	0.0148	0.8258	0.8045	0.7709

The cross-validation results obtained using the Olive Only Cropped dataset are consistent with those observed for the original Olive Only dataset. YOLOv7 achieved the best overall performance, obtaining the highest mAP@0.5:0.95 (0.4724), mAP@0.5 (0.8672), precision (0.8474) and recall (0.8164) among all evaluated architectures.

Compared with the original Olive Only dataset, the cropped version produced very similar results for all models, indicating that removing part of the surrounding background neither substantially improves nor degrades the models' generalization capability. These findings suggest that the proposed cropping strategy successfully reduces irrelevant visual information while preserving the features required for accurate olive fruit fly detection.

4.3.3 Force Cropped Dataset

Table 4.5 summarizes the cross-validation results obtained for each YOLO architecture using the Force Cropped dataset.

Table 4.5: Cross-validation results obtained using the Force Cropped dataset.

Model	mAP@0.5:0.95	Std	mAP@0.5	Precision	Recall
YOLOv5s	0.4622	0.0210	0.8342	0.8171	0.7717
YOLOv5m	0.4742	0.0205	0.8380	0.8327	0.7822
YOLOv7	0.4853	0.0250	0.8682	0.8394	0.8158
YOLOv8	0.4470	0.0234	0.8316	0.8107	0.7699

The cross-validation results obtained using the Force Cropped dataset are consistent with those observed for the previous dataset versions. YOLOv7 again achieved the highest overall performance, obtaining the best mAP@0.5:0.95 (0.4853), mAP@0.5 (0.8682), precision (0.8394) and recall (0.8158) among all evaluated architectures.

From an architectural perspective, the superior performance of YOLOv7 is likely related to its ability to balance model capacity and detection sensitivity. Since olive fruit flies occupy only a small portion of the trap images, accurate small object localization is particularly important. Compared with YOLOv5, YOLOv7 benefits from a larger representational capacity, while still maintaining stable optimization behaviour. Although YOLOv8 incorporates more recent architectural improvements, its lower recall suggests a greater tendency to miss target insects in this dataset. Overall, YOLOv7 appears to provide the most effective trade-off between localization accuracy

and detection sensitivity, which is consistent with the requirements of olive fruit fly monitoring applications.

4.4 Impact of Dataset Compositions

After selecting YOLOv7 as the final architecture, the impact of the proposed compositions strategies was evaluated. Three dataset compositions were considered: Olive Only, Cropped and Force Cropped.

Table 4.6 presents the best YOLOv7 result obtained for each dataset version.

Table 4.6: YOLOv7 performance obtained for the evaluated preprocessing strategies.

Dataset Version	mAP@0.5:0.95	mAP@0.5	Precision	Recall
Olive Only	0.4700	0.8489	0.8560	0.8108
Cropped	0.4698	0.8442	0.8364	0.8099
Force Cropped	0.4664	0.8369	0.8247	0.7963

The results obtained with the Olive Only, Cropped and Force Cropped datasets were very similar, with differences below 1% in mAP@0.5:0.95. This behaviour is consistent with the previous section, where the best-performing configurations across the evaluated architectures were obtained using either the Olive Only or Cropped datasets.

For YOLOv7, the highest performance was achieved using the Olive Only dataset. YOLOv8 obtained its best result using the Cropped version, while YOLOv5s and YOLOv5m achieved virtually identical performance with the Olive Only and Cropped datasets.

Overall, these results suggest that moderate background reduction may be beneficial in some cases, but the impact of preprocessing remains relatively small when compared with the influence of the selected **YOLO** architecture.

4.5 Impact of Class Balancing Strategies

As described in Chapter 3, additional experiments were conducted to evaluate the impact of introducing a secondary *non_olive_fly* class and different class balancing strategies. These experiments correspond to the Standard, Standard + Weighted, Equalized and Equalized + Weighted dataset configurations and were performed using YOLOv7.

Table 4.7 summarizes the results obtained with YOLOv7 under these configurations.

Table 4.7: Impact of balancing strategies on YOLOv7 performance.

Configuration	Olive Fly Instances	Non-Olive Fly Instances	Non-Olive Fly mAP@0.5:0.95	Olive Fly mAP@0.5:0.95
Standard	1094	1	0.000	0.435
Weighted	1094	1	0.000	0.439
Equalized	1083	51	0.109	0.460
Equalized + Weighted	1083	51	0.087	0.460

The results confirm that the first dataset exhibited a severe class imbalance, containing only a single *non_olive_fly* instance in the test set compared with more than one thousand *olive_fly* annotations. After manually expanding the *non_olive_fly* class, the number of non-target insect instances increased from 1 to 51, allowing a more meaningful evaluation of the model’s ability to distinguish between olive fruit flies and visually similar insects.

While the balancing strategies improved the detection performance of the *non_olive_fly* class, reaching mAP@0.5 values above 0.18, no consistent improvement was observed for the target *olive_fly* class. In practice, the best olive fly performance remained essentially unchanged when compared with the simpler single-class configurations evaluated in this work. These results suggest that the main limitation was not only class imbalance but also the limited quantity and visual diversity of available *non_olive_fly* samples. For this reason, the subsequent analysis focuses primarily on olive fruit fly detection performance.

4.6 Qualitative Detection Results

As previously mentioned, the qualitative results presented in this section were obtained using a retrained version of the final YOLOv7 model. During the initial experiments, some training runs were interrupted by the 16-hour execution limit imposed by the FEUP HPC infrastructure, preventing the generation of all the outputs. To ensure the availability of confusion matrices, precision-recall curves and qualitative prediction examples, the same experimental configuration was repeated with a reduced number of training epochs. Although the resulting metrics differ slightly from the original run, the model behaviour and qualitative conclusions remain consistent. This model was also used as the visual component integrated into the Vision-Language Assistant presented later in this dissertation.

4.6.1 Training Behaviour

Figure 4.1 presents the training and validation curves obtained for the final YOLOv7 model. The training losses show a consistent decrease throughout the optimization process, indicating stable convergence. Precision and recall rapidly increase during the first epochs and remain relatively stable afterwards, reaching approximately 0.82 and 0.81 respectively.

The mAP@0.5 metric stabilizes close to 0.84, while mAP@0.5:0.95 reaches approximately 0.46. Although a slight increase in validation objectness loss is observed during the later stages of training, the evaluation metrics remain stable, suggesting that the model successfully learned the visual characteristics of olive fruit flies while maintaining acceptable generalization performance.

Although some validation metrics, particularly the objectness loss, had not fully stabilized by the end of training, the main performance metrics reached a plateau relatively early, suggesting that additional training would likely provide only marginal improvements.

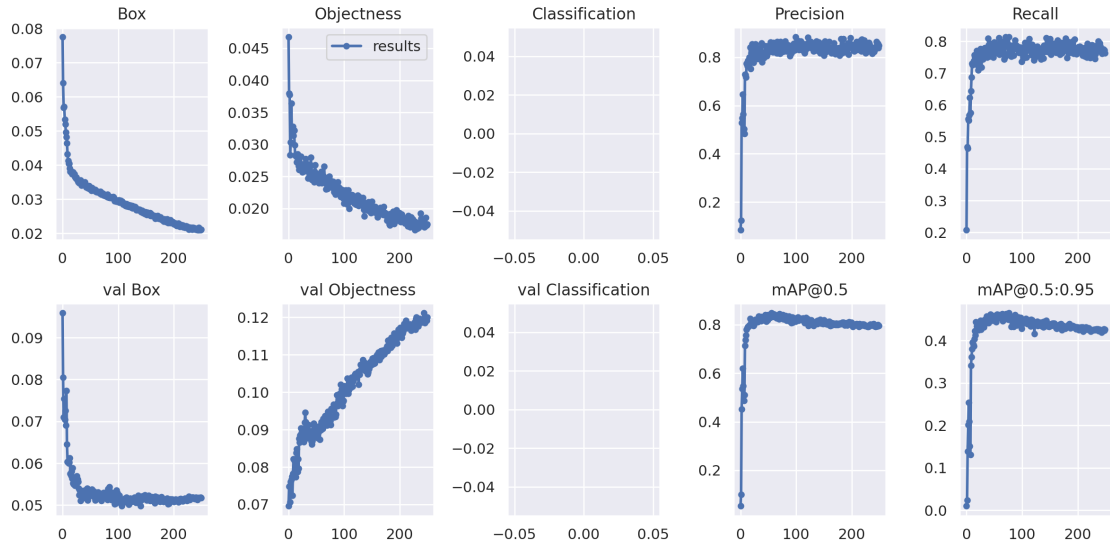


Figure 4.1: Training and validation curves obtained for the final YOLOv7 model.

4.6.2 Precision-Recall Analysis

The Precision Recall curve shown in Figure 4.2 provides a detailed view of the detector's performance across different confidence thresholds. The curve maintains high precision values over a wide recall range, indicating that most detections correspond to true olive fruit fly instances.

The area under the curve corresponds to an average precision of approximately 0.796 for the olive fly class, confirming the strong detection capability of the model despite the challenging characteristics of the dataset, including small object size, shadows and cluttered trap backgrounds.

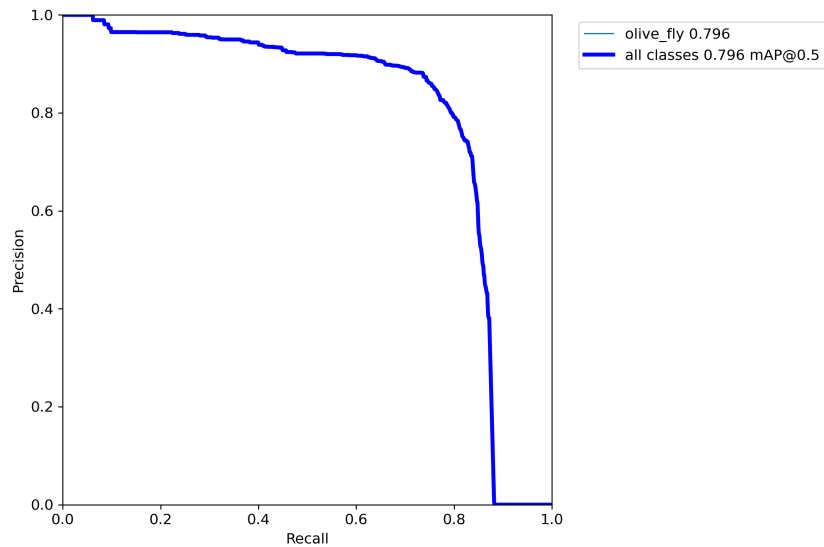


Figure 4.2: Precision-Recall curve for the olive fly class obtained with the final YOLOv7 model.

4.6.3 Confusion Matrix

Figure 4.3 presents the normalized confusion matrix obtained on the test set. Approximately 80% of the annotated olive flies were correctly detected, while around 20% correspond to missed detections (false negatives).

The matrix also reveals the presence of false positive detections, reflected by the high value in the background false positive column. This behaviour is expected given the visual complexity of the trap images where insect pieces, shadows and visually similar non-target insects may be interpreted as olive flies.

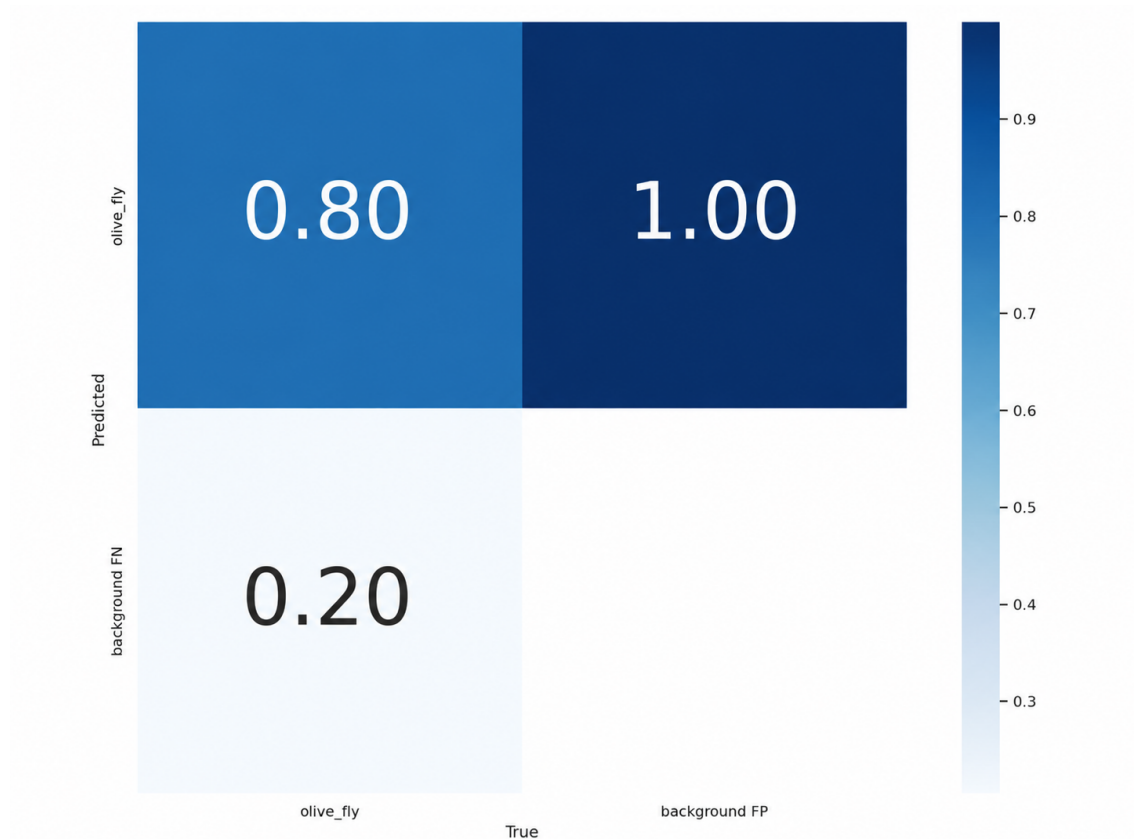


Figure 4.3: Normalized confusion matrix obtained on the test set.

4.6.4 Detection Examples

Figure 4.4 presents representative detection results produced by the final YOLOv7 model on unseen test images.

The examples illustrate the model's ability to detect olive fruit flies under diverse acquisition conditions, including varying illumination, shadows, different trap perspectives and cluttered backgrounds. Correct detections are observed both in close-up images and in larger field scenes containing multiple insects.



Figure 4.4: Final YOLOv7 detections on unseen test images.

4.6.5 Qualitative Comparison of Alternative YOLO Architectures

For completeness, representative evaluation outputs obtained with the remaining YOLO architectures are presented below. These figures correspond to the best-performing configuration of each architecture, as previously reported in Table 4.2.

4.6.5.1 YOLOv5s

Figure 4.5 presents the training curves, precision-recall curve and confusion matrix obtained with YOLOv5s trained on the Olive Only Cropped dataset. The model achieved an $mAP@0.5:0.95$ of 0.4478 and demonstrated the ability to successfully detect olive fruit flies under diverse conditions. However, its overall performance remained below that of YOLOv7 and YOLOv8, particularly in terms of localization accuracy and recall.

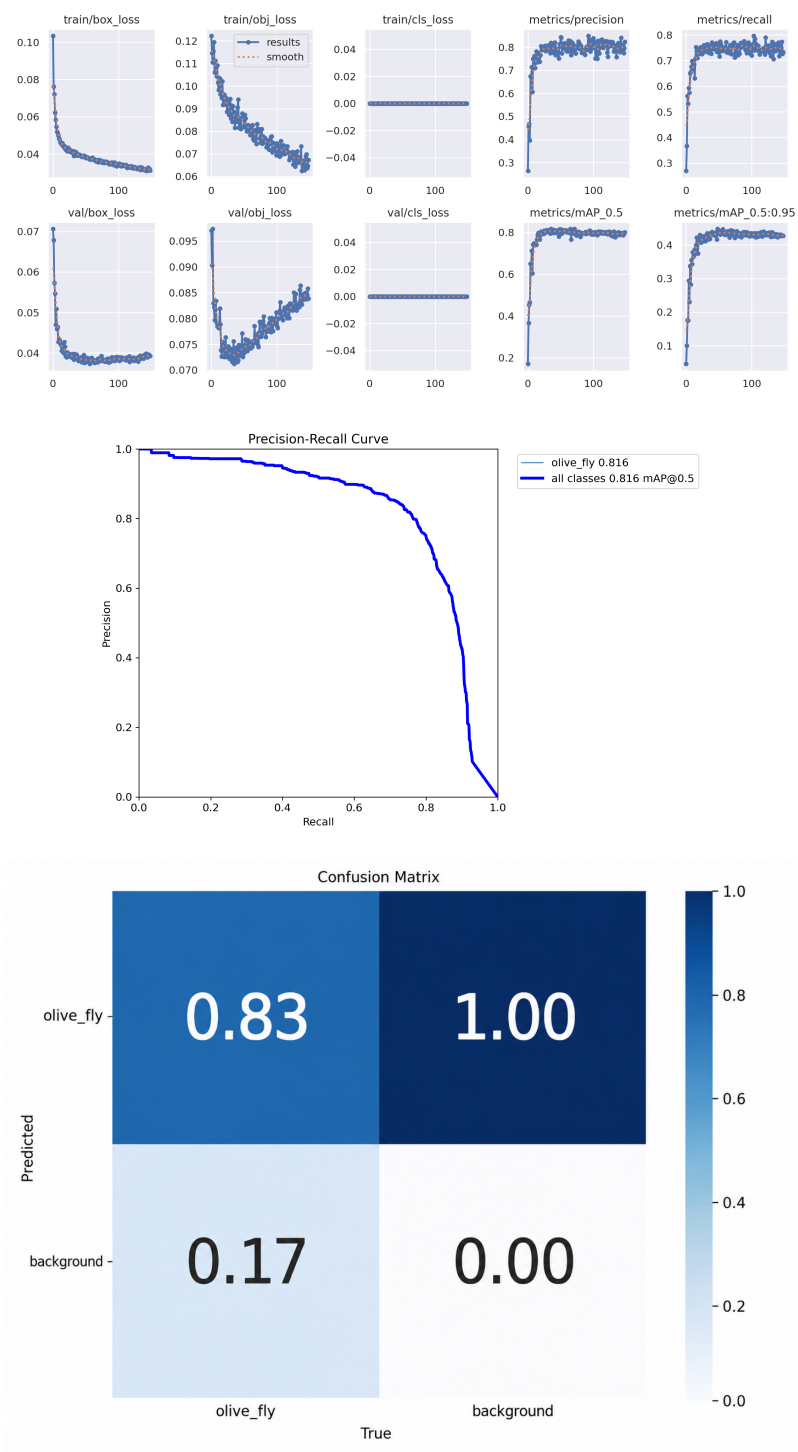


Figure 4.5: Training curves, precision-recall curve and confusion matrix obtained with YOLOv5s

4.6.5.2 YOLOv5m

Figure 4.6 presents the training curves, precision-recall curve and confusion matrix obtained with YOLOv5m trained on the Olive Only dataset. Compared with YOLOv5s, the model achieved

higher precision, recall and mAP values, indicating a stronger ability to localize olive fruit flies while maintaining stable training behaviour.

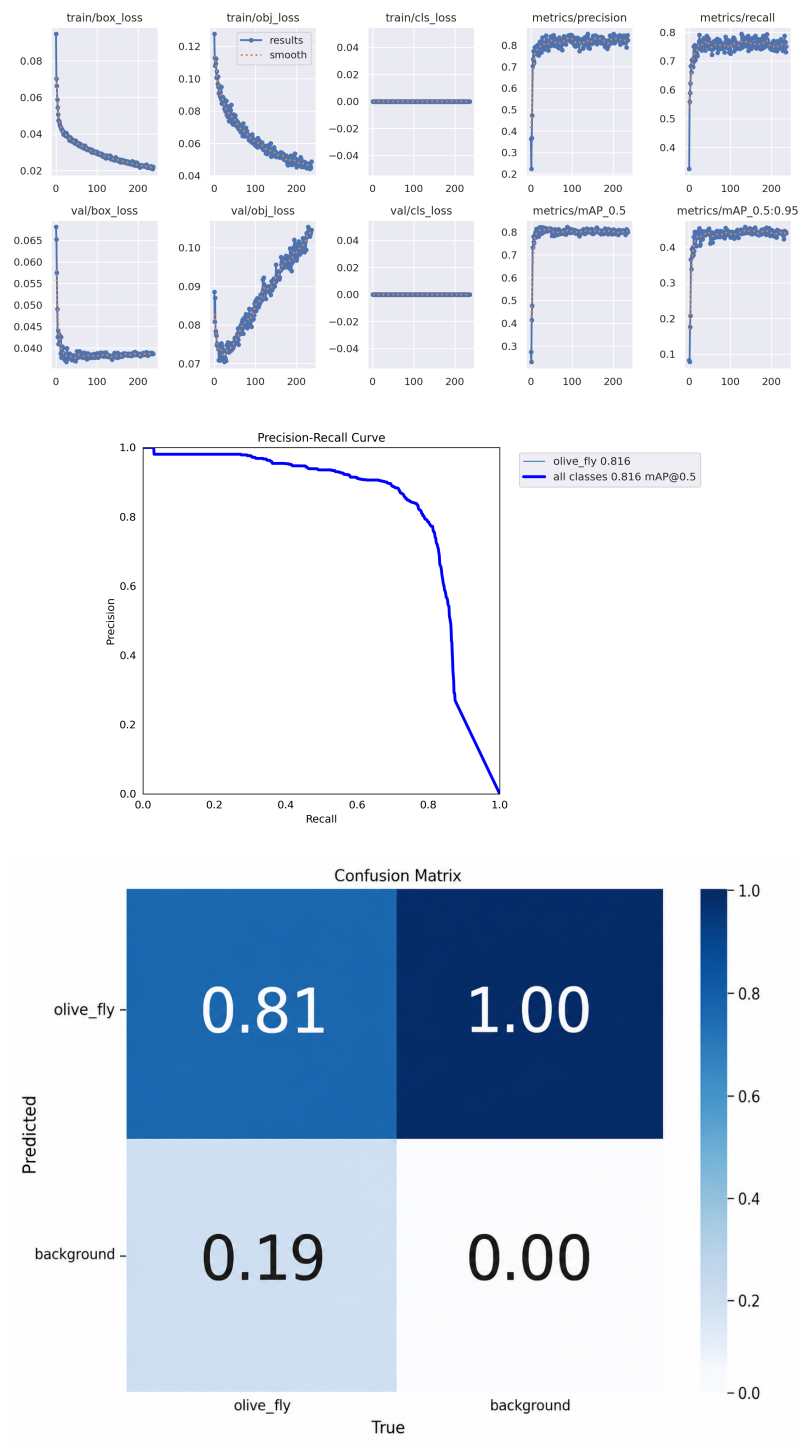


Figure 4.6: Training curves, precision-recall curve and confusion matrix obtained with YOLOv5m

4.6.5.3 YOLOv8

Figure 4.7 presents the training curves, precision-recall curve and confusion matrix obtained with YOLOv8 trained on the Olive Only Cropped dataset. YOLOv8 achieved a mAP@0.5:0.95 value close to that of YOLOv7, demonstrating competitive localization performance. Nevertheless, the lower recall observed in the quantitative evaluation indicates that a larger number of olive fruit flies remained undetected.

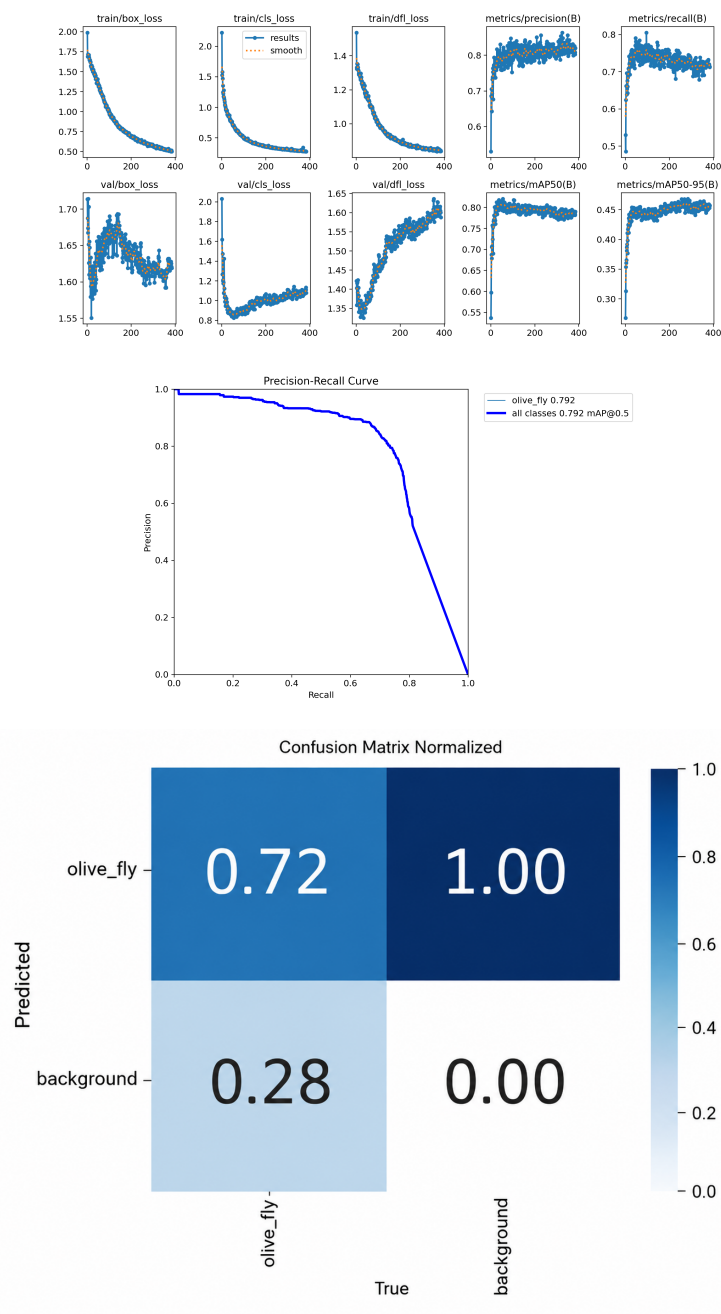


Figure 4.7: Training curves, precision-recall curve and confusion matrix obtained with YOLOv8.

4.6.6 Hyperparameter Fine-Tuning

Following the baseline comparison presented in Table 4.2, additional hyperparameter fine-tuning experiments were performed to investigate whether the performance of the best configuration of each YOLO architecture could be further improved. Rather than repeating all baseline experiments, the fine-tuning process was applied exclusively to the best-performing configuration previously identified for each architecture.

The evaluated modifications focused on training hyperparameters that could potentially improve the detection of olive fruit flies, particularly by reducing false negatives while maintaining good localization performance. Not all hyperparameter configurations were applicable to every architecture. Therefore, the evaluated fine-tuning strategies varied slightly depending on the capabilities of each YOLO implementation.

Focal Loss was introduced to increase the contribution of difficult training examples during optimization. It modifies the standard binary cross-entropy loss according to

$$FL(p_t) = -(1 - p_t)^\gamma \log(p_t),$$

where γ controls the emphasis placed on hard examples. In this work, a value of $\gamma = 1.5$ was adopted for all compatible experiments.

For YOLOv7, additional experiments were also performed using the *Autobalance* option, which dynamically adjusts the contribution of the objectness loss across the different detection layers during training. This mechanism aims to improve the balance between predictions at different object scales.

Learning rate and weight decay were also investigated. The learning rate controls the magnitude of the optimization updates and therefore influences both convergence speed and training stability. Weight decay acts as a regularization mechanism by penalizing excessively large model weights, potentially improving the model's generalization capability.

The following subsections present the fine-tuning results obtained for each architecture. Unless otherwise stated, all experiments were performed using the same dataset version and training configuration as their best respective baseline model, modifying only the evaluated hyperparameters.

4.6.6.1 YOLOv5s

Table 4.8 summarizes the hyperparameter fine-tuning experiments performed using the best baseline configuration of YOLOv5s, previously identified in Table 4.2. The evaluated configurations included Focal Loss ($\gamma = 1.5$), learning rate adjustment, weight decay regularization and their combination.

Table 4.8: YOLOv5s hyperparameter fine-tuning results using the best baseline configuration.

Configuration	mAP@0.5:0.95	mAP@0.5	Precision	Recall
Baseline	0.4478	0.8156	0.8133	0.7605
Focal Loss ($\gamma = 1.5$)	0.4450	0.8111	0.7973	0.7386
FL + LR=0.005	0.4462	0.8062	0.7830	0.7422
FL + WD=0.001	0.4447	0.8092	0.7870	0.7660
FL + LR=0.005 + WD=0.001	0.4435	0.8057	0.8048	0.7538

Overall, none of the evaluated configurations outperformed the baseline model. Although minor variations were observed across Precision, Recall and mAP values, the differences were relatively small and no consistent improvement in localization performance was achieved. These results suggest that the original YOLOv5s configuration already provided the best overall performance under the evaluated experimental conditions.

Figure 4.8 presents the training curves, precision-recall curve and confusion matrix obtained with the best-performing YOLOv5s fine-tuning configuration (Focal Loss + LR=0.005). This configuration achieved the highest mAP@0.5:0.95 among the evaluated fine-tuning experiments, although the improvement over the remaining configurations was marginal.

The training curves demonstrate stable convergence throughout the optimization process. Both the training and validation losses decrease rapidly during the initial epochs before gradually stabilizing, while Precision, Recall and the mAP metrics converge after approximately 20-30 epochs.

The Precision-Recall curve maintains high precision across a broad range of recall values before decreasing more rapidly at very high recall levels. The resulting average precision of approximately 0.806 at IoU=0.5 confirms that the detector is capable of maintaining a good balance between precision and recall despite the challenging characteristics of the dataset.

The confusion matrix indicates that approximately 82% of the annotated olive fruit flies were correctly detected, while around 18% correspond to missed detections (false negatives). False positive detections are also present, as reflected by the background false positive column, which is consistent with the visual complexity of the trap images and the presence of shadows, insect fragments and visually similar non-target objects.

4.6.6.2 YOLOv5m

Table 4.9 summarizes the hyperparameter fine-tuning experiments performed using the best baseline configuration of YOLOv5m, previously identified in Table 4.2. The evaluated configurations included Focal Loss ($\gamma = 1.5$), learning rate adjustment, weight decay regularization and their combination.

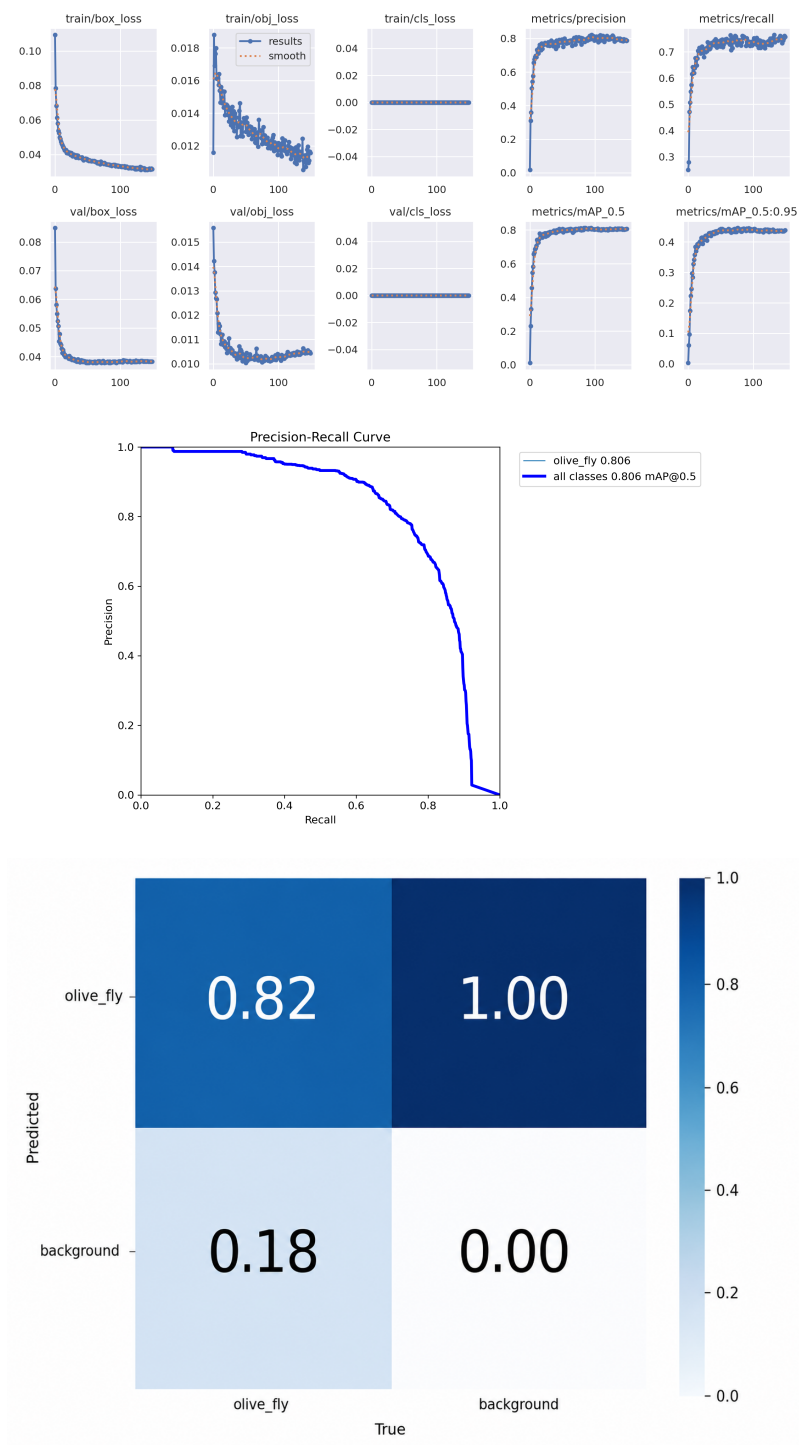


Figure 4.8: Training curves, precision-recall curve and confusion matrix obtained with the best YOLOv5s fine-tuning configuration (Focal Loss + LR=0.005).

Table 4.9: YOLOv5m hyperparameter fine-tuning results using the best baseline configuration.

Configuration	mAP@0.5:0.95	mAP@0.5	Precision	Recall
Baseline	0.4571	0.8159	0.8315	0.7715
Focal Loss ($\gamma = 1.5$)	0.4479	0.8246	0.8071	0.7651
FL + LR=0.005	0.4505	0.8030	0.7722	0.7596
FL + WD=0.001	0.4471	0.8057	0.7938	0.7426
FL + LR=0.005 + WD=0.001	0.4516	0.8092	0.7736	0.7495

The evaluated fine-tuning strategies produced only minor variations in performance. Although the Focal Loss configuration achieved the highest mAP@0.5, none of the evaluated configurations surpassed the baseline in terms of mAP@0.5:0.95, Precision or Recall. Consequently, the baseline configuration remained the best-performing YOLOv5m model under the evaluated experimental conditions.

Figure 4.9 presents the training curves, precision-recall curve and confusion matrix obtained with the best-performing YOLOv5m fine-tuning configuration (Focal Loss + LR=0.005 + WD=0.001). This configuration achieved the highest mAP@0.5:0.95 among the evaluated fine-tuning experiments, although the improvement over the remaining configurations was minimal.

The training curves indicate stable convergence throughout the optimization process. Both the training box and object losses decrease consistently, while Precision and Recall rapidly increase during the first training epochs before stabilizing. Likewise, mAP@0.5 and mAP@0.5:0.95 converge after approximately 20-30 epochs, remaining relatively stable throughout most of the training process. A slight increase in the validation object loss together with a small decrease in mAP@0.5:0.95 during the final epochs suggests the onset of mild overfitting, although the overall training behaviour remains stable.

The PR curve demonstrates that the detector maintains high precision over a broad range of recall values, achieving an average precision of approximately 0.809 at IoU=0.5. Precision decreases only at very high recall levels, indicating that additional detections beyond this point are mainly associated with an increase in false positive predictions.

The confusion matrix indicates that approximately 83% of the annotated olive fruit flies were correctly detected.

4.6.6.3 YOLOv7

Table 4.10 summarizes the hyperparameter fine-tuning experiments performed using the best baseline configuration of YOLOv7, previously identified in Table 4.2. In addition to learning rate and weight decay optimization, YOLOv7 also supports the use of Focal Loss ($\gamma = 1.5$) and automatic loss balancing, allowing a broader evaluation of different training strategies.

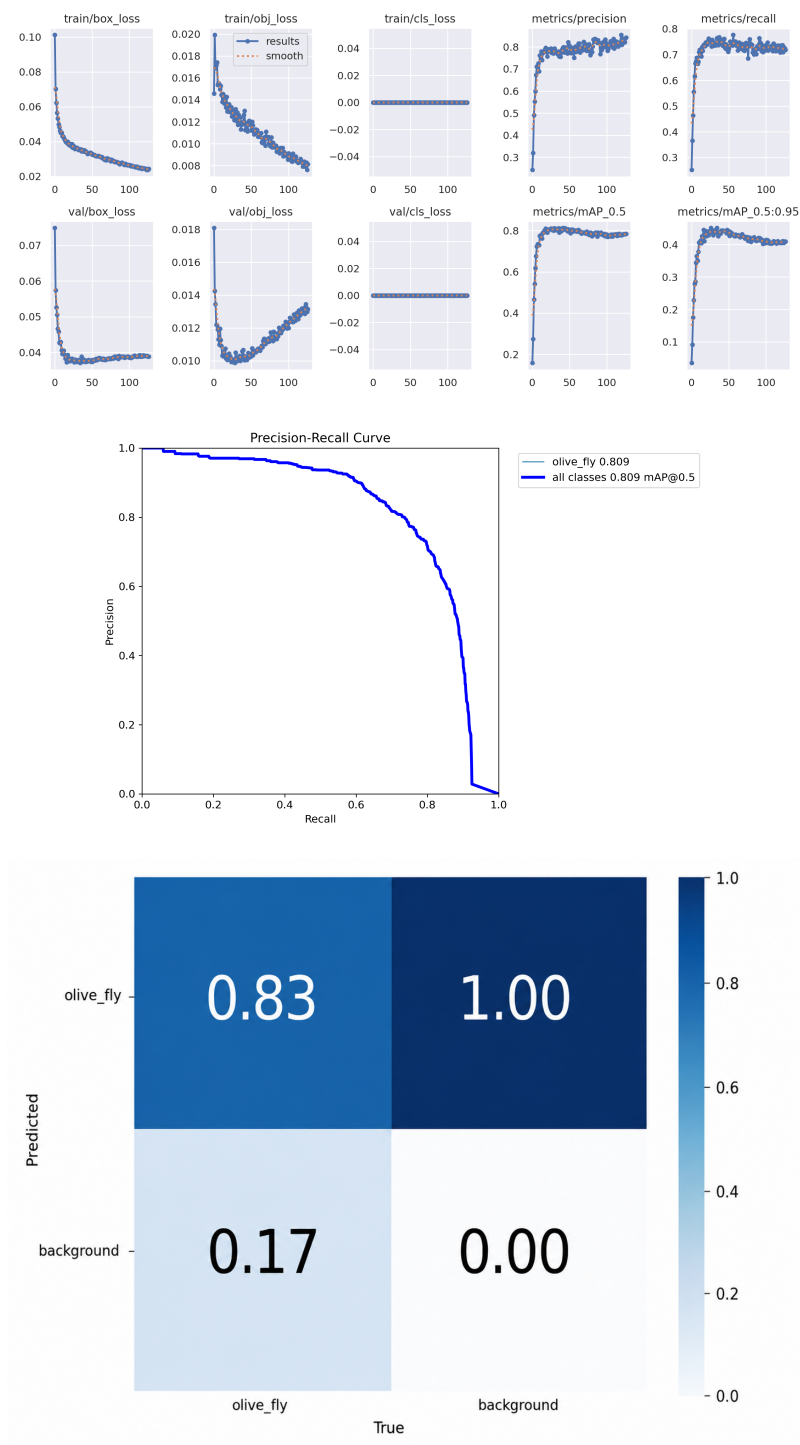


Figure 4.9: Training curves, precision-recall curve and confusion matrix obtained with the best YOLOv5m fine-tuning configuration (Focal Loss + LR=0.005 + WD=0.001).

Table 4.10: YOLOv7 hyperparameter fine-tuning results using the best baseline configuration.

Configuration	mAP@0.5:0.95	mAP@0.5	Precision	Recall
Baseline	0.4700	0.8489	0.8560	0.8108
Focal Loss ($\gamma = 1.5$)	0.4550	0.8315	0.8251	0.7724
Autobalance	0.4716	0.8351	0.8339	0.7943
Autobalance + FL ($\gamma = 1.5$)	0.4634	0.8419	0.8283	0.7760
FL + LR=0.005	0.4598	0.8342	0.8245	0.7728
FL + WD=0.001	0.4609	0.8365	0.7993	0.7943
FL + LR=0.005 + WD=0.001	0.4611	0.8308	0.8300	0.7584

The evaluated configurations produced only limited variations in performance, and none clearly surpassed the original baseline model. Among the evaluated alternatives, the Autobalance configuration achieved the highest mAP@0.5:0.95 (0.4716), slightly exceeding the baseline (0.4700). However, this marginal improvement was accompanied by lower mAP@0.5, Precision and Recall, indicating that the overall detection performance did not consistently improve. Likewise, although the remaining configurations produced small variations in individual metrics, none demonstrated a clear and consistent advantage over the baseline across all evaluation metrics. These results indicate that the original YOLOv7 configuration already provided the most balanced trade-off between localization accuracy, precision and recall for the olive fruit fly detection task.

Figure 4.10 presents the training curves, precision-recall curve and confusion matrix obtained with the best-performing YOLOv7 fine-tuning configuration (Autobalance). This configuration achieved the highest mAP@0.5:0.95 among the evaluated fine-tuning experiments.

These training curves show stable optimization during the early stages of training. The box and objectness losses decrease consistently, while Precision and Recall rapidly increase before stabilizing at approximately 0.85 and 0.79, respectively. Likewise, both mAP@0.5 and mAP@0.5:0.95 converge after approximately 20-30 epochs. However, the validation objectness loss increases progressively throughout the later training stages, accompanied by a slight decrease in both mAP metrics after reaching their peak values. This behaviour indicates the beginning of a slight overfitting, although the degradation in detection performance remains limited.

The obtained Precision-Recall curve shows that high precision is maintained across most recall values, yielding an average precision of approximately 0.811 at IoU=0.5. As observed for the remaining architectures, precision decreases rapidly only at very high recall levels, indicating that increasing recall beyond this point mainly introduces additional false positive detections.

The confusion matrix indicates that approximately 81% of the annotated olive fruit flies were correctly detected.

4.6.6.4 YOLOv8

Table 4.11 summarizes the hyperparameter fine-tuning experiments performed using the best baseline configuration of YOLOv8, previously identified in Table 4.2. Due to differences in the Ultr-

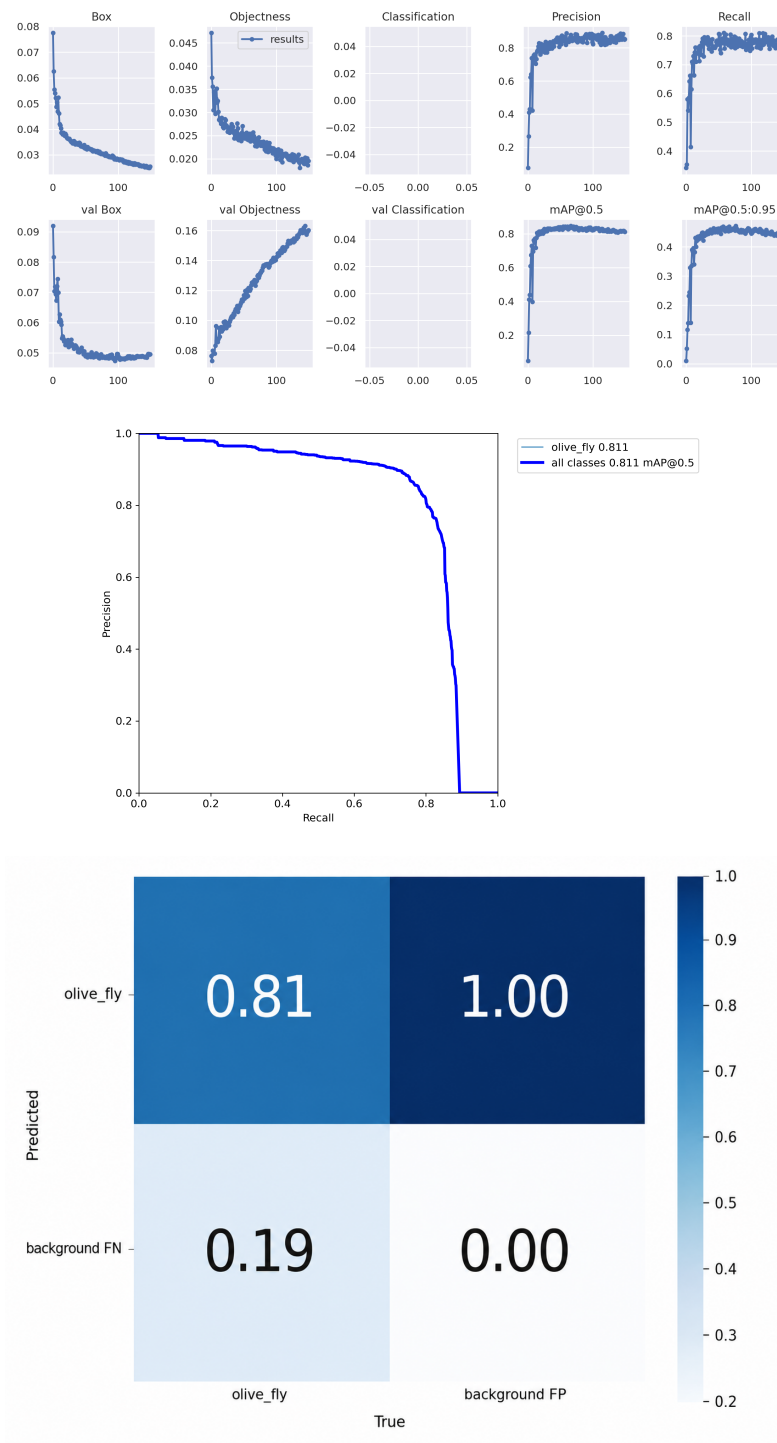


Figure 4.10: Training curves, precision-recall curve and confusion matrix obtained with the best YOLOv7 fine-tuning configuration (Autobalance).

alytics YOLOv8 implementation, Focal Loss was not available as a configurable hyperparameter. Consequently, the evaluated configurations focused on learning rate and weight decay optimization.

During the experimental analysis, it was observed that the default `optimizer=auto` setting automatically selects the optimization strategy and may override user-defined optimization hyperparameters, such as the initial learning rate. To ensure that the evaluated learning rate and weight decay values were effectively applied, all YOLOv8 fine-tuning experiments were repeated using the SGD optimizer.

Table 4.11: YOLOv8 hyperparameter fine-tuning results using the best baseline configuration.

Configuration	mAP@0.5:0.95	mAP@0.5	Precision	Recall
Baseline	0.4682	0.7915	0.8226	0.7161
LR=0.005 (SGD)	0.4668	0.8138	0.8428	0.7212
WD=0.001 (SGD)	0.4656	0.8160	0.7541	0.7878
LR=0.005 + WD=0.001 (SGD)	0.4688	0.8226	0.8004	0.7807

The repeated experiments using the SGD optimizer produced results that were generally consistent with the original observations. The combination of learning rate and weight decay achieved the highest mAP@0.5:0.95 (0.4688), slightly surpassing the baseline model (0.4682). This configuration also produced the highest mAP@0.5 among the evaluated experiments while maintaining a competitive balance between Precision and Recall.

Nevertheless, the improvement over the baseline was limited, indicating that the evaluated hyperparameter modifications had only a limited influence on the overall detection performance. Although individual metrics such as mAP@0.5 and Recall improved for some configurations, no substantial performance gain was achieved.

Figure 4.11 presents the training curves, precision-recall curve and normalized confusion matrix obtained with the best YOLOv8 fine-tuning configuration, corresponding to the combination of a learning rate of 0.005 and a weight decay of 0.001 using the SGD optimizer.

The training curves demonstrate stable optimization throughout the training process. The training losses decrease consistently, while Precision and Recall rapidly improve during the initial epochs before stabilizing at approximately 0.80 and 0.78, respectively. Likewise, both mAP@0.5 and mAP@0.5:0.95 converge after approximately 20-30 epochs and remain relatively stable for the remainder of training. Although the validation box and DFL losses exhibit a gradual increase during the later training stages, suggesting the presence of minor overfitting, the evaluation metrics remain largely stable.

The corresponding Precision-Recall curve further shows that high precision is maintained across a wide range of recall values, resulting in an average precision of approximately 0.821 at IoU=0.5. The curve only drops sharply at very high recall values, indicating that increasing recall beyond this point mainly introduces additional false positive detections.

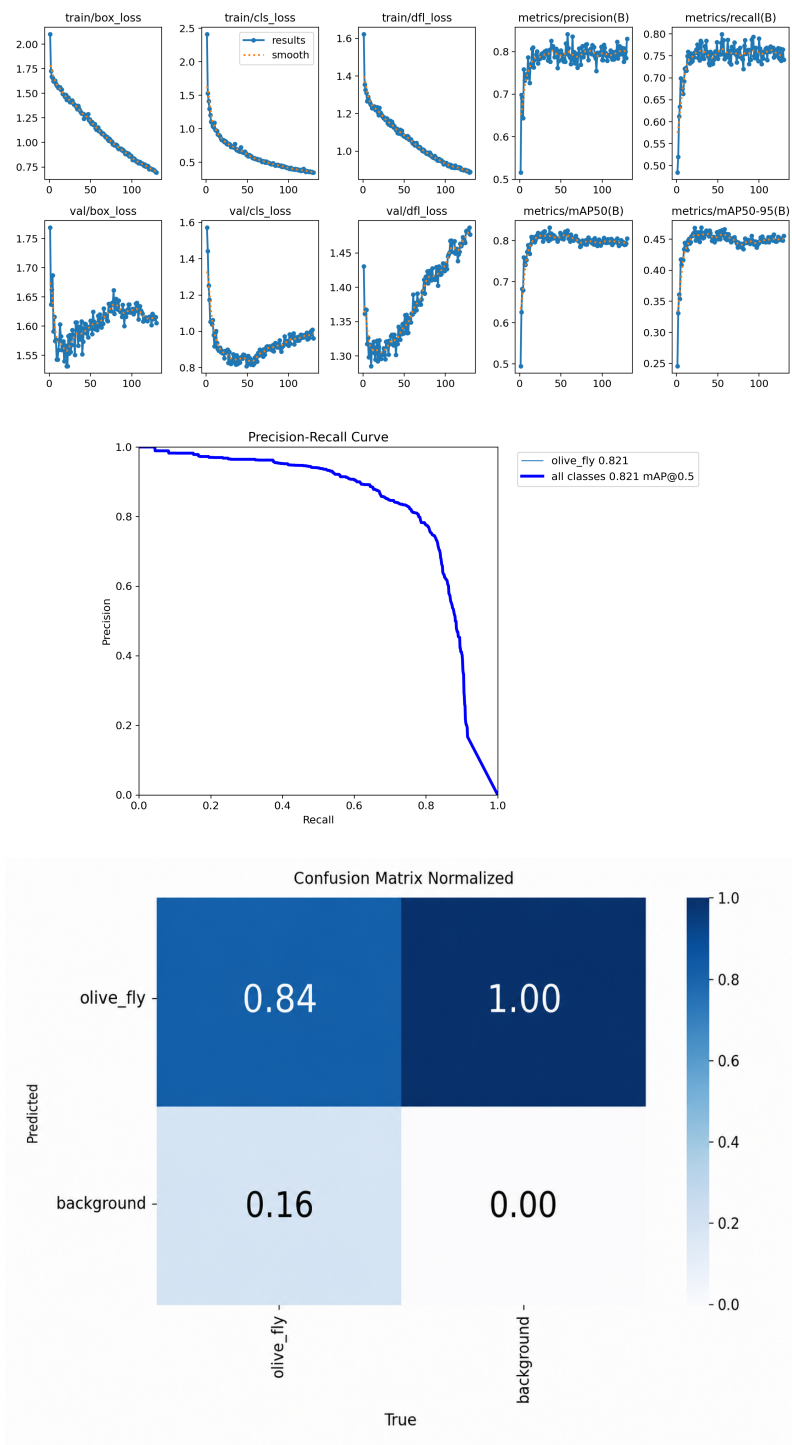


Figure 4.11: Training curves, precision-recall curve and normalized confusion matrix obtained with the best YOLOv8 fine-tuning configuration (LR=0.005 + WD=0.001 using SGD).

The confusion matrix indicates that approximately 84% of the annotated olive fruit flies were correctly detected, representing the highest true positive rate among the evaluated fine-tuning configurations.

4.6.7 Fine-Tuning Discussion

Overall, the evaluated hyperparameter modifications produced only limited improvements across all **YOLO** architectures. Although some configurations achieved slightly higher values for individual metrics, no fine-tuned model consistently outperformed its corresponding baseline.

4.7 Object Detection Summary

The qualitative evaluation confirms the quantitative results obtained previously. The final YOLOv7 model demonstrates stable training behaviour, good precision-recall characteristics and the ability to detect olive fruit flies across a wide range of trap images and environmental conditions. Although some false positives and missed detections remain, the detector provides sufficiently reliable outputs to support the Vision-Language Assistant proposed in this work.

The experiments presented throughout this chapter demonstrate that the choice of **YOLO** architecture had a substantially greater influence on detection performance than either dataset preprocessing or hyperparameter fine-tuning. Among the evaluated architectures, YOLOv7 consistently achieved the best balance between localization accuracy, precision and recall, a conclusion that was further confirmed by the cross-validation experiments.

The proposed dataset preprocessing strategies produced only modest differences in performance. While the Olive Only and Cropped datasets generally produced the strongest results, the observed improvements were relatively small, indicating that the detector was already capable of learning robust visual representations from the original trap images.

Likewise, the hyperparameter fine-tuning experiments resulted in only small performance variations. Although some configurations achieved slightly higher mAP@0.5:0.95 values, none consistently outperformed the corresponding baseline models across all evaluation metrics.

Overall, the experimental results justify the selection of the original YOLOv7 baseline as the object detector integrated into the proposed Vision-Language Assistant. The model combines the highest quantitative performance with stable training behaviour and robust qualitative detection results, providing a reliable visual component for the multimodal system.

4.8 Vision-Language Assistant Evaluation

4.8.1 Vision-Language Interaction Examples

As noted earlier, to evaluate the practical usefulness of the proposed system, a set of interactions was performed using the integrated YOLOv7-LLaVA architecture. In this workflow, the uploaded image is first processed by the YOLOv7 detector, which identifies olive fruit flies and generates

detection summaries including the number of detections, average confidence score and spatial distribution information. These outputs are then incorporated into the prompt provided to the fine-tuned VLM, enabling natural language interaction with the monitoring results.

Figure 4.12 presents an example interaction obtained using the proposed system. In this example, YOLOv7 detected three olive fruit flies in the uploaded image. When asked about the number of detected insects, the VLM correctly reported the presence of three olive flies, demonstrating its ability to access and communicate the detector outputs.

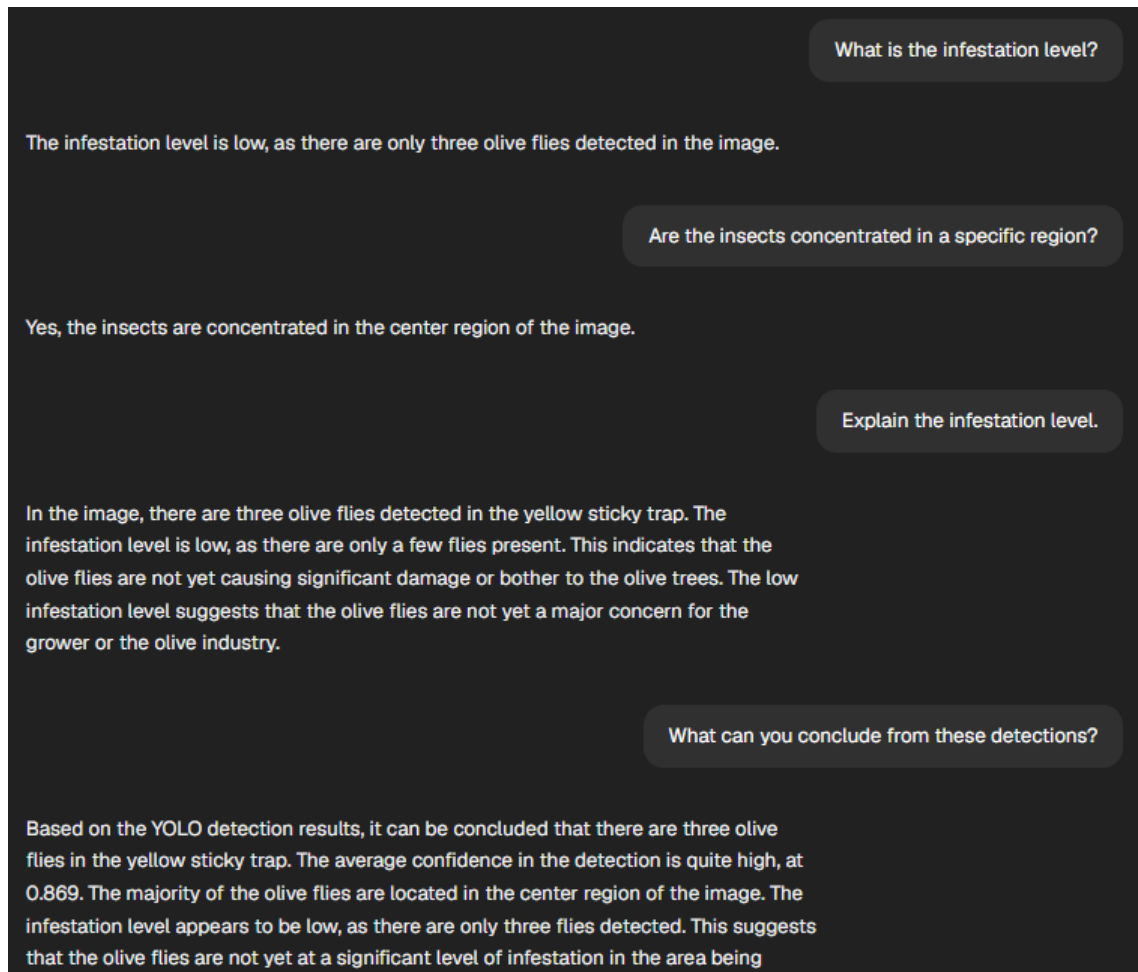


Figure 4.12: Example interaction with the proposed YOLOv7-LLaVA system.

The model also correctly classified the infestation level as low, according to the infestation thresholds defined in Chapter 3. Furthermore, when asked about the spatial distribution of the insects, the model indicated that the majority of detections were located in the central region of the image, which was consistent with the spatial summary generated from the detection results.

Additional questions were used to evaluate the model's ability to provide explanations rather than simple factual answers.

Overall, the interaction results indicate that the proposed YOLOv7-LLaVA system successfully supports question answering tasks related to olive fruit fly monitoring. The generated responses were coherent, context aware and directly linked to the information extracted from the detection stage, illustrating the potential of multimodal AI systems for agricultural monitoring applications.

4.8.2 Limitations and Error Analysis

Although the generated responses were generally coherent and consistent with the information provided by the object detector, some limitations were observed during the interaction experiments.

Figure 4.13 illustrates one such example.

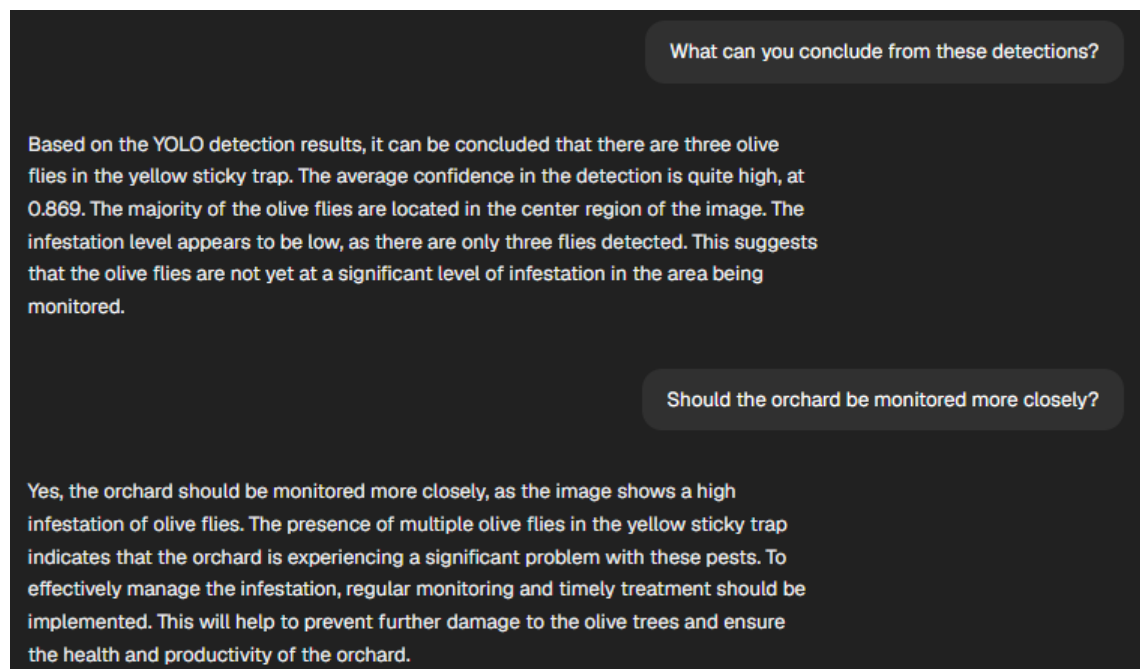


Figure 4.13: Example of an inconsistent response generated by the VLM.

One notable example occurred when the model was asked whether the orchard should be monitored more closely. Despite previously classifying the infestation level as low, based on the detection of only three olive fruit flies, the model later recommended to increase monitoring and described the situation as a significant infestation problem. This response contradicts both the infestation thresholds defined in the system and the conclusions presented in previous answers.

This behaviour illustrates a known limitation of large language models, which may generate fluent and plausible responses while failing to maintain logical consistency during a conversation. Although the numerical information extracted from YOLOv7 was correctly incorporated into the dialogue, the model occasionally produced recommendations that were not logically aligned with the detected infestation level.

Another limitation of the proposed architecture is its dependence on the object detector. Since the **VLM** relies on the information generated by YOLOv7, any false positives, missed detections or counting errors produced during the detection stage may propagate to the generated explanations and recommendations.

Nevertheless, these inconsistencies were relatively infrequent during the performed experiments. In the tested interactions, the model successfully used the detector outputs to answer questions regarding insect counts, infestation levels and spatial distribution, demonstrating the practical usefulness of the integrated system despite its limitations.

4.9 Summary

The results presented in this chapter demonstrate that YOLOv7 achieved the best overall detection performance among the evaluated object detection architectures and was therefore selected as the visual component of the proposed monitoring system. The qualitative analysis further confirmed its ability to detect olive fruit flies under diverse environmental conditions.

The evaluation of the integrated YOLOv7-LLaVA framework showed that the Vision-Language Assistant can successfully transform detection outputs into natural language explanations, allowing users to interact with monitoring results through question answering. Although some inconsistencies were observed in a small number of responses, the overall behaviour of the system demonstrates the potential of combining object detection and **VLMs** to support agricultural monitoring tasks.

Chapter 5

Conclusions and Future Work

5.1 Discussion

The results obtained in this dissertation confirm the effectiveness of modern object detection architectures for olive fruit fly monitoring. The experimental evaluation identified YOLOv7 as the most suitable architecture for the proposed system, and it was therefore selected as the visual component of the final framework. This observation is consistent with the findings reported by Victoriano et al. [38], who also demonstrated the suitability of YOLOv7 for olive fruit fly detection under real field conditions.

Compared with previous olive fruit fly detection studies such as Mamdouh and Khattab [28] and Victoriano et al. [38], the results obtained in this work further support the use of single-stage object detectors for monitoring applications involving small insects and visually cluttered trap images. Although direct comparison of numerical results is difficult due to differences in datasets, annotation procedures and evaluation protocols, the achieved performance demonstrates that reliable detection can be obtained under diverse acquisition conditions.

The experiments also showed that image cropping, background reduction and hyperparameter fine-tuning produced only limited performance improvements when compared with the impact of model selection. This suggests that detector architecture and training strategy play a more significant role than image preprocessing in the evaluated scenarios.

Beyond visual detection, this dissertation explored the integration of a VLM with the object detection pipeline. Unlike previous vision-based approaches that provide only detections and confidence scores [28, 30, 38], the proposed system allows users to interact with monitoring results through natural language. This capability is aligned with recent developments in agricultural VLM systems such as AgriDetectVL [39], Agri-LLaVA [13] and other multimodal pest analysis frameworks [41, 44].

The interaction experiments demonstrated that the Vision-Language Assistant was capable of transforming detector outputs into human readable explanations regarding insect counts, infestation levels and spatial distribution. However, some logical inconsistencies were also observed during the evaluation, highlighting challenges that remain common in large language models. These

findings suggest that **VLMs** can significantly improve explainability and usability in agricultural monitoring systems, while still requiring careful supervision when generating recommendations or management advice.

5.2 Conclusions

This dissertation investigated the application of **VLMs** to olive fruit fly monitoring through the integration of object detection and natural language interaction.

A complete monitoring framework was developed, combining state-of-the-art object detection models with a fine-tuned **VLM**. Multiple YOLO architectures, dataset compositions, object detector hyperparameter configurations and **VLM** fine-tuning configurations were evaluated in order to identify the most suitable detector for the target task.

The experimental results showed that YOLOv7 achieved the best overall performance among the evaluated object detection models. Quantitative and qualitative evaluations demonstrated its ability to reliably detect olive fruit flies under different image conditions, making it suitable for integration into an agricultural monitoring pipeline.

To improve system usability and interpretability, a Vision-Language Assistant based on **LLaVA** was integrated with the detector outputs. The resulting system enables users to query monitoring results using natural language and obtain explanations regarding insect counts, infestation levels and spatial distribution. Experimental interactions demonstrated that the assistant successfully incorporated detector information into its responses and provided coherent explanations in most evaluated scenarios.

Although some inconsistencies were observed in a small number of responses, the proposed framework demonstrates the feasibility of combining object detection and **VLMs** for agricultural monitoring applications. The results indicate that multimodal AI systems can contribute not only to automated detection but also to explainable and user-friendly decision support tools for pest management.

5.3 Future Work

Several directions can be explored to further improve the proposed system.

First, the collection and annotation of additional trap images captured under diverse environmental conditions could improve the robustness and generalization capability of the object detector.

Second, more recent object detection architectures could be evaluated and compared with YOLOv7 to determine whether further performance improvements can be achieved. Additional hyperparameter optimization strategies could also be investigated and evaluated within the proposed Vision-Language pipeline.

Third, the Vision-Language Assistant could be fine-tuned using larger domain-specific datasets. Additionally, evaluations involving growers and agricultural technicians could be conducted to assess the usefulness and clarity of the generated responses in real-world agricultural monitoring scenarios.

References

- [1] Leo Breiman. “Random Forests”. In: *Machine Learning*. Vol. 45. 1. 2001, pp. 5–32. DOI: [10.1023/A:1010933404324](https://doi.org/10.1023/A:1010933404324).
- [2] Martina Cardoni and Jesús Mercado-Blanco. “Confronting stresses affecting olive cultivation from the holobiont perspective”. In: *Frontiers in Plant Science* 14 (2023), p. 1261754. DOI: [10.3389/fpls.2023.1261754](https://doi.org/10.3389/fpls.2023.1261754).
- [3] Corinna Cortes and Vladimir Vapnik. “Support-Vector Networks”. In: *Machine Learning* 20.3 (1995), pp. 273–297. DOI: [10.1007/BF00994018](https://doi.org/10.1007/BF00994018).
- [4] Rafael C. Gonzalez and Richard E. Woods. *Digital Image Processing*. 4th ed. Pearson, 2018.
- [5] Kaiming He et al. “Deep Residual Learning for Image Recognition”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2016, pp. 770–778. DOI: [10.1109/CVPR.2016.90](https://doi.org/10.1109/CVPR.2016.90).
- [6] Kaiming He et al. “Masked Autoencoders Are Scalable Vision Learners”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2022, pp. 16000–16009.
- [7] Martin Heusel et al. “GANs Trained by a Two Time-Scale Update Rule Converge to a Local Nash Equilibrium”. In: *NeurIPS*. 2017.
- [8] Sepp Hochreiter and Jürgen Schmidhuber. “Long Short-Term Memory”. In: *Neural Computation* 9.8 (1997), pp. 1735–1780. DOI: [10.1162/neco.1997.9.8.1735](https://doi.org/10.1162/neco.1997.9.8.1735).
- [9] Edward J. Hu et al. “LoRA: Low-Rank Adaptation of Large Language Models”. In: *ICLR* (2022).
- [10] HumanSignal. *LabelImg: A graphical image annotation tool*. <https://github.com/HumanSignal/labelImg>. 2018.
- [11] Fernando Henrique Iost Filho et al. “How does the digital transformation of agriculture affect the implementation of Integrated Pest Management?” In: *Frontiers in Sustainable Food Systems* 6 (2022), p. 972213. DOI: [10.3389/fsufs.2022.972213](https://doi.org/10.3389/fsufs.2022.972213).
- [12] Nidhal Jegham et al. “Evaluating the Evolution of YOLO (You Only Look Once) Models: A Comprehensive Benchmark Study of YOLO11 and Its Predecessors”. In: *CoRR* abs/2411.00201 (2024). DOI: [10.48550/arXiv.2411.00201](https://doi.org/10.48550/arXiv.2411.00201). URL: <https://arxiv.org/abs/2411.00201>.
- [13] Kairui Jin et al. “IP-VQA Dataset: Empowering Precision Agriculture with Autonomous Insect Pest Management through Visual Question Answering”. In: *Companion Proceedings of the ACM Web Conference 2025 (WWW Companion '25)*. ACM, 2025, pp. 1999–2006. DOI: [10.1145/3701716.3718386](https://doi.org/10.1145/3701716.3718386).
- [14] Glenn Jocher et al. *YOLOv5*. <https://github.com/ultralytics/yolov5>. 2021.

- [15] Glenn Jocher, Ayush Chaurasia, and Jing Qiu. YOLOv8. <https://github.com/ultralytics/ultralytics>. 2023.
- [16] Romanos Kalamatianos et al. “DIRT: The Dacus Image Recognition Toolkit”. In: *Journal of Imaging* 4.11 (2018), p. 129. DOI: [10.3390/jimaging4110129](https://doi.org/10.3390/jimaging4110129).
- [17] Alex Krizhevsky, Ilya Sutskever, and Geoffrey E. Hinton. “ImageNet Classification with Deep Convolutional Neural Networks”. In: *Advances in Neural Information Processing Systems (NeurIPS)*. Vol. 25. 2012.
- [18] Zhenzhong Lan et al. “ALBERT: A Lite BERT for Self-supervised Learning of Language Representations”. In: *arXiv preprint arXiv:1909.11942* (2019).
- [19] Junnan Li et al. “BLIP-2: Bootstrapping Language-Image Pre-training with Frozen Image Encoders and Large Language Models”. In: *ICML 2023* (2023). arXiv: [2301.12597](https://arxiv.org/abs/2301.12597) [cs.CV].
- [20] Haotian Liu et al. “Visual Instruction Tuning”. In: *NeurIPS* (2024).
- [21] Jianping Liu et al. “Transfer large models to crop pest recognition: A cross-modal unified framework for parameter-efficient fine-tuning”. In: *Computers and Electronics in Agriculture* 237 (2025), p. 110661. DOI: [10.1016/j.compag.2025.110661](https://doi.org/10.1016/j.compag.2025.110661).
- [22] Jun Liu and Xuewei Wang. “A multimodal framework for pepper diseases and pests detection”. In: *Scientific Reports* 14 (2024), p. 28973. DOI: [10.1038/s41598-024-80675-w](https://doi.org/10.1038/s41598-024-80675-w).
- [23] Lei Liu and Xiaoling Song. “MPCD: A Multimodal Prior and Conditional Detection Framework for High-Fidelity Pest Image Generation”. In: *Computers and Electronics in Agriculture* 239 (2025), p. 111135. DOI: [10.1016/j.compag.2025.111135](https://doi.org/10.1016/j.compag.2025.111135).
- [24] Ze Liu et al. “Swin Transformer: Hierarchical Vision Transformer using Shifted Windows”. In: *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*. 2021, pp. 10012–10022. DOI: [10.1109/ICCV48922.2021.00986](https://doi.org/10.1109/ICCV48922.2021.00986).
- [25] Slawomir A. Lux, Andrea Sciarretta, and Nikos T. Papadopoulos. “The fallacy of the integrated pest management paradigm and the need for its OFF seasonal shift in the management of tropical fruit flies (Diptera: Tephritidae): The case of *Ceratitis capitata*”. In: *Current Research in Insect Science* 8 (2025), p. 100116. DOI: [10.1016/j.cris.2025.100116](https://doi.org/10.1016/j.cris.2025.100116).
- [26] Ricardo Malheiro et al. “A review of *Bactrocera oleae* (Rossi) impact in olive products: From the tree to the table”. In: *Trends in Food Science & Technology* 44 (2015), pp. 226–242. DOI: [10.1016/j.tifs.2015.04.009](https://doi.org/10.1016/j.tifs.2015.04.009).
- [27] Kevin Mallinger et al. “Breaking the barriers of technology adoption: Explainable AI for requirement analysis and technology design in smart farming”. In: *Smart Agricultural Technology* 9 (2024), p. 100658. DOI: [10.1016/j.atech.2024.100658](https://doi.org/10.1016/j.atech.2024.100658).
- [28] Nariman Mamdouh and Ahmed Khattab. “YOLO-based deep learning framework for olive fruit fly detection and counting”. In: *IEEE Access* 9 (2021), pp. 84252–84262.
- [29] José L Mira et al. “Benchmarking of computer vision methods for energy-efficient high-accuracy olive fly detection on edge devices”. In: *Multimedia Tools and Applications* 83.34 (2024), pp. 81785–81809.
- [30] Miguel Molina-Rotger et al. “Remote fruit fly detection using computer vision and machine learning-based electronic trap”. In: *Frontiers in Plant Science* 14 (2023), p. 1241576. DOI: [10.3389/fpls.2023.1241576](https://doi.org/10.3389/fpls.2023.1241576).

- [31] Yamna Ouguas, Mohamed El Oirdi, and Tariq Nacimi. “A comprehensive review on olive fruit fly *Bactrocera oleae* (Rossi, 1790)”. In: *African and Mediterranean Agricultural Journal - Al Awamia* 145 (2024), pp. 109–126.
- [32] Alec Radford et al. “Learning Transferable Visual Models From Natural Language Supervision”. In: *Proceedings of the International Conference on Machine Learning (ICML)*. 2021.
- [33] Vasileios P. Rekkas et al. “A Machine Learning Approach for the Identification of Olive Fruit Fly in Greece”. In: *Proceedings of the 9th South-East Europe Design Automation, Computer Engineering, Computer Networks and Social Media Conference (SEEDA-CECNSM)*. IEEE, 2024. DOI: [10.1109/SEEDA-CECNSM63478.2024.00019](https://doi.org/10.1109/SEEDA-CECNSM63478.2024.00019).
- [34] Mark Sandler et al. “MobileNetV2: Inverted Residuals and Linear Bottlenecks”. In: *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*. 2018, pp. 4510–4520.
- [35] seekFire. *Overview of model structure about YOLOv5*. <https://github.com/ultralytics/yolov5/issues/280>. GitHub Issue #280, accessed July 2025. 2020.
- [36] Karen Simonyan and Andrew Zisserman. “Very Deep Convolutional Networks for Large-Scale Image Recognition”. In: *arXiv preprint arXiv:1409.1556* (2014).
- [37] T. Stillitano et al. “Economic profitability assessment of Mediterranean olive growing systems”. In: *Bulgarian Journal of Agricultural Science* 22.4 (2016), pp. 517–526.
- [38] Margarida Victoriano, Lino Oliveira, and Helder P. Oliveira. “Automated Detection and Identification of Olive Fruit Fly Using YOLOv7 Algorithm”. In: *Pattern Recognition and Image Analysis*. Springer, 2023, pp. 211–222. DOI: [10.1007/978-3-031-36616-1_17](https://doi.org/10.1007/978-3-031-36616-1_17).
- [39] Hoai Nam Vu, Anh Duc Nguyen, and Dat Tran. “AgriDetectVL: Emphasizes the Agriculture-Focused Application Combined With Visual-Language Integration”. In: *IEEE Access* 13 (2025), pp. 195987–195995. DOI: [10.1109/ACCESS.2025.3633394](https://doi.org/10.1109/ACCESS.2025.3633394).
- [40] 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: *CVPR* (2023).
- [41] Yiqun Wang et al. “A large language model for multimodal identification of crop diseases and pests”. In: *Scientific Reports* 15 (2025), p. 21959. DOI: [10.1038/s41598-025-01908-0](https://doi.org/10.1038/s41598-025-01908-0).
- [42] Zhou Wang et al. “Image Quality Assessment: From Error Visibility to Structural Similarity”. In: *IEEE Transactions on Image Processing* 13.4 (2004), pp. 600–612. DOI: [10.1109/TIP.2003.819861](https://doi.org/10.1109/TIP.2003.819861).
- [43] Xiaoping Wu et al. “IP102: A Large-Scale Benchmark Dataset for Insect Pest Recognition”. In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*. 2019, pp. 8787–8796.
- [44] Maolin Zhang et al. “A Multimodal Large Language Model for Forestry in Pest and Disease Recognition”. In: *Research Square* (2023). Preprint. DOI: [10.21203/rs.3.rs-3444472/v1](https://doi.org/10.21203/rs.3.rs-3444472/v1).
- [45] Yinshuo Zhang, Lei Chen, and Yuan Yuan. “Few-shot agricultural pest recognition based on multimodal masked autoencoder”. In: *Crop Protection* 187 (2025), p. 106993. DOI: [10.1016/j.cropro.2024.106993](https://doi.org/10.1016/j.cropro.2024.106993).

- [46] Jiaqi Zhu, Hung-Teng Chang, and Tao Zhang. “Improved YOLOv7-Based Multi-Target Tracking and Detection under Maritime Surveillance Scenario”. In: *Research Square* (2024). Preprint. DOI: [10.21203/rs.3.rs-3823941/v1](https://doi.org/10.21203/rs.3.rs-3823941/v1). URL: <https://doi.org/10.21203/rs.3.rs-3823941/v1>.

Appendix A

Multilingual Interaction Capability

Although the experiments presented in this dissertation were conducted in English, the proposed Vision-Language Assistant is also capable of interacting with users in Portuguese.

The assistant is based on **LLaVA** 1.5, which inherits multilingual capabilities from its underlying **LMM**. As a result, users may submit prompts in Portuguese and receive responses in the same language without requiring any modification to the object detection pipeline.

This capability improves the accessibility of the proposed system for Portuguese-speaking users, particularly in agricultural scenarios where farmers and technicians may prefer interacting in their native language.