Investigating the Role of Learning Using Privileged Information in Object Detection

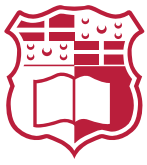
Matthias Bartolo

Supervisor: Dr. Dylan Seychell

Co-Supervisor: Dr. Konstantinos Makantasis

September 2025

Submitted in partial fulfilment of the requirements for the degree of Master of Science in ICT.



L-Università ta' Malta
Faculty of Information & Communication Technology

Abstract

Object detection is widely recognised as a foundational task within computer vision, with applications spanning automation, medical imaging, and surveillance. Although numerous models and methods have been developed, attaining high detection accuracy often requires the utilisation of complex model architectures, especially those based on transformers. These models typically demand extensive computational resources for inference and large-scale annotated datasets for training, both of which contribute to the overall difficulty of the task.

To address these challenges, this work introduces a novel methodology incorporating the Learning Using Privileged Information (LUPI) paradigm within the object detection domain. The proposed approach is compatible with any object detection architecture and operates by introducing privileged information to a teacher model during training. This information is then distilled into a student model, resulting in more robust learning and improved generalisation without increasing the number of model parameters and complexity.

The methodology is evaluated on general-purpose object detection tasks and a focused case study involving litter detection in visually complex, highly variable outdoor environments. These scenarios are especially challenging due to the target objects' small size and inconsistent appearance. Evaluation is conducted both within individual datasets and across multiple datasets to assess consistency and generalisation. A total of 120 models are trained, covering five well-established object detection architectures. Four datasets are used in the evaluation: three focused on Unmanned Aerial Vehicle (UAV)-based litter detection and one drawn from the Pascal VOC 2012 benchmark to assess performance in multi-label detection and generalisation.

Experimental results consistently demonstrate improvements in detection accuracy across all model types and dataset conditions when employing the LUPI framework. Notably, the approach yields increases of 0.02 to 0.15 in the strict mean Average Precision (mAP)@50–95 metric, highlighting its robustness across both general-purpose and domain-specific tasks. In nearly all cases, we observed performance improvements, indicating that the proposed methodology achieves such results without increasing the number of parameters or altering the model architecture. This supports its viability as a lightweight and effective modification to existing object detection systems.

Keywords: Learning Using Privileged Information, Object Detection, Knowledge Distillation, Computer Vision, Litter Detection

Matthias Bartolo was awarded the
MDIA Pathfinder Scholarship Fund to support his Master's studies.

Let this work stand not as a testament to the greatness of man, but to the greatness of God. Without His guidance and strength, this dissertation would not have taken the shape it holds today.

"Let your light so shine before men, that they may see your good works and glorify your Father in heaven." **– Matthew 5:16**

Acknowledgements

While this journey is presented as a single document encompassing a year's worth of work, to me it signifies something far greater. It marks a turning point–a period of profound personal transformation and growth in ways I had not anticipated. Though this dissertation bears the name of a single author, it was made possible through the invaluable support of many individuals. I dedicate these few pages to all those who enabled this transformation in me, and who supported me in discovering and nurturing my passion.

First and foremost, I would like to thank God. There were days when I felt alone, confused, and incapable of making further progress–when even writing a single line of code felt overwhelmingly difficult amid the mental and physical strains. Yet, during these moments, I experienced spiritual growth that gave me the courage to persevere. Even when the cognitive burden felt paralysing and clarity eluded me, I found inspiration and strength in the following verse:

"In the beginning was the Word, and the Word was with God, and the Word was God. He was with God in the beginning. Through him all things were made; without him nothing was made that has been made. In him was life, and that life was the light of all mankind. The light shines in the darkness, and the darkness has not overcome it." **– John 1:1–3**

Without His influence and steady guidance, this dissertation would not have taken the form it has today, nor would its direction and clarity have been fully realised.

Secondly, I wish to express my deepest gratitude to my supervisors, Dr. Dylan Seychell and Dr. Konstantinos Makantasis, for their guidance and support throughout this process. Beyond directing my work, they helped cultivate my understanding of scientific methodology and encouraged my growth as an independent researcher. I am sincerely thankful for the guidance they provided from their experience (privileged information), their thoughtful feedback, and their unwavering commitment through countless meetings and extensive correspondence.

I am also grateful to Mr. Gabriel Hili and Mr. Jonathan Attard for their technical support in configuring the hardware infrastructure provided by the Department of Artificial Intelligence at the University of Malta. Their assistance was instrumental in enabling the training of the models developed as part of this dissertation.

Finally, I extend heartfelt thanks to my family–my backbone, and the foundation of all I strive for. Their wisdom, encouragement, and constant support sustained me through

the most difficult moments. To my mother, who has supported me every step of the way and ensured I had the resources I needed to continue my studies, and to my father, who consistently reminds me to remain humble–I am eternally grateful. To my grandparents, both living and departed, and to my grandaunts and granduncles: thank you for instilling in me enduring values and a sense of simplicity in service to others. To my uncles, aunts, and cousins: thank you for your love, encouragement, and for always being there.

Just as the following words from Scripture reminded me of God's constant presence in times of trial, they also echo the steadfast love and support my family has shown me throughout this journey:

"But now, thus says the Lord, who created you, …, and He who formed you, …: 'Fear not, for I have redeemed you; I have called you by your name; You are Mine. When you pass through the waters, I will be with you; And through the rivers, they shall not overflow you. When you walk through the fire, you shall not be burned, Nor shall the flame scorch you.'" **– Isaiah 43:1–2**

"I was one way… and now I am completely different. And the thing that happened in between… was Him." **– Mary Magdalene, The Chosen**

Contents

List of Figures

List of Tables

List of Abbreviations

AdaBoost Adaptive Boosting.

Adam Adaptive Moment Estimation.

AGL Above Ground Level.

AI Artificial Intelligence.

AP Average Precision.

API Application Programming Interface.

AWIGS Aerial Waste Identification and Geolocation System.

BDW Bottle Detection in the Wild.

BRIEF Binary Robust Independent Elementary Features.

CAM Class Activation Mapping.

CNNs Convolutional Neural Networks.

COCO Common Objects in Context.

CUDA Compute Unified Device Architecture.

CV Computer Vision.

DCNs Deformable Convolutional Networks.

DETR DEtection TRansformer.

DINO DETR with Improved deNoising anchOr boxes.

DL Deep Learning.

DoG Difference of Gaussians.

DPM Deformable Part Model.

EXIF Exchangeable Image File Format.

FAST Features from Accelerated Segment Test.

FCOS Fully Convolutional One-Stage Object Detection.

FN False Negative.

FP False Positive.

FPN Feature Pyramid Network.

FPS Frames Per Second.

GFLOPS Giga Floating Point Operations Per Second.

GNSS Global Navigation Satellite System.

GPS Global Positioning System.

GPU Graphics Processing Unit.

GSD Ground Sampling Distance.

HOG Histogram of Oriented Gradients.

IoU Intersection over Union.

JKSE Joint Kernel Support Estimation.

KL Kullback–Leibler.

LUPI Learning Using Privileged Information.

mAP Mean Average Precision.

mAR Mean Average Recall.

ML Machine Learning.

NAS Neural Architecture Search.

NMS Non-Maximum Suppression.

OBBs Object-Oriented Bounding Boxes.

ORB Oriented FAST and Rotated BRIEF.

R-CNN Regional Based Convolutional Neural Network.

R-FCN Region-based Fully Convolutional Network.

RF Random Forests.

RGB-D Red-Green-Blue-Depth.

RL Reinforcement Learning.

RoI Region of Interest.

RPN Regional Proposal Network.

RPU Remote Processing Unit.

RRPN Rotation Region Proposal Network.

RT-DETR Real-Time DEtection TRansformer.

SAHI Slicing Aided Hyper Inference.

SDMP Sequential Decision Making Problem.

SGD Stochastic Gradient Descent.

SIFT Scale-Invariant Feature Transform.

SODA Small Objects from Different Altitudes.

SSD Single Shot MultiBox Detector.

SURF Speeded-Up Robust Features.

SVMs Support Vector Machines.

TACO Trash Annotations in Context.

TN True Negative.

TP True Positive.

TPU Tensor Processing Unit.

UAV Unmanned Aerial Vehicle.

VOC Visual Object Classes.

YOLO You Only Look Once.

Glossary of Symbols

α Weighting factor controlling the impact of privileged information on the student's training.

b Bounding box containing coordinates: $\{b_x, b_y, b_w, b_h\}$.

$c(x)$ Classification function, responsible for predicting the class labels for all detected objects in the image.

D Cosine distance loss measuring similarity between teacher and student latent representations.

$f(x)$ Object detection prediction function that outputs bounding box coordinates and class labels for all detected objects.

H Height of an image.

I Original input image.

I_{max} Maximum pixel intensity across the dataset for specific input channel.

I_{min} Minimum pixel intensity across the dataset for specific input channel.

l Predicted class label for a single bounding box.

L Finite set of class labels used in detection, specific to the problem being tackled.

$\mathcal{L}$ The number of layers in the neural network architecture, which applies to both the teacher and student models.

$L_{\mathcal{S}}$ Student loss incorporating both task-specific and distillation components.

N Number of predicted detections in a given image.

$\mathcal{N}$ Total number of input channels, including both RGB and any privileged information channels.

O Predicted set of bounding boxes and class labels for an image.

$r(x)$ Regression function, responsible for predicting the bounding box coordinates for all detected objects.

f_{student} Student network trained with standard input features and soft labels from the teacher.

f_{teacher} Teacher network trained with both standard input and privileged information.

$\mathcal{D}_{\text{train}}$ Training set consisting of input-output pairs, with privileged information available only during training.

W Width of an image.

x Input features (available at both training and testing stages).

x^* Privileged information (available only during training).

y Target output (ground truth labels).

Chapter 1 Introduction

"The real voyage of discovery consists not in seeking new landscapes, but in having new eyes."

– Marcel Proust

1.1 Introduction

Object detection is considered a foundational problem within the field of Computer Vision (CV), central to numerous applications ranging from medical analysis [1, 2] and autonomous systems [3, 4] to the monitoring of environmental degradation [5, 6], including the identification and classification of litter [7, 8]. In recent years, the field has seen a marked expansion in computational depth, with models capable of detecting increasingly complex visual patterns across several domains. Nevertheless, the issue of achieving consistently high detection accuracy persists. Many high-performing models are heavily dependent on intricate architectures [9, 10] or extensive labelled datasets [11], both of which introduce significant practical constraints. Deep models typically require prolonged training cycles and considerable computational power, while large-scale datasets necessitate laborious annotation procedures that are both costly and time-consuming [12]. In contexts where labelled data is scarce or prohibitively expensive, such as detecting environmental waste across irregular natural terrain or within densely populated urban settings, the limitations of current methods become especially pronounced [7, 13]. Furthermore, deploying such models in practical scenarios often demands rapid inference and reliable performance under limited computational resources–requirements that many conventional approaches are ill-equipped to satisfy [11].

Given these challenges, alternative learning strategies that can improve performance without imposing additional demands on model complexity or dataset scale have become increasingly relevant. One such strategy is Learning Using Privileged Information (LUPI), a training paradigm in which supplementary information is made available exclusively during the learning phase but not during inference [14]. The privileged data may take various forms: detailed texture maps, depth cues, high-resolution images, or domain-specific expert input [14–16]. Crucially, the model learns to internalise these richer signals during training, thereby producing a more refined feature representation that ultimately bolsters its predictive capabilities under normal test-time conditions. By

improving generalisation and accelerating convergence, LUPI provides an opportunity to compensate for sparse annotations or imbalanced datasets without increasing inference time or architectural depth.

This becomes especially significant in environmental monitoring contexts, where object detection is applied to dynamic, cluttered, and often unpredictable scenes [13, 17]. Automated litter detection, for instance, requires the localisation and classification of debris in varied lighting, terrain, altitude, and background conditions [7, 13]. Models must learn to distinguish waste from non-waste in a manner that is both accurate and efficient, particularly where rapid deployment and scalability are necessary. In such scenarios, the integration of privileged information may offer substantial improvements.

1.2 Motivation

Within the broader effort to improve object detection performance in challenging visual contexts, automated systems designed for the identification of environmental waste have become as a growing area of interest within applicable Artificial Intelligence (AI) technology. Although several datasets and models have emerged to address this task [7, 8, 18], consistently accurate detection remains elusive. Litter manifests in a wide range of shapes, sizes, materials, and colours. Often, it appears partially obscured or embedded within complex backgrounds. These conditions demand not only high spatial sensitivity but also strong contextual reasoning, both of which are difficult to achieve with conventional architectures trained on limited or imbalanced data.

Notably, an increasingly explored direction involves the use of aerial imagery, captured via unmanned aerial vehicles, to expand the scale and coverage of litter detection systems [13, 19, 20]. While promising in terms of spatial reach and operational efficiency, aerial viewpoints introduce their own set of challenges. As the altitude of image capture increases, litter would start to appear at much smaller scales, becoming less distinct and occupying only a few pixels in high-resolution images. This reduction transforms the detection problem into one of identifying small objects, which remains an ongoing challenge in computer vision due to low signal-to-noise ratios and ambiguous boundaries [21, 22].

In addition to these technical considerations, the practical deployment of drone-based detection systems is also governed by legal and operational frameworks. To gain a deeper understanding of the regulatory aspects surrounding drone usage, the author obtained both A1/A3 and A2 drone operating licenses through a certification process administered by Transport Malta.

Alongside the functional limitations faced by current detection systems lies a pressing concern regarding environmental sustainability [23]. Improvements in accuracy

are frequently achieved by expanding model depth or increasing computational complexity [9, 10], both of which carry significant energy demands. While such strategies may yield stronger performance on benchmark tasks, their long-term viability becomes questionable when deployed at scale, especially in scenarios where energy efficiency is a priority [24]. As such, there is a clear need for approaches that can maintain or improve predictive performance without exacerbating computational costs.

1.3 Problem Definition

In light of the identified problem, this study seeks to improve the accuracy of both general object detection and litter detection, without significantly increasing the computational costs associated with such systems. Recent advancements in the field often associate improved accuracy with increasingly complex architectures. However, such models typically incur higher energy consumption, which may undermine the environmental objectives of litter detection systems.

To address this, the dissertation explores the potential of the LUPI paradigm within object detection. This training framework introduces auxiliary data available during learning, which may guide the model more effectively without requiring any modification to its architecture or parameters during inference. The potential of LUPI resides in its ability to bolster decision-making by incorporating supplementary information during the training phase while maintaining an efficient and streamlined inference process.

Although LUPI has been applied in other contexts, its relevance to object detection has yet to be thoroughly examined. This work proposes its application not only in the context of general object detection but also in the specific context of litter detection from aerial imagery. The latter remains a complex and largely unresolved problem due to the visual ambiguity, occlusion, and scale variation of litter objects in natural environments.

Furthermore, to evaluate the proposed method, a set of experiments will be conducted across several object detection architectures. These will assess the performance of the proposed approach on established datasets that reflect varied and realistic environmental conditions. The ultimate aim is, first, to derive a rigorous methodology for integrating privileged information into object detection models; and second, to determine whether incorporating LUPI within object detection pipelines can lead to improved performance without necessitating the use of computationally expensive architectures that increase inference time.

1.4 Aims and Objectives

This study aims to explore the potential of integrating the LUPI paradigm with object detection models to improve the accuracy and efficiency of both general and litter detection in various environmental contexts. The primary goal is to develop models that could identify and classify objects in diverse settings while minimising computational costs. This study aims to achieve this by leveraging additional privileged information during the training phase to improve model robustness without increasing the complexity of the detection architecture. To meet these objectives, the research will focus on the following aims:

Objective (O1) Derive a rigorous methodology for integrating the LUPI paradigm into object detection models in the context of litter detection, where its use remains largely unexplored.

Objective (O2) Evaluate this methodology across a variety of renowned object detection architectures to determine its adaptability and performance.

Objective (O3) Test the proposed approach on widely recognised litter detection datasets, examining both within-dataset and cross-dataset evaluation.

Objective (O4) Assess the trade-off between detection accuracy and computational cost, and explore the method's broader applicability through testing on other detection datasets.

1.5 Main Contributions

Introduction of LUPI to Object Detection This dissertation demonstrates that integrating the LUPI paradigm into object detection, particularly for litter detection, enhances performance. This methodology, applied across five prominent object detection models, does so without altering model architecture or increasing inference time.

Improved Litter Detection and Localisation This research establishes that the application of LUPI significantly elevates litter detection accuracy, particularly in the detection of smaller objects. Notably, more substantial gains are observed in binary detection (object localisation) compared to multi-label detection, although advancements are evident in both areas.

Model-Agnostic Performance Improvement The proposed approach is shown to be model-agnostic, achieving strong performance without increasing model parameters or

inference time. Although training time increases because of the additional requirement to train the teacher model, computational efficiency during deployment remains unaffected.

Generalisation Across Litter Detection Datasets This dissertation also demonstrates that the proposed methodology generalises effectively both within the SODA dataset [13] and across other litter detection datasets, including BDW [17] and UAVVaste [19]. Extensive experimentation underscores the trained models' improved ability to detect small and partially occluded objects when applied to different contexts beyond those they were trained.

Generalisation Across Object Detection Datasets Finally, this dissertation highlights its contribution beyond litter detection, particularly in the broader context of object detection. The proposed methodology demonstrated improved multi-label detection performance when evaluated on the Pascal VOC 2012 dataset [25], which includes all 20 object classes. However, performance tends to decline as the number of classes increases.

1.6 Publications

During the writing of this dissertation, the key milestones of the study, including work conducted in the area of object detection, were documented in papers and published in internationally recognised, peer-reviewed conferences and journals:

1. M. Bugeja, M. Bartolo, M. Montebello, and D. Seychell, "MRTMD: A Multi-Resolution Dataset for Evaluating Object Detection in Traffic Monitoring Systems," *IEEE Access*, vol. 13, pp. 134460–134483, 2025. doi: 10.1109/ACCESS.2025.3585986. [26]

2. M. Bartolo, K. Makantasis, and D. Seychell, "Learning Using Privileged Information for Litter Detection," in *Proc. 2025 13th European Workshop on Visual Information Processing (EUVIP)*, 2025. [27]

1.7 Dissertation Overview

The dissertation is structured as follows:

Chapter 2 establishes the foundational concepts of object detection and examines key detection models. It provides an in-depth exploration of learning using privileged information, followed by a review of literature on litter detection approaches. The chapter concludes with a review of the application of learning

using privileged information to computer vision, as well as the performance metrics used for evaluation.

Chapter 3 details the study's methodology, including the problem definition, theoretical framework, and the proposed system architecture. Additionally, it discusses the implementation of privileged information, knowledge distillation, and the experimental setup, including model selection, data pre-processing techniques, and training parameters.

Chapter 4 evaluates the proposed system through a series of experiments on the SODA, BDW, UAVVaste, and Pascal VOC 2012 datasets, including a preliminary optimal tiling experiment. It compares the performance of various models across these experiments, discussing the findings and their implications, along with comparing their visual predictions and interpretability.

Chapter 5 concludes the dissertation by summarising the aims and objectives, highlighting potential applications of the study, and discussing its limitations and possible directions for further research.

1.8 Conclusion

This chapter introduced the motivation behind the study, offering a summary of the research undertaken and the defined problem. It then presented the aims and objectives of the dissertation, followed by an overview of the key contributions and publications, and a brief outline of the chapters that follow.

Chapter 2 Background and Literature Review

"We are like dwarfs sitting on the shoulders of giants."

– Bernard of Chartres

2.1 Introduction

This chapter opens by outlining the object detection problem and the primary difficulties it entails, such as the presence of cluttered backgrounds, scale variation, and the challenge of accurately detecting small objects. This section is then followed by an overview of prominent methods developed before the rise of deep learning, encompassing both traditional computer vision techniques and early machine learning models. This is followed by a section that explores deep learning-based approaches, including one-stage and two-stage detectors, transformer-based architectures, and other notable frameworks. The subsequent section presents an overview of the learning using privileged information paradigm, detailing its problem formulation and associated techniques. Following this, a review of prominent litter detection approaches is presented, encompassing both UAV and non-UAV methods, along with an overview of approaches leveraging the LUPI paradigm within the field of computer vision. The chapter concludes by introducing relevant evaluation metrics that establish the criteria for model performance assessment.

2.2 Object Detection

Object detection can be considered a pivotal problem within the field of computer vision, encompassing the task of identifying and locating objects within an image from a predefined set of categories [28]. At its core, the problem involves determining *what* objects are and *where* they are located in an image [29]. This can be further broken down into two interrelated tasks: object localisation, which focuses on pinpointing the position of objects within an image, and object classification, which identifies the type or category of each detected object [28, 30, 31].

2.2.1 Object Localisation

To determine the position and presence of an object within an image, a bounding box (a rectangular box) is used. Each object of interest is enclosed within such a box, drawn as closely as possible around its outline to minimise the surrounding background. It is within this context that, for every image, the task of object localisation requires outputting a set of these bounding boxes, each defined by four parameters: b_x, b_y, b_w, and b_h. The first two parameters, b_x and b_y, specify the coordinates of the bounding box's centre, while b_w and b_h represent its width and height (see Figure 2.1) [30].

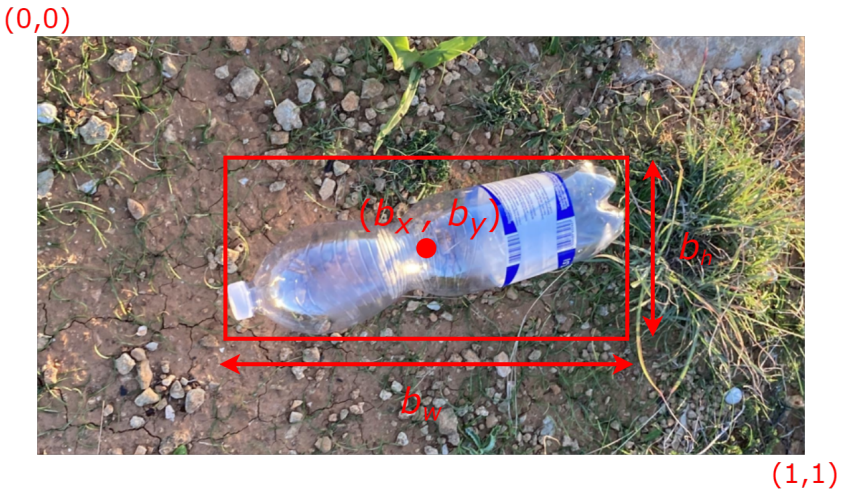


Figure 2.1 Visual representation of the bounding box coordinate system on a modified image taken from the SODA dataset. (Source: [13]).

2.2.2 Object Classification

While a bounding box indicates the location of an object, identifying the type of object detected is equally important. The result of the object classification task is usually presented as a label, which denotes the class or category assigned to the detected object from a predefined set of categories. Typically, alongside the label, the outputs from detection models also include a confidence score, indicating the degree of certainty that the object belongs to the assigned category [30]. Figure 2.2 provides a visual example of the classification and localisation tasks, illustrating multiple detections across different categories.

2.2.3 Challenges in Object Detection

Although the object detection problem can be divided into two subproblems and may initially seem straightforward, it is far from simple. Developing models capable of robustly detecting multiple objects across a variety of images with diverse backgrounds is

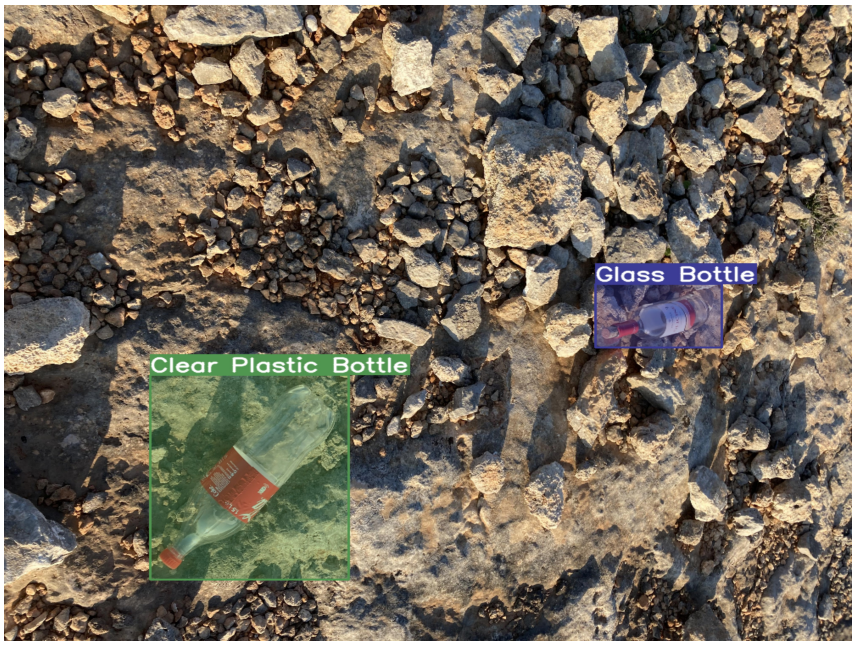


Figure 2.2 Visual representation of the classification and localisation tasks, showing multiple detections across various categories, based on an image from the SODA dataset. (Source: [13]).

a highly complex task. Furthermore, it is crucial to consider the demands of computational efficiency and real-time performance, as these factors are integral to the practical application of detection models. As highlighted by [11], the object detection problem encompasses several key challenges:

1. **Complex Backgrounds and Interferences**: In real-world situations, especially in outdoor environments, backgrounds tend to be complex and cluttered [11, 30]. Natural settings introduce factors such as weather conditions, lighting changes, shadows, and occlusions, which blur the distinction between objects and their surroundings, complicating the detection process [29]. Furthermore, the wide range of object categories and shapes adds another layer of difficulty, as different detectors may categorise the same object in varying ways. In dynamic environments, objects within the same category can appear in different poses, necessitating that object detectors generalise effectively [11].

2. **Scale Variability**: Objects in real-world images often appear at varying scales. For instance, in aerial imagery, litter may appear much smaller compared to close-up photographs. Additionally, the size disparity between different types of litter, such as a glass bottle and a cotton bud, further complicates detection. Designing an algorithm that can effectively manage multi-scale variations and generate multi-scale feature representations remains a significant challenge. To this end, such tasks typically require techniques like multi-scale input processing for robust detection [11, 29].

3. **Object Occlusion**: Object occlusion presents another common challenge in detection tasks and can be classified into *partial occlusion* and *complete occlusion* [11, 32, 33]. Complete occlusion is particularly problematic, as it prevents the detector from extracting sufficient feature information, thus diminishing detection accuracy [11]. In contrast, partial occlusion, while less severe, still results in parts of the object being obscured by noise, which leads to a loss of important feature information and a weaker feature representation.

4. **Class Imbalance and Dataset Bias**: In supervised learning, object detection models rely on labelled datasets, where annotators identify object locations using bounding boxes. These labelled images are then employed to train the detection algorithm. However, datasets often suffer from class imbalances, with certain object categories being over-represented while others are under-represented. This imbalance can impair model performance, particularly when detecting less frequent objects [11]. As noted by [34], class imbalance manifests in two forms: *foreground-background imbalance*, where background pixels dominate over object pixels, and *foreground-foreground imbalance*, where some object categories are more prevalent than others. Both types of imbalance can hinder the model's ability to accurately detect objects, especially in complex scenes [11, 34].

5. **Small Object Detection**: Small objects occupy only a small number of pixels within an image, which makes it difficult for models to extract meaningful features. Accurate localisation is essential, as even minor deviations in bounding box predictions can result in detection failures [11]. Furthermore, the limited feature representation of small objects, coupled with a lack of sufficient contextual information and an inadequate number of positive examples (*foreground-background imbalance*), exacerbates the challenge of detecting such objects [21, 29].

2.2.4 Object Detection Before the Rise of Deep Learning

In efforts to address the object detection problem and its associated challenges, numerous approaches were developed using traditional and machine learning methods, preceding the widespread adoption of deep learning techniques.

2.2.4.1 Traditional Methods

Early methods for object detection were heavily reliant on creative heuristics and the efficiency of basic computational techniques. Simple strategies, such as background subtraction [35], sought to identify objects by detecting fluctuations in pixel intensity across an image. As research progressed, more sophisticated feature-driven approaches

emerged. One notable example was the development of Haar cascade classifiers [36, 37], designed particularly for face detection. These classifiers exploited Haar-like features, which captured essential edge and line structures within images. Another prominent method introduced at the time was the Histogram of Oriented Gradients (HOG) descriptor [38, 39], which was widely applied to tasks such as pedestrian detection. HOG focused on the distribution of local gradient directions, capturing the shape and appearance of objects through structured edge information.

Alongside these developments, several feature extraction techniques became foundational. The Scale-Invariant Feature Transform (SIFT) [40] algorithm identifies distinctive keypoints within an image by detecting local extrema in a scale-space constructed from Difference of Gaussians (DoG), allowing for reliable feature matching under variations in scale and rotation. The Speeded-Up Robust Features (SURF) algorithm [41] built upon these ideas, offering a faster alternative while maintaining reliable performance under transformations such as viewpoint and illumination changes. Later, the Oriented FAST and Rotated BRIEF (ORB) algorithm [42] emerged, combining the efficiency of the Features from Accelerated Segment Test (FAST) keypoint detector with the compactness of the Binary Robust Independent Elementary Features (BRIEF) descriptor. ORB provided a computationally inexpensive, rotation-invariant solution, serving as a competitive alternative to SIFT and SURF in environments requiring faster performance.

2.2.4.2 Machine Learning Approaches

In tandem with traditional methods, notable advances in object detection were made through the adoption of Machine Learning (ML), a branch of AI focused on enabling systems to learn patterns from data [43]. During this period, Support Vector Machines (SVMs) [44] emerged as a pivotal tool, providing a strong classification framework, specifically when combined with feature descriptors such as HOG [45]. Other machine learning strategies also contributed meaningfully to the field's development. Boosting algorithms, such as Adaptive Boosting (AdaBoost), proved instrumental in the Viola-Jones face detection framework [46], while ensemble methods, including Random Forests (RF) [47] and techniques utilising Red-Green-Blue-Depth (RGB-D) information [48], further expanded detection capabilities. Although these approaches improved performance by allowing models to learn directly from data, they continued to rely heavily on manually engineered features. Additional progress was marked by the introduction of Deformable Part Models (DPM) [49], which conceptualised objects as collections of interconnected parts, thus enabling more effective modelling of variations in object appearance. Collectively, these machine learning-based methods significantly advanced object detection, laying the essential foundations for the subsequent development of deep learning ap-

proaches.

2.2.5 Object Detection in the Era of Deep Learning

The introduction of Deep Learning (DL), most notably Convolutional Neural Networks (CNNs), brought about a fundamental shift in the field of object detection [50]. In contrast to earlier methods that depended heavily on labour-intensive, hand-crafted feature engineering, CNNs are capable of automatically extracting features from large datasets, thereby streamlining the detection process and improving overall efficiency [51–53]. This progression led to the development of a new generation of object detectors that demonstrated greater accuracy and reliability than their traditional counterparts. Consequently, numerous object detection architectures have been introduced, each offering distinct advantages and facing particular limitations.

CNN-based object detectors are generally categorised into two principal types: one-stage detectors and two-stage detectors [54]. More recently, transformer-based models, originally proposed for natural language processing tasks [55], have been successfully adapted for computer vision applications, including object detection [9, 10], pushing performance boundaries even further. Beyond these mainstream approaches, emerging deep learning models increasingly incorporate visual attention mechanisms to strengthen feature representation and contextual reasoning within images [56, 57]. Moreover, alternative paradigms, such as Reinforcement Learning (RL), have also been explored to dynamically optimise object detection strategies [58, 59].

2.2.5.1 One Stage Detectors

One-stage detectors address the problem of object detection by concurrently performing localisation and classification within a single network, predicting bounding boxes and corresponding labels in a single forward pass. OverFeat [60], introduced in 2013, marked an early application of deep learning to object detection. It employed a multi-scale sliding window method wherein a classifier generated class labels and confidence scores at each spatial location. Prediction quality was subsequently improved through adjustments in resolution and the merging of bounding boxes.

The You Only Look Once (YOLO) series significantly reshaped one-stage detection by reframing object detection as a regression problem. The original YOLO model [61], released in 2016, partitioned images into an $S \times S$ grid, with each cell responsible for predicting bounding boxes, confidence scores, and class probabilities (refer to Figure 2.3). Later versions introduced several refinements: YOLOv2 [62] integrated batch normalisation and optimised anchor boxes; YOLOv4 [63] adopted a suite of training techniques commonly referred to as bag-of-freebies [64]; YOLOv5 [65] incorporated au-

tomatic anchor learning; and YOLOX [66] shifted towards anchor-free methodologies. More recently, YOLOv12 [67] introduced attention-driven designs, leveraging linear attention mechanisms and residual-efficient layer aggregation networks.

Further adaptations have continued to emerge. For example, YOLO-NAS [68] improved detection efficiency by employing Neural Architecture Search (NAS) within a quantisation-friendly framework. Meanwhile, YOLO-World [69] extended the model's capabilities to open-vocabulary detection, integrating vision-language path aggregation and region-text contrastive loss, achieving notable performance boosts in both speed and accuracy over contemporary models.

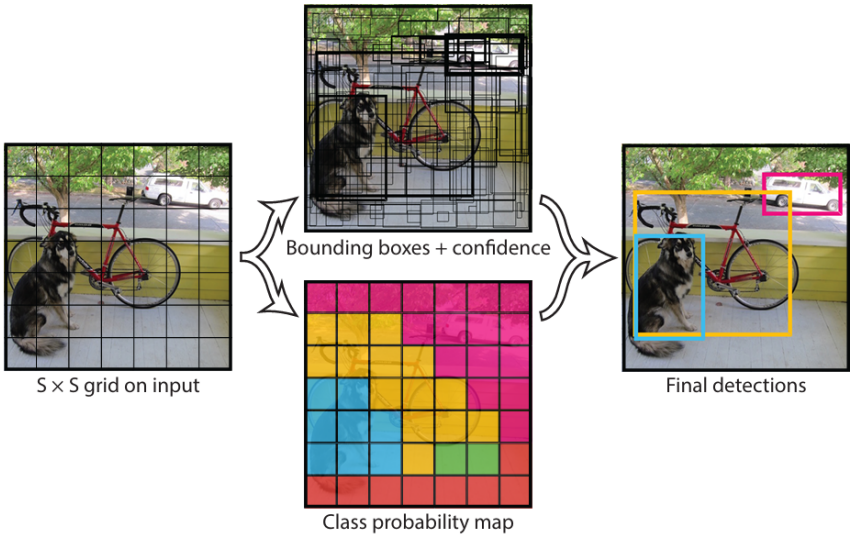


Figure 2.3 YOLOv1 pipeline. (Source: [61])

In parallel with developments in YOLO, other notable one-stage detectors have emerged. Among them, the Single Shot MultiBox Detector (SSD) [70], released in 2016, introduced an architecture that bypasses region proposal networks. Instead, it predicts object categories and bounding box refinements directly over a fixed set of predefined anchor boxes, known as default boxes. SSD builds on a modified VGG-16 backbone, where the fully connected layers are removed and replaced with additional convolutional layers. These progressively reduce spatial resolution, resulting in multiple feature maps of decreasing size. Each feature map is responsible for detecting objects at a specific scale, supporting effective multi-scale detection.

To accommodate objects of varying sizes, SSD deploys default boxes with multiple aspect ratios and scales across each feature map. At every level, lightweight convolutional heads independently predict class labels and refine bounding box coordinates. These operations are performed in parallel across all scales, enabling the model to classify and localise objects within a single forward pass simultaneously. This design achieves an effective compromise between computational efficiency and detection ac-

curacy. The complete architecture is illustrated in Figure 2.4.

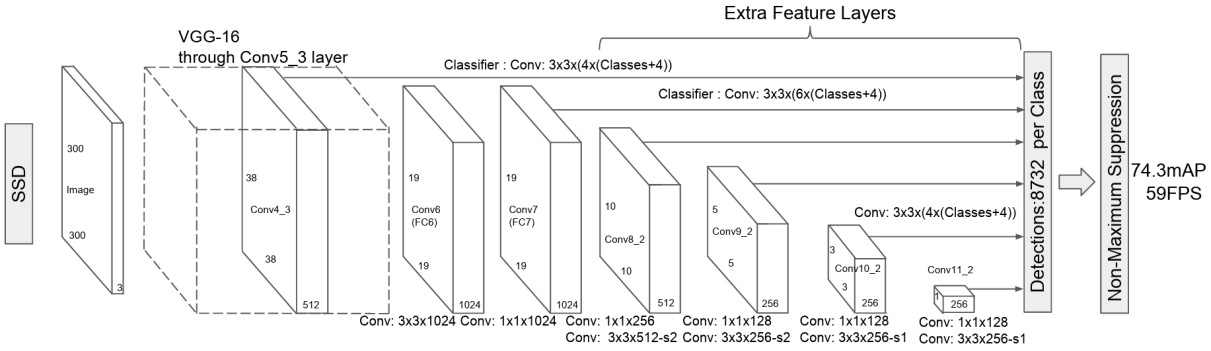


Figure 2.4 SSD architecture. (Source: [70])

Other prominent one-stage detectors include RetinaNet [71], introduced in 2017 to mitigate the foreground–background class imbalance that commonly undermines the performance of dense object detectors. Unlike two-stage methods such as Faster R-CNN, RetinaNet performs classification and localisation in a single forward pass (see Figure 2.5). It employs a ResNet backbone combined with a FPN to extract multi-scale features, followed by two parallel subnetworks: one for predicting object classes and another for refining bounding box coordinates. A key innovation is the use of focal loss, which shifts training focus toward difficult, misclassified examples by down-weighting easy negatives. To detect objects of different shapes and sizes, RetinaNet utilises anchor boxes with varying scales and aspect ratios across feature levels. This design allows the model to achieve accuracy comparable to that of many two-stage detectors, while maintaining the efficiency advantages of a single-stage framework.

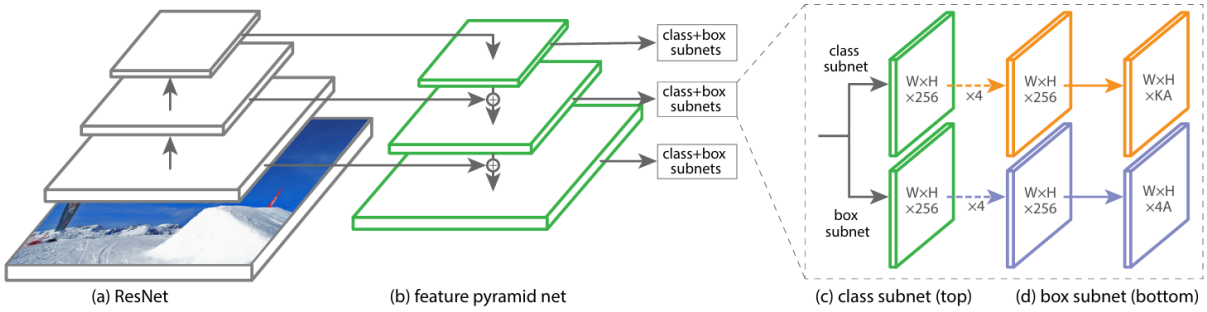


Figure 2.5 RetinaNet architecture. (Source: [71])

Building on the foundation of SSD, SSDLite was introduced in 2018 as a lighter variant designed for mobile and resource-constrained devices [72]. It replaces the original SSD's VGG-16 backbone with MobileNetv2, which employs inverted residual layers and linear bottlenecks to balance efficient feature extraction with model capacity. This change significantly reduces computational demand. Furthermore, standard convolutional layers in the detection heads are substituted with depthwise separable convolutions (see Figure 2.6), which decrease parameter count and speed up inference. Later

iterations incorporated MobileNetv3 as the backbone, adding squeeze-and-excitation modules and the h-swish activation function for improved efficiency and accuracy (see Figure 2.7). These adaptations make SSDLite highly suitable for deployment in environments with limited computational resources.

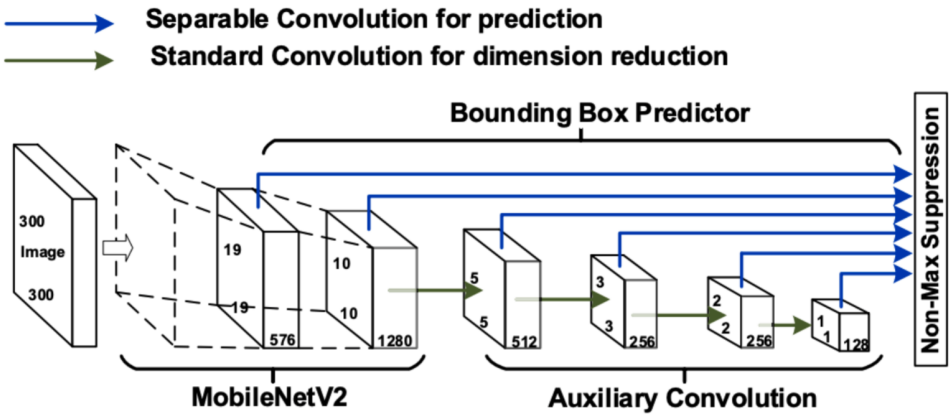


Figure 2.6 MobileNetV2-SSDLite architecture. (Source: [73])

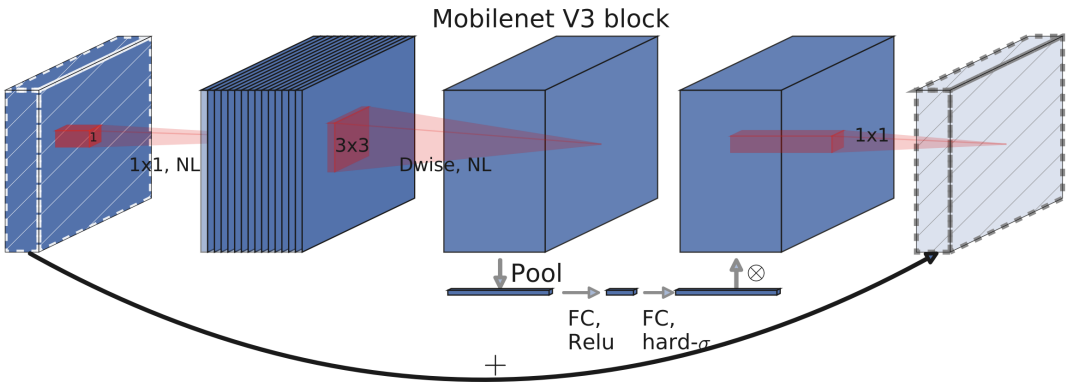


Figure 2.7 MobileNetv3 architecture. (Source: [74])

In contrast, FCOS [75], released in 2019, departs from traditional anchor box approaches entirely. It adopts a fully convolutional architecture that directly predicts object centre points and bounding box coordinates, simplifying the detection pipeline. One distinctive feature is the addition of a centre-ness branch, which estimates how close a predicted box is to the actual object centre, helping suppress low-quality detections and reducing false positives (see Figure 2.8). Built on a ResNet-FPN backbone, FCOS leverages multi-scale feature extraction to improve detection across object sizes. Its output head consists of three branches: regression for bounding boxes, classification for object categories, and centre-ness for quality estimation, enabling competitive performance without complex anchor matching.

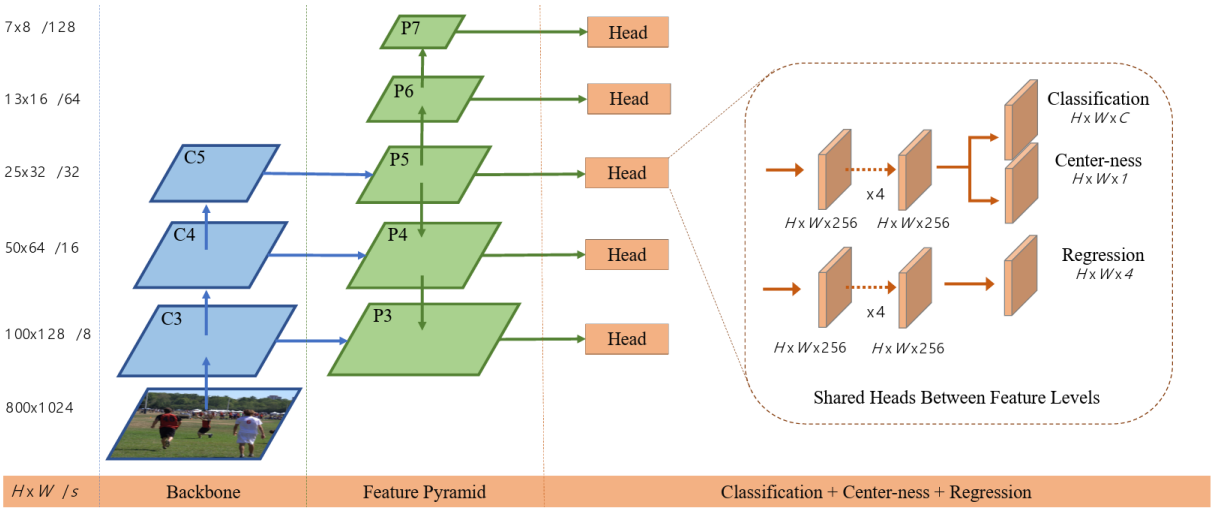


Figure 2.8 FCOS architecture. (Source: [75])

2.2.5.2 Two Stage Detectors

In contrast to one-stage detectors, two-stage detectors separate region proposal from classification and localisation. The R-CNN family was instrumental in this approach, beginning with the original R-CNN model [76]. This model used a selective search to generate candidate regions, which were then passed individually through AlexNet [77] to extract features. Each region was classified and refined through fully connected layers. Although effective, this process was slow due to repeated feature extraction for each region (see Figure 2.9). Fast R-CNN [78] improved efficiency by computing a convolutional feature map over the entire image once, and then applying region proposals on this shared feature map. This reduced redundant computations while maintaining detection accuracy.

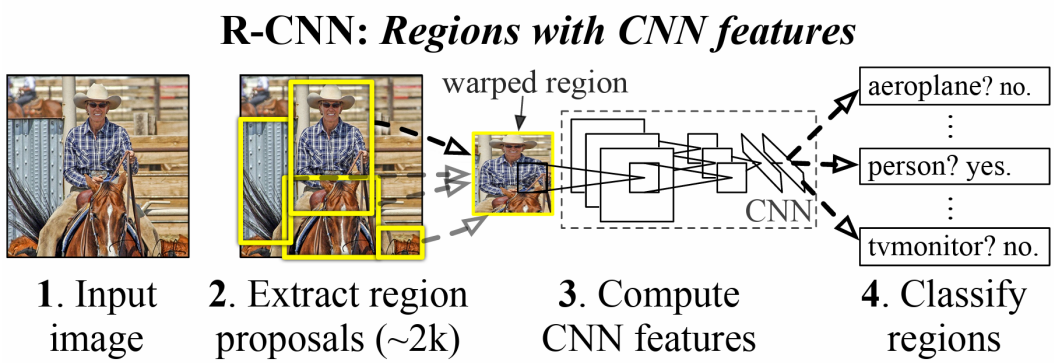


Figure 2.9 R-CNN architecture. (Source: [76])

A more pivotal advancement came with Faster R-CNN [79], which introduced the Regional Proposal Network (RPN). This component removed the dependency on external region proposal algorithms by allowing the network to learn object proposals

directly from feature maps using anchor boxes of varying scales and aspect ratios. The RPN shares convolutional features with the detection network, enabling efficient and integrated region proposal generation. After region proposals are obtained, a shared head performs both classification and bounding box regression, streamlining the process within a unified framework.

Subsequent improvements to Faster R-CNN incorporated an FPN, significantly enhancing the model's ability to detect objects at multiple scales by enriching feature representations across layers. This advancement became standard in later implementations, exemplified by the ResNet-FPN backbone architecture. Here, a shared network efficiently performs both region proposal and detection tasks. After extracting region proposals, a unified head simultaneously executes classification and bounding box regression, preserving architectural simplicity while boosting efficiency. Building upon this foundation, Mask R-CNN [80] added a parallel branch to predict segmentation masks for each Region of Interest (RoI), enabling instance segmentation alongside detection.

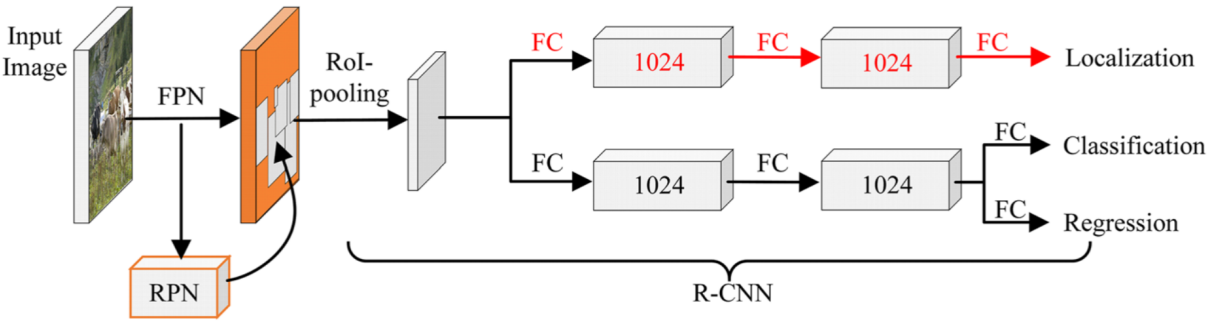


Figure 2.10 Faster R-CNN architecture with ResNet-50 backbone and integrated FPN, demonstrating a unified two-stage detection pipeline for proposal generation, classification, and bounding box regression. (Source: [81])

Complementing these technological advancements, FPN [82] improved detection across all network layers by constructing a feature pyramid for improved multi-scale detection. This pyramid included bottom-up pathways that captured semantic information and top-down pathways that refined these maps by merging high-level features with spatially rich information. The Region-based Fully Convolutional Network (R-FCN) [83] integrated both classification and localisation tasks, utilising position-sensitive score maps to classify and refine bounding box coordinates. This approach struck a balance between speed and accuracy by enabling shared computation across regions. Building on these two-stage frameworks, EfficientDet [84] incorporated an EfficientNet [85] backbone and employed a weighted bi-directional FPN for efficient multi-scale feature fusion (see Figure 2.11). Furthermore, it introduced a compound scaling method that uniformly scaled resolution, depth, and width across the backbone, feature network, and prediction networks.

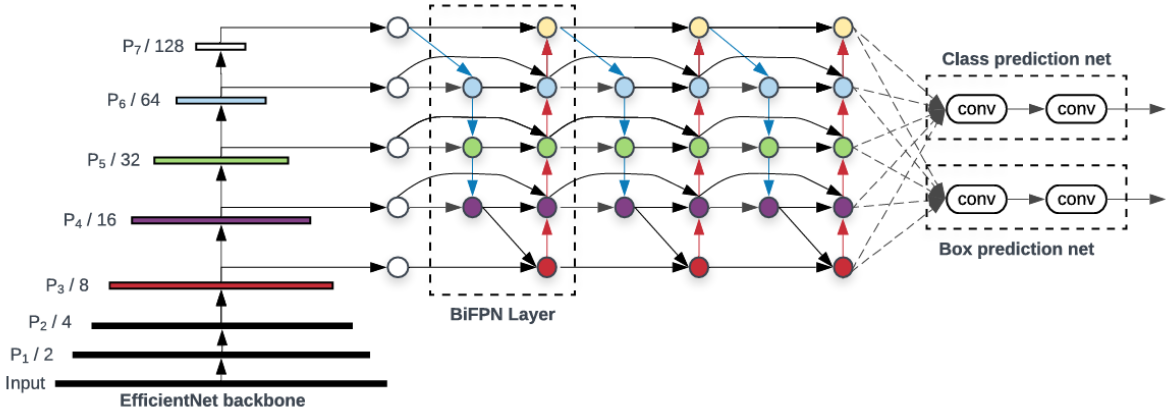


Figure 2.11 EfficientDet architecture. (Source: [84])

2.2.5.3 Transformer-Based Detectors

Transformers, originally developed for natural language processing [55], have garnered significant attention in computer vision [86], especially for object detection tasks. The DEtection TRansformer (DETR) [9], introduced in 2020, marked a pioneering shift in transformer-based object detection by framing the task as a set prediction problem. DETR removed the need for region proposals, employing a bipartite matching loss that aligns predicted boxes with ground-truth boxes. Its encoder-decoder architecture processes input data through self-attention mechanisms, enabling it to distinguish between individual instances successfully. This architecture supports both object detection and panoptic segmentation tasks, and the architecture for DETR is illustrated in Figure 2.12.

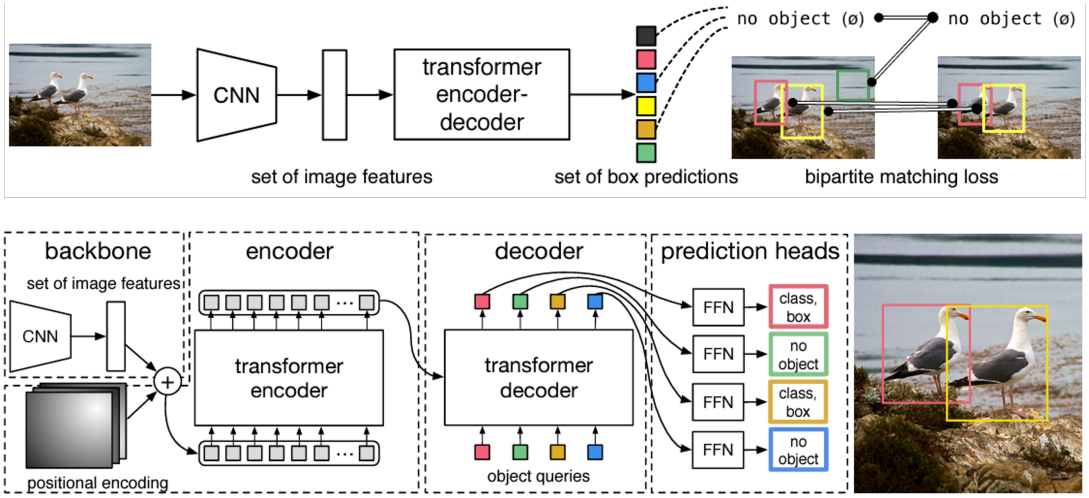


Figure 2.12 DETR architecture. (Source: [9])

Several models have built upon the foundation laid by DETR. The DETR with Improved deNoising anchOr boxes (DINO) framework [87–89] improved both training efficiency and overall performance. Grounding DINO [90] advanced zero-shot object

detection by enabling the identification of objects without prior training on specific categories. This was achieved through the use of natural language queries and the multimodal fusion of textual and visual information. Real-time transformer-based detection has also emerged as a significant advancement. The Real-Time DEtection TRansformer (RT-DETR) [10] addressed the limitations of non-maximum suppression by incorporating a hybrid encoder and high-quality initial queries. This made the model more efficient and adaptable, eliminating the need for retraining. RT-DETRv2 [91] further optimised training strategies and integrated bag-of-freebies techniques, enhancing real-time performance.

Vision-language models have also adopted transformer architectures for object detection. PaliGemma [92] combined a vision transformer for image encoding with a transformer decoder to merge textual and visual data, framing object detection as a set prediction task. Its successor, PaliGemma 2 [93], improved efficiency by incorporating Gemma 2 language models with the SigLIP vision encoder, enabling it to handle multiple input resolutions more effectively. Similarly, Florence-2 [94] utilised a unified, prompt-based architecture with an image encoder and a multi-modality encoder-decoder to address a range of vision-language tasks, including object detection, captioning, and segmentation (refer to Figure 2.13). These transformer-based approaches collectively represent a paradigm shift in object detection, offering end-to-end solutions that eliminate traditional components, such as non-maximum suppression, while enabling more flexible, multimodal, and zero-shot capabilities.

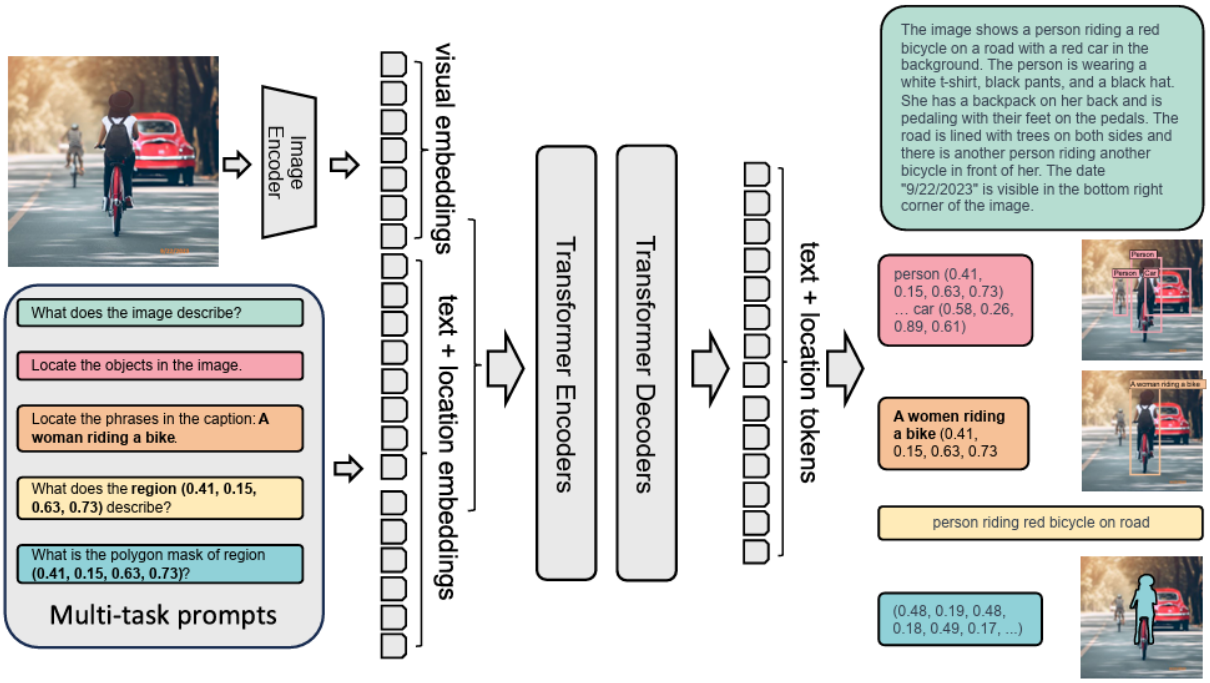


Figure 2.13 Florence-2 architecture. (Source: [94])

2.2.5.4 Other Deep Learning Approaches

Deep learning methodologies that go beyond traditional one-stage, two-stage, and transformer detectors have emerged to tackle specific challenges in object detection. These approaches can be categorised based on their underlying principles and technical innovations. One prominent paradigm is reinforcement learning-based detection frameworks, starting with the approach by Caicedo et al. [59], which used class-specific Deep Q-Network agents to iteratively refine object localisation by framing detection as a Sequential Decision Making Problem (SDMP). Subsequent studies have expanded on this foundation by incorporating multitask learning [95], integrating saliency ranking [58], and developing other complementary strategies [96, 97] to enhance the reinforcement learning paradigm for object detection. An example architecture of such a reinforcement learning-based framework for iterative object localisation is depicted in Figure 2.14.

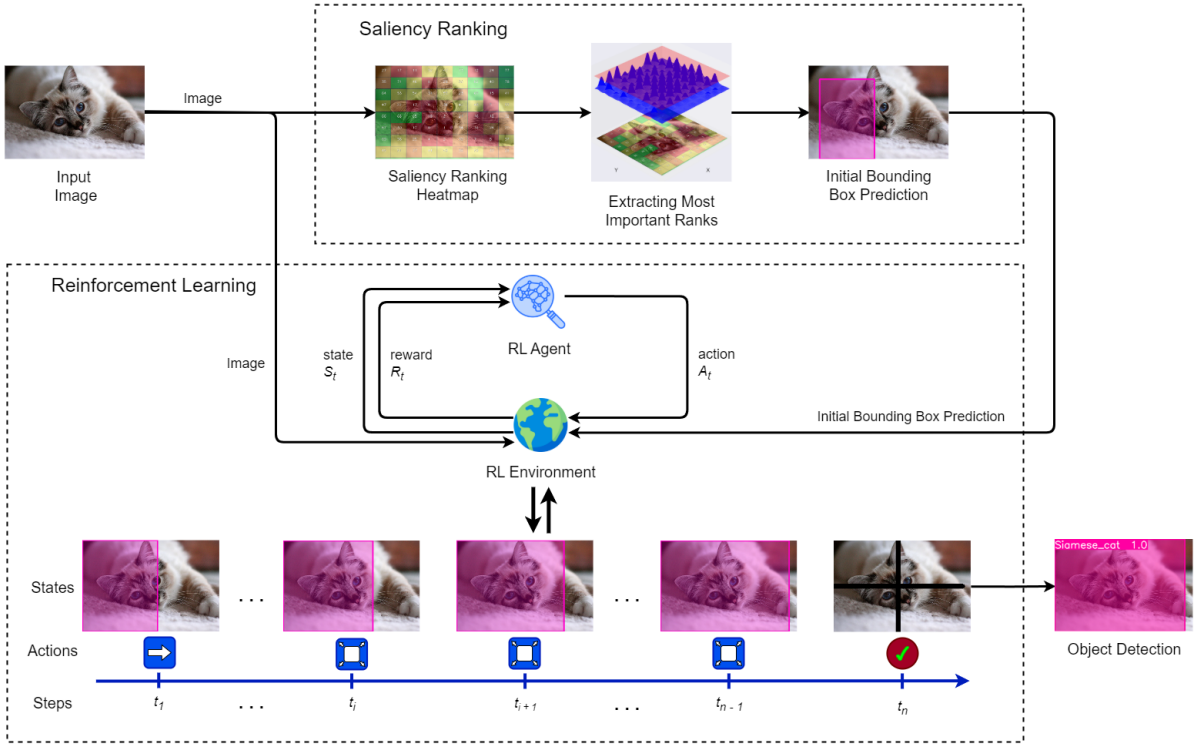


Figure 2.14 Architecture of a reinforcement learning-based object detection framework for iterative object localisation. (Source: [58])

Feature enhancement techniques represent another significant area, exemplified by Deformable Convolutional Networks (DCNs) introduced in 2017 [98]. DCNs improves feature extraction across various detector architectures by incorporating deformable convolutional layers and RoI pooling, enabling adaptive sampling of input features with learnt offsets. This approach better models spatial transformations and bolsters the detection of objects with varying shapes and sizes. Point-based and slicing-based approaches offer alternative detection paradigms. CenterNet [99], models ob-

jects as single points (the centres of their bounding boxes) and uses keypoint estimation to identify these centres while regressing to other properties such as size and orientation. This results in an end-to-end differentiable system that is simpler, faster, and often more accurate than traditional bounding box-based detectors. The architecture of CenterNet is illustrated in Figure 2.15. Furthermore, the Slicing Aided Hyper Inference (SAHI) framework [100] improves small object detection by dividing input images into overlapping patches during both fine-tuning and inference. This increases pixel coverage for small objects, which are then reassembled through non-maximum suppression. These diverse approaches complement mainstream detection architectures by addressing specific limitations and introducing novel perspectives on the object detection problem.

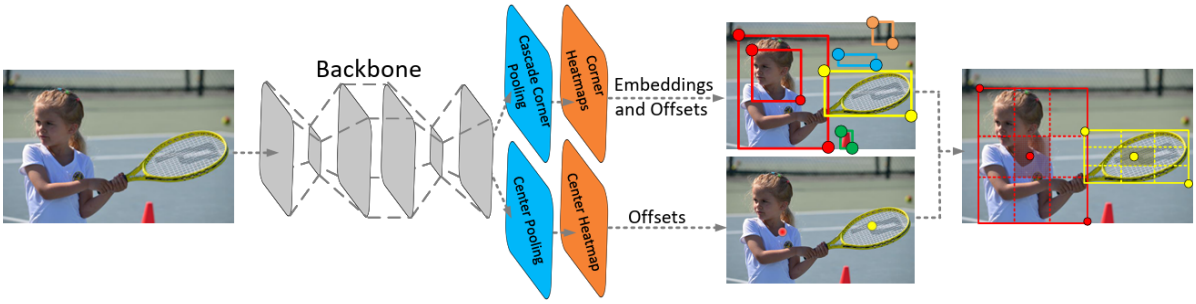


Figure 2.15 CenterNet architecture. (Source: [99])

2.3 Learning Using Privileged Information

Unlike the conventional machine learning approach, which selects the most suitable function from a predefined set based solely on input-output training examples, the LUPI paradigm introduces a richer structure. Proposed by Vapnik and Vashist [14, 101], LUPI introduces an inherent asymmetry between training and inference: additional informative inputs, referred to as privileged information, are available during training but not at test time [15, 102]. This distinction aims to accelerate the learning process by guiding the model with supplementary insights that cannot be exploited during deployment.

LUPI is inspired by human learning, reflecting the idea that a single day with a great teacher can be more valuable than a thousand days of independent study [14, 103]. Following the intuition that a student gains more than just examples, the teacher provides explanations, comparisons, and context, which enhance understanding [14, 103, 104]. In much the same way, LUPI incorporates additional information, known as privileged information (x^*), during training. This data, which is not available during testing, offers a deeper understanding beyond the standard input-output (x, y) pairs. As outlined in [14], the standard supervised learning framework is formally expressed by a training

set composed of input-output pairs:

$$(x_1, y_1), \ldots, (x_l, y_l), \quad x_i \in X, y_i \in Y. \tag{2.1}$$

Assuming that the data pairs $(x,\ y)$ are generated according to a fixed but unknown probability distribution $P(x, y)$. The task is to select the function $\hat{y} = f(x; \theta^*)$, with parameter $\theta^* \in \Theta$, from a specified class of functions which minimises the probability of misclassification. In this setting, the input vector $x_i \in X$ describes an instance, and y_i denotes its corresponding target label. The predicted output $\hat{y}$ is the model's estimate of the true label y given the input x. In comparison, within the LUPI framework, training data can be represented as a sequence of triplets:

$$(x_1, x_1^*, y_1), \ldots, (x_l, x_l^*, y_l), \quad x_i \in X, x_i^* \in X^*, y_i \in Y. \tag{2.2}$$

These samples are assumed to be drawn from an unknown but fixed probability distribution $P(x, x^*, y)$. The task is to select, from a specified class of functions $f(x, \theta)$, where $\theta \in \Theta$, the function $y = f(x, \theta^*)$ that minimises the probability of classification errors. Although the aim remains the same as in the conventional supervised learning setting, the LUPI paradigm introduces an additional element during training. Rather than learning from pairs (x, y), the model receives triplets (x, x^*, y), where $x^* \in X^*$ represents privileged information. This privileged component belongs to a separate space X^*, which need not coincide with the original input space X.

In addition, the LUPI framework can be extended to align with the concept of knowledge distillation [105], as proposed in [103]. Both approaches can be viewed as instances of a broader idea in which one model provides guidance to another, a process referred to as *generalised distillation*. This unified perspective treats Hinton's distillation [105] and Vapnik's use of privileged information [14] as complementary forms of machine-driven instruction. In this framework, a teacher model is first trained on standard input-output $(x^*,\ y)$ pairs. The teacher then produces soft labels, which act as enriched supervisory signals. These soft labels are used in Hinton's distillation method to train a student model. The student learns from the soft labels generated by the teacher, benefiting from the additional information embedded in the teacher's predictions, even though it only receives standard input-output $(x,\ y)$ pairs [103].

Within practical implementations, LUPI in deep learning is generally realised through a teacher–student structure embedded in conventional training pipelines. The teacher network is trained using both regular inputs and privileged data. Its outputs, such as soft predictions or intermediate feature representations, are used to supervise the student through additional loss functions [104]. Kullback–Leibler (KL) divergence is often used for classification tasks, while cosine similarity is more suitable when aligning continuous

outputs or internal vector representations [102, 105]. The architectures of the teacher and student are usually similar, though the teacher is adapted to accept the extra input during training [102, 103]. This setup integrates into existing deep learning workflows with minimal modification. Various strategies have been investigated for distilling knowledge, such as matching output logits [106], aligning attention maps [107], transferring hidden features [102, 108], and combining multiple forms of guidance to strengthen the student's learning process [103, 109]. The application of LUPI within computer vision is examined in greater detail in Section 2.5.

2.4 Review of Litter Detection Methodologies

Within the broader context of object detection, the task of identifying litter poses particular challenges, especially when employing an UAV to capture expansive outdoor scenes. The difficulty lies in the nature of the environments: litter often appears amid textured and irregular backgrounds such as rocky outcrops or dense vegetation, where visual contrast is minimal [7, 18, 110]. Despite these difficulties, several studies have addressed this problem by creating a dataset or proposing varied and increasingly refined approaches. In what follows, the review concentrates on the most prominent and widely cited works, offering a representative view of the prevailing methodologies adopted in recent research.

2.4.1 Bottle Detection in the Wild Using Low-Altitude UAVs

To address the challenge of bottle detection, the BDW dataset, introduced in 2018, was developed to identify plastic bottles across varied environments using UAV imagery, with the broader aim of supporting recycling initiatives. The dataset comprises of images captured from a DJI Phantom 4 Pro quadcopter equipped with a 3-axis stabilised gimbal. The footage was taken at AGL altitudes ranging from 10 to 30 metres, with a resolution of 5472×3078 pixels. To bolster dataset diversity and simulate real-world conditions, the dataset includes images featuring eight distinct background types: bush forest land, step, flat land, sand land, wasteland, mixture, plastic stadium, and grassland, as illustrated in Figure 2.16. This diversity of different backgrounds, presented as part of the dataset engineering process, accounts for the complexities associated with varied backgrounds in terms of generalising to real-world data. Furthermore, the authors highlight that the plastic bottles in the dataset are relatively small, with sizes less than 50×50 pixels, and are often transparent, which allows the background to be visible through the bottles, thereby significantly increasing the detection difficulty [17].

The BDW dataset contains 25,407 annotated images with 34,791 object instances, all belonging to a single category: *plastic bottles*. Annotations in the BDW dataset utilise

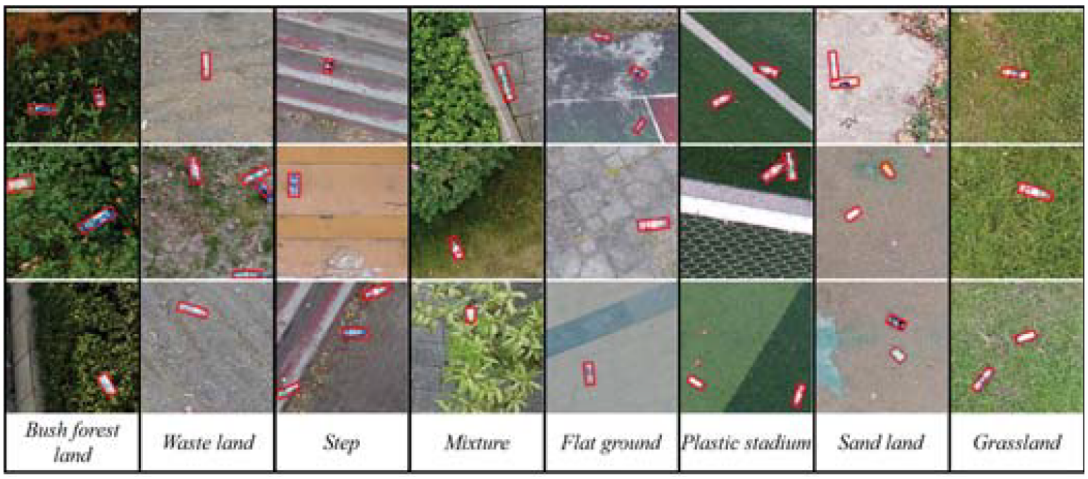


Figure 2.16 Showcasing the different backgrounds found in the BDW dataset. (Source: [17])

Object-Oriented Bounding Boxes (OBBs), which include traditional bounding box coordinates along with additional parameters: centre coordinates (c_x, c_y), height (h), width (w), and orientation angle (θ), where θ represents the angle from the horizontal axis. Additionally, the dataset was split randomly into training (64%), validation (16%), and testing (20%) subsets [17].

In their study, the authors utilise the created BDW dataset in multiple experiments to train popular object detection models, including Faster R-CNN, SSD, YOLOv2, and a Modified Rotation Region Proposal Network (RRPN) from [111]. Among these models, RRPN uniquely predicts OBBs, whereas the others predict standard axis-aligned bounding boxes. Wang et al. report that these experiments demonstrate that OBB regression is crucial for oriented object detection. In addition, RRPN achieved superior localisation accuracy while minimising false alarms and false positives, making it the most robust model in the study [17].

2.4.2 Optimising Beached Litter Monitoring through Aerial Imagery

Building on the concept of litter detection proposed by [17], Deidun et al. (2018) introduced an optimised system for monitoring beach litter through aerial imagery [112]. Their study focused on three coastal stretches within the North-East Marine Protected Area of the Maltese Islands. The monitored areas included Baħar iċ-Ċagħaq, specifically the western and eastern flanks of its rocky peninsula. The data collection process utilised a DJI Phantom 4 Pro drone, configured with a gimbal angle of -90 degrees and flown at an AGL altitude of 30 metres. This altitude was empirically determined to balance image quality and spatial resolution, following tests conducted at heights ranging from 20 to 50 metres. Images were captured at varying ground resolutions, ranging from 2.5 to 50 centimetres per pixel, to provide detailed visual data [112].

The collected footage was processed using OpenDroneMap software [113], which facilitated the creation of point clouds and texture maps. Georeferenced orthophoto maps with a resolution of 1 centimetre per pixel were generated using Global Positioning System (GPS) metadata embedded in the Exchangeable Image File Format (EXIF) data of each image file. These orthophotos were subsequently tiled and visualised in Google Earth [112].

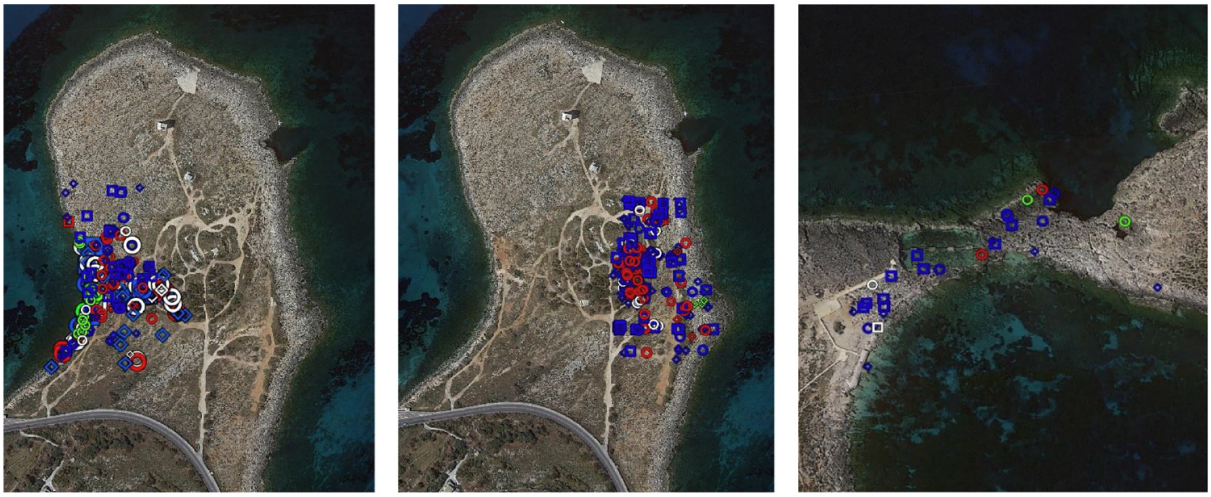


Figure 2.17 Showcasing snapshots from the digitised marine litter database, highlighting litter detected in the west (left) and east (middle) bays of Baħar iċ-Ċagħaq, and Qawra Point (right). Legend: blue = plastics; green = rope; red = wood; black = rubber; white = other non-natural materials. (Source: [112])

The digitised database of detected marine litter, shown in Figure 2.17, was created using 473 annotated images containing 608 labelled instances of litter. These instances were categorised into five litter types: *plastics*, *rope*, *wood*, *rubber*, *and non-natural items*. While the study did not involve testing object detection algorithms, its key contribution lies in presenting a robust data collection protocol and improving problem understanding [112].

2.4.3 SuperDock: Automated Floating Trash Monitoring System

To address the environmental issue of trash in rivers, Niu et al. (2019) proposed an automated river trash monitoring system called SuperDock [114]. This system consists of a Remote Processing Unit (RPU), a docking station, and a UAV. SuperDock enables the UAV to land precisely on the docking station, where automated battery replacement is performed. This allows the UAV to resume its monitoring tasks without significant wasted time. SuperDock incorporates a deep learning-based trash detection module, leveraging the YOLOv3 architecture. As illustrated in Figure 2.18, the system includes three key components: the RPU, the docking station, and the UAV. The object detection

process uses data collected from a consumer-grade UAV flying at an AGL altitude of 5 to 10 meters [114].

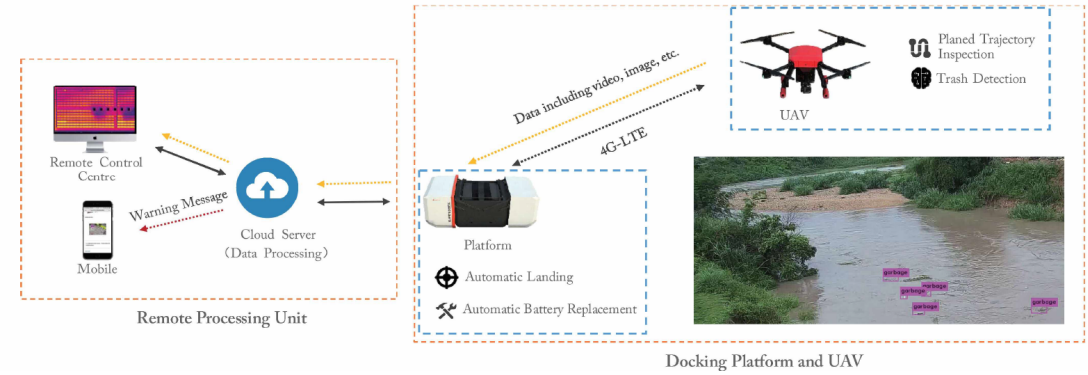


Figure 2.18 Block diagram of the automated SuperDock system. (Source: [114])

The dataset features 100 annotated images, divided into 80% for training and 20% for testing, with a total of 312 object instances. All types of litter in the dataset are annotated under a single category: *garbage*. Data augmentation techniques were applied, increasing the dataset size by a factor of five. Additionally, the authors used Microsoft AirSim (Aerial Informatics and Robotics Simulation) to experiment with the gathered data. The dataset was used to train and test three object detection models: Faster R-CNN, YOLOv3, and YOLOv3 with an improved loss function. Among these, the improved YOLOv3 model demonstrated the best performance in terms of both accuracy and processing time. While the study did not focus on creating a specialised litter detection dataset, its primary contribution lies in developing an automated trash monitoring system [114].

2.4.4 Detection and Monitoring of Styrofoam Litter using UAV Imagery

To analyse the patterns of waste generation and distribution, as well as the factors contributing to its inflow, Bak et al. (2019) proposed an automated floating trash monitoring system based on deep learning [115]. Their study focused on Heungnam Beach in Geoje, situated in Korea's marine climate zone on the South Sea. The system utilises SegNet [116], an instance segmentation model based on a convolutional encoder-decoder structure developed by the University of Cambridge, to detect beach litter. While the authors did not specify the exact number of annotated images in the dataset, UAV imagery was collected using DJI's MAVIC 2 PRO, a multi-rotor UAV. The UAV operated at an altitude of 15 meters, chosen to account for the size of the beach litter and the Ground Sampling Distance (GSD). Orthoimages were generated using Pix4D and divided into 224×224-pixel segments for neural network input, with positional information recorded for each segment [115].

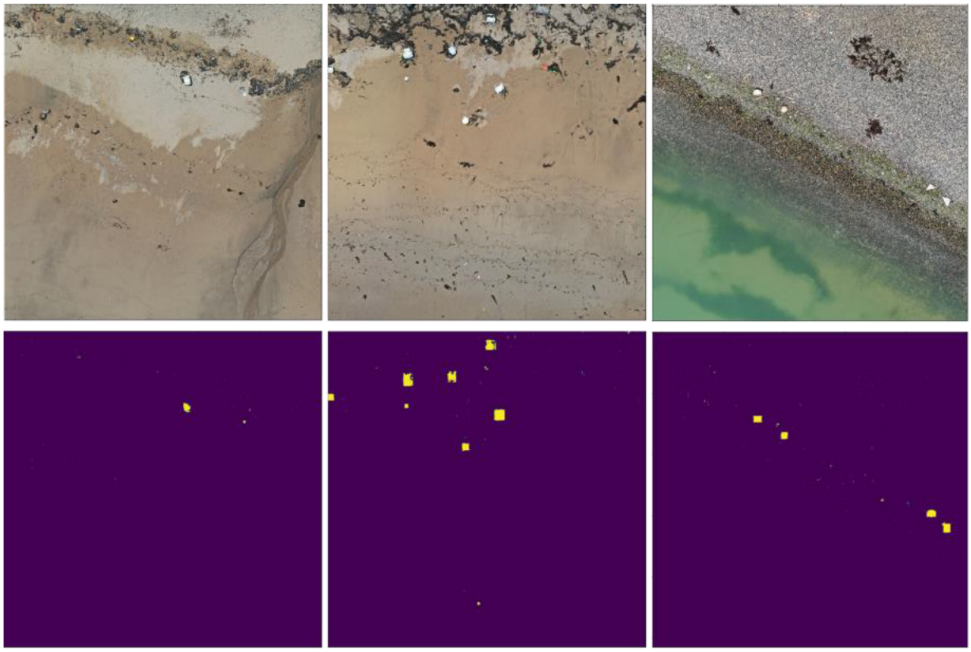


Figure 2.19 Styrofoam detection results using the trained neural network. (Source: [115])

The developed dataset only contained a single litter category, focusing on detecting *Styrofoam* waste, and was notably imbalanced, as the majority of pixels represented the background, with less than 5% occupied by detected objects. To mitigate this imbalance, data augmentation techniques were employed. Despite these challenges, the system achieved an impressive detection accuracy of 98.2%, demonstrating its efficacy in identifying Styrofoam litter on beaches, as depicted in Figure 2.19 [115].

2.4.5 Small Litter Detection in Highly Variable Backgrounds

In 2019, Schembri and Seychell proposed a case study focusing on litter detection through aerial imagery to address challenges in small object detection in highly variable backgrounds [110]. This study explored techniques for small object localisation using CNNs to detect litter in outdoor, non-urban imagery. The dataset for this research was collected using a consumer-grade UAVs at altitudes ranging from 5 to 10 metres AGL. Compiled from land surveys conducted within the Maltese Islands, the dataset includes 744 annotated images with all types of litter objects classified under a single *litter* category [110]. The algorithmic pipeline proposed by the authors for small litter detection, is illustrated in Figure 2.20. A sliding window technique was used to subdivide the entire scene into tiles of 224×224 pixels with a 32-pixel overlap, and each tile was processed individually. Non-relevant objects such as the sky, sea, and humans were masked during a scene filtering stage to improve accuracy. The litter detection problem was tackled using a VGG-16 CNN model, pre-trained on ImageNet [117], and fine-tuned with

the collected dataset. Data augmentation techniques were also applied to improve the model's robustness [110].

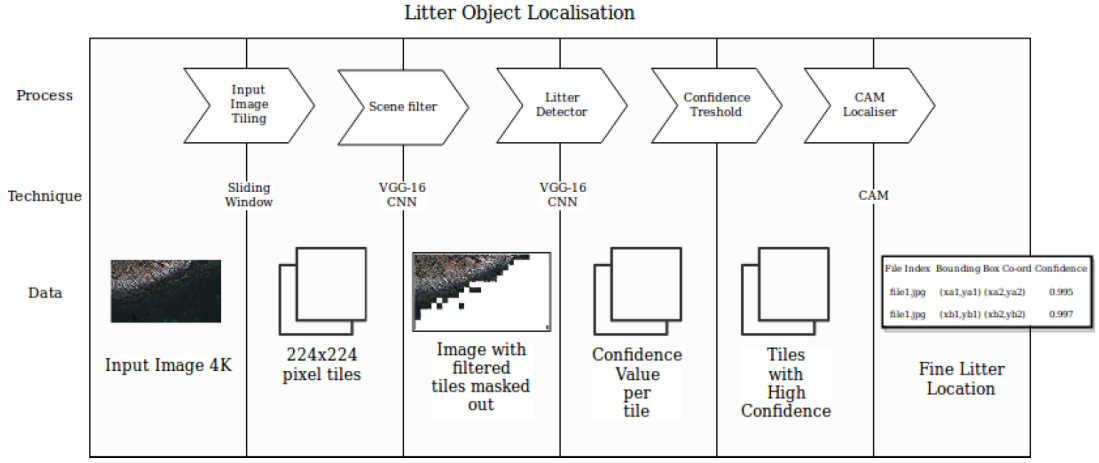


Figure 2.20 Data flow and process diagram of the litter detection algorithm. (Source: [110])

Due to the dataset's foreground-to-background sample ratio of 1:20, a confidence threshold was used to refine the detection results. CAM was employed to improve the Intersection over Union (IoU) score by highlighting relevant foreground areas. Thresholding and normalising the heatmap allowed for generating masks that effectively located the target objects. An IoU threshold of 0.01 for Non-Maximum Suppression (NMS) was used to facilitate the detection of very small objects. Furthermore, the study identified that stones and vegetation were among the classes frequently misclassified as litter. To address this issue, the authors proposed a two-stage model incorporating a terrain filter to automate the detection of misclassifications, which provided improved results. They also concluded that litter with more defined shapes was easier to detect for both human annotation and algorithmic detection. Finally, the authors also trained the Faster R-CNN detection model on the same dataset and observed lower performance metrics, which were attributed to its inability in handling highly variable background representations [110].

2.4.6 TACO: Trash Annotations in Context for Litter Detection

Detecting litter in natural environments presents considerable challenges due to the variability of litter, which can be deformable, transparent, aged, fragmented, occluded, and camouflaged. Furthermore, models must contend with the wide variety of features found in natural landscapes. To tackle these issues, Proença and Simões introduced the TACO dataset in 2020 [7]. This dataset was created to feature images captured from a variety of global environments, as shown in Figure 2.21, including beaches and urban

areas, with litter segmented and annotated according to a hierarchical classification system [7].

Figure 2.21 Annotated images from the TACO dataset, with litter objects marked using polygon masks for instance segmentation. (Source: [7])

The TACO dataset includes 60 distinct litter categories, organised into 28 top-level (super) categories, as observed in Figure 2.22. Unlike UAV-based datasets, TACO consists of images captured from ground-level natural environments. At the time of release, the dataset comprised 1,500 annotated images and 4,784 instances of litter, with an additional 3,918 new images currently awaiting annotation. The dataset uses polygon masks for annotation in an attempt to tackle the problem of both object detection, and instance segmentation [7]. The authors conducted several experiments using Mask R-CNN to evaluate the dataset's performance. Two configurations were tested:

- **TACO-1:** Identifying a single class of litter.
- **TACO-10:** Identifying ten distinct litter classes.

The authors noted that the detection performance was notably low due to challenges in detecting small items, particularly cigarette butts, which led to a high rate of false positives and negatives. This was attributed to their small size, with most images resized to 1024×1024 pixels, causing many small objects (less than 20×20 pixels) to be missed. Larger objects, such as cans and bottles, were detected more accurately, though a considerable number of bottles were misclassified as cans. Confusion was also observed between *plastic bags* and *other litter*, which is understandable given the material similarities between these categories [7].

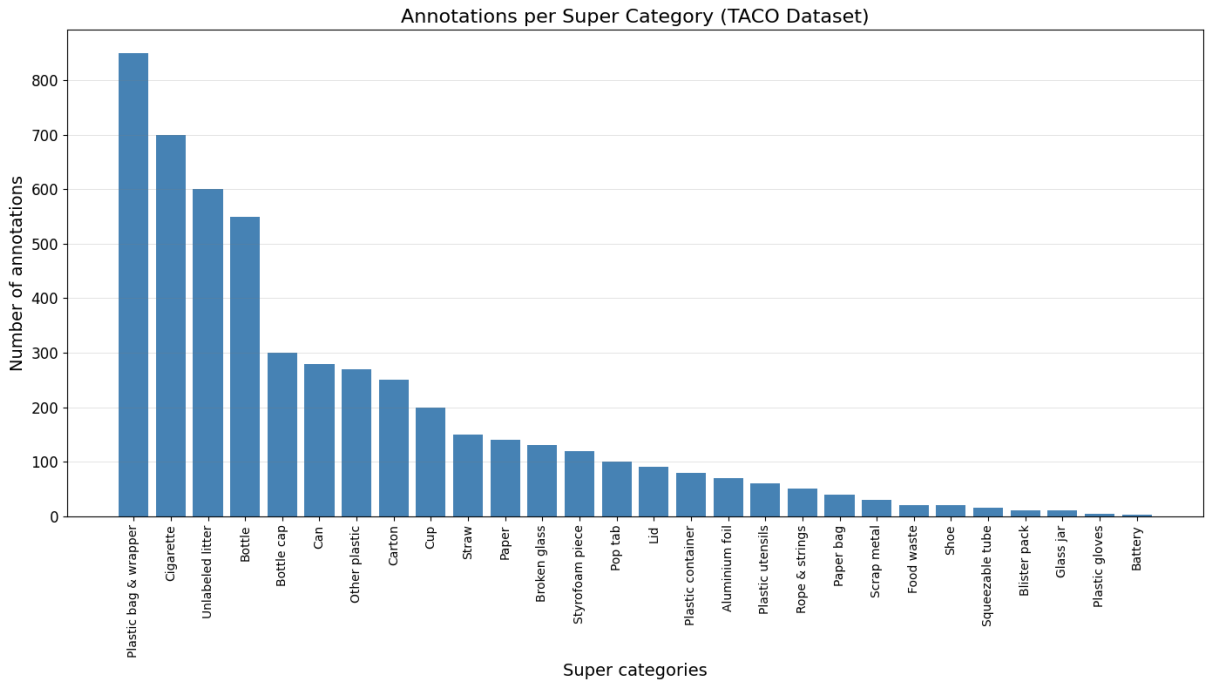


Figure 2.22 Number of annotations per super category in the published version of the TACO dataset. (Source: [7])

2.4.7 Multi-Level Approach to Waste Object Segmentation

In 2020, Wang et al. also sought to address the challenge of waste object segmentation through a multi-level approach [118]. To create their dataset, the authors collected waste items from a university campus, photographed individuals holding these items, and then annotated the images. The MJU-Waste dataset focuses solely on a single category, classifying all items under the label of *waste*. The MJU-Waste dataset contains 2,475 annotated images and 2,525 instances of litter, all of which are annotated using polygon masks. Unlike UAV-based datasets, the MJU-Waste dataset consists of images captured from controlled environments. The dataset includes segmentation masks, along with both raw and processed depth maps for all litter instances, captured using a Microsoft Kinect RGB-D camera. However, due to sensor limitations, the depth frames contain missing values, notably at reflective surfaces, occlusion boundaries, and distant regions. To address this, a median filter was applied to fill in the missing data and ensure the depth images were of high quality. Each image in the dataset was annotated with pixel-wise masks of the waste objects, and examples of colour frames, ground-truth annotations, and depth frames are provided in Figure 2.23. Alongside the semantic segmentation ground truths, object instance masks are also available [118].

The authors evaluate their proposed method against state-of-the-art semantic segmentation baselines. Their approach adopts a multi-level strategy for waste object segmentation, involving three key stages: first, scene-level parsing for initial coarse

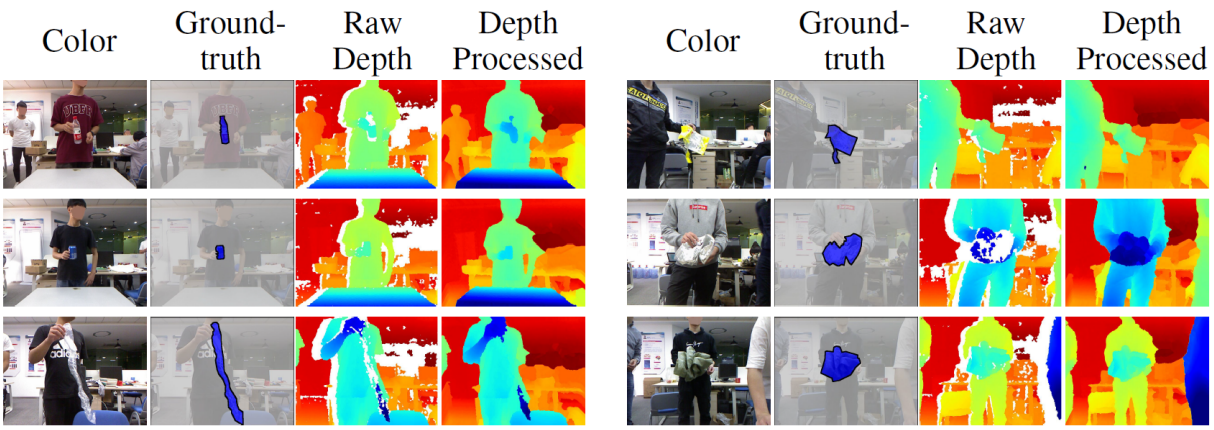


Figure 2.23 Sample RGB images, ground-truth annotations, and depth frames from the MJU-Waste dataset. (Source: [118])

segmentation; second, object-level parsing to refine object region proposals; and third, pixel-level refinement using colour, depth, and spatial affinities. By combining inference from all three levels, the method achieves a coherent and accurate final segmentation. In addition, the study focuses on two particularly challenging scenarios for waste object localisation: the hand-held setting, which is crucial for applications such as service robot interactions and smart trash bins, and waste objects found in natural environments. A significant challenge noted by the authors in both cases is the extreme variation in object scale, which leads to suboptimal performance from standard segmentation algorithms [118]. In comparison to the TACO dataset, the MJU-Waste dataset is split as follows: 60% for training, 10% for validation, and 30% for testing. The evaluation metrics used include IoU, mean IoU, and pixel precision. The models tested include FCN-8s [119], PSPNet [120], CCNet [121], and DeepLabv3 [122]. Among these, DeepLabv3 performed best in terms of IoU and mIoU, while CCNet excelled in pixel precision. Notably, DeepLabv3 also outperformed other models when tested on the TACO dataset. The authors also conclude that their multi-level approach proved effective, with scene-level segmentation, object-level segmentation, and pixel-level refinement working together to produce high-quality localisation results [118].

2.4.8 UAVVaste: Vision-Based Trash and Litter Detection in Low Altitude Aerial Images

In 2021, Kraft et al. addressed the challenge of detecting litter and the associated issue of mapping detected litter geographically [19]. Their proposed system employed a UAV equipped with onboard sensors to process low-altitude aerial imagery. A key feature of the system was its reliance on low-cost, commercially available components, which were integrated into a self-contained solution capable of autonomous operation during UAV patrol missions. A notable contribution of their research was the introduction

of an application-specific dataset, UAVVaste. This dataset consists of 772 images with 3,716 bounding box annotations, as illustrated in Figure 2.24, and is dedicated to a single category: litter, labelled as *rubbish*. The creation of this dataset was motivated by the lack of domain-specific data tailored to UAV-based litter detection. Compared to the TACO dataset, the UAVVaste dataset features a marked distribution shift, characterised by smaller object sizes, reflecting the unique challenges of detecting litter in aerial imagery [19].

Figure 2.24 Sample images from the UAVVaste dataset with litter objects annotated using bounding boxes. (Source: [19])

The UAV hardware used in the study included a Pixhawk 2 autopilot controller, a Here2 GPS/Global Navigation Satellite System (GNSS) module equipped with a barometric pressure sensor, and a downward-facing camera stabilised using a gimbal. The computational platform featured multiple components, including an Nvidia Xavier NX, a Google Coral USB (Tensor Processing Unit (TPU)), and a Raspberry Pi 4. A variety of neural network architectures were evaluated for litter detection. These included SSD detectors with lightweight backends optimised for TensorFlow Lite, which operated efficiently on the Google Coral TPU, as well as YOLOv3 and YOLOv4 and their lightweight variants. For models deployed on the Nvidia Xavier NX, TensorRT was utilised to optimise performance. Additionally, EfficientDet with a MobileNet backend was implemented. Training these models involved transfer learning, leveraging pre-trained weights from the COCO dataset to improve performance [19].

A significant portion of this study also involved developing a geolocation algorithm to map detected litter using UAV sensor data. Camera calibration and distortion correction were performed to ensure accurate mapping, assuming the camera faced directly downward. The algorithm used the UAV's altitude and camera angles to estimate

ground coverage, with localisation accuracy largely dependent on altitude measurement precision. Alongside this spatial mapping component, model performance was evaluated using standard detection metrics. The evaluation included the Mean Average Precision (mAP) scores from the COCO dataset, calculated at IoU thresholds of 0.5 and 0.95. Separate mAP and recall scores were also reported for small, medium, and large objects. Among the models tested, YOLOv4 achieved the highest mAP, while EfficientDet excelled in recall, particularly for smaller objects [19].

2.4.9 ZeroWaste: Towards Automated Waste Recycling

In 2022, Bashkirova et al. introduced the first industrial-grade, in-the-wild waste detection and segmentation dataset, ZeroWaste, making a significant contribution to computer-aided waste detection [8]. The dataset features four categories of litter: *cardboard*, *soft plastic*, *rigid plastic*, and *metal*. Unlike UAV-based datasets, the images in ZeroWaste were captured in real-world settings. The ZeroWaste dataset includes 10,715 annotated images, collected from a high-quality paper conveyor at a single-stream recycling facility in Massachusetts, containing 27,744 instances of litter, all annotated in polygon format, as illustrated in Figure 2.25. Data augmentation was also applied to the dataset to improve model generalisation by artificially increasing the variety of training samples. Furthermore, this four-class labelled dataset captures various waste items identified during the sorting process at the facility, which aims to separate high-quality paper from contaminants such as metal, plastic, brown paper, and cardboard [8].

Figure 2.25 Sample images (left) and ground truth polygon annotations (right) in the ZeroWaste dataset. (Source: [8])

The data was gathered using two compact recording systems placed at the start

and end of the conveyor belt during the facility's routine operations. The footage consists of 12 video sequences with a total duration of 95 minutes and 14 seconds, recorded at a frame rate of 120 FPS and a resolution of 1920×1080. To ensure high-quality data, the footage was processed to correct optical distortions, crop unnecessary elements, and remove motion blur. The ZeroWaste dataset was divided as follows: 65% for training, 15% for validation, and 20% for testing. The evaluation metrics used include Average Precision (AP) and mAP at various thresholds, IoU, and pixel precision. Additionally, the models tested include RetinaNet, Mask R-CNN, TridentNet [123], and DeepLabv3, and a series of experiments were conducted, including fully, semi, and weakly supervised learning. The results revealed that the ZeroWaste dataset posed considerable challenges for state-of-the-art detectors such as Mask R-CNN. However, TridentNet emerged as the top-performing detector, while DeepLabv3 achieved the best results in terms of segmentation [8].

2.4.10 PlastOPol: Litter Detection with Deep Learning

To address the issue of litter accumulation, Córdova et al. investigated the efficacy of lightweight neural networks for detecting litter in real-world environments with complex and crowded image backgrounds [18]. The study also evaluated the feasibility of deploying these models on mobile devices with limited memory capacity. Their 2022 publication presented two key contributions: a comparative analysis of state-of-the-art deep learning techniques for image-based litter and waste detection and the introduction of a novel dataset, PlastOPol, comprising 2,418 images captured in real-world conditions with 5,300 litter annotations. The PlastOPol dataset includes a single class, with all litter items categorised under the label *litter*. The dataset consists of images from natural settings rather than being derived from UAV data. These images were sourced using the Marine Debris Tracker and are publicly available under a Creative Commons Attribution licence. The dataset was annotated in a bounding box format and focused on the problem of object detection. Of the 5,300 annotated instances, 90.98% are large objects, 8.40% are medium objects, and only 0.62% are small objects [18].

To assess the performance of various deep learning models in detecting litter, the authors used both the proposed PlastOPol dataset and the publicly available TACO dataset. For the PlastOPol dataset, the data was split into 80% for training and 20% for testing. Evaluation metrics included AP, mAP at IoU threshold 0.5, F1 score, and recall. The models tested included EfficientDet, Faster R-CNN, Mask R-CNN, RetinaNet, and YOLOv5. As shown in Figure 2.26, the visual detection results on the PlastOPol dataset highlight the performance of these models in identifying litter objects across different environments. The experiment results demonstrated that YOLOv5 was the best-performing model overall. The authors also observed that YOLOv5 and Efficient-

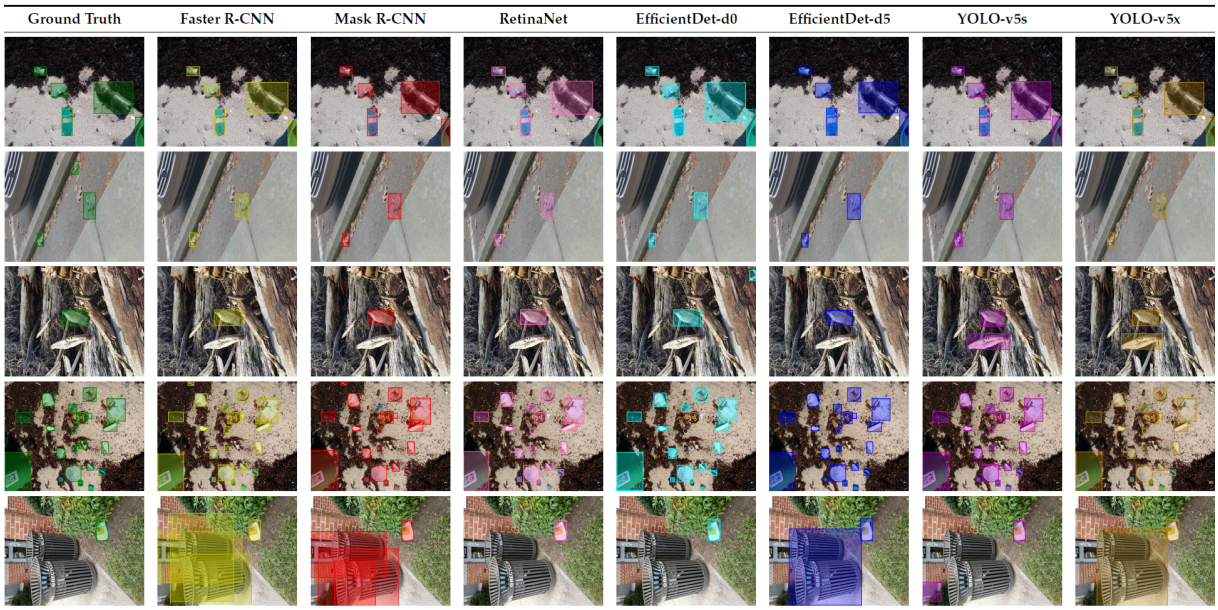


Figure 2.26 Visual detection results on the PlastOPol dataset. (Source: [18])

Det were the fastest models, even when deployed on mobile devices. However, they noted that all methods struggled with detecting small objects, such as cigarette butts. Notably, YOLOv5x outperformed Faster R-CNN and Mask R-CNN in terms of speed [18].

2.4.11 Real-Time UAV Trash Monitoring System

In 2022, Liao and Juang developed a marine litter detection system using UAVs [124]. Their goal was to reduce reliance on manual efforts and offer a more efficient method for identifying marine debris. The real-time UAV trash monitoring system consisted of nine components: UAVs, a message queuing system, a database, a video streaming server, a connector, a UAV control station, a web service, a UAV map, and a data analysis module, as illustrated in Figure 2.27. The ground station incorporated key elements, such as the video streaming server, Kafka, the Kafka connector, MongoDB, and the web service. The system addressed two key challenges: object detection and geolocation, and as part of the system's development, the authors introduced the HAIDA dataset. This dataset, collected on the NTOU campus, was used to train the object detection model and contains two types of litter objects: *garbage* and *bottles*. The data was gathered using a UAV equipped with a Pixhawk controller and an NVIDIA Jetson Xavier NX. The UAV operated at altitudes ranging from 1 to 10 metres AGL [124].

The HAIDA dataset comprises 1,319 annotated images, with 6,475 instances of litter, including 3,904 garbage objects and 2,571 bottle objects. Additionally, 456 images in the dataset contain no litter. The authors utilised the dataset to train a YOLO object detection model (version unspecified). The results showed that the model was able to

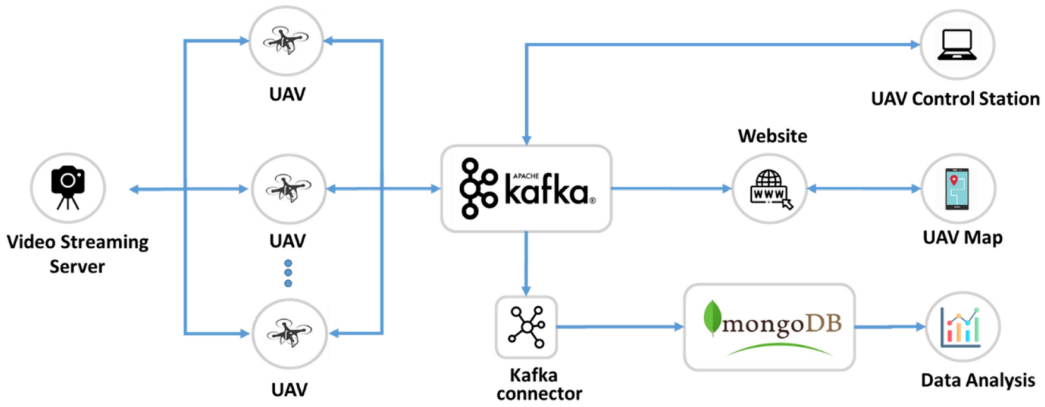


Figure 2.27 The system architecture of the real-time UAV trash monitoring system. (Source: [124])

identify trash objects with an AP of over 70% at a 0.5 IoU threshold. Notably, as shown in Figure 2.28, the detection confidence for small objects was high, with confidence scores of 98.27%, 64.65%, and 91.86% for different categories of litter. The real-time UAV trash monitoring system operated by first collecting UAV footage as the drone flew across predetermined waypoints. The footage was then transmitted to a server via Kafka, where it was stored in an online database such as Oracle, MySQL, or MongoDB. Using this stored data, the server applied the trash detection model to identify litter both in the images and on a map. Control stations, both mobile and desktop, could make requests to the server to retrieve the map data, which displayed the locations of detected litter [124]. To estimate litter geolocation, the authors followed an approach similar to that in [19], calculating pollution area based on UAV altitude, camera angles, and object size in the image. This enabled a reliable approximation of both the position and extent of detected waste [124].

(a) (b)

Figure 2.28 UAV trash monitoring results on the NTOU campus. (a) Detection results from the UAV. (b) Detected litter plotted on a map, showing real-time monitoring and detection. (Source: [124])

2.4.12 Outdoor Trash Detection in Natural Environment

As populations in underdeveloped nations continue to grow, the challenges associated with trash production and disposal have become more urgent. Aware of the time-consuming and potentially hazardous nature of manual litter classification, Das et al. (2023) aimed to tackle this issue by developing automated methods for more efficient and safer waste management [125]. To achieve this, they focused on collecting data that accurately captured the complexities of litter in Bangladesh, creating an outdoor dataset, and incorporating OpenLitterMap to broaden its scope. This dataset comprises images from natural settings, not UAV-based data, and includes ten categories of litter: *tissue paper*, *plastic*, *medical waste*, *rope*, *paper*, *cigarette butts and boxes*, *metal*, *glass*, *organic waste*, and *textiles*. Initially consisting of 1,283 annotated images, the dataset was expanded to 4,418 by integrating data from OpenLitterMap, a global database containing litter and plastic images. The Bangladeshi dataset contains 6,178 instances of litter, which was extended to 8,077 while retaining the ten distinct classes [125].

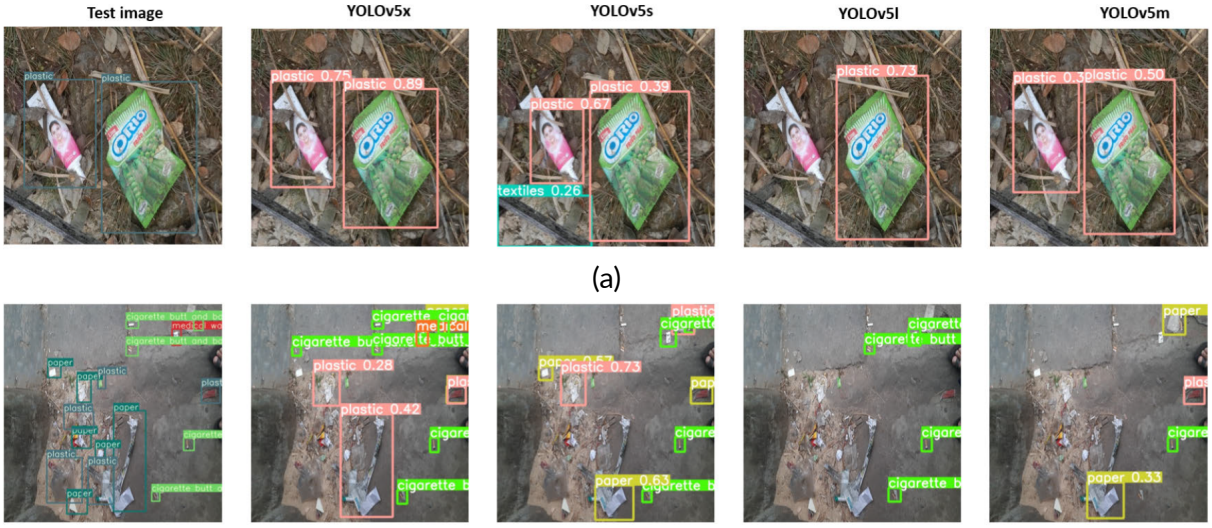


Figure 2.29 Litter detection results using different versions of the YOLOv5 model on the Bangladeshi dataset. (a) 2 instances of litter; (b) 17 instances of litter. (Source: [125])

The experimental setup used for this study involved a comparison with the TACO and PlastOPol datasets. The Bangladeshi dataset was divided as follows: 80% for training, 10% for validation, and 10% for testing. The evaluation metrics included AP and mAP at IoU thresholds from 0.5 to 0.95, as well as F1 score and recall. Several models were tested, including different versions of YOLOv5, YOLOv6, YOLOv8, and Faster R-CNN. The authors also employed data augmentation techniques during training and optimised model performance by adjusting hyperparameters. In their study, the authors reported that YOLOv5x was the best-performing model, as visually demonstrated

in Figure 2.29, which shows a comparison of the detection bounding boxes on the images, demonstrating the improved mAP for the expanded dataset. However, single-class detection experiments on the TACO and PlastOPol datasets yielded better results compared to the Bangladeshi dataset. Das et al. also highlighted that in future work, they plan to expand the dataset to include a broader range of categories, such as ocean waste, which is currently absent. Additionally, other object detection methods, such as SSD, Mask R-CNN, and EfficientDet, could also be evaluated using their dataset [125].

2.4.13 Use of UAVs and Deep Learning for Beach Litter Monitoring

In 2023, Pfeiffer et al. proposed a framework for an autonomous beach litter monitoring and retrieval system based on drone surveys and object detection using deep learning techniques [126]. Their research combined drone footage collected from the islands of Malta and Gozo, Sicily (Italy), and the Red Sea coast with publicly available litter datasets, which were subsequently used to train an object detection model for identifying litter in the captured footage. Figure 2.30 illustrates the overall framework of the proposed beach litter monitoring system. To address both object detection and geolocation challenges, the authors compiled a UAV dataset collected with a DJI Phantom 4 Pro 2.0 and a DJI Mavic 2 UAVs. The dataset covered altitudes ranging from 10 metres above ground level to 60 metres above sea level and included 67 types of litter, which were categorised into seven meta-classes: *clothing & fabric*, *glass*, *small objects*, *metal litter*, *paper & cardboard*, *plastic*, and *other waste*. Additionally, the collected Beach Litter Dataset contains 4,126 annotated images and 10,611 litter instances, including 1,154 images with no litter present. The dataset was divided as follows: 60% for training, 20% for validation, and 20% for testing. The models were evaluated using metrics such as AP,mAP (at 0.5–0.95), F1 score, and recall, with the YOLOv5 model being the primary model used for training [126].

The authors also created a geolocation algorithm to link detected litter with a debris-retrieving robot, requiring that detection coordinates be translated into real-world positions. This process used GPS-tagged footage and involved calculating pixel dimensions, spatial distances, and converting image offsets into geographic coordinates. Accuracy was further refined by incorporating UAV height and camera angle, ensuring reliable localisation for robotic retrieval [126]. The results of their experiments using the YOLOv5 model on the labelled dataset were as follows: precision of 0.695, recall of 0.288, mAP@50 of 0.314, and mAP@50-95 of 0.2. The model performed most reliably on the following ten classes: *plastic bottle*, *metal can*, *plastic bottle cap*, *plastic container*, *shoe*, *cardboard*, *pop tab*, *rope & string*, *wood*, and *glass bottle* [126].

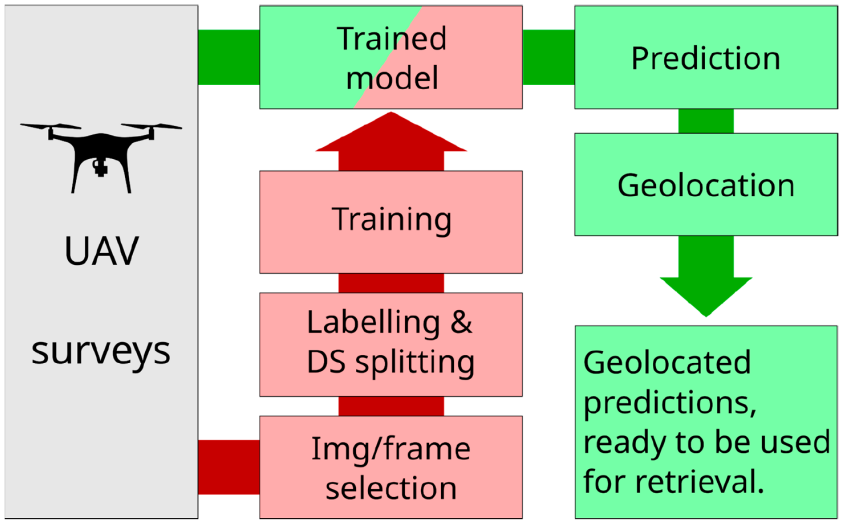


Figure 2.30 The system architecture of the real-time UAV beach litter monitoring and geolocation system, highlighting the two main pipelines: training (in red) and prediction (in green). (Source: [126])

2.4.14 TrashNet: Real-Time Object Detection for Trash Classification

In 2024, Veeravadivel Santhanalakshmi, and Nguyen introduced TrashNet, an object detection model designed to classify images of waste in real time. The dataset employed to train the TrashNet network comprises six primary categories of litter: *cardboard*, *glass*, *metal*, *paper*, *plastic*, and *general waste*. This dataset, derived from an image classification dataset on Kaggle, does not utilise UAV-based data. Furthermore, the TrashNet dataset comprises 2,524 annotated images, each featuring a single object instance. The authors explain that the dataset was divided into three subsets for model development: 70% for training, 20% for testing, and 10% for validation. In this study, the YOLOv5 object detection model was used, and its performance was evaluated using metrics such as mAP, precision, recall, and F1 score. The authors highlight that the primary aim of this study was to develop a real-time system to assist individuals in identifying the correct bin for various types of waste near rubbish disposal areas, as seen in Figure 2.31. Notably, the model successfully fulfilled this objective, achieving an accuracy of 90% on both the validation and testing sets [127].

2.4.15 SODA: Small Objects at Different Altitudes

Modern UAVs are equipped with high-resolution cameras capable of capturing detailed pictures and videos of their surroundings. However, when the UAV is flying at higher altitudes, objects in the images may look very tiny, occupying only a few pixels. This poses substantial hurdles for spotting such objects. To tackle this issue, Pisani et al. (2024) introduced the SODA dataset, designed to aid research focused on small object

Figure 2.31 Litter detection results generated by the TrashNet detection model. (Source: [127])

detection in aerial imagery [13]. The dataset features six primary types of litter, derived from the TACO dataset: *clear plastic bottles*, *other plastic bottles*, *glass bottles*, *glass jars*, *drinking cans*, and *drinking cartons*. Figure 2.32 illustrates examples of litter categories used in the SODA dataset. Furthermore, the images were collected using a combination of DJI Mini 2 and DJI Air 2S UAVs at various altitudes, including 1m, 5m, 10m, 15m, 20m, 25m, and 30m [13, 20].

Figure 2.32 The six object categories featured in the SODA dataset. (Source: [13])

The SODA dataset consists of 829 annotated images, with 452 (54.52%) taken

at 1 metre and 377 (45.48%) captured from altitudes of 5 metres or more. The SODA dataset supports multi-class classification, with each object annotated with a distinct label using polygons rather than bounding boxes. The authors explain that polygons offer a more accurate fit around the object, whereas bounding boxes often capture additional background. Additionally, they clarify that polygons provide precise alignment during augmentations, such as rotation. Techniques for annotating with polygons and applying these augmentations are outlined in [128]. However, although the SODA dataset is annotated using polygons, the authors focus on the challenge of object detection rather than instance segmentation. To address the challenge of detecting small objects across multiple categories of litter, the authors employed three different training strategies. The first approach involved segregating the training dataset according to altitude. In the second approach, the dataset was merged while maintaining multi-class labels. The third approach also merged the dataset, consolidating all categories into a single *litter* class. Notably, a key feature of this study was the application of a *tiling methodology* to improve the detection of small objects. This technique divides an image into a grid and splits it into smaller, equally sized sections, which were used in both the training and inference stages. The authors note that the grid size can significantly influence detection performance, with smaller grids, such as 2×2 (refer to Figure 2.33), used at lower altitudes and larger grids, like 5×5, applied at higher altitudes. For the SODA dataset, a 5×5 grid was used across all altitudes and images [13, 20, 129].

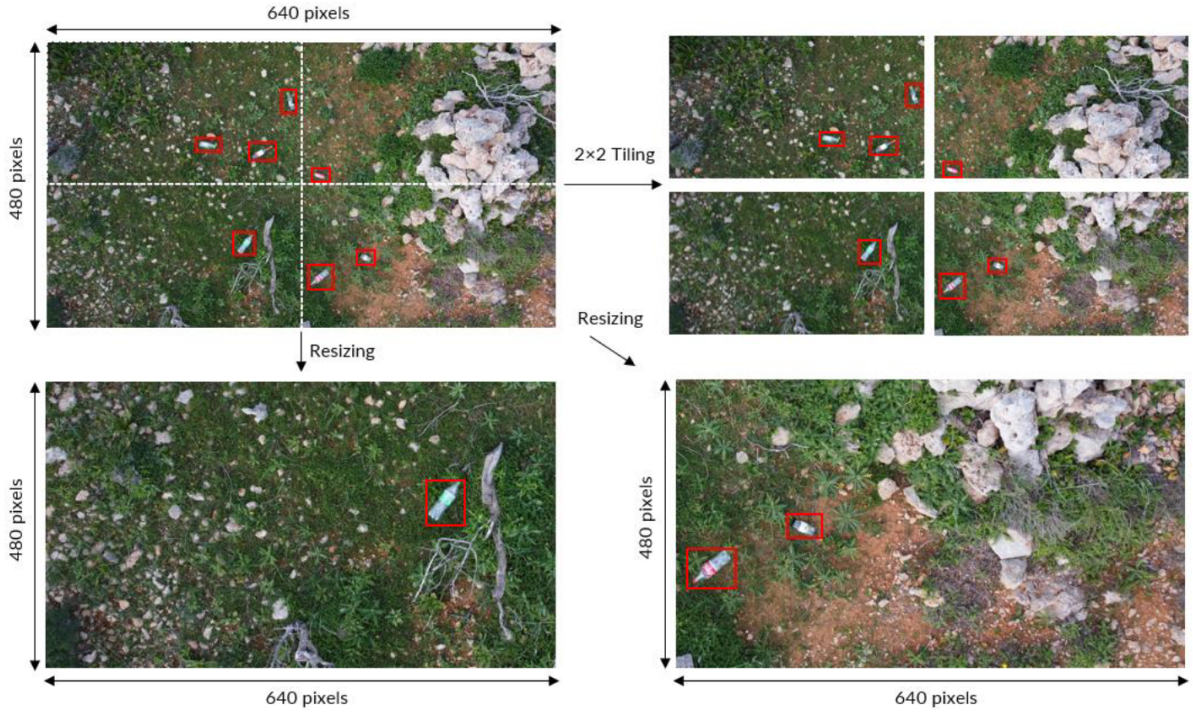


Figure 2.33 Application of a 2×2 tiling grid to aerial imagery captured at an altitude of 5 metres. (Source: [20])

Three object detection models were trained for this study: YOLOv5 and YOLOv8

(both one-stage detectors) and Faster R-CNN (a two-stage detector). Models trained using the second and third approaches were evaluated on the BDW dataset. The results indicated that the third approach, which merged all classes into a single litter category, outperformed the second approach. Additionally, the evaluation revealed that the SODA dataset was lacking a sufficient number of images of clear plastic bottles. In terms of mAP, the results for the different models were as follows: Faster R-CNN achieved a mAP of 0.921, YOLOv5 had a mAP of 0.854, and YOLOv8 performed with a mAP of 0.728 [13, 20]. Conclusively, this study contributes to the use of UAVs for capturing aerial imagery and applying computer vision techniques to detect and classify objects within these images. The authors emphasise its significance as a crucial phase and a stepping stone towards the automation of litter collection for cleanup efforts [13, 20, 129].

2.4.16 Discussion

The reviewed datasets and approaches highlight the growing potential, and popularity of UAV-based AI systems for litter detection and management. While certain datasets were originally developed for recycling classification in indoor or controlled settings, others target outdoor environments, which present significantly greater challenges. Datasets that rely on UAVs are particularly affected by the difficulty of identifying small objects from aerial perspectives. The variation across datasets reveals a wide range of technical and contextual difficulties associated with litter detection, as shown in Table 2.1. For example, datasets such as TrashNet [127], MJU-Waste [118], and ZeroWaste [8] were created under artificial conditions. In contrast, datasets like PlastoPol [18], TACO [7], and several UAV-oriented collections [13, 17, 19, 112, 114, 124, 126] contain real-world data collected in natural and urban environments.

Honing in on UAV-based litter deteciton, a recurring limitation evident in UAV datasets is the restricted number of annotated categories. In many cases, only one category is included. To this end, when multiple categories are included in a dataset, a common practice observed was to conduct an experiment by merging all classes into a single general litter category. Another challenge lies in the height at which data are collected. Most UAV-based datasets capture images at around 10 metres AGL or less. Exceptions such as BDW [17] and SODA [13] extend this altitude to 30 metres, while the Beach Litter Dataset [126] reaches up to 60 metres. Notwithstanding this, from among the UAV litter datasets only the SODA dataset [13] provides altitude-specific subsets of data. Notably, the diversity of litter datasets and public accessibility is a key factor in supporting further development of AI-based litter detection systems. Although several of the reviewed datasets have been featured in international peer-reviewed conferences and journals, only a limited number are available to the public. As illustrated

in Table 2.1, the publicly accessible datasets include BDW [17], TACO [7], MJU-Waste [118], UAVVaste [19], ZeroWaste [8], PlastoPol [18], HAIDA [124], TrashNet [127], and SODA [13], with just four of these specifically developed for UAV applications.

In terms of detection methods, *data augmentation* is frequently employed to improve model performance. This is a common practice across both UAV and ground-based approaches. Techniques such as image *tiling* or *slicing* are especially important to facilititate UAV-based litter detection. High-resolution drone footage, often recorded in 4K, provides greater visual detail, which helps in identifying smaller objects. However, standard object detection models typically require downscaled inputs, ranging between 640 and 1280 pixels. This reduction can enable small litter objects even harder to detect. Splitting the original image into smaller tiles helps to preserve object size and minimise the loss of pixel detail. Nevertheless, proper experimentation of this method must be ensured, as adopting this process increases the computational load.

Name	Year	No of. Images	UAV	AGL Altitudes	Dataset Details	No of. Categories	Available
BDW Dataset [17]	2018	25,407	Yes	10m–30m	Detection	1 (Litter)	Yes
UM Geo. Survey [112]	2018	472	Yes	30m	Data Collection	5 (Litter)	No
SuperDock [114]	2019	100	Yes	5m–10m	Detection	1 (Litter)	No
Styrofoam Monitoring [115]	2019	N/S[1]	Yes	15m	Detection, Segmentation	1 (Litter)	No
Small Litter Detection [110]	2019	744	Yes	5m–10m	Detection	1 (Litter)	No
TACO Dataset [7]	2020	1,500	No	N/A[2]	Detection, Segmentation	60 (Litter) [28 Super]	Yes
MJU-Waste Dataset [118]	2020	2,475	No	N/A[2]	Segmentation	1 (Litter)	Yes
UAVVaste Dataset [19]	2021	772	Yes	low-altitude	Detection, Geolocation	1 (Litter)	Yes
ZeroWaste Dataset [8]	2022	10,715	No	N/A[2]	Detection, Segmentation	4 (Litter)	Yes
PlasOPol Dataset [18]	2022	2,418	No	N/A[2]	Detection	1 (Litter)	Yes
HAIDA Dataset [124]	2022	1,319	Yes	1m–10m	Detection, Geolocation	2 (Litter)	Yes
Bangladeshi Dataset [125]	2023	4,418	No	N/A[2]	Detection	10 (Litter)	No
Beach Litter Dataset [126]	2023	4,126	Yes	10m–60m	Detection, Geolocation	67 (Litter) [7 Super]	No
TrashNet [127]	2024	2,524	No	N/A[2]	Detection	6 (Litter)	Yes
SODA Dataset [13]	2024	829	Yes	1m, 5m–30m	Detection, Segmentation	6 (Litter) [4 Super]	Yes

Table 2.1 Comparison of datasets and approaches, systematically organised, with litter-related images captured from both UAV and non-UAV data.

In addressing the problem of litter detection, most studies rely on deep learning models. The most commonly used architectures include variants of YOLO [61, 65, 130], along with Faster R-CNN [79], RetinaNet [71], SSD [70], and EfficientDet [84]. A typical approach involves using these general object detection models as out-of-the-box solutions for the task [13, 18, 20, 126]. Other studies, such as those by [110, 115], have

[1] N/S: The amount of images was not specified.
[2] N/A: Not applicable for non-UAV-based litter detection datasets.

explored modifications to the architecture and incorporated techniques such as background removal and CAM to improve results. Additionally, some methods [7, 8, 118] approach the problem not just as an object detection problem but also as one of litter instance segmentation. In this regard, models like DeepLabv3 [122] and Mask R-CNN [80] have gained prominence. Interestingly, other UAV-based approaches [19, 124, 126] aim to address litter geolocation or georeferencing. These methods develop algorithms based on drone properties and mathematical formulas, allowing integration with the detection outputs to determine the precise locations of the detected litter.

2.5 Review of Learning Using Privileged Information in Computer Vision

At the time of writing this dissertation, the application of the LUPI paradigm within object detection remains underexplored. One early attempt is the work of Feyereisl et al. (2014) [131], who employed segmentation masks and SURF features as privileged information for object localisation. Their approach enabled a Structural SVM+ algorithm to incorporate privileged information on the Caltech-UCSD Birds 2011 dataset [132]. However, this study is relatively old, addressed only the problem of general localisation, and did not involve modifications to deep learning–based object detection models. The reported improvements from privileged information were also marginal. A later study by Sun et al. (2018) [133] also used the Caltech-UCSD Birds dataset to examine localisation with privileged information. In their work, segmentation masks were applied to a subset of images where the birds appeared relatively large and well-defined. The method, based on the Joint Kernel Support Estimation (JKSE) algorithm [134] and evaluated on this subset, achieved only limited improvements in localisation accuracy as measured by the IoU metric. These works highlight that research on LUPI in detection and localisation remains at an early stage. More broadly, in computer vision, applications of LUPI are still sparse, even though many tasks involve an asymmetry between the information available during training and that at test time [135], a setting where LUPI is particularly relevant.

Sharmanska et al. [135, 136] explore four distinct forms of privileged information for image classification: semantic attributes, bounding boxes, textual descriptions, and annotator rationale. Their experiments apply these types of privileged information within a rank transfer framework compatible with SVM solvers, focusing on the SVM+ algorithm. Their findings indicate a measurable improvement in classification performance when LUPI is integrated into the learning process [135, 136]. In a related contribution, Wang et al. [15] also address multi-label image classification by incorporating privileged information during training. Their approach employs similarity constraints derived from

privileged inputs, along with ranking constraints informed by multiple labels, to develop a more effective classifier. High-resolution images and associated image tags serve as the privileged data, available solely during training. The method is evaluated across several benchmark datasets, and the results suggest that the inclusion of privileged information, alongside an awareness of label dependencies, contributes meaningfully to improved model performance [15].

Although not directly related to LUPI, a substantial body of work aligns conceptually through the use of knowledge distillation techniques within computer vision [108, 109]. Knowledge distillation is a prominent method in machine learning, facilitating the transfer of information from a high-capacity model to a smaller, more efficient counterpart [105]. This process enables the distilled model to retain essential predictive capabilities while reducing computational overhead.

In the context of vision-based tasks, various distillation methods have been explored, including those based on output responses, internal feature representations, and inter-feature relationships [109]. These techniques are applicable across a range of visual domains, including image classification, object detection, and multimodal frameworks. Specific to object detection, two widely adopted strategies are feature imitation and logit matching, which aim to replicate intermediate activations and output distributions respectively [108]. A noteworthy development is the concept of localisation distillation, as proposed in [108], which refines the transfer process by concentrating on spatial regions deemed informative for both classification and localisation. This targeted distillation allows the student model to better capture salient visual patterns by selectively attending to regions that contribute meaningfully to detection accuracy.

2.6 Metrics

To objectively assess the performance of object detection systems, it is necessary to rely on standardised evaluation protocols. These metrics enable a consistent comparison between different models and methodologies, independent of the datasets or specific applications involved. In the context of object detection, evaluation typically involves comparing predicted bounding boxes and class labels against annotated ground truth data using metrics derived from the IoU criterion. This section outlines the most widely used performance metrics in vision-based detection tasks.

Performance is typically measured by comparing predicted bounding boxes and class labels against annotated ground truth using the IoU metric, which quantifies the overlap between predictions and actual annotations. The IoU threshold defines the minimum required overlap for a detection to be accepted as a true positive. Lower thresholds (e.g., 0.5) allow for more leniency, while higher thresholds (e.g., 0.75) enforce stricter ac-

curacy requirements [137]. This threshold informs the calculation of standard indicators: True Positives (TP), which represent correctly detected objects; False Positives (FP), indicating incorrect or insufficient detections; and False Negatives (FN), which refer to missed objects that appear in the ground truth. True Negatives (TN) are generally excluded from these evaluations, as they pertain to correct predictions of object absence.

IoU, or Jaccard's Index, serves as a metric for evaluating the overlap between the predicted bounding box and the ground truth, offering a quantitative measure of the precision in object localisation. As demonstrated in Equation 2.3, b denotes the predicted bounding box, while g refers to the ground truth. A higher IoU value, with a maximum of 1, signifies greater accuracy in the predictions, indicating a smaller deviation between the predicted bounding box and the ground truth.

$$\text{IoU}(b, g) = \frac{\text{area}(b \cap g)}{\text{area}(b \cup g)}. \tag{2.3}$$

Precision, as outlined in Equation 2.4, measures the proportion of relevant items retrieved by the model. It is determined by the ratio of TP, which are objects correctly identified as relevant, to the sum of TP and FP, which denotes the total number of objects that were mistakenly identified as relevant.

$$\text{Precision} = \frac{\text{TP}}{\text{TP} + \text{FP}} = \frac{\text{IoU}(b, g) > \text{threshold}}{\text{IoU}(b, g) > \text{threshold} + \text{FP}}. \tag{2.4}$$

Recall, as presented in Equation 2.5, evaluates the model's ability to identify all relevant items. Recall is calculated as the ratio of TP to the sum of TP and FN, with FN representing relevant objects that the model failed to detect.

$$\text{Recall} = \frac{\text{TP}}{\text{TP} + \text{FN}} = \frac{\text{IoU}(b, g) > \text{threshold}}{\text{IoU}(b, g) > \text{threshold} + \text{FN}}. \tag{2.5}$$

The F1 Score, represented in Equation 2.6, is the harmonic mean of precision and recall, offering a balanced evaluation of both. It is computed by taking the product of precision and recall, doubling it, and dividing by the sum of precision and recall, thereby reflecting the model's overall ability to identify relevant objects.

$$\text{F1 Score} = 2 \times \frac{\text{Precision} \times \text{Recall}}{\text{Precision} + \text{Recall}}. \tag{2.6}$$

Average Precision (AP), as outlined in Equation 2.7, evaluates the overall precision at various recall levels, R_k, where k represents a recall threshold between 0 and 1. AP reflects the balance between FP and FN, providing a more nuanced evaluation of model performance. It can also be interpreted as the area under the precision-recall curve

(AUC), with higher values signifying better model performance.

$$\text{Average Precision (AP)} = \sum_{k=0}^{n} (R_k - R_{k-1}) \cdot P_k. \tag{2.7}$$

The Mean Average Precision (mAP), as represented in Equation 2.8, computes the average precision across all N classes by summing the individual average precision values (AP_i) and dividing by the total number of classes. mAP values range from 0 to 1, with higher values indicating better model performance.

$$\text{Mean Average Precision (mAP)} = \frac{1}{N} \sum_{i=1}^{N} \text{AP}_i. \tag{2.8}$$

The Mean Average Recall (mAR), as defined in Equation 2.9, measures the model's average recall across all N classes. For each class, recall is computed at several IoU thresholds, typically ranging from 0.50 to 0.95 in steps of 0.05. These per-class recall values are first averaged across IoU levels, then across all categories. This results in a single mAR score that reflects the model's ability to detect objects across varying overlap conditions. Additionally, mAR can be reported at different detection counts, such as mAR@1, mAR@10, and mAR@100. These settings restrict the evaluation to the top 1, 10, or 100 predicted boxes per image, respectively [137].

$$\text{Mean Average Recall (mAR)} = \frac{1}{N} \sum_{i=1}^{N} \text{AR}_i(\text{IoU}). \tag{2.9}$$

The confusion matrix, shown in Equation 2.10, is a metric that summarises a classification network's performance by reporting TP, FP, FN, and TN for each class. In the context of object detection, a background class (denoted as BG) is added to account for regions without objects. The matrix includes all object categories the model is trained to detect, collectively represented as the set C. A strong diagonal in the matrix suggests high accuracy, indicating that predicted labels closely match the actual ones.

$$\underset{c \in \{BG, 1, \ldots, C\}}{\text{Confusion Matrix}} = \begin{array}{c|cc} & \text{Predicted Positive} & \text{Predicted Negative} \\ \hline \text{Actual Positive} & \text{TP}_c & \text{FN}_c \\ \text{Actual Negative} & \text{FP}_c & \text{TN}_c \end{array}. \tag{2.10}$$

2.7 Conclusion

This chapter has provided a structured foundation for understanding the task of object detection, describing its core challenges and progressing through the evolution of methodologies, from traditional techniques to contemporary deep learning models. It

has also outlined the conceptual basis and definition of the LUPI framework, highlighting its potential to address information asymmetries during training by incorporating privileged data and distilling knowledge effectively. The discussion then turned to existing approaches in litter detection, considering both aerial and ground-based methods, as well as the broader application of LUPI within computer vision. Collectively, these sections provide the necessary foundation for a focused examination of how models trained using the LUPI framework may improve both the performance and robustness of litter detection systems. To complete the framework, relevant evaluation metrics were introduced, setting the stage for the rigorous assessment of these models in subsequent chapters.

Chapter 3 Methodology

"We cannot solve problems with the same thinking we used when we created them."

– Albert Einstein

3.1 Introduction

This chapter opens with a formal mathematical formulation and theoretical context for the problem under investigation, specifically addressing how LUPI may be incorporated into object detection. This initial section establishes the conceptual basis before proceeding to define the role of privileged information within this context. Subsequently, the chapter presents an overview of the object detection models selected for this study. This is followed by a detailed account of the modifications made to these models to facilitate the construction of the teacher and student networks. An architectural overview is provided to clarify the interrelation between these components and the general configuration adopted. The following section outlines the implementation details, including the training parameters employed during the experiments outlined in the next chapter. Furthermore, this section also covers the pre-processing and post-processing steps applied uniformly across all models. The discussion then turns to the datasets employed in this study. Both UAV-based litter detection datasets and the Pascal VOC 2012 dataset are examined, highlighting the dataset pre-processing techniques required for each. The structure and nature of the data are also considered in relation to the proposed evaluation methodology.

3.2 Problem Definition

To properly investigate the role of learning using privileged information in object detection and assess its feasibility, as per the first objective (**O1**), it is first necessary to understand the core problem being tackled and the required outputs. As outlined in Section 2.2, the object detection problem can be decomposed into two distinct components: localisation and classification. With respect to the localisation component, the objective is to predict a bounding box (b) that encapsulates the object of interest. This

bounding box can be formally represented as a set of coordinates:

$$b = \{b_x, b_y, b_w, b_h\}, \quad \text{where } 0 \leq b_x < W, 0 \leq b_y < H, 0 \leq b_w \leq W, 0 \leq b_h \leq H, \tag{3.1}$$

where b_x and b_y denote the coordinates of the box's centre, while b_w and b_h represent the width and height of the box, respectively, constrained by the image dimensions W (width) and H (height). In contrast, the classification component aims to assign a categorical label to the detected object. Let l denote this predicted class label for a single object of interest:

$$l \in L, \tag{3.2}$$

where L represents the set of all possible class labels defined during the training process. For instance, in the case of COCO, trained models predict 80 distinct labels, i.e., $L = \{l_1, l_2, \ldots, l_{80}\}$, while for Pascal VOC, the label set is smaller, with $L = \{l_1, l_2, \ldots, l_{20}\}$. Thus, for multi-label object detection, the predicted output for a given image can be defined as a set of detections (O), whereby each tuple in the set consists of a bounding box and a class label:

$$O = \{(b_1, l_1), (b_2, l_2), \ldots, (b_i, l_i)\}_{i=1}^{N}, \quad \text{where } N \in \mathbb{N}. \tag{3.3}$$

Here, N represents the total number of detections (or predictions) made for the given image, which is typically determined by the model's output layer and the number of objects detected in the scene.

After establishing the output, the next step is to focus on the predicting function $f(x)$, where x represents the input image fed into the network. In the context of object detection, this prediction can be split into two tasks: predicting the bounding box coordinates through a regression process and predicting the class label through a classification task. Therefore, the prediction function can be expressed as:

$$f(x) = r(x) \oplus c(x). \tag{3.4}$$

In this formulation, $r(x)$ the regression component is responsible for estimating the bounding box coordinates, while $c(x)$ the classification component is responsible for assigning a class label to the image. The operator "$\oplus$" denotes set element addition, combining the outputs from both the regression and classification components to form the final output set, denoted by O.

The object detection task, as formulated in the LUPI paradigm, incorporates both standard input data (x) and privileged information (x^*), where the latter is only available during training. This combination enhances the model's performance by providing additional context. Formally, the object detection problem is defined using a training set

consisting of triplets, as follows:

$$\mathcal{D}_{\text{train}} = (x_i, x_i^*, y_i)_{i=1}^N, \quad x_i \in X, x_i^* \in X^*, y_i \in Y. \tag{3.5}$$

In this formulation, X represents the space of input images, X^* denotes the space of privileged information instances, and Y comprises the space of bounding boxes with their associated class labels. Given a teacher model defined as:

$$f_{\text{teacher}} : X \cup X^* \rightarrow Y, \tag{3.6}$$

which accurately predicts y based on both x and x^*, the objective is to develop a student model:

$$f_{\text{student}} : X \rightarrow Y, \tag{3.7}$$

that effectively maps X to Y by leveraging not only the intrinsic information in X, but also the knowledge encoded within f_{teacher}. In other words, during training, f_{student} learns to map X to Y through knowledge distillation from f_{teacher} and the information contained in the labelled examples from the training set $\mathcal{D}_{\text{train}}$.

3.3 Proposed Approach

Given the problem definition, the proposed approach aims to integrate LUPI into object detection. To this end, both the f_{teacher} and f_{student} models are implemented as deep neural networks, each consisting of $\mathcal{L}$ layers. These models are represented as a series of transformations:

$$f_{\text{teacher}} = f_1^{(t)} \circ f_2^{(t)} \circ \cdots \circ f_l^{(t)} \circ \left\{ \begin{array}{l} f_r^{(t)} \circ f_{r+1}^{(t)} \circ \cdots \circ f_{r+k}^{(t)} \\ f_c^{(t)} \circ f_{c+1}^{(t)} \circ \cdots \circ f_{c+m}^{(t)} \end{array} \right\} \circ f_{\mathcal{L}}^{(t)}, \tag{3.8}$$

$$f_{\text{student}} = f_1^{(s)} \circ f_2^{(s)} \circ \cdots \circ f_l^{(s)} \circ \left\{ \begin{array}{l} f_r^{(s)} \circ f_{r+1}^{(s)} \circ \cdots \circ f_{r+k}^{(s)} \\ f_c^{(s)} \circ f_{c+1}^{(s)} \circ \cdots \circ f_{c+m}^{(s)} \end{array} \right\} \circ f_{\mathcal{L}}^{(s)}, \tag{3.9}$$

where "$\circ$" denotes function composition, and $f_i^{(t)}$ and $f_i^{(s)}$ represent the i-th transformation layer in the teacher and student networks, respectively. The l-th layer of both networks, $f_l^{(t)}$ and $f_l^{(s)}$, is constrained to have identical dimensionality, ensuring that the shared latent representation is consistent across both models. This shared representation serves as the foundation for deriving both the regression and classification outputs. Specifically, the regression branch, consisting of layers $f_r^{(s)}, f_{r+1}^{(s)}, \ldots, f_{r+k}^{(s)}$, predicts the bounding box coordinates, while the classification branch, consisting of layers $f_c^{(s)}, f_{c+1}^{(s)}, \ldots, f_{c+m}^{(s)}$, assigns class labels to the detected objects. The results from both branches are then combined at the final layer $f_{\mathcal{L}}^{(s)}$ to form the final detection output (O).

To enable knowledge transfer between the models, a distance function $D(f_l^{(t)}, f_l^{(s)})$ is minimised, thereby aligning the internal representations of the teacher and student at layer l. For each training triplet $(x_i, x_i^*, y_i) \in \mathcal{D}_{train}$, where x_i denotes the standard input, x_i^* denotes the privileged information, and y_i is the target label, the teacher computes a more expressive latent representation due to access to both x_i and x_i^*. The student, restricted to x_i, is guided to approximate this representation via distillation.

The student network is optimised using a composite loss function (L_S) that incorporates both supervised learning and representational distillation. The overall loss is defined as:

$$L_S = (1 - \alpha) \cdot L(f_{\text{student}}(x, y)) + \alpha \cdot D(f_i^{(t)}, f_i^{(s)}), \tag{3.10}$$

where $L(f_{\text{student}}(x, y))$, represents the standard supervised loss for the student network, which encompasses both regression and classification tasks. The term $D(f_i^{(t)}, f_i^{(s)})$ denotes the distillation loss, measuring the difference between the teacher and student networks' internal representations at the i-th layer. The scalar parameter $\alpha \in [0, 1]$ controls the balance between standard supervision and distillation. Specifically, α determines how much the student network relies on the teacher's internal representation as opposed to learning from labelled data. When $\alpha = 0$, the student learns solely from labelled data, while $\alpha = 1$ implies that the student depends entirely on the teacher's internal representations for guidance.

To compute the dissimilarity between the latent feature representations of the student and teacher (D), the cosine distance function is employed:

$$D(f_i^{(t)}, f_i^{(s)}) = 1 - \frac{\sum_{j=1}^{d} f_{i,j}^{(t)} f_{i,j}^{(s)}}{\sqrt{\sum_{j=1}^{d} (f_{i,j}^{(t)})^2} \sqrt{\sum_{j=1}^{d} (f_{i,j}^{(s)})^2}}. \tag{3.11}$$

Here, $f_i^{(t)}$ and $f_i^{(s)}$ represent the feature vectors of the teacher and student networks at layer i, each of length d, with j indexing the components of the vectors. Here, the cosine distance captures the angular difference between these representations. By encouraging alignment at a shared intermediate layer l, the student can benefit from the privileged information available during training, even though only the input x is accessible at inference.

Given this general formulation of how we integrated LUPI within the context of object detection, the next step, in alignment with the stated objectives, is to delve into the nature and role of the privileged information itself.

3.4 Privileged Information for Object Detection

Determining the appropriate form for privileged information that can be effectively used for object detection is challenging. Identifying the relevant information that enables detection networks to achieve improved results requires extensive research. Recent studies have delved into similar aspects of this issue, aiming to understand how humans detect objects in a scene [138]. In [138], it was highlighted that *physical reasoning*, such as the *centre of mass*, is a key factor humans rely on when locating objects. This suggests that physical reasoning and vision are intertwined, rather than one process solely depending on the other. The relationship between the two is sufficiently strong that physical reasoning can predict the location of objects more effectively than processes like spatial memory and attention. In related studies in deep learning, saliency [139, 140] and depth prediction [141, 142] outputs were analysed to determine which sources most strongly correlate with object detection [56]. The results showed that saliency prediction had a stronger correlation with the object detection problem [56]. Building on these findings, a preliminary experiment was conducted to assess several renowned saliency methods, including the Itti [139] and DeepGaze IIE [140] models, as well as popular depth prediction models such as Depth Anything [141] and DPT-Large [142]. Figures A.1 and A.2 illustrate the findings. However, these single-channel privileged information sources did not prove to be particularly successful in improving detection performance.

To further investigate the types of privileged information that could be beneficial in this regard, inspiration was drawn from psychological concepts related to human perception [143, 144]. Specifically, the spotlight principle was investigated in the context of identifying privileged information that could potentially improve detection accuracy [144]. According to [144], spatially directed attention significantly improves visual perceptual processing. Drawing from this attention spotlight principle in the human visual cortex [144], this study explored how a similar concept might be applied to object detection, albeit in a different format. The underlying idea is straightforward: *given an image, how can an object detection network be guided to focus on areas containing the objects of interest?* Although the network already receives guidance through the ground-truth bounding boxes during training, the study considered whether an additional channel could be integrated to streamline and aid the search process. It is within this context that this study proposes a bounding box mask as a potential solution.

The bounding box mask image proposed in this study serves as a structured form of privileged information, developed to support object detection networks across both localisation and classification tasks. The generated image consists of a black background, representing non-object regions. Over this, bounding boxes are drawn to indicate the locations of objects, thereby addressing the localisation problem. To simultaneously encode class-specific information, each bounding box is filled with a distinct shade of grey

(a) (b)

Figure 3.1 Visual illustration of generated privileged information: (a) a three-channel RGB image from the SODA dataset [13], and (b) the corresponding generated privileged information represented by a single-channel bounding box mask image.

corresponding to its class label in order to tackle the classification problem. This approach is visually exemplified in Figure 3.1.

Moreover, a single-channel grayscale format in the range $[0, 255]$ was used to represent the generated privileged information image. This kept the representation simple, preserving essential structure and meaning, without adding extra computational load. This single-channel format was selected in preference to RGB representations, as it simplifies the data without compromising the richness of the information conveyed. The algorithm used to generate this form of privileged input is presented in Algorithm 1. Special attention was given to reducing object occlusion within the privileged mask. To achieve this, a sorting step was introduced before rendering the bounding boxes onto the black mask image. The boxes were ordered in descending size, allowing the larger ones to be drawn first. This approach helped to reduce the likelihood of smaller objects being obscured.

Algorithm 1 Generating Bounding Box Mask (Privileged Information) for Object Detection

Input: Bounding boxes, labels, image width, height, number of classes
Output: Single-channel grayscale bounding box mask
1. Initialise a blank grayscale image of size $(\text{width}, \text{height})$
2. Load annotations: bounding boxes and class labels
3. Assign distinct grayscale values in range $[55, 255]$ to each class label to avoid overlap
4. Sort annotations by bounding box area (descending order) to reduce occlusion

for each annotation i **do**
   5. Extract bounding box coordinates (b_x, b_y, b_w, b_h) and label l
   6. Retrieve grayscale value corresponding to l
   7. Fill the box region in the image with this value

end for
8. Save as a single-channel grayscale mask image

In addition to this, polygon or segmentation masks are often regarded in the literature [128, 131, 133] as offering greater precision in object representation. However, in this study, the bounding box mask was preferred over these alternatives for two principal reasons. The first is that the majority of object detection datasets are annotated using bounding boxes rather than polygon formats. Consequently, obtaining a polygon mask from a bounding box would require the use of a segmentation model such as [80, 122, 145, 146] to predict the polygon mask. Such a prediction cannot be regarded as ground truth and would therefore introduce an additional error surface.

The second reason concerns the control of the error margin. The aim was to improve object detection accuracy without adding unnecessary complexity by introducing shape details that could confuse the model. Preliminary experiments with polygon masks indicated further challenges, particularly in handling overlapping objects. Since these were not represented as squares, identifying the object's corners and thereby locating its centre proved difficult. Nevertheless, the potential of polygon or segmentation masks as privileged information for object detection merits further investigation in future studies.

3.5 Deep Learning-Based Object Detection Architectures

A structured methodological framework was devised to investigate the influence of LUPI in object detection and assess whether it provides measurable improvements in accuracy, without necessitating increasingly complex or computationally demanding architectures, to address the first objective (**O1**). This framework relied on utilising pre-trained models with few structural adjustments, as briefly hinted in the preceding sections, thereby presenting a methodology capable of accommodating the integration of LUPI into existing detection pipelines to tackle the second objective (**O2**). To ensure scientific rigour and relevance, a selection of renowned models was employed to test this hypothesis–models that feature prominently in object detection literature and are also commonly used in litter detection (see Section 2.4). Five architectures were selected for their distinct algorithmic principles and practical relevance: Faster R-CNN [76], SSD [70], RetinaNet [71], SSDLite [72], and FCOS [75]. All of these models are open-source and publicly available through the `torchvision`[1] library, with pre-trained weights sourced from the COCO dataset. Furthermore, these pre-trained configurations provided a uniform baseline for all subsequent experimental comparisons.

Although the study drew upon the full suite of pre-trained object detection models offered through the `torchvision` Application Programming Interface (API), these five were selected particularly for their representational breadth and relevance across differ-

[1] https://pytorch.org/vision/main/models.html#object-detection

ing detection paradigms. Together, these models cover both one-stage and two-stage detection paradigms, as well as lightweight and computationally intensive architectures, providing a balanced basis for assessing the contribution of LUPI. Their selection ensured that any findings are not confined to the characteristics of a single architecture but instead speak to broader trends observable across different model types.

3.6 Learning Using Privileged Information for Object Detection

Having established the architectures and defined the privileged information for this task, the next stage involved integrating this information into each model through a unified training pipeline. This is done by adapting each architecture to support a teacher–student training setup. In this pipeline, the teacher model is trained using RGB images (x), privileged information (x^*), and the corresponding labels (y). The student model, sharing the same architecture, is trained only with RGB images and labels, mimicking the input conditions expected at deployment. This setup ensures that the student does not rely on any modality unavailable at test time.

Knowledge transfer between the teacher and student models is performed through distillation, which allows the student to learn from the teacher's enriched representations derived from bounding box mask privileged information. To ensure methodological consistency and isolate the effect of privileged information, both the teacher and student models employ identical architectures. This approach simplifies the alignment of intermediate features or output logits during training and eliminates architectural differences as a confounding factor. Consequently, any observed improvements in detection performance can be confidently attributed to the incorporation of privileged information rather than variations in model design. For example, a RetinaNet teacher model is exclusively paired with a RetinaNet student model, avoiding cross-architecture training scenarios such as pairing a Faster R-CNN student with a RetinaNet teacher.

3.6.1 Teacher Network

To incorporate privileged information, the teacher network was modified to accept both the RGB image and its corresponding bounding box mask. Since conventional models typically process three-channel RGB inputs, the input layer was adapted to accept a four-channel tensor, combining the RGB data with the single-channel mask. More generally, the architecture was extended to handle inputs with $\mathcal{N}$ channels, accommodating additional privileged sources. This modification affected the initial convolutional layer, requiring reinitialisation of its weights. To maintain consistency with pre-trained param-

eters and ensure stable training, Kaiming Normal initialisation [147] was applied to the adjusted layer.

3.6.2 Student Network

With the teacher network established, the subsequent step involved defining the student network. While the student learns under the supervision of the teacher, it does not receive any privileged information as input. Consequently, the input layer of the student network remained unaltered, allowing the direct use of a standard pre-trained object detection model. To incorporate teacher supervision during training, however, the student's loss function had to be adjusted to include a distillation term, as described in Equation 3.10.

To enable meaningful feature-level distillation, this approach required the selection of a shared internal layer from both networks, one that captures semantically rich representations and allows for direct comparison. In this study, the final backbone layer was selected for distillation, in line with existing approaches in this area [102, 103, 108]. Specifically, for architectures such as Faster R-CNN, RetinaNet, and FCOS, which rely on the ResNet-FPN backbone, this corresponded to the final convolutional layer preceding the FPN. The feature layer used for distillation in SSD, and SSDLite was taken just before the auxiliary detection heads, corresponding to the final layer of the VGG-16 and MobileNetv3 backbones, respectively.

While previous studies have commonly employed the final backbone layer for knowledge transfer, it is important to acknowledge that this choice may not be optimal across all network architectures. The varying designs of FPNs and detection heads introduce complexities that might be better addressed by distillation applied at multiple levels or different points within the model. Intermediate layers, for example, may contain distinct or complementary features that the final backbone layer does not capture, potentially bolstering the knowledge transfer process. Due to practical limitations and to preserve consistency across experiments, this investigation did not evaluate alternative distillation layers or experiment with different architectural points for knowledge transfer. Nonetheless, identifying the most suitable layer or combination of layers for knowledge transfer in each architecture represents a valuable avenue for future investigation. Given the variability in architectural design, their response to distillation may differ, and further research could refine and improve the current approach.

Once the appropriate layer was selected, the corresponding latent feature representations were extracted during training from both the student and the teacher. These outputs were then flattened into vectors, allowing for a similarity comparison using the Cosine Distance (D) function. This metric quantifies the angular divergence between the two feature vectors, offering a precise scalar measure of their alignment in feature space.

The resulting distance value was then integrated into the student's total loss function. As outlined in Equation 3.10, an α parameter controls the influence of this distillation loss, balancing the learning from direct supervision against the soft guidance offered by the teacher.

3.6.3 Proposed Architecture

Bringing together all the elements described in this section, the incorporation of privileged information into any object detection model can be formally represented through the architecture depicted in Figure 3.2. This method rests on two core alterations: adapting the teacher network to process both RGB images and the corresponding privileged input and introducing a loss adjustment in the student model to enable distillation. These adjustments are deliberately kept minimal to prevent unnecessary architectural complexity or increased computational demands. The required changes are confined to modifying the input layer of the teacher model and integrating a distillation mechanism during the student training. Moreover, generating the privileged information, such as

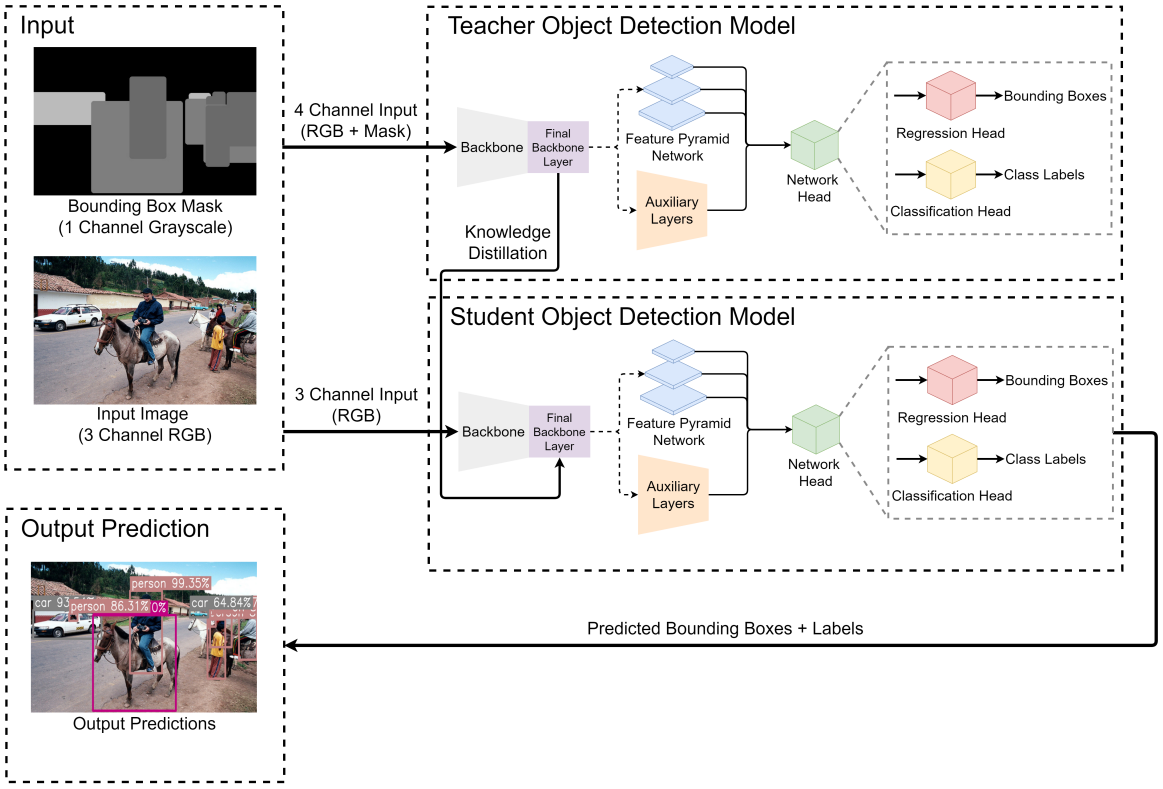


Figure 3.2 General architecture for integrating the LUPI paradigm into any object detection model. The diagram illustrates the incorporation of both RGB images and bounding box masks as inputs to the teacher network, the use of a standard RGB input for the student network, the selection of the final backbone layer for knowledge distillation, and the generation of output predictions from the student model.

bounding box masks, remains a simple and direct process.

Training an optimised detection model in this study began with the teacher network, which was configured to learn from a richer four-channel input composed of RGB data and privileged information. Once the teacher was trained, its parameters were frozen, and it served as a guide for the student model. During this training phase, the student processed only the standard RGB images, ensuring no modifications to the existing detection architecture while strictly preventing any data leakage from privileged information into the student model or inference process. Feature representations were drawn from the final layer of the backbone in both networks. Using identical architectures for the teacher and student simplifies implementation, as it guarantees that the resulting feature vectors are dimensionally aligned and directly comparable.

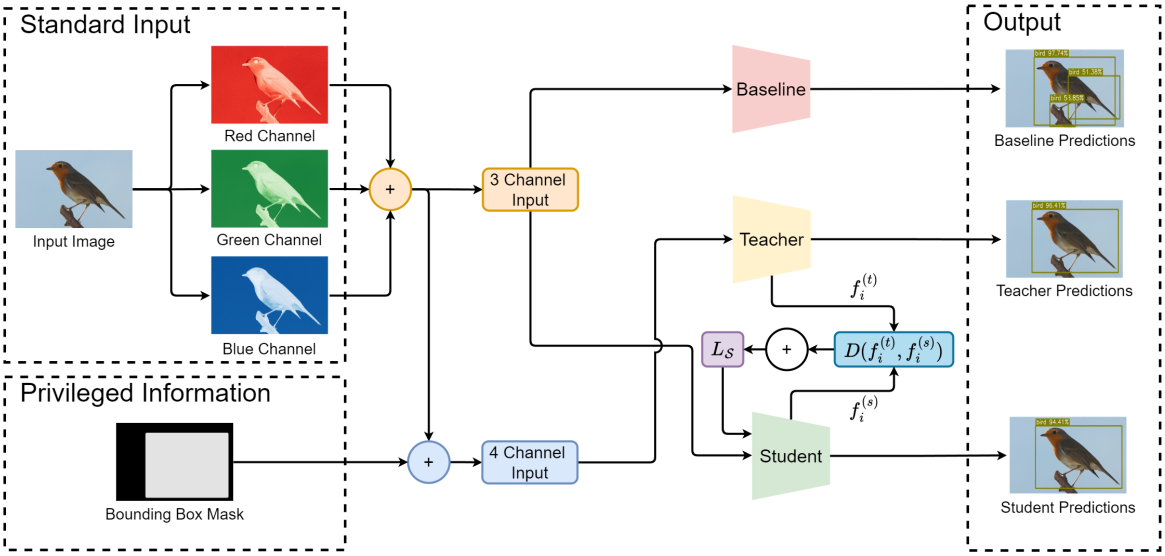


Figure 3.3 Detailed architecture of the training setup showing the teacher network receiving both RGB images and privileged input channels. The student network processes only RGB data but is trained with additional supervision through knowledge distillation. A baseline RGB-only model is included for comparison.

The similarity between the latent feature representations—obtained by compressing feature maps into continuous vectors—was measured using a cosine distance function (see Equation 3.11), producing a distillation loss that captures how closely the student aligns with the teacher. This loss was backpropagated through the student model alongside standard detection losses such as regression and classification, as well as additional losses specific to each architecture, including the FPN loss for Faster R-CNN and the centerness loss for FCOS. The influence of the teacher's guidance on the overall training objective was controlled by a tunable parameter, α, which determines the weight given to the distillation term. Once training was complete, the teacher model was no longer needed, as privileged information, such as bounding box ground truth,

cannot be accessed during inference when detections must be produced autonomously. Nevertheless, the student model, having learned under the teacher's supervision, retains the benefits of this privileged knowledge despite relying solely on standard RGB inputs at test time.

To complement the overview presented in Figure 3.2, Figure 3.3 offers a more detailed and structured illustration of the training framework. This expanded diagram does not revisit the mathematical details but clarifies the data flow and knowledge distillation process. It highlights that the teacher model receives a richer input, combining standard RGB channels with additional privileged information. Meanwhile, both teacher and student models operate in parallel, with the loss computation (refer to Equation 3.10) abstracted into a simplified and more interpretable visual form.

The baseline model, relying exclusively on standard RGB input, serves as a conventional point of reference within this framework. The emphasis remains on contrasting this baseline with the student model, which, although identical in architecture during inference, achieves improved performance through the additional supervision provided by the teacher during training.

3.7 Training Parameters

To evaluate the proposed methodology across the five selected object detection models, a uniform experimental training protocol was employed throughout, in alignment with the second objective (**O2**). The primary aim was to determine whether the integration of the LUPI framework could yield improvements in detection accuracy without imposing additional computational demands on the baseline architectures. To ensure that any observed improvements in performance could be attributed solely to the inclusion of LUPI, the training configuration was intentionally kept simple and consistent. Each model was trained for 100 epochs using the Adaptive Moment Estimation (Adam) optimiser [148], with a fixed learning rate of 1×10^{-3}. Preliminary tests compared alternative optimisers and learning rates [149]. Although the Stochastic Gradient Descent (SGD) optimiser was initially considered, Adam achieved comparable convergence to the same minimum but with greater efficiency. Various learning rates were also explored, including $0.1, 0.01, 0.001, 1\times10^{-3}$, and 1×10^{-4}. Among these, 1×10^{-3} consistently provided the most stable convergence across the selected models. In addition, an early stopping callback was applied based on validation loss, with a patience threshold of eight epochs. A model checkpointing mechanism was also used to retain the weights corresponding to the best-performing epoch, thereby minimising the risk of overfitting. Moreover, following standard practice when training object detection models for a given application or dataset [13, 67, 79, 150], all models were initialised with pre-trained weights from the

COCO dataset. The detection heads were subsequently adapted to match the number of target classes present in the specific dataset used during training.

3.8 Pre-processing Steps

In addition to the training parameters, a consistent set of preprocessing steps was applied across all experiments to maintain comparability of results. These transformations applied to the input images of the detection networks include image normalisation, resizing, and per-channel standardisation, each of which is outlined in detail below.

3.8.1 Min-Max Normalisation

The initial step involves normalising the image pixels to a uniform scale. This is done to ensure equal weighting across all image inputs and channels. Both the RGB inputs and the additional privileged information channels are scaled using Min-Max normalisation [151], as described below:

$$I_{\text{norm}} = \frac{I - I_{\text{min}}}{I_{\text{max}} - I_{\text{min}}}. \tag{3.12}$$

Here, I refers to the original image, I_{min} and I_{max} are the minimum and maximum pixel intensity values computed across the entire dataset for each respective channel. The inputs are scaled to lie within the range $[0, 1]$, thereby normalising the input space for both the teacher and student networks. Crucially, normalisation is performed separately for the RGB images and the privileged information channels to maintain the distinct statistical properties of each modality.

3.8.2 Image Resizing

Following normalisation, all images are resized to a fixed resolution of 800×800 pixels. This resizing step allows for the formation of input tensors with consistent dimensions, a prerequisite for batch training in CNNs. The resolution of 800 pixels was selected as a compromise across the models employed in this study. Although models such as SSD and SSDLite typically operate on smaller input resolutions (e.g., 300×300 or 320×320), detectors like Faster R-CNN, RetinaNet, and FCOS adopt a minimum resize of 800 pixels. The choice of a larger image resolution also contributes to preserving fine-grained features, which is especially relevant for detecting small-scale objects that might otherwise lose distinctive characteristics if downsampled excessively.

3.8.3 Channel-Wise Standardisation

The final step in the pre-processing pipeline involves statistical normalisation by subtracting the channel-wise mean and dividing by the standard deviation. These parameters are computed over the training dataset and are applied individually to each channel, including both RGB and privileged input. This transformation ensures zero-mean and unit-variance inputs, which helps stabilise training and accelerate convergence [151]. This standardisation is carried out automatically by the model during training, based on mean and standard deviation values calculated from the specified dataset. The computed mean and standard deviation values for each training dataset and input channel are summarised in Table 3.1, with the corresponding datasets explored in Section 3.10.

Dataset	Channel	Mean	Standard Deviation
SODA Litter (1-Metre Altitude)	Red	0.434	0.261
	Green	0.406	0.249
	Blue	0.320	0.238
	Bounding Box Mask	0.144	0.135
SODA Litter (All Altitudes)	Red	0.467	0.255
	Green	0.430	0.240
	Blue	0.357	0.233
	Bounding Box Mask	0.021	0.129
Pascal VOC 2012	Red	0.452	0.275
	Green	0.431	0.273
	Blue	0.399	0.284
	Bounding Box Mask	0.142	0.216

Table 3.1 Channel-wise mean and standard deviation values for the Red, Green, Blue, and Bounding Box Mask channels, computed over the training sets of the Pascal VOC 2012 and SODA Litter datasets.

Overall, these pre-processing transformations were kept deliberately minimal, consistent, and aligned with the general standards in object detection research to ensure that the influence of the LUPI methodology could be fairly and transparently evaluated without extraneous confounding factors.

3.9 Post-processing Steps

In addition to the pre-processing techniques applied to the input images, a consistent post-processing step was introduced to ensure uniformity across all five selected models. This step employed non-maximum suppression [152], a widely used method for refining object detection results by eliminating redundant predictions. The suppression

process ranks all predicted bounding boxes according to their confidence scores. When two or more boxes significantly overlap, only the one with the highest score is retained. In this study, the significant overlap was defined using an IoU threshold of 0.5. Any box exceeding this level of overlap with a higher-scoring prediction was removed by the NMS. Applying this method consistently was essential for a fair evaluation. Some of the chosen models already had NMS integrated internally, while others required an external implementation. By enforcing a common post-processing step, the outputs became directly comparable, regardless of any differences in model architecture.

3.10 Datasets

The object detection problem, which is regarded as a supervised learning task, necessitates the use of datasets for both training and validating detection models. These datasets typically consist of image sequences annotated with ground truth bounding boxes and class labels, specifying the type and location of each object [25, 137]. To evaluate the proposed methodological architecture in alignment with the third objective (**O3**), UAV-based litter detection was chosen as the target problem to assess the feasibility of the proposed approach. This problem presents a considerable challenge due to the need to detect small objects in varied terrain and under differing altitudes [13]. Moreover, the complexity of this task makes it a suitable benchmark for evaluating both the generalisability and precision of the proposed framework.

Currently, only four UAV-based litter detection datasets are available (see Table 2.1): BDW [17], UAVVaste [19], HAIDA [124], and SODA [13]. Among these datasets, BDW[2], UAVVaste[3], and SODA[4] are accessible through the Roboflow dataset annotation platform. Although the HAIDA dataset is publicly available on Github[5], it could not be used due to inaccessible annotations. As a result, the evaluation focused on the three datasets available through Roboflow. In addition, the well-established Pascal VOC 2012 dataset [25], also accessible on Roboflow[6], was included to give a broader perspective. It was chosen to test the proposed approach for multi-label detection across a larger variety of classes.

3.10.1 SODA: Small Objects at Different Altitudes

As discussed in Subsection 2.4.15, SODA serves as a representative UAV-based litter detection dataset. It includes six litter classes and contains image subsets captured at

[2]https://universe.roboflow.com/bottles-in-the-wild
[3]https://universe.roboflow.com/mcast/uavvaste-avcle
[4]https://universe.roboflow.com/soda-dataset/
[5]https://github.com/LiaoSteve/HAIDA-Trash-Dataset-High-Resolution-Aerial-image
[6]https://public.roboflow.com/object-detection/pascal-voc-2012/1

a range of altitudes, which reflects more realistic operating conditions for UAV-based detection. However, as shown in Table 3.2, one of the key issues with this dataset is class imbalance. Certain categories, such as *drink can* and *clear plastic bottle*, contain significantly more annotations than others. There is also an uneven distribution of images across altitudes, with noticeably fewer samples at higher elevations. Despite these limitations, SODA remains a valuable dataset in this area of research. Among the three datasets selected in this study, SODA covers the widest range of altitudes.

AGL (m)	Total Images	Total Annotations	Drink Can	Clear Plastic Bottle	Glass Bottle	Other Plastic Bottle	Drink Carton	Glass Jar
1	452	712	256	180	96	71	54	55
5	202	1839	837	429	183	152	136	102
10	50	1138	491	268	128	106	88	57
15	35	956	434	232	93	87	67	43
20	30	763	324	199	68	79	57	36
25	30	709	327	174	46	79	57	26
30	30	602	270	149	36	82	55	10
Total (Incl. 1m)	**829**	**6719**	**2939**	**1631**	**650**	**656**	**514**	**329**
Total (Excl. 1m)	**377**	**6007**	**2683**	**1451**	**554**	**585**	**460**	**274**

Table 3.2 Summary of annotated objects across different altitudes AGL in the SODA dataset. (Source: [13])

In the original publications that introduced the SODA dataset and used it for training detection models [13, 20], the authors adopted a tiling technique [100, 153]. This method aimed to improve the detection of small objects, particularly in images taken at higher altitudes, by magnifying specific regions. In [20], they applied a 5×5 tiling strategy, resizing each tile to 640×640 pixels to match the input dimensions required by the YOLOv8 [130] network employed in their study.

To understand why such an approach becomes necessary, visual exploration of the data proves useful. Figure 3.4 illustrates this clearly. Subfigure (a) presents an annotated image captured at 15 metres altitude, which is half the maximum altitude in the dataset. If the entire image is resized directly to 640×640 pixels without tiling, as shown in subfigure (c), the result is a significant loss in feature detail. This version represents the full image as it would be input to the detection network in one pass. Applying tiling, shown in subfigure (b), divides the original image into smaller sections. Taking the 13^{th} tile–resized and displayed in subfigure (d)–as an example, it shows litter with greater clarity and scale. This makes detection of litter from UAV footage easier. A side-by-side comparison with the corresponding region from the non-tiled image (subfigure (e)) highlights the reduction in granularity that occurs without tiling. However, this improved resolution comes at the cost of longer inference times [153], since the model must process several images rather than one. As a result, selecting appropriate tiling parameters

(a)

(b)

(c)

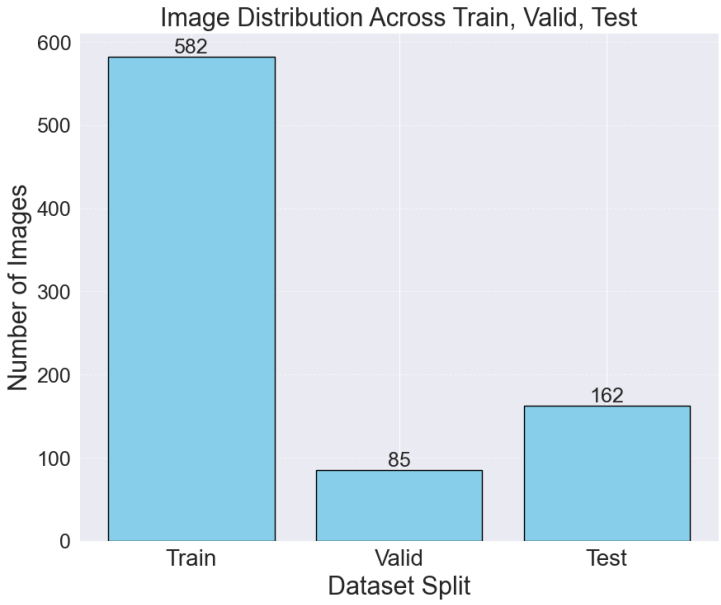
(d)

(e)

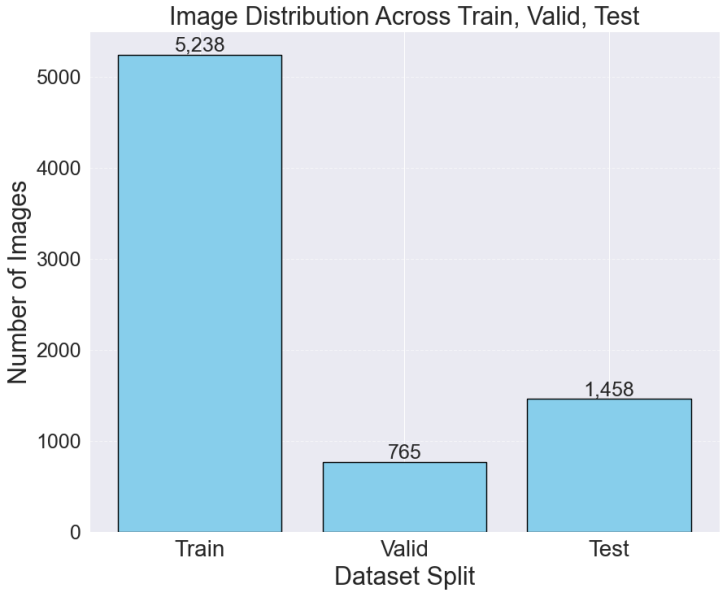
Figure 3.4 Visual illustration highlighting the need for tiling. (a) Annotated image of litter captured at 15 metres altitude from the SODA dataset [13]; (b) 5×5 tiling of the original image, with each tile resized to 640×640 pixels; (c) original image resized directly to 640×640 pixels; (d) the 13th tile from the tiling approach, resized; (e) the corresponding region extracted from the non-tiled image.

involves balancing detection accuracy with computational efficiency.

For this study, a 3×3 grid was used to tile the SODA dataset. This choice is supported by the experimental findings discussed in Section 4.4. Additionally, a resize dimension of 800 pixels was applied, based on the rationale presented in Subsection 3.8.2. No data augmentation techniques were introduced to maintain simplicity and ensure that any observed improvements could be attributed directly to the proposed methodology. As a result, the total number of images and annotations increased sub-



(a)

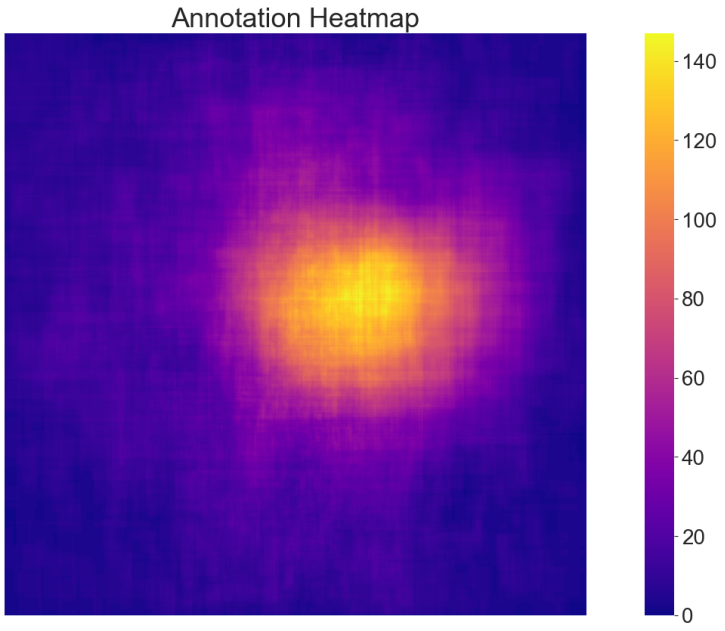


(b)

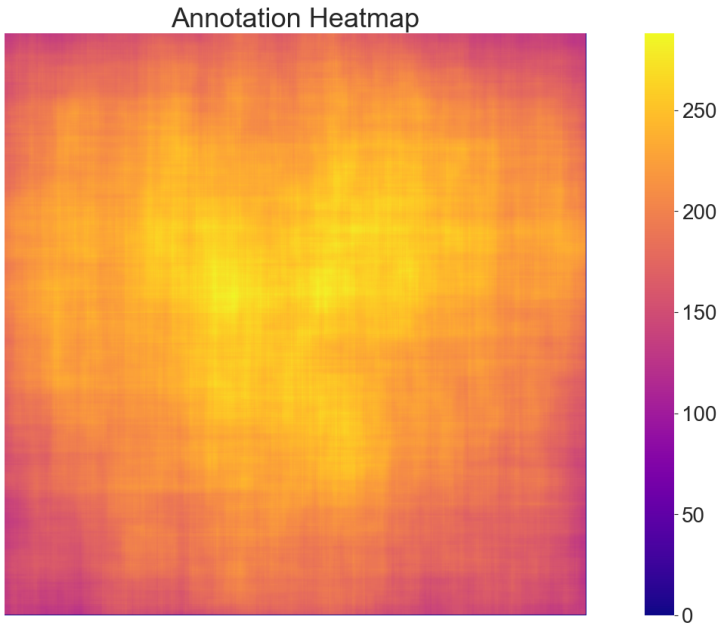
Figure 3.5 SODA dataset image distribution: (a) before 3×3 tiling, and (b) after 3×3 tiling.

stantially. The original version of the SODA dataset contained 829 images. After tiling, this figure rose to 7,461. The image distribution across the category split is presented in Figure 3.5. Notably, despite the application of these pre-processing steps, the original ratio between the training, validation, and testing sets remains consistent before and after tiling.

Additionally, with the use of tiling, the spatial distribution of object annotations



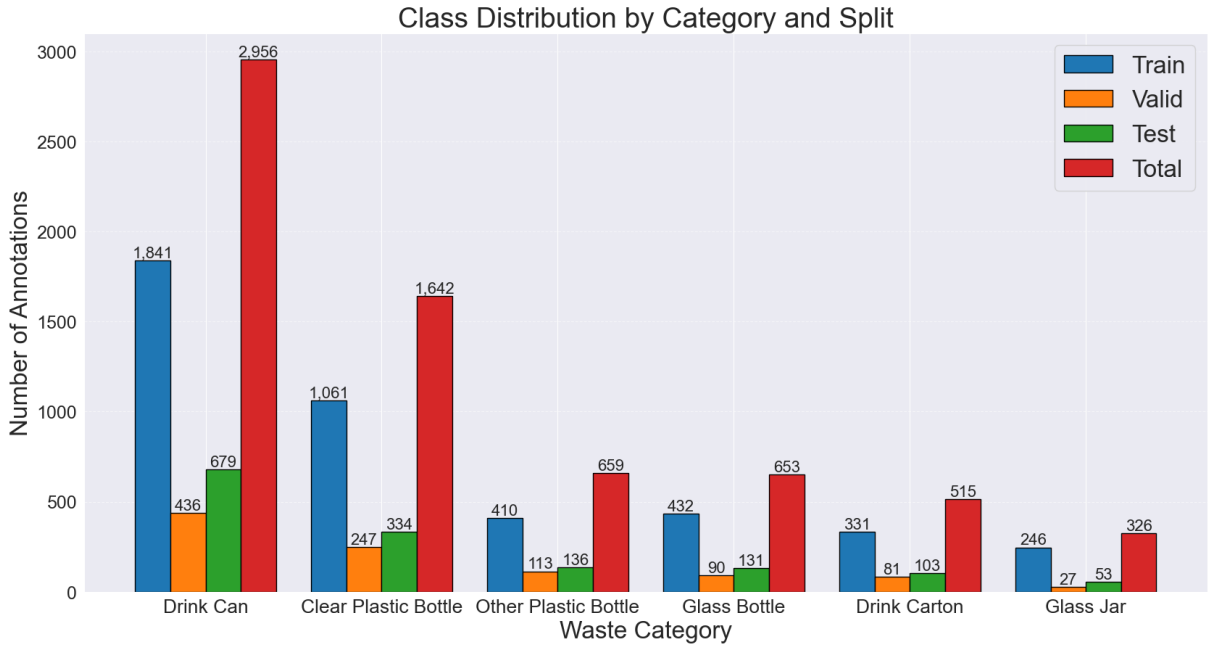
(a)



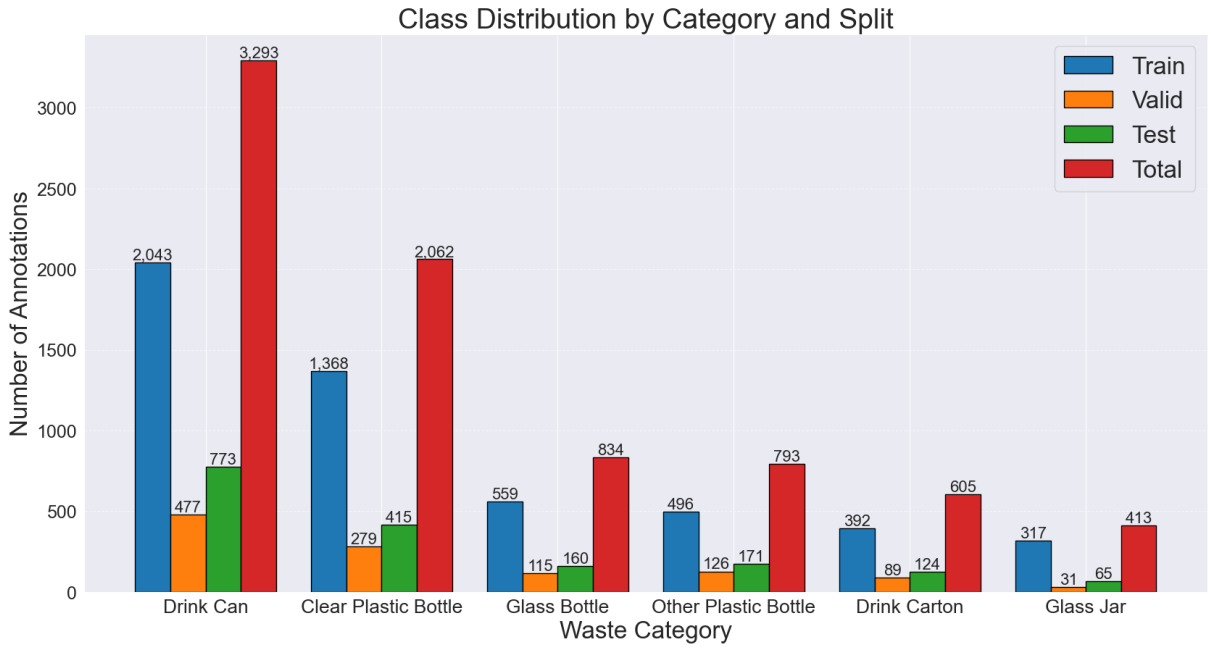
(b)

Figure 3.6 SODA dataset annotation heatmap: (a) before 3×3 tiling, and (b) after 3×3 tiling.

changed significantly. As the number of images increased, the objects became larger and more defined, helping to mitigate the annotation bias present in the original SODA dataset. This is illustrated in Figure 3.6. In subfigure (a), the heatmap before tiling shows a strong centre bias. After tiling, shown in subfigure (b), this bias is less pronounced, with a more even distribution across the heatmap, though some focus on the centre remains. The pixel intensities also increased due to the larger number of images and annotations.



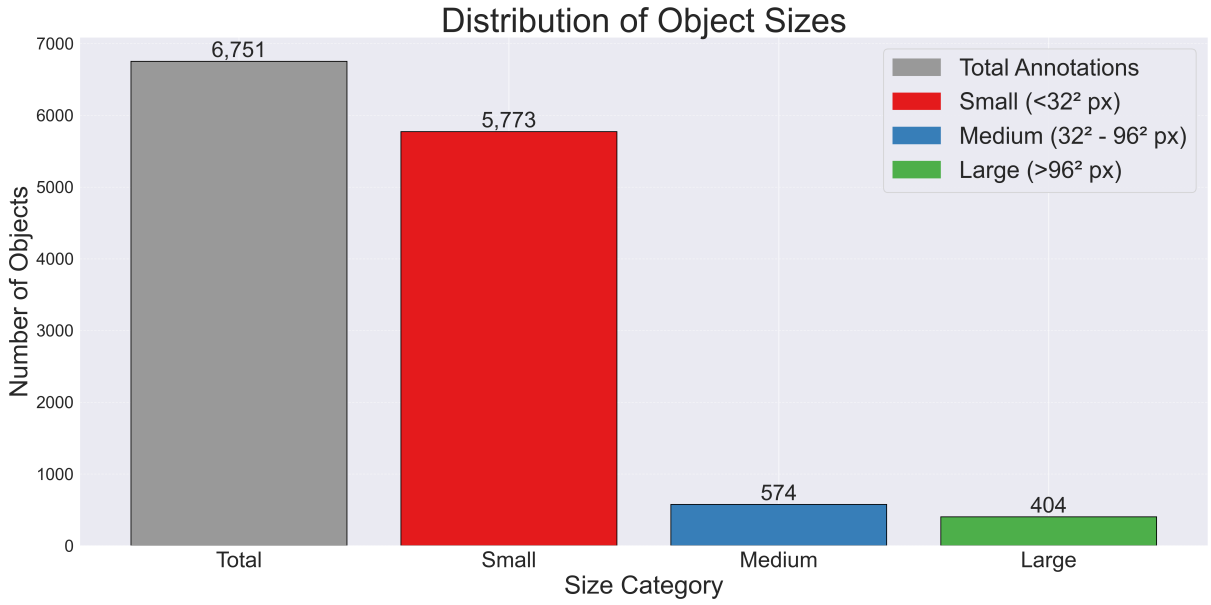
(a)



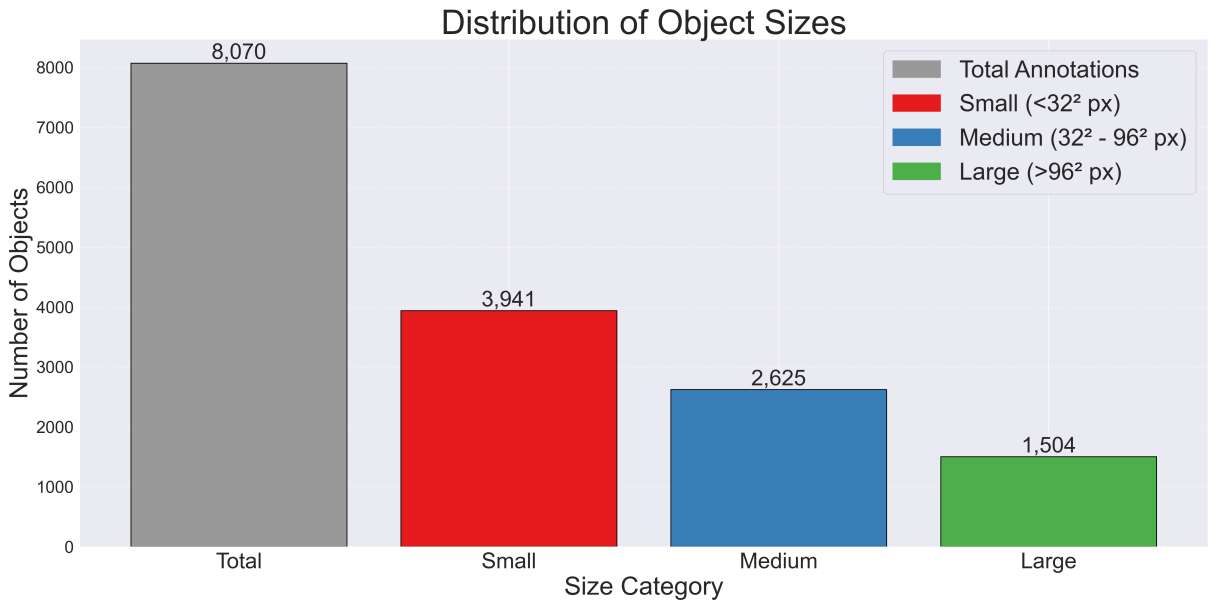
(b)

Figure 3.7 SODA dataset annotation distribution per category and split: (a) before 3×3 tiling, and (b) after 3×3 tiling.

Tiling unintentionally worsened the class imbalance. As images were split into smaller sections, some object bounding boxes were also divided. This was especially the case for objects near the edges or those covering a larger area. Common categories like *drink can* and *clear plastic bottle*, which were already more frequent, appeared even more often after tiling. This shift is reflected in the category distribution shown in Figure 3.7. Meanwhile, less frequent classes such as *glass jar* and *drink carton* showed a marginal increase in representation. Interestingly, *glass bottle* surpassed Other *plastic bottle* in the number of annotations, reversing their relative frequencies before tiling.



(a)



(b)

Figure 3.8 SODA dataset object size distribution: (a) before 3×3 tiling, and (b) after 3×3 tiling.

The increase in annotations due to tiling has had a significant impact on the overall distribution of object sizes. Using the object size classifications from the COCO dataset [137], where objects are defined as small (less than 32×32 pixels), medium (ranging from 32×32 to 96×96 pixels), and large (greater than 96×96 pixels), a clear shift in object size distribution can be observed. As shown in Figure 3.8, while the total number of objects has increased, the proportion of small objects has decreased, with a corresponding rise in both medium and large objects. This shift in distribution suggests an improvement in the dataset's size balance, which should improve the detection of previously smaller objects. With the larger objects now more prominent, the detection models can benefit from improved size constraints, making these objects easier to detect.

In summary, tiling has proven to be quite beneficial in improving the detection of objects in the SODA dataset, particularly for smaller objects. While it helped reduce some of the bias in annotations, it also introduced new challenges, especially regarding class imbalance. This tiling setup has been utilised to magnify and address the issue of small litter detection for the chosen case study, which will be used in the experiments discussed in Section 4.5.

3.10.2 BDW: Bottle Detection in the Wild

The BDW dataset [17], mentioned in Subsection 2.4.1, contains 25,407 images taken at altitudes between 10 and 30 metres, but it focuses only on detecting bottles. For this study, this dataset will be used solely for evaluation, as it only covers the representation of bottles, thus limiting the generalisability of the results. Therefore, only the testing set, consisting of 5,078 images, will be used to evaluate the models. Although the dataset claims to include images captured at higher altitudes, the objects in these images, as shown in Figure 3.9, appear relatively large and clear. This suggests that an image-splitting technique may have been applied during data collection.

Figure 3.9 Sample images from the test subset of the BDW dataset, illustrating the general nature of the data. (Source: [17])

3.10.3 UAVVaste

UAVVaste [19], the final publicly available UAV-based litter detection dataset chosen for this study, and as discussed in Subsection 2.4.8, contains 772 images captured at unspecified low altitudes, with various forms of litter annotated under a single waste category. In this study, the dataset will be used exclusively for evaluation, as its focus on low-altitude imagery limits generalisation. Only the testing set, containing 77 images, will be used to evaluate the trained models. A visual inspection of the dataset, as shown in Figure 3.10, suggests the images were likely captured at altitudes between 5 and 10 metres AGL.

Figure 3.10 Sample images from the test subset of the UAVVaste dataset, illustrating the general nature of the data. (Source: [19])

3.10.4 Pascal Visual Object Classes

Pascal VOC is a widely used benchmark dataset in object detection, featuring 20 object categories including people, animals, vehicles, and various household items [25]. Fig-

Figure 3.11 Sample images from the Pascal VOC 2012 dataset, showcasing the diversity in object categories included in the dataset. (Source: [25])

ure 3.11 shows sample images that illustrate the diversity of the dataset. The dataset includes images annotated with object class labels, bounding boxes, and pixel-level segmentation masks, enabling its use across multiple computer vision tasks. In this study, the Pascal VOC 2012 dataset is used for both training and validation to test the proposed methodology beyond the context of UAV-based litter detection, per the fourth objective (**O4**). This helps assess whether the approach remains effective across a broader set of object classes.

The dataset utilised in this study, sourced from Roboflow, contains 17,112 images in total. Of these, 13,690 are assigned to the training set, and 3,422 are designated for validation. However, the Pascal VOC 2012 dataset also suffers from class imbalance, as shown in Figure 3.12. There is a disproportionately high number of instances labelled as *person* compared to other categories. While the remaining classes show smaller gaps among themselves, class imbalance is still present, with *bus* being the least represented category.

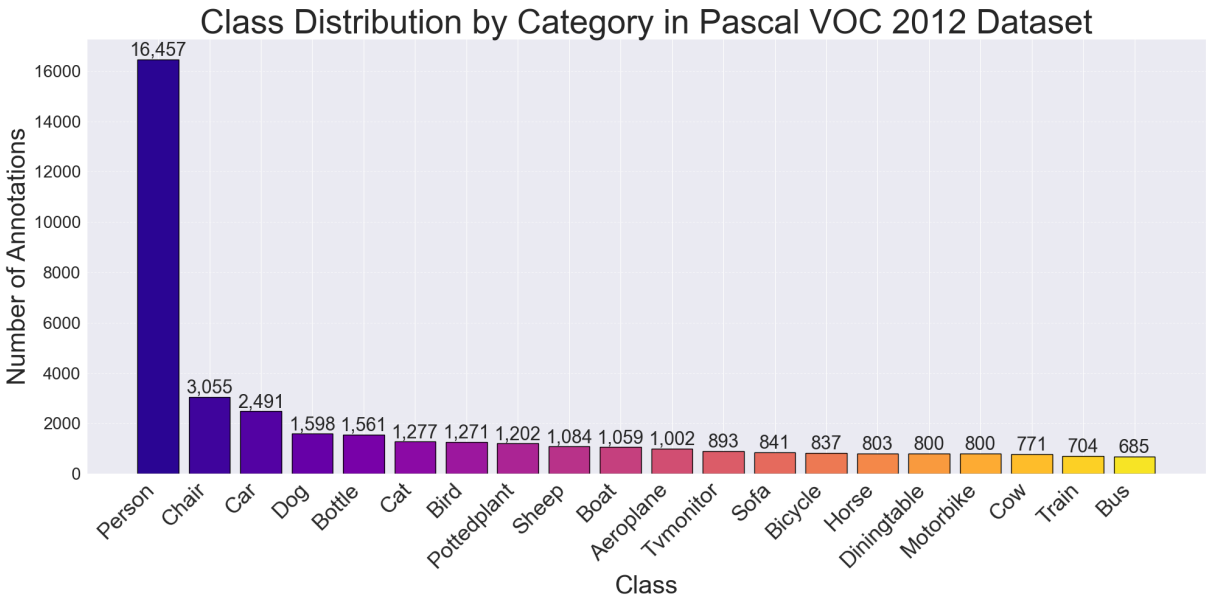


Figure 3.12 Class distribution of the Pascal VOC 2012 dataset, highlighting the imbalance across different categories, with the number of objects per class from both the training and validation splits.

3.11 Conclusion

This chapter has provided a structured framework for understanding the integration of LUPI in object detection, beginning with a formal mathematical formulation and the theoretical context surrounding the problem. It has established the conceptual basis for incorporating privileged information and defined its role within this context. The discussion then turned to the object detection models selected for this study, outlining the

modifications made to facilitate the construction of teacher and student networks. The implementation details, including training parameters and the consistent application of pre-processing and post-processing steps, were also covered. Furthermore, the chapter examined the datasets used in this research, highlighting the necessary pre-processing steps involved and analysing the structure of the data in relation to the proposed evaluation methodology.

Chapter 4 Evaluation

"It is not the strongest of the species that survive, nor the most intelligent, but the one most responsive to change."

– Charles Darwin

4.1 Introduction

This chapter begins by outlining the hardware setup and evaluation strategy that form the foundation for the experiments. It then proceeds to describe the optimal tiling experiments conducted on the SODA dataset, a necessary preliminary step to achieving the set objectives. The following section focuses on the within-dataset evaluation, specifically UAV-based litter detection, covering three experiments carried out on the SODA dataset. Next, the chapter moves on to the cross-dataset evaluation, where experiments on the BDW and UAVVaste datasets are presented to assess the generalisability of the trained models. An additional section examines the detection performance in relation to object size, with special focus on determining whether the proposed LUPI approach improves the recognition of objects at small scales. The subsequent section details the Pascal VOC 2012 evaluation, which tests the methodology's ability to perform multi-label detection across a large number of categories. This is followed by an ablation study on the alpha parameter, a visual comparison of model predictions across all evaluated datasets, and an analysis of the model's interpretability. The chapter then discusses the limitations encountered in generating effective privileged information to guide the model's internal feature representations. Finally, the chapter concludes with a synthesis of the results and a discussion on whether the research question was successfully addressed to the stated objectives.

4.2 Hardware Setup

All experiments involving deep learning models were conducted using two primary hardware configurations. Training was performed on systems equipped with an NVIDIA GeForce RTX 4070 and an RTX 4090 Graphics Processing Unit (GPU), both supporting Compute Unified Device Architecture (CUDA)-enabled acceleration. These setups were used across different experiments as needed, enabling parallel computation and

helping to reduce training times significantly. Furthermore, a batch size of 16 was consistently used across all experiments for training the deep learning models.

4.3 Evaluation Strategy

To accomplish the research objectives, a comprehensive evaluation strategy was devised to systematically assess the proposed methodology. The evaluation comprised a series of experiments using five distinct object detection architectures (refer to Section 3.5). For each architecture, both baseline models and their modified versions trained under the LUPI framework were compared. This comparison provided insight into the feasibility and adaptability of the proposed approach across diverse detection models.

The primary application domain for evaluation was UAV-based litter detection, a challenging problem due to the small size of target objects and the complexity of aerial imagery. Experiments leveraged several datasets outlined in Section 3.10. A within-dataset evaluation was conducted using different subsets of the SODA dataset to examine performance under controlled conditions. A cross-dataset evaluation was also carried out on the BDW and UAVVaste datasets to test the generalisation capabilities of the trained models on other UAV-based litter detection tasks. To assess the broader applicability of the method, experiments were also carried out on the Pascal VOC 2012 dataset, which featured a larger variety of object categories and greater complexity.

Evaluation metrics were standard within object detection research, as detailed in Section 2.6. These included mAP calculated at different IoU thresholds, along with precision, recall, and F1 score. Additionally, mAR was employed to provide a comprehensive perspective on detection performance. The evaluation also incorporated COCO-style metrics, which were essentially variations of mAP computed separately for small, medium, and large objects. These metrics offered a more granular understanding of model performance across object sizes, which was particularly relevant given the emphasis on detecting small-scale litter objects.

Key experiments included validating the tiling preprocessing strategy on SODA to ensure appropriate input representation; conducting within-dataset and cross-dataset evaluations to test detection performance and generalisation; analysing performance based on object size to determine whether the LUPI framework improves small-object detection; and evaluating generalisability on the diverse Pascal VOC 2012 dataset. An ablation study examined the effect of the α parameter, a key element of the proposed methodology, on detection performance and computational trade-offs. Finally, qualitative analyses, including visual comparisons and interpretability studies using Grad-CAM variants, were performed to complement quantitative metrics by examining model decision-making processes and attentional focus.

This evaluation strategy was carefully designed to cover all the key objectives: developing, evaluating and refining the methodology (Objectives **O1** and **O2**), testing it across various litter detection datasets (Objective **O3**), and examining both the computational trade-offs and wider generalisation of the approach (Objective **O4**). A summary of how each experiment relates to these goals is detailed in Table 4.1.

Experiment	Dataset/s	Metric/s	Objective/s	Purpose
Optimal Tiling	SODA	Small Object Ratio per Image	preliminary for **O1**, **O2**, **O3**	Validate tiling preprocessing; no reason presented in literature
Within-Dataset Evaluation	Subsets of SODA	mAP, Precision, Recall, F1 Score, mAR, Confusion Matrix, Training Time, Model Size, FPS, GFLOPS	**O1**, **O2**, **O3**, partially **O4**	Test methodology on UAV-based litter detection
Cross-Dataset Evaluation	BDW, UAVVaste	mAP, Precision, Recall, F1 Score	**O3**	Assess generalisation to other UAV litter detection datasets
Object Size Performance Analysis	Subsets of SODA, BDW, UAVVaste	mAP (small, medium, large)	**O2**, **O3**	Evaluate small-object detection improvements
Pascal VOC 2012 Evaluation	Pascal VOC 2012	mAP, Precision, Recall, F1 Score, mAR, Confusion Matrix, Training Time, Model Size, FPS, GFLOPS	**O4**	Test generalisability on a larger dataset with a broader range of object categories
Ablation Study on Alpha (α)	Subsets of SODA, Pascal VOC 2012	mAP, F1 Score	**O2**, partially **O4**	Assess impact of critical alpha (α) parameter
Visual Comparison	All datasets	Qualitative comparison	—	Visually inspect improvements beyond metrics
Visual Interpretability	Primarily SODA, but also evaluated on other datasets with similar results	Grad-CAM variants	—	Explore model decision processes and attention

Table 4.1 Summary of the evaluation strategy, detailing the conducted experiments, datasets used, key metrics, aligned research objectives, and the purpose of each experiment.

4.4 Optimal Tiling Experiments

To determine the most suitable tiling parameter for pre-processing the SODA dataset, which serves as an essential preliminary step in addressing objectives **O1**, **O2**, and **O3**, a series of experiments was conducted. The objective was to identify an appropriate grid size for effective tiling. Although the authors in [20] proposed a 5×5 configuration, they offered no clear justification for selecting that specific size. Tiling is applied in this context to magnify the size of small litter objects, which appear relatively diminished due to the high-altitude perspective (see Figure 3.4). As the grid becomes finer, these objects occupy a larger proportion of each tile. This can potentially improve detection accuracy. A larger grid size, therefore, appears beneficial in this respect.

However, increasing the grid size introduces new challenges. The most significant issue is the additional burden on processing and inference time. More tiles are produced, which in turn means more data needs to be handled during model inference. This added complexity can lead to slower performance if not carefully managed [153]. Thus, choosing a grid size involves balancing two competing goals. One is to make small objects more detectable. The other objective is to prevent a significant rise in computational cost. Identifying an optimal balance between object visibility and processing efficiency requires further exploration.

In this study, an experiment was carried out to identify the optimal tiling grid size by measuring the rate of change in the number of small objects and the number of resulting images as the grid size increases. The first step involved establishing a classification scheme for object sizes. For this purpose, the definitions provided by the COCO dataset [137] were adopted, and are summarised below:

Small: Objects occupying an area of less than 32×32 pixels.

Medium: Objects with sizes ranging from 32×32 to 96×96 pixels.

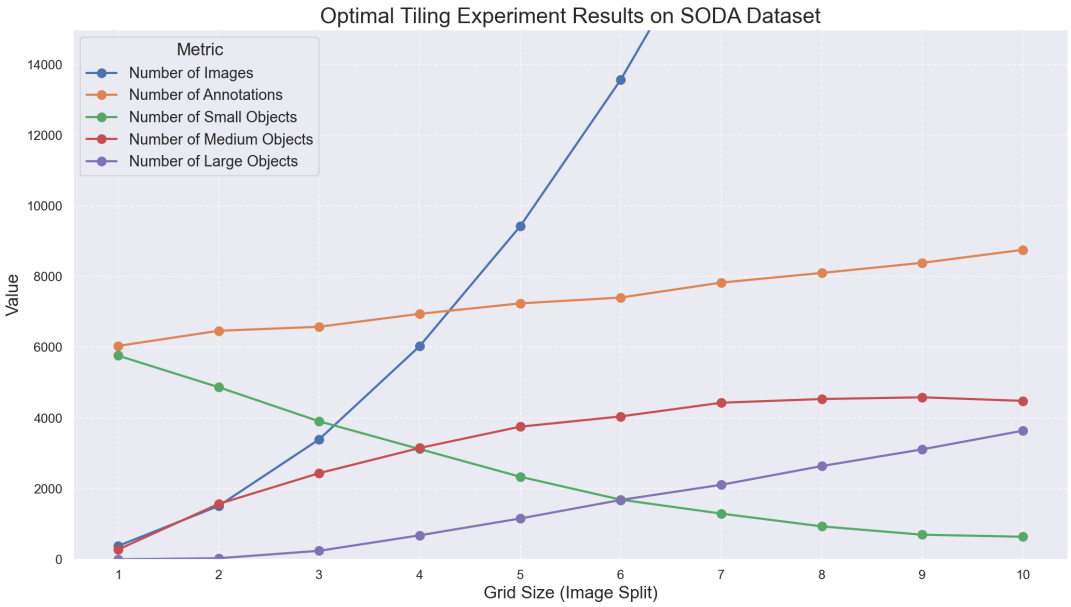
Large: Objects exceeding 96×96 pixels in area.

With these definitions in place, the experiment proceeded by testing a range of grid sizes from 1 to 10. A grid size of 1 represents the original image with no tiling applied, while a grid size of 10 results in each image being divided into 100 tiles. For each configuration, the rate of change was recorded for small, medium, and large objects, as well as the total number of annotations and images. A key consideration was the treatment of annotations that spanned tile boundaries. In such cases, these annotations were duplicated across the relevant tiles, leading to an expected increase in annotation count. This effect was deliberately preserved to reflect how tiling may influence the volume of training data. Each tile was also resized to 800 by 800 pixels to ensure consistency in size across all tiling configurations. This resizing was essential for maintaining uniform object scale during analysis, as 800 pixels was the chosen model input size throughout this study (refer to Subsection 3.8.2).

Two separate experiments were conducted. The first included the full SODA dataset, covering altitudes from 1 metre to 30 metres. The second considered a subset with altitudes ranging from 5 metres to 30 metres. These configurations were selected to examine whether including very close-range images (such as those at 1 metre) would affect the optimal grid size. The ultimate objective is to develop a single detection model capable of operating effectively across various altitudes. The outcomes of these experiments are illustrated in Figure 4.1.

The results of both experiments demonstrate that, as grid size increases, the number of images and annotations also increases. Conversely, the number of small objects decreases, while the count of medium and large objects rises. In the experiment cover-

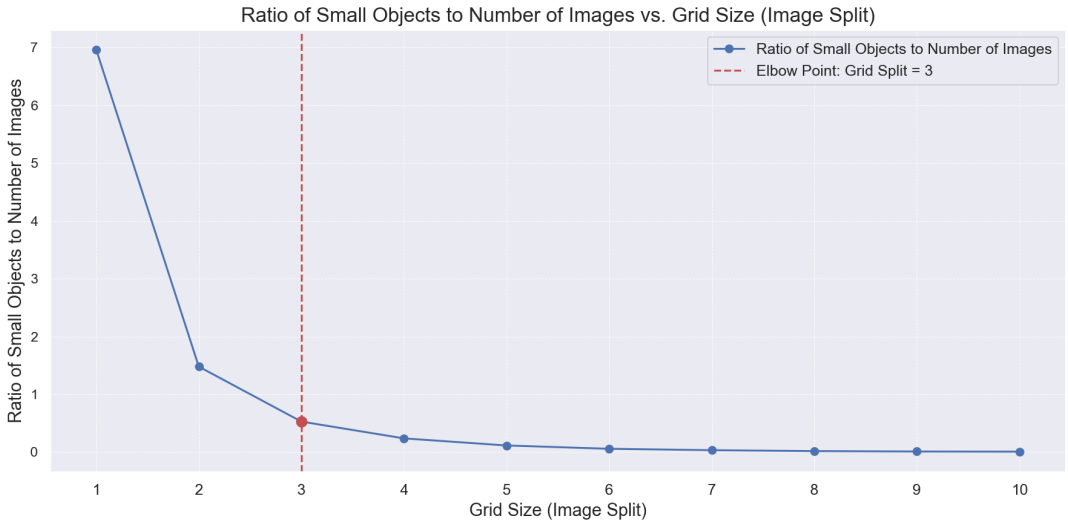
(a)



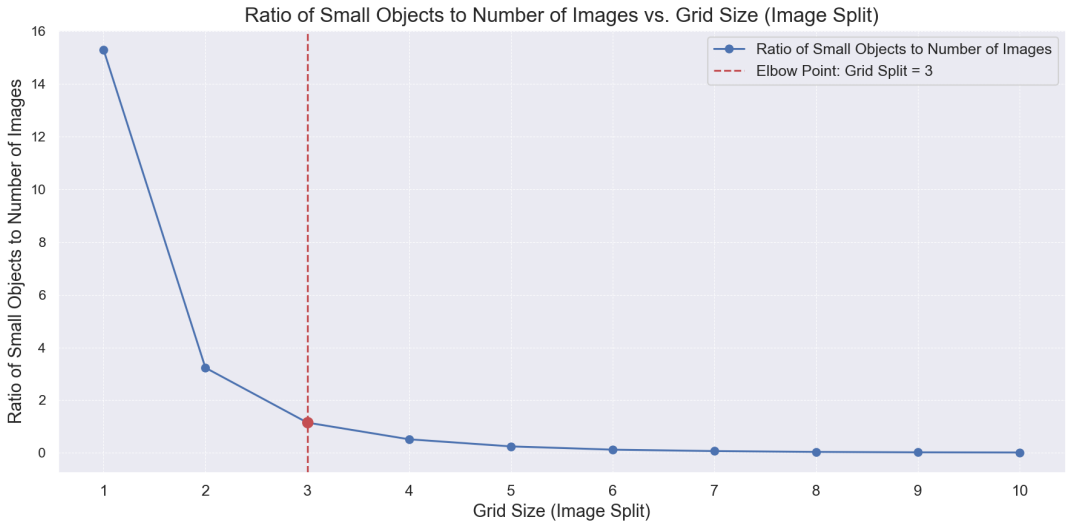
(b)

Figure 4.1 Optimal tiling experiment results on the SODA dataset, illustrating the increase in the number of images and annotated objects as the grid size increases. (a) Results over the full altitude range (1–30 metres), and (b) results on a subset of altitudes (5–30 metres) from the SODA dataset.

ing all altitudes, the rate of change was more pronounced, largely due to the presence of large objects captured at the 1-metre altitude. In contrast, the experiment excluding 1-metre data showed more gradual, consistent changes. Despite this difference, both experiments exhibited the expected trends based on the tiling methodology. However,



(a)



(b)

Figure 4.2 Ratio of small objects to the number of images plotted against tiling grid size, highlighting the observed elbow point in red. (a) Results over the full altitude range (1–30 metres), and (b) results on a subset of altitudes (5–30 metres) from the SODA dataset.

these outcomes alone do not provide a direct conclusion regarding the optimal grid size. The goal remains to identify a configuration that reduces the proportion of small objects without introducing excessive computational overhead from processing a high number of tiles. To this end, the ratio of small objects to the number of generated tiles was examined across grid sizes, as shown in Figure 4.2.

To determine the optimal grid size, the elbow point [154] method was applied. This approach involved calculating the first and second derivatives of the curve, identi-

fying the point at which the rate of change begins to level off. The second derivative was essential in locating the inflection point, where diminishing returns set in. Interestingly, both experiments pointed to a 3×3 grid as the optimal configuration. It is worth noting, however, that although the trends across both plots were similar, the ratio of small objects per tile was higher in the all-altitudes experiment. This suggests that including close-range imagery may influence the density of small object annotations. As a result of this experiment, the tiling configuration used for dataset pre-processing in this study was set to a 3×3 grid for the SODA dataset (refer to Subsection 3.10.1), in line with the findings outlined above.

4.5 Within-Dataset Evaluation

To assess the proposed integration of the LUPI paradigm within object detection, UAV-based litter detection was selected as a case study, supporting the analysis of objectives **O1**, **O2**, **O3**, and partially addressing **O4**. This task presents a particularly challenging scenario, largely due to the frequent presence of small objects, which considerably complicates the detection process. To examine the effectiveness of the proposed methodology under such conditions, a within-dataset evaluation was conducted using the SODA dataset.

The SODA dataset was purposefully chosen as the principal dataset for training UAV-based litter detection models. Its selection stems from its capacity to represent a diverse range of altitudes and litter types–characteristics not fully captured by the other available UAV-based litter datasets. The within-dataset evaluation comprised three core experiments: (i) binary litter detection using only the 1-metre altitude subset with no tiling; (ii) binary litter detection on the full SODA dataset, tiled using a 3×3 grid and covering all altitudes; and (iii) multi-label litter detection using the same 3×3 tiled version across all altitudes. Each of these configurations was chosen for a specific purpose: the first experiment targets detection at close range, the second expands the scope across variable altitudes, and the third introduces additional classification complexity by extending from binary to multi-label tasks, distinguishing between various litter types and the background.

For each of these experiments, the five object detection architectures chosen in Section 3.5 were trained: Faster R-CNN, SSD, RetinaNet, SSDLite, and FCOS. For each architecture, a baseline model, a teacher model, and four student models were trained with varying levels of teacher guidance. Consequently, 30 distinct models were trained for each experiment.

The central objective was to determine whether student models trained using the LUPI framework, with access to privileged training signals, could outperform their

respective baselines, which were trained using only standard input data. For each architecture, a teacher model was first trained, followed by the training of student models using five distinct α values: 0, 0.25, 0.5, 0.75, and 1. These values match those used in related work in LUPI [102]. An α of 0 represents the baseline, where the student is trained without any guidance from the teacher, while $\alpha = 1$ reflects full reliance on the teacher's output during training. The intermediate values allowed a controlled study of how varying degrees of teacher influence affected model performance.

Importantly, this study did not involve cross-architecture distillation. In other words, teacher models were used only to train student models of the same architecture. This approach was adopted to avoid the added complexity of aligning latent feature representations between different model types, which would have required additional mechanisms and exceeded the scope of this work.

All models were trained using the designated training subsets, while validation and testing were carried out on their respective set partitions. The evaluation employed the detection metrics outlined in Section 2.6, providing a consistent and coherent basis for comparison. A uniform pre-processing pipeline and identical training configurations were applied across all models, ensuring fairness in evaluation. Given this diverse range of detection architectures and the controlled experimental setup, the resulting outcomes offer a reliable and concrete basis for assessing the influence of LUPI in object detection.

4.5.1 Binary Litter Detection on the SODA Dataset at 1-Metre Altitude

The first experiment aimed to assess the benefits of using the LUPI framework for close-range litter detection. To this end, a subset of the SODA dataset at a 1-metre altitude was selected, consisting of 452 images. These images were divided into 316 for training, 46 for validation, and 90 for testing. In this experiment, detection models were trained to classify all litter into a single category, distinguishing between litter (foreground) and background.

The results of this experiment, comparing the baseline model with the best student model across all five selected detection architectures, are presented in Figure 4.3. In all architectures, the student model outperformed the baseline, showing a notable improvement in detection accuracy, particularly in the challenging mAP@50–95 metric. Additionally, improvements were seen in other key detection metrics, including precision, recall, and F1 score. Among the models, the RetinaNet student achieved the highest performance, followed by the Faster R-CNN and FCOS student models. Across all metrics, the student models demonstrated an accuracy improvement ranging from 0.05 to 0.15. Interestingly, when comparing the baseline models with the student models, RetinaNet and SSD showed the highest improvements in detection accuracy, with the student models outperforming their baseline counterparts. The other architecture

types, such as Faster R-CNN and FCOS, also saw improvements, though the performance boosts were somewhat smaller in comparison to those of RetinaNet and SSD.

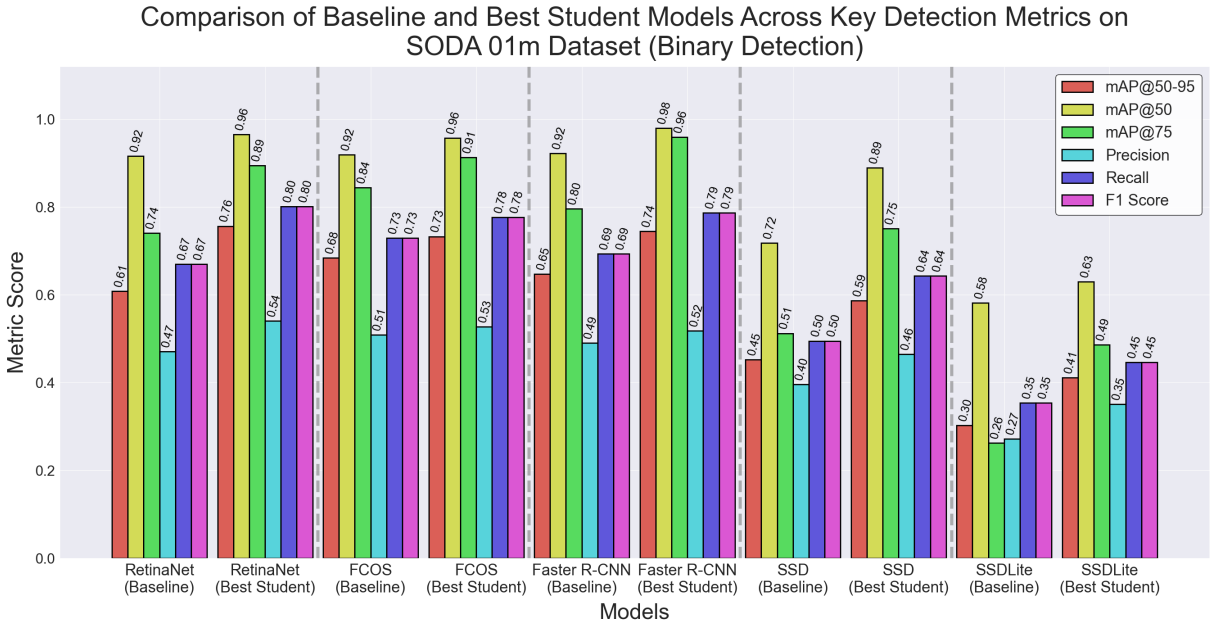


Figure 4.3 Comparison between the baseline and the best-performing student models across key detection metrics on the SODA dataset at a 1-metre altitude for binary litter detection.

While RetinaNet, FCOS, and Faster R-CNN delivered the best results in this experiment, which can be attributed largely to their complex architectures and the use of the ResNet-FPN backbone, notable improvements were also observed with the SSD and SSDLite architectures. A closer examination suggests that this performance boost is closely linked to the role of LUPI in refining the representational quality of the ResNet-FPN within the teacher model. In the course of training, the privileged spatial information enables the teacher to acquire a more precise spatial understanding of the scene, thereby increasing the spatial information captured by the backbone FPN and leading to higher-quality proposals, which are subsequently transferred to the student during distillation. This suggests that the LUPI method can improve litter detection performance at close range, even in complex environments with backgrounds such as rocks and plants. The results indicate that, regardless of the architecture, LUPI significantly improves detection accuracy in close-range binary detection.

In addition to the comparison between the baseline and student models, a further analysis was conducted to evaluate the performance of the different teacher models, using the same evaluation metrics and experimental setup. The results are summarised in Table 4.2, which presents the detection accuracy achieved by each teacher model architecture. It can be observed that Faster R-CNN, FCOS, and RetinaNet stood out as the highest-performing teacher models. Most of their detection metrics approached 1.0, suggesting near-ideal detection performance across the evaluated scenes.

Model	mAP@50–95	mAP@50	mAP@75	mAR@1	mAR@10	mAR@100	Precision	Recall	F1 Score
RetinaNet	0.94	0.98	0.96	0.63	0.96	0.96	0.93	0.99	0.96
FCOS	**0.96**	0.98	0.97	0.63	0.97	0.97	0.81	0.99	0.89
Faster R-CNN	**0.96**	**0.99**	**0.98**	**0.63**	**0.98**	**0.98**	**0.99**	**0.99**	**0.99**
SSD	0.78	0.96	0.94	0.54	0.81	0.81	0.65	0.99	0.79
SSDLite	0.61	0.73	0.72	0.48	0.63	0.63	0.02	0.99	0.03

Table 4.2 Comparison of teacher model performance across key detection metrics, trained on the SODA dataset at a 1-metre altitude for binary litter detection.

On the other hand, the SSD and SSDLite teacher models, while still achieving reasonable scores, demonstrated noticeably lower accuracy. Even when trained with privileged information, these architectures did not attain the same level of conceptual understanding, indicating limitations in adapting to this particular detection task. Combined with the student model results in Figure 4.3, this suggests that these architectures may be less suited to scenarios requiring high sensitivity to fine-grained object features.

Interestingly, while Faster R-CNN proved to be the top-performing teacher model, it was not associated with the best-performing student. That distinction went to the RetinaNet student model. Given that both architectures share a similar ResNet-FPN backbone and were distilled at equivalent feature extraction layers, this outcome suggests that RetinaNet may be structurally more compatible with learning using privileged information in this context. Across all teacher models, recall was generally high, while precision tended to be lower. This implies that the teacher models were primarily geared toward minimising false negatives.

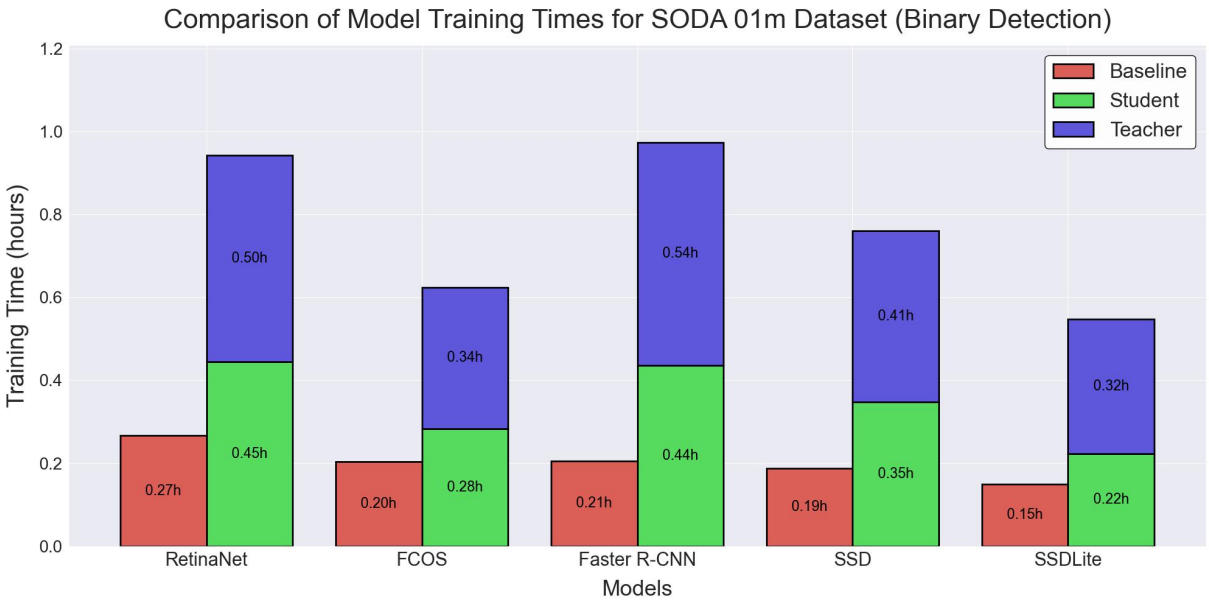


Figure 4.4 Comparison of model training times on the SODA dataset at a 1-metre altitude for binary litter detection.

While the results clearly show that LUPI improves detection accuracy in this particular experiment, it is also important to acknowledge that this improvement does not come without cost. Training within the LUPI framework requires an additional training phase, wherein a teacher model must be trained before the student. Moreover, each model must process privileged inputs during training, resulting in larger tensors and increased computational demands, despite only minor adjustments to the teacher architecture and none to the student.

To examine the impact of these changes on training duration, Figure 4.4 presents a comparison of training times across all models involved in this experiment. As shown in the figure, both student and teacher models required substantially more time to train than the baseline models. This increase is due in part to the two-stage nature of the training process and, for student models specifically, the added computation needed to process the teacher's outputs and compute the distillation loss. The time taken by the teacher during inference directly contributes to the student's overall training time.

In terms of model-specific performance, FCOS and SSDLite exhibited the shortest training durations, whereas RetinaNet and Faster R-CNN incurred the highest training times, especially under the LUPI setup. Notably, teacher models consistently recorded the longest training durations across all architectures.

Despite the increase in training time, a comparison of model size and parameter count between the baseline and student models, as shown in Table 4.3, shows that they

Model Configuration	Type	Size (MB)	Parameters (M)	GFLOPS	FPS
Baseline	RetinaNet	122.72	32.17	254.58	32.84
	FCOS	122.32	32.06	251.77	30.16
	Faster R-CNN	157.54	41.30	268.56	31.31
	SSD	90.58	23.75	61.05	107.22
	SSDLite	8.42	2.21	0.87	52.35
Student	RetinaNet	122.72	32.17	254.58	31.43
	FCOS	122.32	32.06	251.77	32.07
	Faster R-CNN	157.54	41.30	268.56	31.56
	SSD	90.58	23.75	61.05	105.70
	SSDLite	8.42	2.21	0.87	52.20

Table 4.3 Comparison of model configurations for trained baseline and student models on the SODA dataset at a 1-metre altitude for binary litter detection, including model type, size in megabytes, number of parameters (in millions), computational complexity in GFLOPS, and inference speed in FPS.

remain identical across all architectures. This suggests that student models achieve improved detection accuracy through LUPI without increasing model size or computational demands during inference. This is supported by the identical GFLOPS values, which represent the number of billions of floating-point operations the model performs per second, and comparable FPS measurements, indicating the number of frames processed per second during inference—both of which are consistent across models, as expected. In contrast to many modern detection models that require larger storage and are computationally intensive at deployment [9, 10, 91, 155], the student models preserve the compactness of their baseline versions.

It is also important to point out that Faster R-CNN, FCOS, and RetinaNet occupy more memory overall, as they rely on larger architectural designs. On the other hand, SSD and SSDLite remain comparatively lightweight. Even so, the main observation remains clear. In the context of binary litter detection, where lightweight architectures are essential due to hardware limitations, LUPI enables improved detection accuracy without increasing model size. This makes it a practical solution for situations where both storage and computational efficiency are important.

4.5.2 Binary Litter Detection on the SODA Dataset Across All Altitudes

The second experiment aimed to build upon the context of the first by further exploring the benefits of using the LUPI framework for litter detection at higher altitudes, addressing the challenges posed by more complex detection scenarios. To this end, the entire SODA dataset was tiled in a 3×3 configuration at all altitudes to reflect a more complex, real-world scenario for UAV-based litter detection. This dataset included 7,461 images, with 5,238 allocated for training, 765 for validation, and 1,458 for testing. In this experiment, detection models were trained to classify all litter into a single category, focusing on distinguishing small litter from variable and challenging backgrounds.

The results of this experiment, comparing the baseline model with the best student model across all five selected detection architectures, are presented in Figure 4.5. In most cases, the student models performed better than their baseline counterparts, showing noticeable improvements in detection accuracy, particularly in the more demanding mAP@50–95 metric and precision, recall, and F1 score metrics. However, the SSD and SSDLite architectures exhibited little to no improvement. Compared to the earlier experiment shown in Figure 4.3, the results here reflect a tougher detection setting, which generally led to lower overall performance. Even so, models trained with the LUPI framework still showed clear advantages.

Among all models, the student version of Faster R-CNN delivered the best overall results, followed by FCOS and RetinaNet. This marks a shift from the previous experiment, where RetinaNet had achieved the highest scores. The performance of Faster

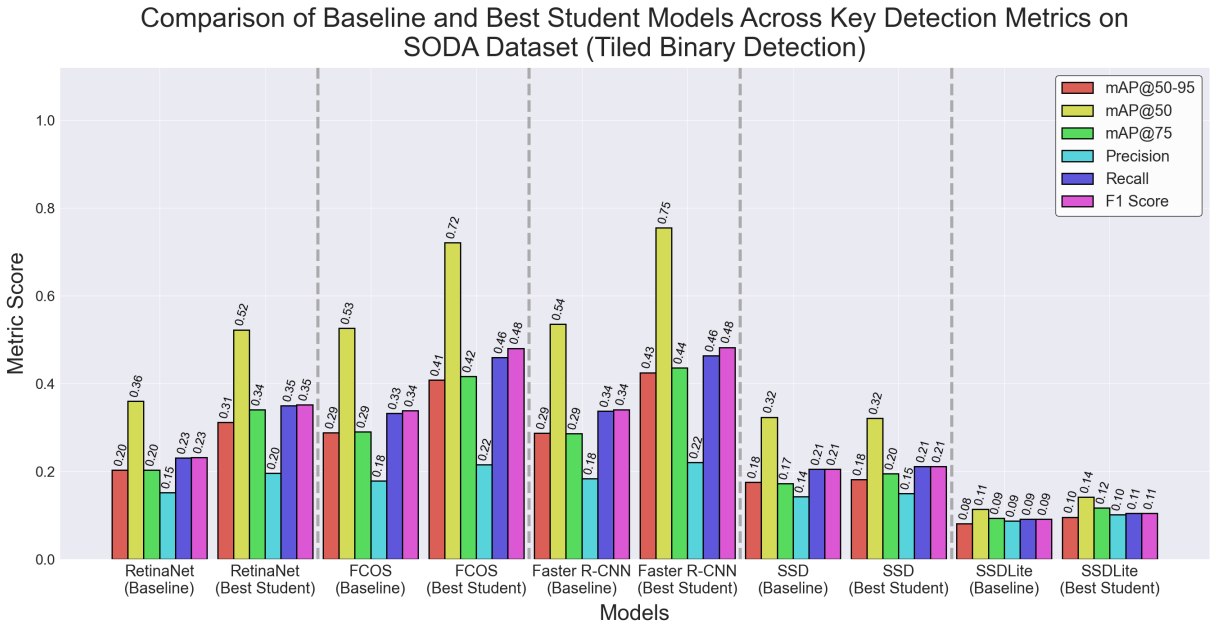


Figure 4.5 Comparison between the baseline and best-performing student models across key detection metrics on the 3×3 tiled SODA dataset across all altitudes for binary litter detection.

R-CNN suggests that its two-stage architecture may be better suited to the demands of this task than the one-stage approaches. Furthermore, the student models outperformed their baseline counterparts across all evaluated metrics, with accuracy improvements ranging from 0.01 to 0.11. FCOS and Faster R-CNN recorded the most noticeable performance boosts compared to their baselines. RetinaNet and SSDLite also benefitted, though to a lesser degree. For SSD, the improvement was limited, appearing only in the mAP@75 metric.

Moreover, the findings further support the selection of Faster R-CNN, FCOS, and RetinaNet as more effective detection architectures in this context, especially when compared to the lighter alternatives, owing to their structural complexity and performance consistency. This continues to suggest that the individual components of these networks, particularly the FPN, when trained with privileged information, enable a more precise spatial representation, which in turn facilitates improved detection outcomes.

In addition to this comparison, a further analysis was carried out to assess the efficacy of different teacher models, following a similar approach to the previous experiment. The outcomes, presented in Table 4.4, detail the detection accuracy attained by each teacher architecture. Once again, Faster R-CNN, FCOS, and RetinaNet emerged as the top performers, with several of their metrics approaching ideal values, though with slight reductions in some areas. These results indicate that, despite the increased scene complexity, these models continue to demonstrate high detection accuracy.

The SSD and SSDLite teacher models, while producing acceptable results, exhibited a noticeable decline in detection accuracy relative to the other models. This drop

Model	mAP@50–95	mAP@50	mAP@75	mAR@1	mAR@10	mAR@100	Precision	Recall	F1 Score
RetinaNet	0.90	0.95	0.94	0.34	0.83	0.91	0.41	0.98	0.58
FCOS	0.89	0.94	0.93	0.34	0.82	0.90	**0.97**	0.96	0.96
Faster R-CNN	**0.96**	**0.99**	**0.98**	**0.35**	**0.87**	**0.97**	0.96	**0.99**	**0.98**
SSD	0.49	0.62	0.59	0.27	0.51	0.51	0.16	0.97	0.27
SSDLite	0.18	0.23	0.19	0.17	0.19	0.19	0.00	0.79	0.01

Table 4.4 Comparison of teacher model performance across key detection metrics, trained on the 3×3 tiled SODA dataset across all altitudes for binary litter detection.

was especially evident when assessing performance through mAP and F1 score. Moreover, their results showed a significant reduction compared to the first experiment, further emphasising the limitations in their ability to generalise, particularly given the added complexity introduced by small object detection.

In contrast to the first experiment conducted at 1 metre altitude, Faster R-CNN emerged as the best-performing teacher and student model in this multi-altitude experiment. RetinaNet, which incorporates focal loss to address class imbalance between foreground and background, underperformed relative to both Faster R-CNN and FCOS, with some detection metrics showing a gap of 0.1 or more in terms of the models outlined in Figure 4.5. This suggests that RetinaNet may not be ideally suited for detecting litter in scenes characterised by visually complex and variable backgrounds. FCOS, leveraging *centre-ness* loss to improve spatial accuracy, delivered results on par with Faster R-CNN across both baseline and student models, reinforcing its reliability under these conditions.

Notably, the consistent use of the ResNet-FPN backbone across Faster R-CNN, FCOS, and RetinaNet likely contributed to their relative success. The FPN's capacity for multi-scale object detection aligns well with the demands of UAV-based litter detection at varying altitudes. This architectural feature appears to bolster compatibility with learning via privileged information.

Across the teacher models, recall remained consistently high, while precision tended to be lower. Once again, this suggests that the models were primarily geared toward reducing false negatives, which is generally beneficial in domains where missing detections is costly. However, in the context of litter detection, this emphasis may be less appropriate, as an increase in false positives can also undermine model reliability.

The training durations for all models employed in this experiment are presented in Figure 4.6. The outcomes differ slightly from those of the first experiment (refer to Figure 4.4). In certain instances, the student model required marginally less time to train than its baseline counterpart, though this difference was observed solely in the case of RetinaNet and was negligible.

Due to the increased number of training images, the overall time required rose

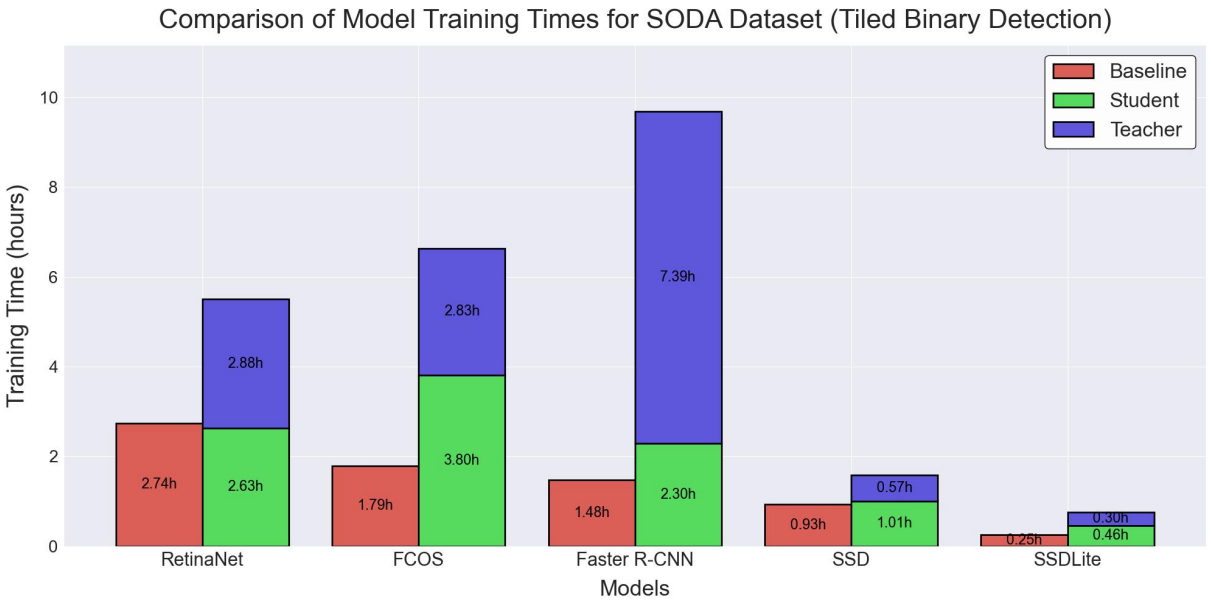


Figure 4.6 Comparison of model training times on the 3×3 tiled SODA dataset across all altitudes for binary litter detection.

across all models. For the SSD and SSDLite models, both the student and baseline versions required roughly the same duration to complete training. Nevertheless, two further observations stand out. The FCOS student model took longer to train than its teacher, which might be explained by the added computational cost of the *centre-ness* loss function. This component introduces further complexity to the optimisation process and could account for the increased training time. Additionally, the Faster R-CNN teacher exhibited a comparatively long training duration. However, the difference in training time between the Faster R-CNN student and its baseline counterpart was minimal.

Interestingly, the Faster R-CNN student model required less training time than the RetinaNet student, despite achieving the highest accuracy among all student models. This performance could be partly explained by the versatility of the region proposal mechanism used by the network.

A comparison of model sizes, as shown in Table 4.5, reveals results identical to those in Table 4.3 from the first experiment. This consistency is due to the use of binary detection models, meaning the model heads remained the same. Despite this, the observed improvement in detection accuracy across most model architectures indicates that student models benefit from incorporating LUPI without incurring additional costs in model size or computational load during inference, with only marginal variations in FPS.

Overall, this experiment sought to introduce a more challenging setting in which to evaluate the proposed methodology by extending the scope of the 1-metre experiment to incorporate altitude as an additional practical factor. This added dimension in-

Model Configuration	Type	Size (MB)	Parameters (M)	GFLOPS	FPS
Baseline	RetinaNet	122.72	32.17	254.58	38.30
	FCOS	122.32	32.06	251.77	40.21
	Faster R-CNN	157.54	41.30	268.56	36.29
	SSD	90.58	23.75	61.05	169.63
	SSDLite	8.42	2.21	0.87	108.80
Student	RetinaNet	122.72	32.17	254.58	38.31
	FCOS	122.32	32.06	251.77	39.94
	Faster R-CNN	157.54	41.30	268.56	36.15
	SSD	90.58	23.75	61.05	169.73
	SSDLite	8.42	2.21	0.87	104.89

Table 4.5 Comparison of model configurations for trained baseline and student models on the 3×3 tiled SODA dataset across all altitudes for binary litter detection, including model type, size in megabytes, number of parameters (in millions), computational complexity in GFLOPS, and inference speed in FPS.

creased the task's difficulty. Despite this, the use of LUPI continued to show improved accuracy in most models, although not universally. A noticeable decline was observed across all metrics, reflecting the increased complexity of the problem being tackled.

4.5.3 Multi-label Litter Detection on the SODA Dataset Across All Altitudes

The third experiment aimed to build upon the context established in the second, focusing again on multi-altitude UAV-based litter detection. However, this time, the objective was to test the method's generalisability in a multi-label detection setting. The experimental setup remained consistent with the previous trials, though the model heads were reconfigured to classify six distinct litter categories, as defined in the 3×3 tiled SODA dataset. This dataset comprised 7,461 images, of which 5,238 were used for training, 765 for validation, and 1,458 for testing. This experiment posed a significantly greater challenge for the detection models. The task required accurate identification of small, visually similar litter types, often set against complex and varied backgrounds. The difficulty of distinguishing between these categories introduced a new layer of complexity.

The results, comparing the baseline models with the best-performing student models across the five selected detection architectures, are illustrated in Figure 4.7. In most cases, the student models outperformed their baselines, with improvements par-

ticularly evident in the more demanding mAP@50–95 metric and precision, recall, and F1 scores. However, these improvements were notably smaller than those observed in the previous experiments. Given this added complexity, detection accuracy declined across all models, with low metric scores observed throughout. The SSD and SSDLite architectures in particular showed little to no improvement under these conditions, further exacerbating their limitations in the context of UAV-based detection.

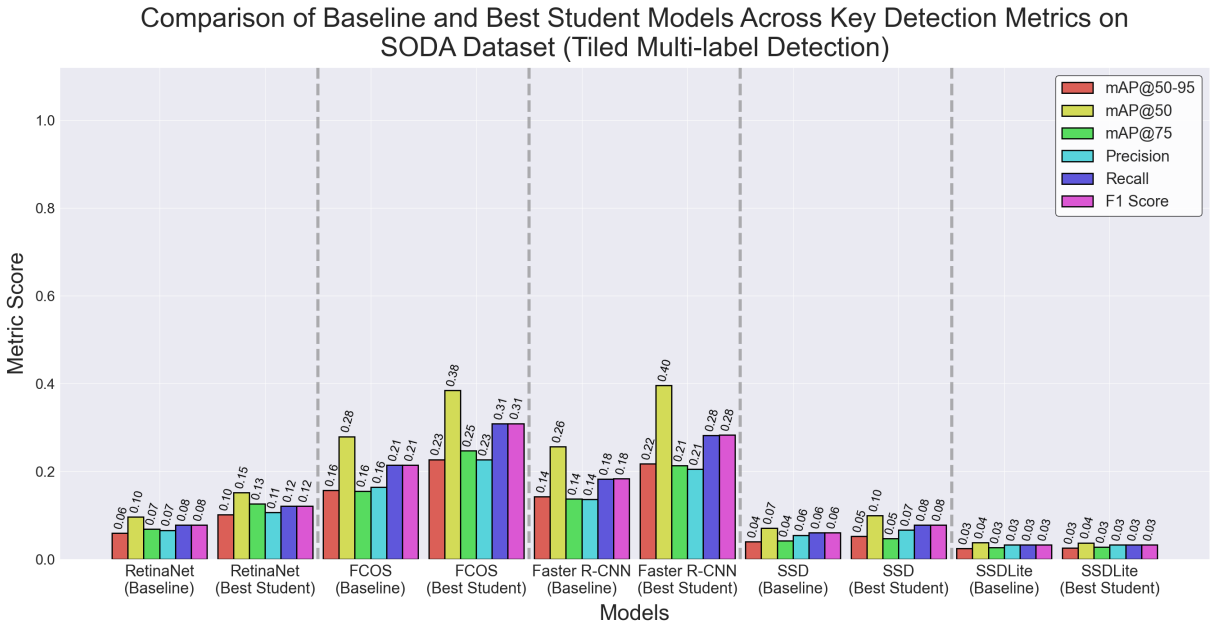


Figure 4.7 Comparison between the baseline and best-performing student models across key detection metrics on the 3×3 tiled SODA dataset across all altitudes for multi-label litter detection.

Compared to the earlier experiments, this setup presented a more difficult detection scenario, leading to an overall drop in performance. Nevertheless, models trained using the LUPI framework continued to show a measurable advantage in most cases. Among all the models, the student version of FCOS delivered the best overall results, followed by Faster R-CNN and RetinaNet. While this represents a shift from previous experiments, the general trend suggests that architectures using the ResNet-FPN backbone, which benefits from acquiring richer and more precise spatial information, tend to be more effective for this litter detection task. For all-altitude binary litter detection, Faster R-CNN produced the best results, but in the multi-label detection experiment, FCOS was more successful. However, the differences between the two were minimal in both experiments. Furthermore, the student models outperformed their baseline counterparts across all evaluated metrics, with accuracy improvements ranging from 0.01 to 0.08. Although these improvements gradually decreased, they still had a noticeable impact. FCOS and Faster R-CNN showed the most significant performance boosts compared to their baselines. RetinaNet and SSD also showed some benefit, though to a lesser degree. In contrast, SSDLite showed no improvement in any of the metrics.

Having addressed multi-label detection, a class-specific analysis of detection accuracy was also conducted. Figure 4.8 presents this evaluation using the AP@50–95 metric. As seen, the same trend in model performance is maintained across architecture types. FCOS and Faster R-CNN again emerged as the best-performing models, followed by RetinaNet, SSD, and SSDLite. The advantage offered by the LUPI framework is especially pronounced here, with student models outperforming their baseline counterparts in most individual AP scores.

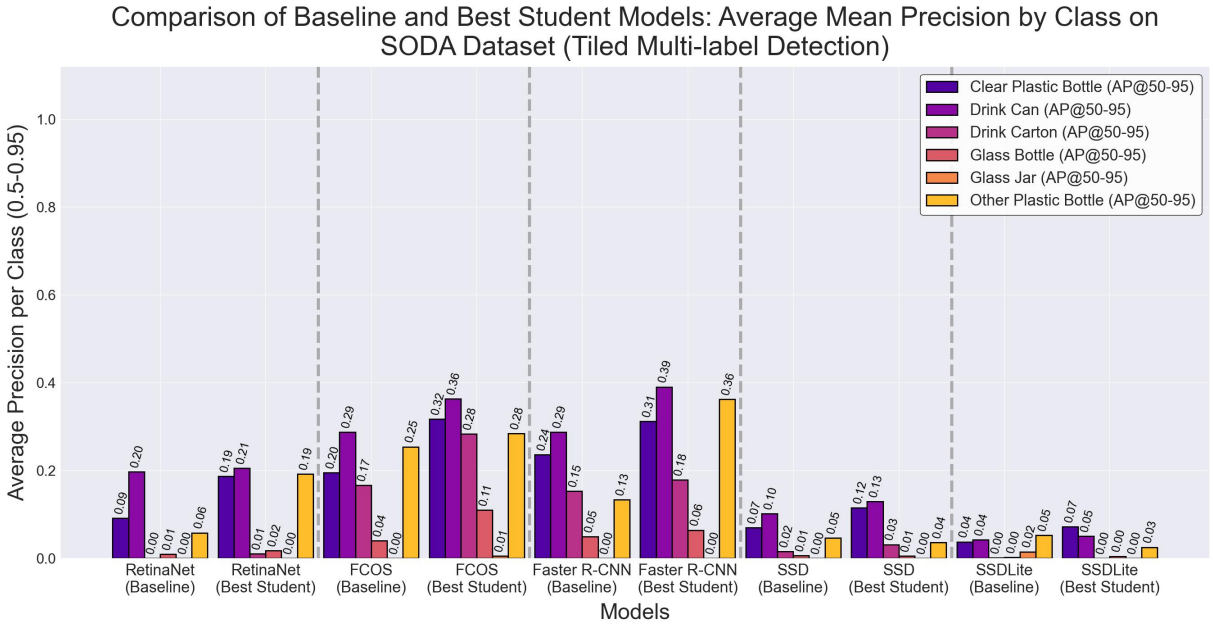


Figure 4.8 Comparison between the baseline and best-performing student models based on mean average precision by class on the 3×3 tiled SODA dataset across all altitudes for multi-label litter detection.

When this performance is compared against the class imbalance found in the SODA dataset, as shown in Figure 3.7, a consistent pattern is observed. The classes *drink can* and *clear plastic bottle*, which are most frequent in the dataset, tend to record the highest AP scores. Curiously, although *glass bottle* is more common than several other classes, no detector in this study successfully identified it. This could be attributed to it being misclassified as *glass jar*, due to the visual similarity between the two, leading to incorrect predictions.

Unexpectedly, the models also performed well in detecting *other plastic bottle*, despite its relatively low frequency in the dataset. This suggests that, in the context of litter detection across varied backgrounds, this particular class may present more distinct visual features, making it easier to detect than more transparent items, which blend more easily with their surroundings.

In addition to the class-specific results, the confusion matrix for this approach provides further insight, as shown in Figure 4.9. This figure presents the normalised confusion matrix for the best-performing model, FCOS. Both baseline and student ver-

sions exhibit a broadly diagonal structure, which is expected in successful classification scenarios. However, a considerable number of litter items remain unclassified, indicating a high rate of missed detections. Moreover, a portion of the litter is misclassified under incorrect categories, suggesting a degree of confusion between visually similar classes.

Notably, the student model demonstrates an improvement in this regard. The confusion matrix reveals stronger diagonal values across most categories, indicating better classification accuracy. Additionally, there are fewer unclassified instances, though a slight increase in misclassifications is also evident. These patterns support the overall finding that adopting the LUPI framework contributes positively to model performance. Finally, the matrix highlights the underlying issue of class imbalance in the dataset. As previously outlined, categories with higher representation tend to achieve better scores.

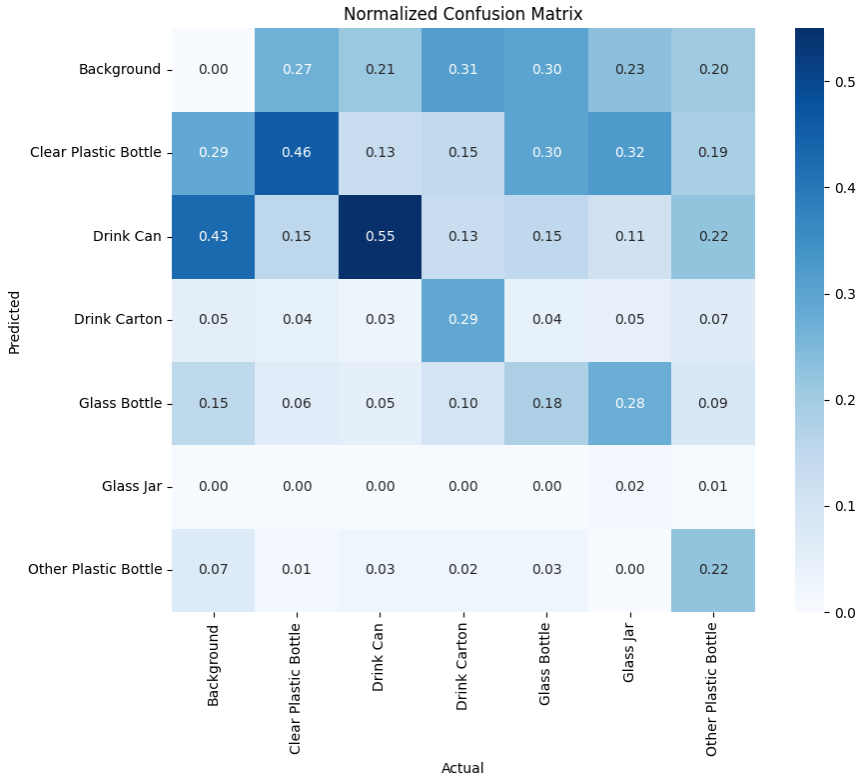
In addition to the above comparisons, an extended analysis was conducted to evaluate the accuracy of different teacher models, following a procedure consistent with the earlier experiments. The results, summarised in Table 4.6, report the detection accuracy achieved by each teacher architecture. As previously observed, Faster R-CNN, FCOS, and RetinaNet remained the strongest performers, with many of their metrics approaching values that suggest effective learning of the target concept. While minor declines were observed across several metrics, the increase in the mAR@1 score may reflect improved top-ranked predictions, though it does not necessarily indicate a broader improvement in recall performance. These findings imply that, even with the added complexity of the scenes, these models continue to perform reliably in terms of detection accuracy.

Model	mAP@50–95	mAP@50	mAP@75	mAR@1	mAR@10	mAR@100	Precision	Recall	F1 Score
RetinaNet	0.88	0.92	0.91	0.66	0.89	0.89	0.76	0.97	0.85
FCOS	0.91	0.95	0.94	0.68	0.92	0.92	0.91	0.97	0.94
Faster R-CNN	**0.95**	**0.99**	**0.98**	**0.70**	**0.96**	**0.96**	**0.96**	**0.99**	**0.97**
SSD	0.36	0.49	0.45	0.33	0.41	0.41	0.59	0.76	0.63
SSDLite	0.11	0.13	0.13	0.13	0.13	0.13	0.00	0.37	0.01

Table 4.6 Comparison of teacher model performance across key detection metrics, trained on the 3×3 tiled SODA dataset across all altitudes for multi-label litter detection.

The SSD and SSDLite teacher models once again displayed a marked drop in detection accuracy, both in comparison to other architectures and relative to their performance in the preceding experiment. This further exacerbates their limited capacity to generalise within this more demanding detection context. As observed previously, Faster R-CNN remained the most effective teacher model overall. However, despite being the second-best teacher, FCOS produced the highest-performing student in this multi-label setup. RetinaNet continued to lag behind both Faster R-CNN and

(a)



(b)

Figure 4.9 Normalised confusion matrices for multi-label litter detection on the 3×3 tiled SODA dataset across all altitudes, using the best-performing models of the FCOS architecture: (a) baseline model, (b) student model.

FCOS, showing comparatively weaker performance. Among all teacher models, recall scores were consistently high, while precision varied more significantly. Notably, SSDLite recorded a precision score of zero, whereas the remaining models achieved more acceptable precision values, ranging from 0.59 to 0.96.

The training durations for all models used in this experiment are shown in Figure 4.10. These results contrast significantly with those from the previous experiment (see Figure 4.6). Overall, training times increased across the board. Notably, the student models required substantially more time to train compared to their baseline counterparts, marking a sharp rise relative to the earlier experiment.

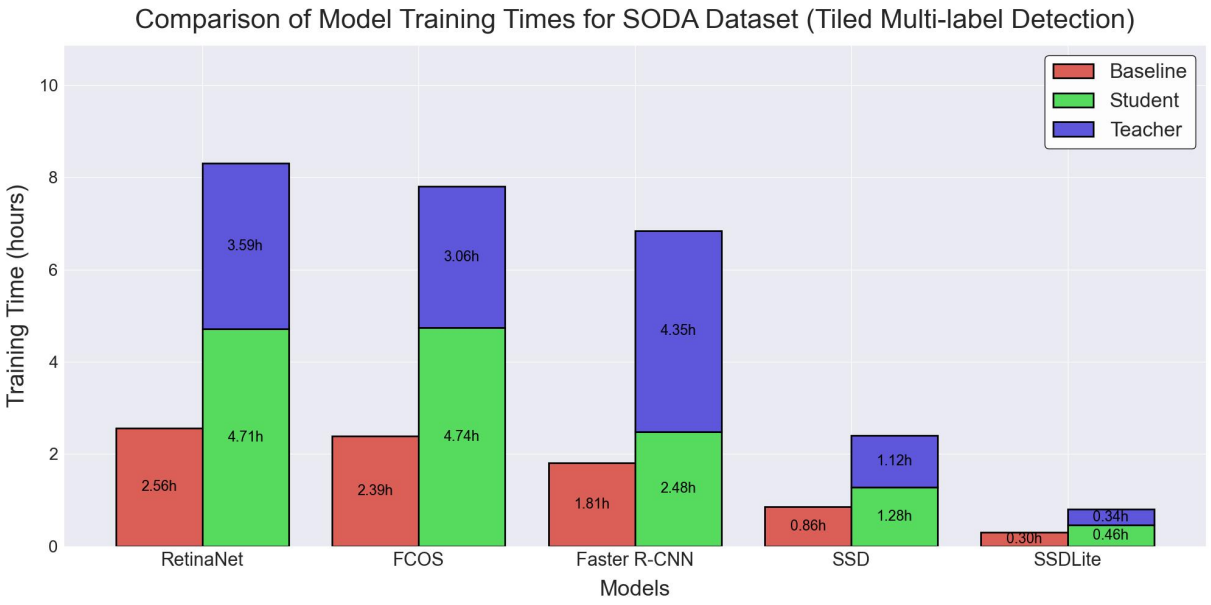


Figure 4.10 Comparison of model training times on 3×3 tiled SODA dataset across all altitudes for multi-label litter detection.

Interestingly, the teacher models now required less training time than their corresponding student models, a trend observed across all architectures except for Faster R-CNN. The FCOS student model, despite requiring more training time than RetinaNet, achieved the best performance overall. In contrast, Faster R-CNN trained faster than RetinaNet, and both Faster R-CNN and FCOS required less cumulative time for training (teacher and student combined) than the RetinaNet model. Meanwhile, SSD and SSDLite remained the fastest to train, maintaining a similar trend to that observed in previous experimental setups.

A comparison of model sizes and inference speeds, as outlined in Table 4.7, reveals results that differ slightly from those observed in the previous experiments. This variation stems from the shift to multi-label detection, which required modifying the model head to predict a broader range of categories. Consequently, this adjustment led to an expected increase in the parameter count within the detection head, consistent with standard object detection practices. Nonetheless, the overall trend of improved

detection accuracy across most model architectures suggests that student models continue to benefit from the application of LUPI, without incurring any added cost in terms of model size or computational demand during inference, as also evident by the similar GFLOPS and FPS values.

Model Configuration	Type	Size (MB)	Parameters (M)	GFLOPS	FPS
Baseline	RetinaNet	123.11	32.27	257.35	40.38
	FCOS	122.36	32.08	252.08	40.88
	Faster R-CNN	157.64	41.32	268.61	37.45
	SSD	93.13	24.41	61.55	147.09
	SSDLite	8.68	2.28	0.89	92.65
Student	RetinaNet	123.11	32.27	257.35	38.30
	FCOS	122.36	32.08	252.08	39.26
	Faster R-CNN	157.64	41.32	268.61	36.65
	SSD	93.13	24.41	61.55	142.24
	SSDLite	8.68	2.28	0.89	97.64

Table 4.7 Comparison of model configurations for trained baseline and student models on the 3×3 tiled SODA dataset across all altitudes for multi-label litter detection, including model type, size in megabytes, number of parameters (in millions), computational complexity in GFLOPS, and inference speed in FPS.

Across these experiments, which span various realistic contexts and model architectures, it has been demonstrated that incorporating LUPI in object detection results in improved performance in the majority of cases. In instances where little to no improvement was observed, this can generally be attributed to the limitations of the architecture itself, which may not be optimised to handle the specific detection task. Additionally, while there is a trade-off in terms of increased training time due to the necessity of training both a teacher and a student model, the overall model parameters and sizes remain unchanged with the adoption of this approach.

4.6 Cross-Dataset Evaluation

In addition to the within-dataset evaluation, a separate cross-dataset examination was conducted to verify whether the models trained on the SODA dataset exhibited more consistent learning when trained with LUPI, compared to their baseline counterparts, in alignment with objective **O3**. To this end, cross-dataset testing was carried out using

the two other available UAV-based litter detection datasets: BDW [17] and UAVVaste [19]. For the BDW dataset, models trained on the SODA dataset at 1-metre altitude were selected, given their closer alignment with the characteristics of the BDW data (see Figure 3.9). For the UAVVaste evaluation, the binary litter detection models trained on the 3×3 tiled version of the SODA dataset were used, again due to the similarity in visual layout and scene composition (refer to Figure 3.10). In both cases, the models applied were trained specifically for binary litter detection.

4.6.1 Binary Litter Detection on the BDW Dataset

The results on the BDW dataset, as shown in Figure 4.11, demonstrate that student models trained using the proposed LUPI-based approach consistently outperformed their baseline counterparts across all architectures. For each detection metric, student models showed measurable improvements. In the more stringent mAP@50–95, performance increased by between 0.04 and 0.13, while even larger margins were observed in metrics such as mAP@50. Interestingly, although RetinaNet had been the top performer in the within-dataset evaluation for the 1-metre altitude experiment, it was FCOS that achieved the best results on the BDW test subset, closely followed by RetinaNet and Faster R-CNN. SSD also returned relatively high results, whereas SSDLite continued to trail behind the others.

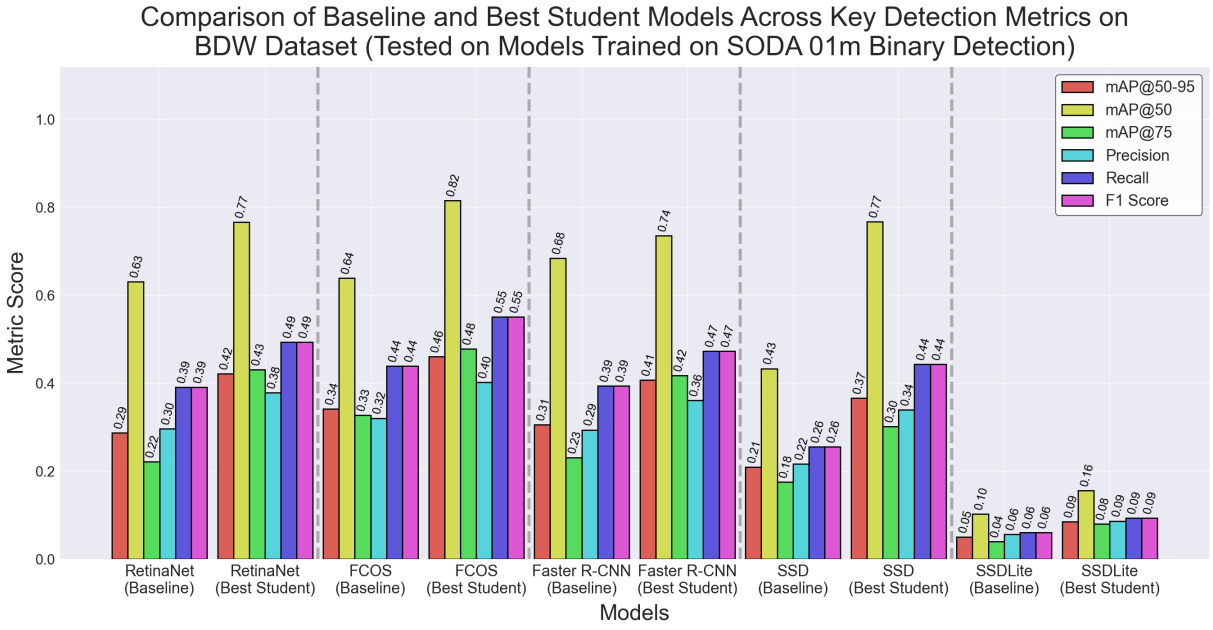


Figure 4.11 Comparison of baseline and best-performing student models on key detection metrics using the BDW dataset. The evaluated models were trained on the SODA dataset captured at 1-metre altitude for binary litter detection.

Models using the ResNet-FPN backbone continued to perform reliably well, suggesting that this architecture suits the challenges of UAV-based litter detection. Over-

all, the results point to the value of integrating LUPI, which seems to produce models that adapt better and generalise more effectively across datasets. While training time is slightly worse, these benefits come without adding to the model's size, which retains the method's efficiency and ease of deployment.

4.6.2 Binary Litter Detection on the UAVVaste Dataset

The cross-dataset evaluation results on the UAVVaste dataset, shown in Figure 4.12, largely confirm the earlier trend: the proposed approach incorporating LUPI tends to improve detection performance for student models over their baseline counterparts. In this setup, the binary detection models trained on the 3×3 tiled SODA dataset were evaluated on the UAVVaste test set.

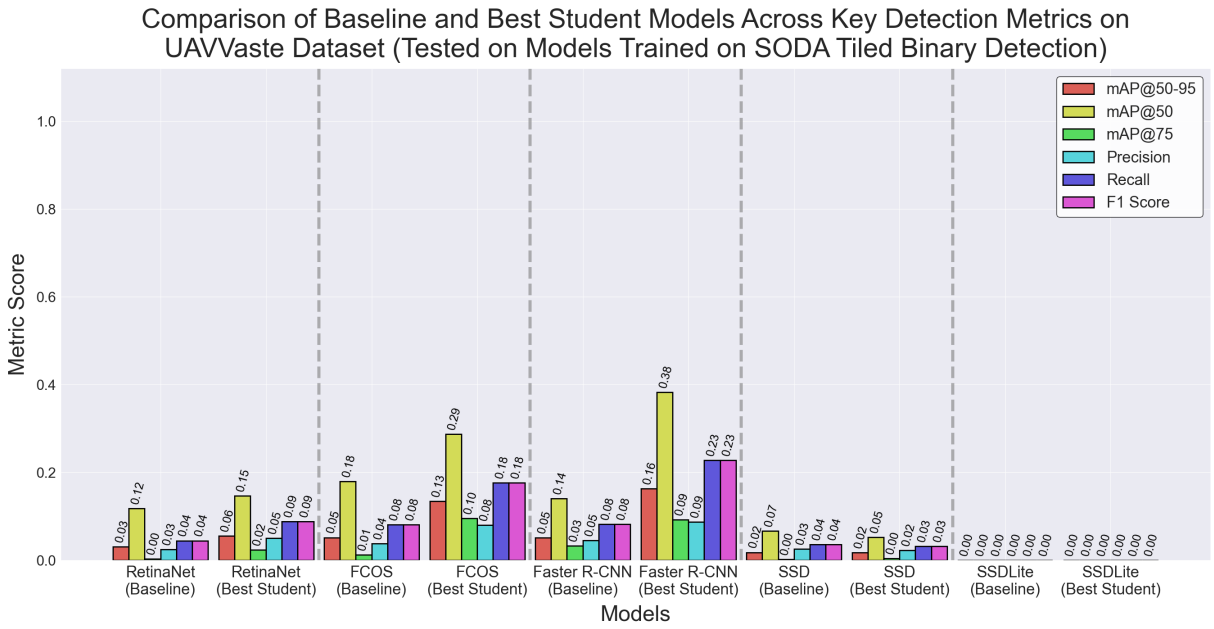


Figure 4.12 Comparison of baseline and best-performing student models on key detection metrics using the UAVVaste dataset. The evaluated models were trained on the 3×3 tiled SODA dataset across all altitudes for binary litter detection.

Faster R-CNN, FCOS, and RetinaNet all produced better results under the student setting, showing an improvement of around 0.03 to 0.11 in the more demanding mAP@50–95 metric. In contrast, SSDLite failed to register any successful detections, and SSD yielded identical mAP@50–95 scores between the student and baseline models, though with slightly lower performance in the other metrics for the student variant.

The weaker outcomes from SSDLite and SSD once again highlight their unsuitability for UAV-based litter detection, suggesting that the limited improvements observed with LUPI stem from the inherent inefficiency of these architectures in addressing this task. The drop in performance across all detectors likely stems from how litter is represented in each dataset. As shown in Figure 3.10, the UAVVaste dataset contains smaller

and visually distinct litter representations compared to the SODA dataset. Even so, several models produced accurate predictions, with most student models outperforming their baseline counterparts.

4.7 Object Size-Based Performance Analysis

In continuation of the within-dataset and cross-dataset evaluations, which have consistently demonstrated the effectiveness of the proposed approach compared to baseline models, it is important to highlight a fundamental challenge posed by the problem of UAV-based litter detection: its inherent focus on small object detection. While improvements in overall metrics such as mAP and mAR have been observed, a more detailed analysis is required to determine whether incorporating the LUPI framework into existing object detection architectures specifically enhances the detection of small objects.

Accordingly, this section examines this aspect in greater detail, in line with objectives **O2** and **O3**. The COCO evaluation metrics are utilised, which expand upon the standard metrics introduced in Section 2.6 by calculating mAP and mAR separately for small, medium, and large objects. These size categories follow the conventions established by the COCO dataset, with definitions consistent with those outlined earlier in Section 4.4. Essentially, this involves applying the standard metrics to subsets of objects grouped by size. Furthermore, the analysis focuses on results from both within- and cross-dataset evaluations, aiming to determine whether the proposed methodology offers distinct improvements across object sizes, with an emphasis on small object detection.

Model	Type	mAP@50–95	mAP^{Small}	mAP^{Medium}	mAP^{Large}	mAR^{Small}	mAR^{Medium}	mAR^{Large}
RetinaNet	**Baseline**	0.61	–	0.44	0.64	–	0.50	0.71
	Best Student	**0.76**	–	**0.62**	**0.79**	–	**0.68**	**0.83**
FCOS	**Baseline**	0.68	–	0.51	0.72	–	0.59	0.76
	Best Student	**0.73**	–	**0.62**	**0.76**	–	**0.67**	**0.80**
Faster R-CNN	**Baseline**	0.65	–	0.43	0.69	–	0.50	0.73
	Best Student	**0.74**	–	**0.67**	**0.76**	–	**0.70**	**0.81**
SSD	**Baseline**	0.45	–	0.09	0.54	–	0.11	0.58
	Best Student	**0.59**	–	**0.50**	**0.61**	–	**0.55**	**0.66**
SSDLite	**Baseline**	0.30	–	0.00	0.37	–	0.00	0.43
	Best Student	**0.41**	–	**0.01**	**0.50**	–	**0.01**	**0.54**

Table 4.8 Comparison of the baseline and best-performing student models across COCO detection metrics on the SODA dataset at 1-metre altitude for binary litter detection, with results reported separately for each object size category.

Examining the results for the SODA dataset at 1-metre altitude for binary litter detection, as detailed in Table 4.8, it is important to note that both mAP^{Small} and

mAR^{Small} are undefined, since this subset does not contain any small objects. However, for the remaining object sizes, a substantial improvement is evident, with the best student models significantly outperforming the baseline models. This is especially noticeable for RetinaNet, Faster R-CNN, and FCOS architectures, where consistent improvements across all metrics are observed, underscoring their enhanced ability to detect medium to large objects. While improvements for larger objects were expected and are consistent throughout the experiments, it is interesting to observe that SSD achieved the largest increase in mAP^{Medium} compared to the other models, whereas SSDLite exhibited the smallest gains, a trend that is mirrored in the mAR^{Medium} metric as well.

Model	Type	mAP@50–95	mAP^{Small}	mAP^{Medium}	mAP^{Large}	mAR^{Small}	mAR^{Medium}	mAR^{Large}
RetinaNet	**Baseline**	0.20	0.08	0.26	0.41	0.10	0.29	0.47
	Best Student	**0.31**	**0.11**	**0.43**	**0.59**	**0.14**	**0.50**	**0.65**
FCOS	**Baseline**	0.29	0.15	0.37	0.49	0.19	0.44	0.54
	Best Student	**0.41**	**0.27**	**0.50**	**0.62**	**0.34**	**0.57**	**0.67**
Faster R-CNN	**Baseline**	0.29	0.13	0.36	0.55	0.16	0.44	0.62
	Best Student	**0.43**	**0.28**	**0.48**	**0.68**	**0.33**	**0.55**	**0.74**
SSD	**Baseline**	0.18	**0.02**	0.25	0.43	**0.03**	0.30	0.49
	Best Student	**0.18**	0.01	**0.25**	**0.49**	0.01	**0.30**	**0.56**
SSDLite	**Baseline**	0.08	0.00	0.01	0.39	0.00	0.00	0.47
	Best Student	**0.10**	0.00	**0.01**	**0.47**	0.00	**0.01**	**0.54**

Table 4.9 Comparison of the baseline and best-performing student models across COCO detection metrics on the 3×3 SODA dataset across all altitudes for binary litter detection, with results reported separately for each object size category.

Tackling a more challenging problem with binary litter detection on the SODA dataset, the application of a 3×3 tiling preprocessing step enables more detailed analysis, as shown in Table 4.9. Unlike before, this dataset subset now includes small objects, thus mAP^{Small} and mAR^{Small} are defined. Notably, RetinaNet, FCOS, and Faster R-CNN demonstrate substantial improvements across object sizes. However, these advances are less apparent for the SSD and SSDLite architectures. As expected, the highest scores are generally observed for medium and large objects, but meaningful improvements in mAP^{Small} and mAR^{Small} are also visible, particularly for models with ResNet-FPN backbones. The other architectures tend to perform worse in this regard, consistent with observations from Subsection 4.5.2. Interestingly, SSD is the only case where the baseline outperforms the student model on small object metrics, which may be attributed to the architectural limitations of SSD in capturing fine-grained features for small objects, resulting in less effective knowledge transfer through LUPI compared to the other models.

Continuing to increase the complexity of the problem by targeting multi-label detection on the 3×3 tiled SODA dataset, the difficulty of the task naturally rises, and the magnitude of improvements begins to diminish. Nonetheless, Table 4.10 continues to

illustrate that the proposed approach generally improves multi-label small object detection. Specifically, marginal improvements were noted for Faster R-CNN, FCOS, and to a lesser extent SSD. However, for RetinaNet and SSDLite, the baseline models outperformed the student models in terms of small object detection. Despite this, the student models consistently outperformed the baselines on medium and large object metrics. These findings align with observations from Subsection 4.5.3, where a significant drop in mAP and mAR scores was recorded due to the increased complexity. Still, even with this heightened challenge, a marginal benefit from adopting the proposed methodology remains evident.

Model	Type	mAP@50–95	mAP^{Small}	mAP^{Medium}	mAP^{Large}	mAR^{Small}	mAR^{Medium}	mAR^{Large}
RetinaNet	**Baseline**	0.06	**0.02**	0.09	0.14	**0.02**	0.11	0.18
	Best Student	**0.10**	0.01	**0.14**	**0.26**	0.01	**0.16**	**0.31**
FCOS	**Baseline**	0.16	0.11	0.20	0.24	0.14	0.26	0.30
	Best Student	**0.23**	**0.13**	**0.30**	**0.35**	**0.17**	**0.38**	**0.46**
Faster R-CNN	**Baseline**	0.14	0.08	0.20	0.22	0.10	0.24	0.29
	Best Student	**0.22**	**0.14**	**0.26**	**0.33**	**0.19**	**0.33**	**0.39**
SSD	**Baseline**	0.04	0.01	0.07	0.11	0.00	0.08	0.17
	Best Student	**0.05**	**0.01**	**0.10**	**0.11**	**0.01**	**0.12**	**0.18**
SSDLite	**Baseline**	0.03	0.00	0.00	0.12	0.00	0.00	0.17
	Best Student	**0.03**	0.00	0.00	**0.13**	0.00	0.00	**0.18**

Table 4.10 Comparison of the baseline and best-performing student models across COCO detection metrics on the 3×3 SODA dataset across all altitudes for multi-label litter detection, with results reported separately for each object size category.

Model	Type	mAP@50–95	mAP^{Small}	mAP^{Medium}	mAP^{Large}	mAR^{Small}	mAR^{Medium}	mAR^{Large}
RetinaNet	**Baseline**	0.29	–	0.14	0.35	–	0.20	0.47
	Best Student	**0.42**	–	**0.26**	**0.49**	–	**0.32**	**0.56**
FCOS	**Baseline**	0.34	–	0.14	0.42	–	0.27	0.51
	Best Student	**0.46**	–	**0.26**	**0.54**	–	**0.39**	**0.62**
Faster R-CNN	**Baseline**	0.31	–	0.13	0.37	–	0.18	0.48
	Best Student	**0.41**	–	**0.24**	**0.47**	–	**0.29**	**0.55**
SSD	**Baseline**	0.21	–	0.07	0.26	–	0.09	0.32
	Best Student	**0.37**	–	**0.19**	**0.43**	–	**0.25**	**0.52**
SSDLite	**Baseline**	0.05	–	0.00	0.07	–	0.00	0.08
	Best Student	**0.09**	–	**0.01**	**0.12**	–	0.00	**0.13**

Table 4.11 Comparison of the baseline and best-performing student models across COCO detection metrics on the BDW dataset, with models trained on the SODA dataset captured at a 1-metre altitude for binary litter detection, and results reported separately for each object size category.

The results of the across-dataset experiments, segregated by object sizes, con-

tinue to align with the previously observed trends. As shown in Tables 4.11 and 4.12 for the BDW and UAVVaste datasets, respectively, the findings further support this notion. For the BDW dataset, small object detection metrics are undefined due to the absence of small objects. In this case, improvements are consistently seen in the medium and large object categories, with student models outperforming their baseline counterparts. The most pronounced improvements are observed for large objects, which is expected. While all models demonstrate this improvement, some achieve it to a greater extent than others, consistent with the observations in Subsection 4.6.1.

For the UAVVaste dataset, a relatively clear improvement in small object detection can be observed for the majority of models, with only a few outliers. This pattern mirrors the findings in Subsection 4.6.2. While the improvements for small objects are marginal, the largest performance boosts remain in the large object category, as previously emphasised. Overall, the across-dataset evaluations confirm that the proposed methodology enables student models to outperform baseline models, a trend that persists across all experiments.

Model	Type	mAP@50–95	mAP^{Small}	mAP^{Medium}	mAP^{Large}	mAR^{Small}	mAR^{Medium}	mAR^{Large}
RetinaNet	**Baseline**	0.03	0.02	0.10	0.04	0.02	0.16	0.20
	Best Student	**0.06**	**0.04**	**0.22**	**0.25**	**0.05**	**0.29**	**0.25**
FCOS	**Baseline**	0.05	0.04	0.15	0.00	0.05	0.24	0.00
	Best Student	**0.13**	**0.09**	**0.34**	**0.40**	**0.13**	**0.42**	**0.40**
Faster R-CNN	**Baseline**	0.05	0.03	0.21	0.11	0.03	0.31	**0.45**
	Best Student	**0.16**	**0.13**	**0.39**	**0.40**	**0.17**	**0.50**	0.40
SSD	**Baseline**	0.02	0.01	0.08	0.09	0.01	**0.17**	0.35
	Best Student	**0.02**	**0.01**	**0.09**	**0.12**	**0.01**	0.15	**0.35**
SSDLite	**Baseline**	0.00	0.00	0.00	0.00	0.00	0.00	0.00
	Best Student	0.00	0.00	0.00	0.00	0.00	0.00	0.00

Table 4.12 Comparison of the baseline and best-performing student models across COCO detection metrics on the UAVVaste dataset, with models trained on the 3×3 SODA dataset across all altitudes for binary litter detection, and results reported separately for each object size category.

Taken together, these findings suggest that the proposed methodology is capable of delivering improvements across datasets and object size categories, albeit with benefits that are at times modest. The recurrence of such outcomes under varied experimental conditions points to a measure of robustness in the approach. Although not uniformly transformative, the observed enhancements indicate that the method holds meaningful potential for tackling a range of object detection scenarios.

4.8 Pascal VOC 2012 Evaluation

Although it has consistently been shown across both within-dataset and cross-dataset evaluations that integrating LUPI into object detection allows student models to outperform their baselines, proven even in demanding scenarios such as that of UAV-based litter detection and small object detection. A final experiment was conducted to verify the generalisability of the proposed approach across larger object detection datasets with a larger number of categories, addressing fully objective **O4**. To this end, the Pascal VOC 2012 dataset was selected for its broader range of object categories and increased complexity. With 20 object classes, the dataset provides a more challenging test environment. As detailed in Subsection 3.10.4, Pascal VOC is a widely recognised benchmark for object detection. It contains 17,112 images in total, with 13,690 used for training and 3,422 for validation in this study.

The results of this experiment, comparing the baseline model with the best student model across five selected detection architectures, are shown in Figure 4.13. In most architectures, the student model outperformed the baseline, demonstrating a marginal improvement in detection accuracy when compared to previous experiments, especially in the challenging mAP@50–95 metric. Among the models tested, the SSDLite student achieved the highest performance, followed by Faster R-CNN, FCOS, and SSD student models. Across all metrics, the student models showed an accuracy improvement ranging from 0.02 to 0.05, although slightly less when compared to the other models presented in this study.

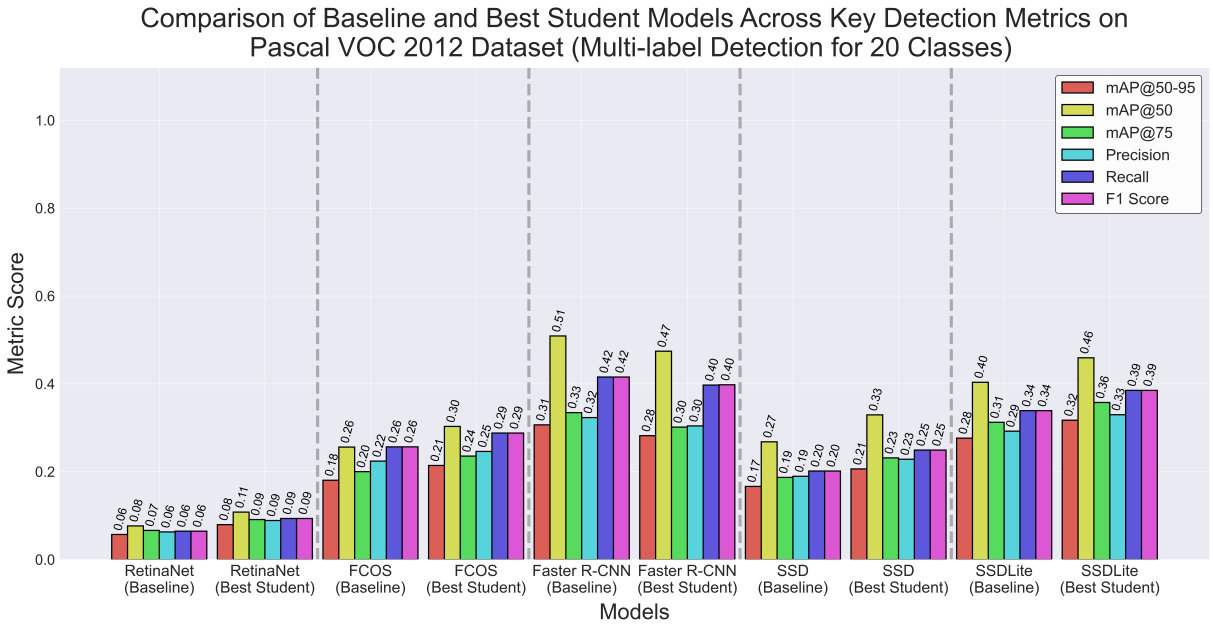


Figure 4.13 Comparison between the baseline and best-performing student models across key detection metrics on the Pascal VOC 2012 dataset for multi-label detection of 20 object classes.

Unfortunately, the Faster R-CNN student architecture performed worse than its baseline counterpart, while SSD and SSDLite models showed the largest improvements. Interestingly, the architectures using the ResNet-FPN architecture performed relatively worse than the others, in contrast to the litter detection experiments. This can be attributed to the nature of the data, as Pascal VOC contains images at relatively smaller resolutions. This, combined with the initialisation of pre-trained weights from models trained on the COCO dataset (which contains images of higher resolution), may have contributed to this performance discrepancy. It can also be the case that, in scenes with overlapping objects, the nature of the privileged information confuses the RPN in Faster R-CNN, leading to less reliable proposal generation and contributing to its poorer performance in comparison to the baseline. On the other hand, SSD and SSDLite take low-resolution images as input, which makes them more suited for this type of data and likely explains their superior performance. In addition to this, Faster R-CNN showed limited improvement, which may be due to the teacher model providing additional region proposals to reduce false positives, a strategy believed to improve detection accuracy.

Upon reviewing the class-specific metrics for each model, as detailed in Figure 4.14, a notable observation becomes apparent. Despite the pronounced class imbalance in favour of the *person* category, which is by far the most represented in the dataset (see Figure 3.12), the average precision scores for this class were comparatively low. This stands in contrast to categories with fewer examples, such as *aeroplane*, *bus*, *train*, and *tv monitor*, which frequently yielded higher scores in this evaluation.

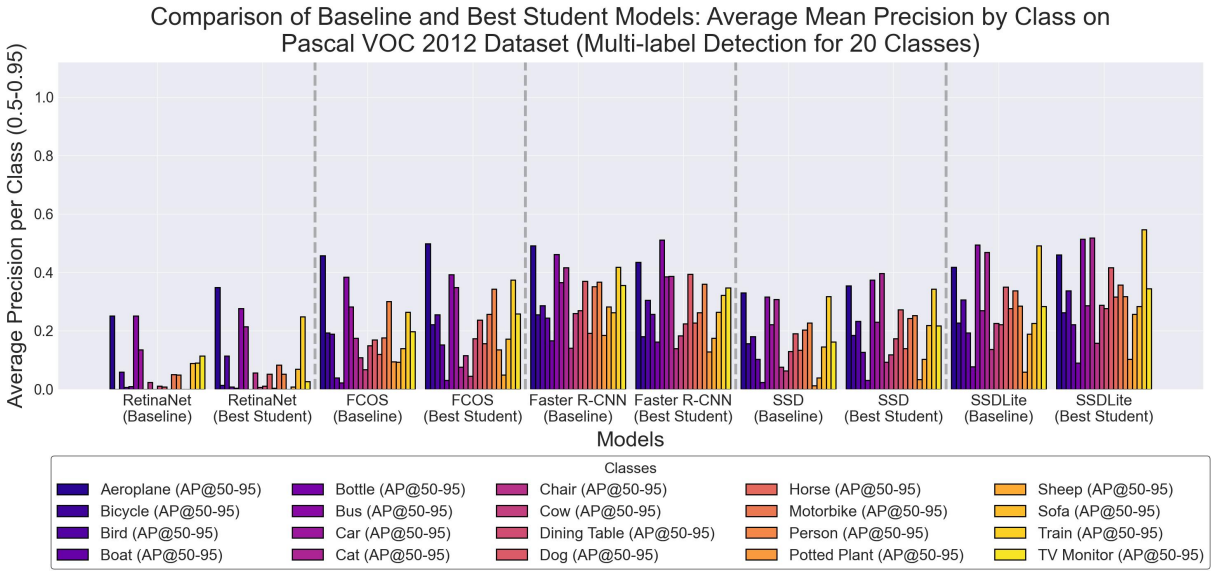


Figure 4.14 Comparison between the baseline and best-performing student models based on mean average precision by class on the Pascal VOC 2012 dataset for multi-label detection of 20 object classes.

Several factors may explain this disparity. Although a large number of *person* instances might be expected to improve detection performance, the high variability within

this class likely introduces ambiguity, making accurate classification more difficult. In contrast, the less frequent classes may have featured more visually distinct or consistently structured examples, which likely contributed to stronger detection outcomes.

This variation in performance was consistent across the evaluated models. RetinaNet, for instance, showed only limited detection capabilities when compared to the others. In contrast, Faster R-CNN and SSDLite recorded the highest overall scores. While the student version of Faster R-CNN generally performed worse than its baseline, certain classes saw an improvement in AP scores under the student model, though this improvement was not consistent. Similar fluctuations were observed in other models, where student variants outperformed their baselines for some classes and underperformed for others.

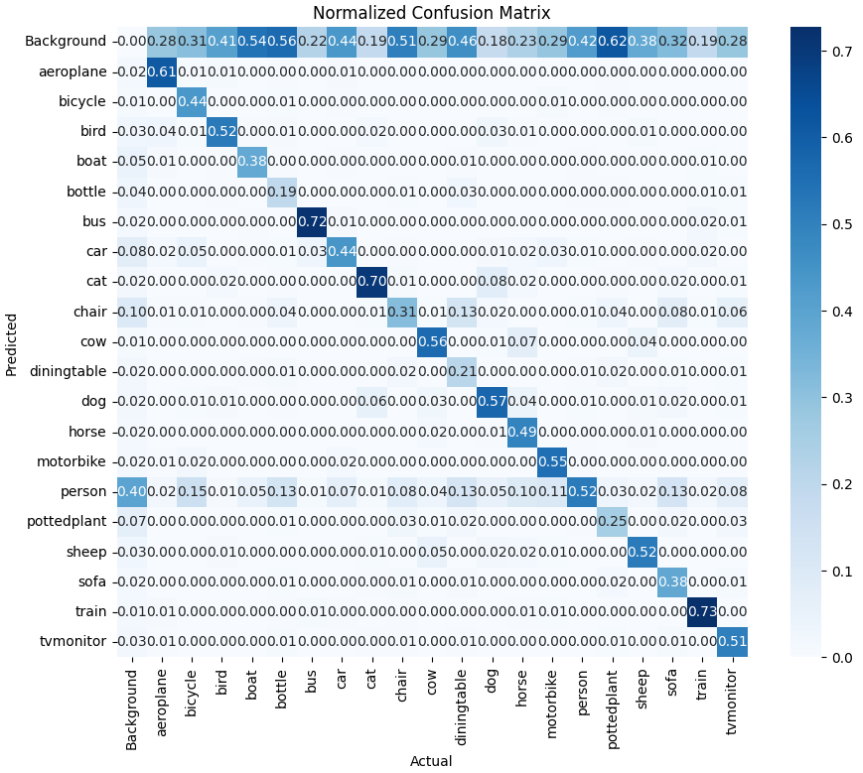
Nevertheless, despite the variation and inconsistency, the broader pattern indicates that student models frequently maintained or slightly improved per-class scores. Given the complexity of detecting across 20 classes, even marginal improvements suggest meaningful progression. This pattern is also evident in the normalised confusion matrix for the best-performing model, SSDLite, shown in Figure 4.15. Here, both the baseline and student versions show a largely diagonal structure, which is expected in effective classification. However, a significant portion of objects remains unclassified, highlighting ongoing challenges with missed detections and a tendency to label uncertain regions as background.

The student model shows some improvement. Its confusion matrix presents clearer diagonal lines across most categories, indicating more accurate predictions. There are also fewer unclassified examples overall. The classes most reliably identified by the student model are *train*, *cat*, and *bus*. These appear to be among the simpler categories to detect, likely because they have more distinct and consistent visual features than others.

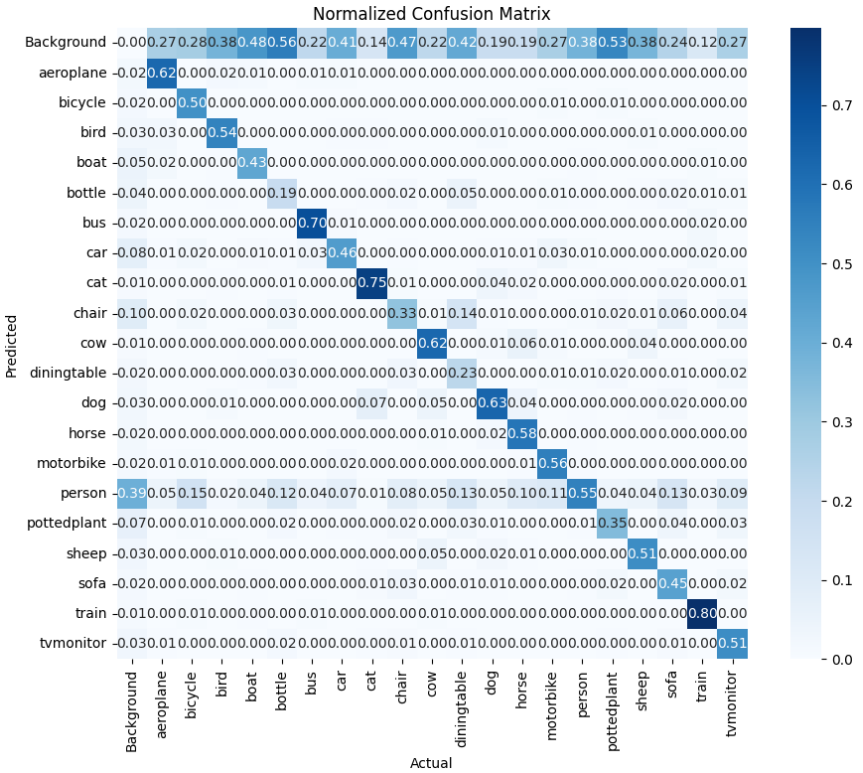
Building on the earlier comparisons, an additional analysis was undertaken to examine the performance of various teacher models, using the same experimental setup as in the previous tests. The outcomes, detailed in Table 4.13, reflect the detection accuracy achieved by each teacher architecture. Interestingly, Faster R-CNN, FCOS, and RetinaNet continued to yield the best results, though their performance metrics were lower than the near-ideal values reported in earlier evaluations.

A notable observation is that, despite their weaker performance as teacher models, both SSD and SSDLite produced relatively robust student models. This outcome contrasts with RetinaNet and FCOS, which, despite achieving better detection scores as teacher models, produced less accurate student versions.

FCOS, in particular, stood out as a strong teacher, closely following Faster R-CNN and even outperforming it in some metrics. A consistent pattern across all teacher models was the high recall values, while precision scores showed greater variation. SSDLite,



(a)



(b)

Figure 4.15 Normalised confusion matrices for multi-label detection of 20 object classes on the Pascal VOC 2012 dataset, using the best-performing models of the SSDLite architecture: (a) baseline model, (b) student model.

Model	mAP@50–95	mAP@50	mAP@75	mAR@1	mAR@10	mAR@100	Precision	Recall	F1 Score
RetinaNet	0.77	0.86	0.79	0.60	0.81	0.81	0.26	0.90	0.38
FCOS	**0.80**	0.88	**0.82**	**0.61**	**0.84**	**0.84**	0.43	**0.91**	0.56
Faster R-CNN	0.77	**0.91**	**0.82**	0.59	0.82	0.82	**0.56**	**0.91**	**0.68**
SSD	0.42	0.56	0.49	0.41	0.48	0.48	0.25	0.69	0.36
SSDLite	0.49	0.61	0.54	0.46	0.55	0.55	0.04	0.79	0.07

Table 4.13 Comparison of teacher model performance across key detection metrics, trained on the Pascal VOC 2012 dataset for multi-label detection of 20 object classes.

in this context, had the lowest precision, whereas the other models achieved values between 0.25 and 0.56. These figures were somewhat lower than anticipated, especially considering the use of privileged information. The results also suggest that increasing the number of classes introduces added complexity, which negatively affects both teacher and student performance across all detection metrics.

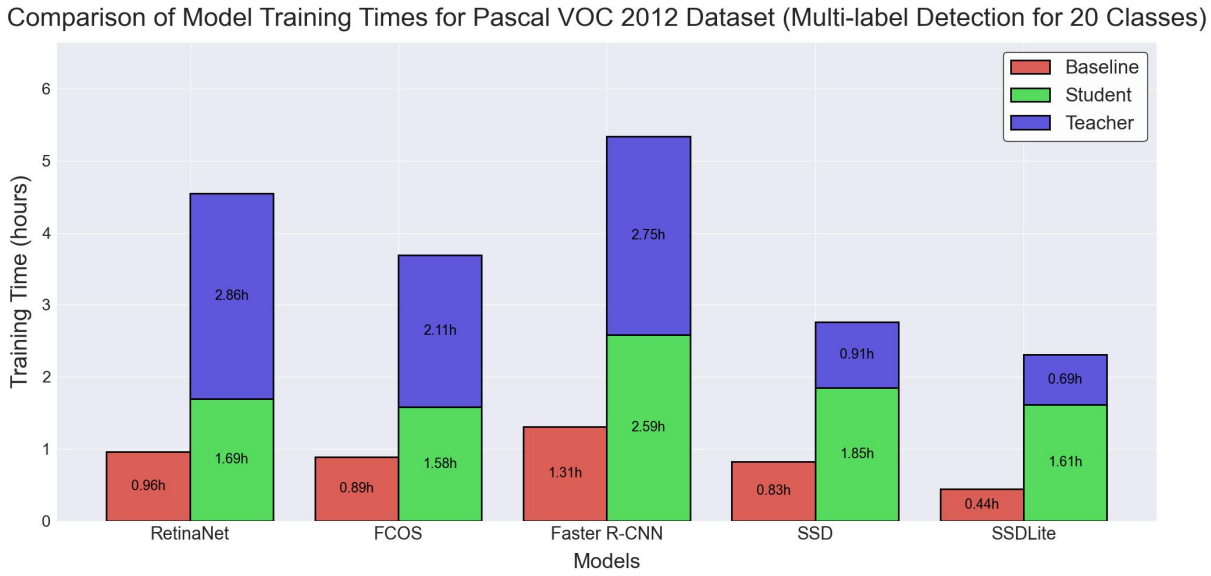


Figure 4.16 Comparison of model training times on the Pascal VOC 2012 dataset for multi-label detection of 20 object classes.

The training times for all models used in this experiment are presented in Figure 4.16. Compared to earlier experiments, these models generally required less time to train. Notably, most models in this study exhibited a more uniform rate of convergence, in contrast to the more varied patterns seen in earlier experiments.

RetinaNet and Faster R-CNN had the longest training times for their student models. On the other hand, SSD and SSDLite, which completed training more quickly, produced the strongest student performances in this experiment. Across all models, student training took longer than the corresponding baselines, while teacher models consistently required the most time overall.

A comparison of model sizes and inference speeds, as detailed in Table 4.14, re-

veals some differences from the findings of previous experiments. The models in this study now feature more parameters and a larger overall size, primarily due to the increased number of classes to 20. This expansion requires additional parameters in the prediction head, which in turn increases the total model size.

Model Configuration	Type	Size (MB)	Parameters (M)	GFLOPS	FPS
Baseline	RetinaNet	124.22	32.56	265.10	39.74
	FCOS	122.48	32.11	252.94	39.55
	Faster R-CNN	157.92	41.40	268.75	30.66
	SSD	100.27	26.29	62.94	67.44
	SSDLite	9.42	2.47	0.95	36.74
Student	RetinaNet	124.22	32.56	265.10	38.00
	FCOS	122.48	32.11	252.94	34.65
	Faster R-CNN	157.92	41.40	268.75	30.39
	SSD	100.27	26.29	62.94	67.67
	SSDLite	9.42	2.47	0.95	36.75

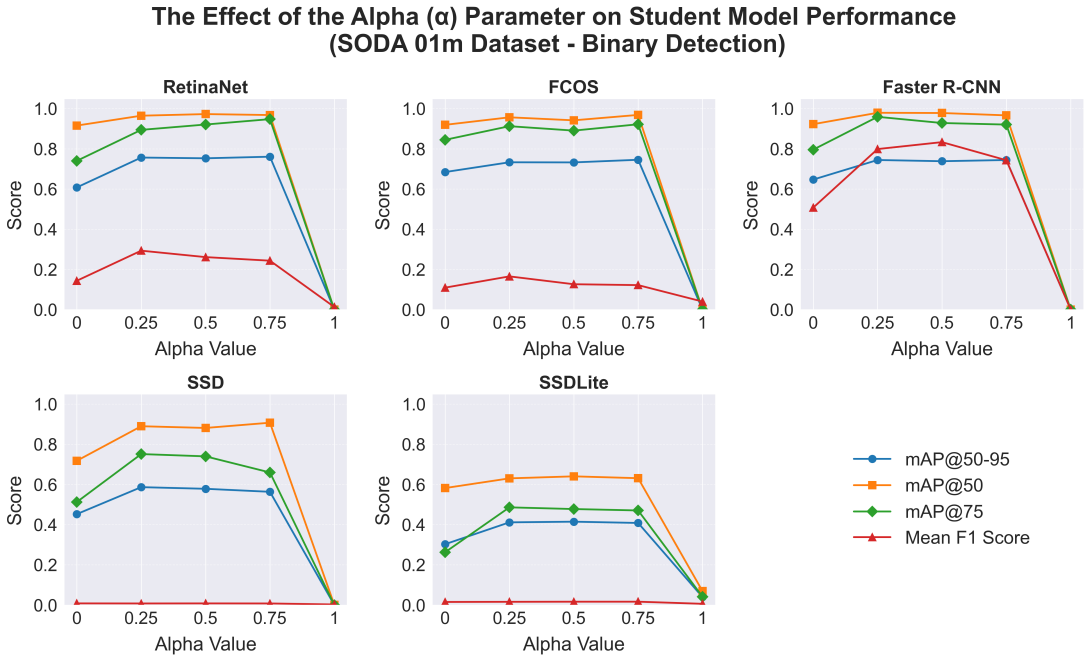
Table 4.14 Comparison of model configurations for trained baseline and student models on the Pascal VOC 2012 dataset for multi-label object detection, including model type, size in megabytes, number of parameters (in millions), computational complexity in GFLOPS, and inference speed in FPS.

Despite this growth, the results demonstrate that applying the proposed methodology to any detector leads to improved detection accuracy. Importantly, this improvement is realised without relying on more complex or computationally intensive architectures. Through various experimental setups, each conducted with strict controls and repeated testing, the method consistently delivered better performance while keeping both model size and parameter count under control, with GFLOPS remaining equivalent and FPS showing similar results during inference.

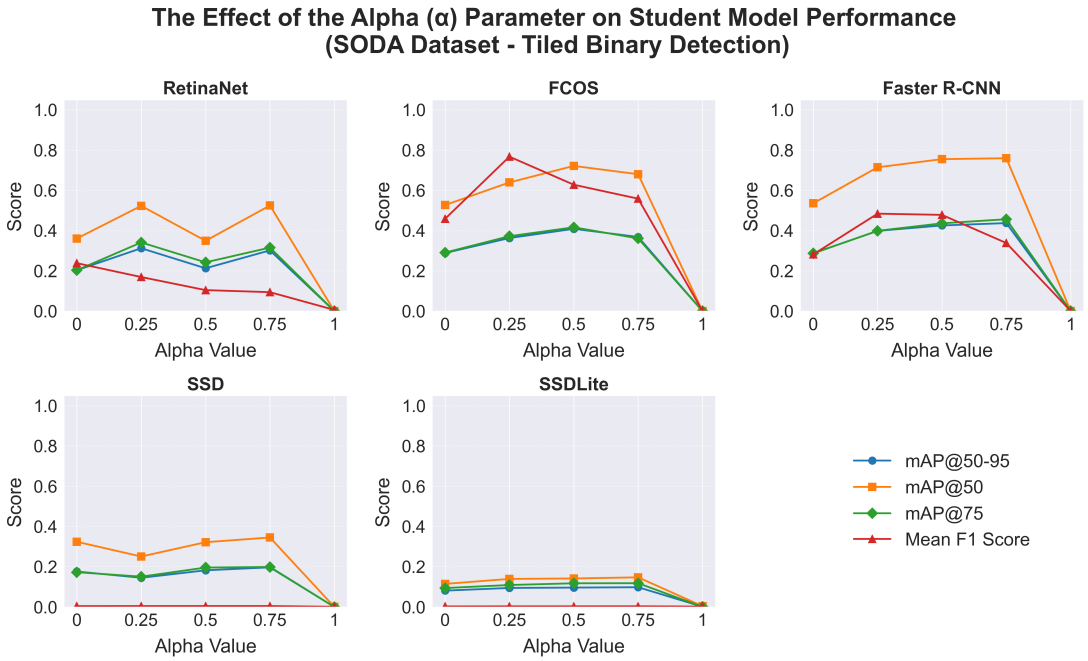
4.9 Ablation Study on the Alpha Parameter

An ablation study was conducted to evaluate the effect of the α parameter, which determines the importance given to the teacher network when distilling knowledge to the student model. Following the approach outlined in [102], α values of 0, 0.25, 0.5, 0.75, and 1 were tested to systematically assess the model's behaviour across the full spectrum from relying solely on labelled data to relying entirely on the teacher's representa-

tions. The results of this study, presented in Figures 4.17 and 4.18, assess which α value best supports the distillation process. These figures compare the outcomes of both the within-dataset experiments and the models trained on the Pascal VOC 2012 dataset.

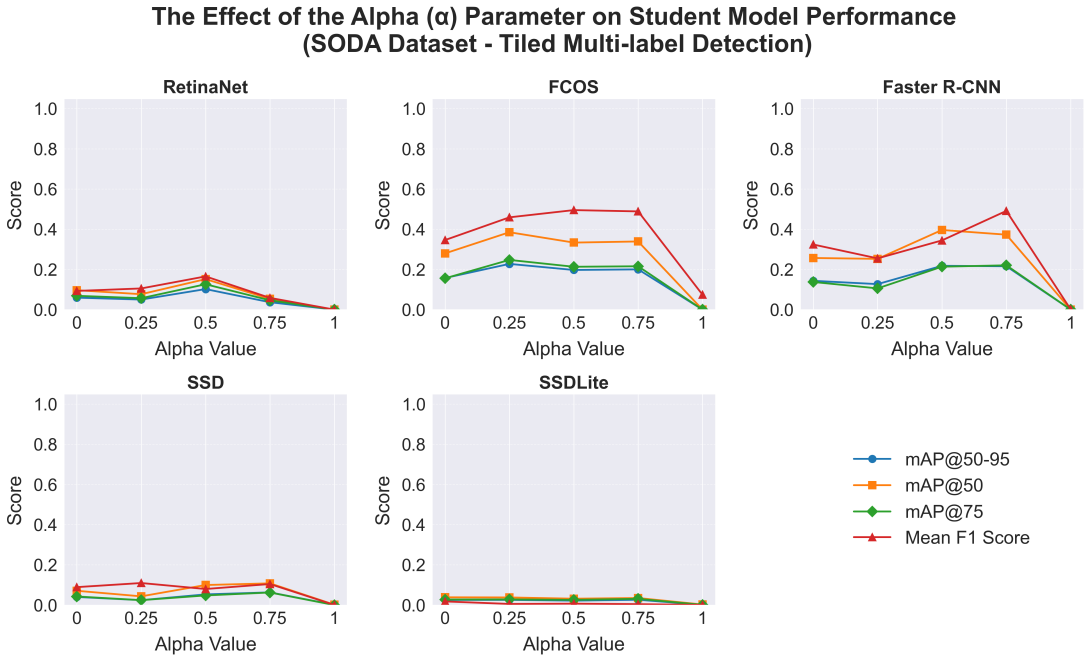


(a)

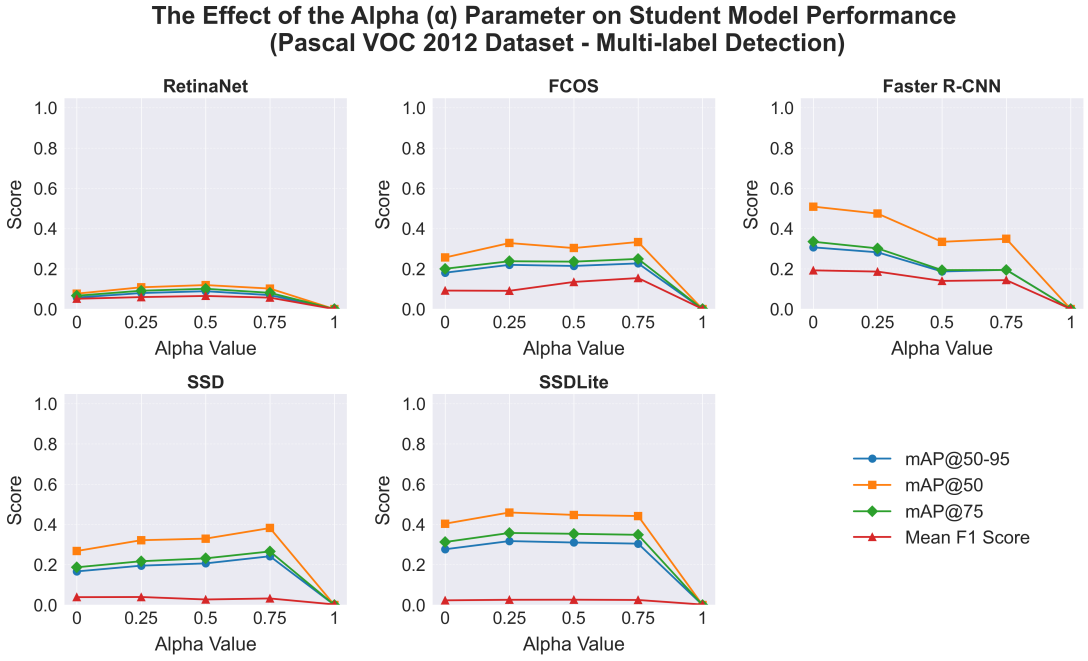


(b)

Figure 4.17 Assessing the impact of the alpha (α) parameter on student model performance. (a) shows results for the SODA dataset at a 1-metre altitude for binary litter detection, while (b) presents results on the 3×3 tiled SODA dataset across all altitudes for binary litter detection.



(a)



(b)

Figure 4.18 Assessing the impact of the alpha (α) parameter on student model performance. (a) shows results on the 3×3 tiled SODA dataset across all altitudes for multi-label litter detection, while (b) presents results on the Pascal VOC 2012 dataset for multi-label object detection.

The principal finding from this analysis is that, across all experiments, the most effective performance in terms of the primary object detection metric (mAP@50–95) was observed when α values were set between 0.25 and 0.5. There were a few isolated

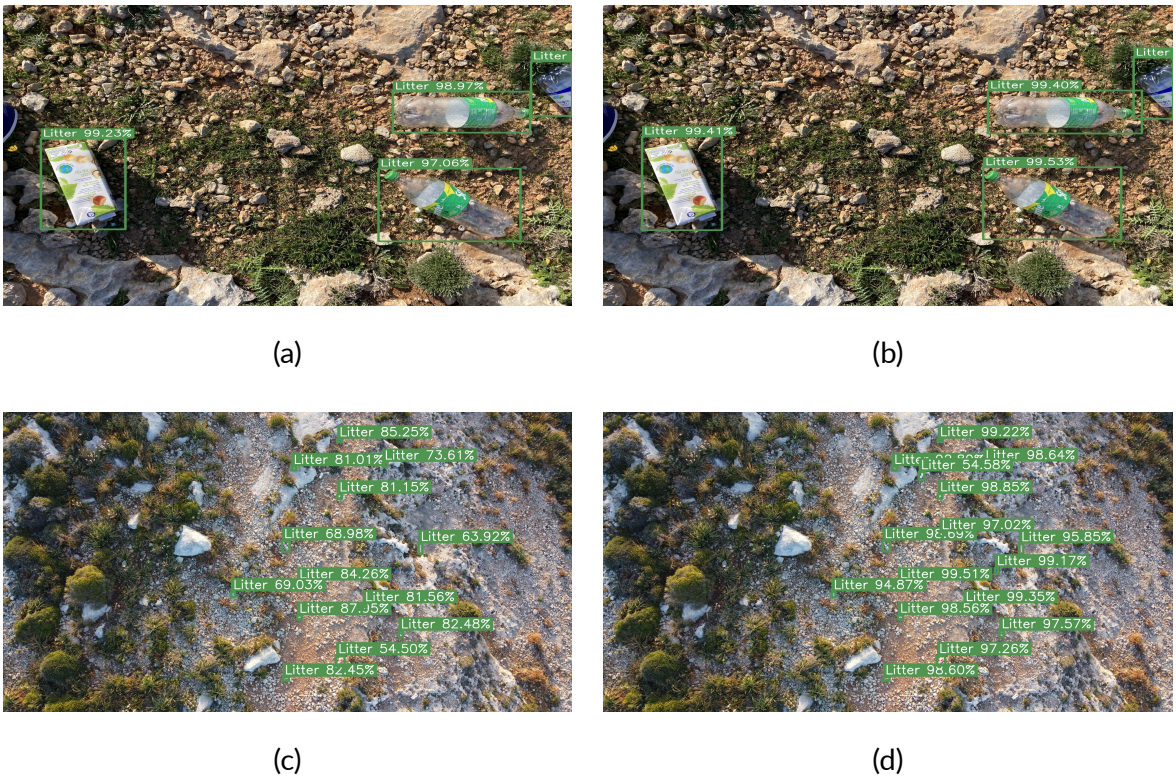
instances in which a value of 0.75 produced comparable results, but these were exceptions rather than the norm. In general, when α was set to 1, which gives full control to the teacher, all metrics showed near-zero scores. For F1 scores, the trend for achieving high object classification performance was observed at α values of 0.25 and 0.5. However, there were a few exceptions in the multi-label litter detection experiment, where FCOS and Faster R-CNN performance favoured α = 0.75.

Further analysis revealed that for RetinaNet, on average, α values of 0.5 and 0.75 yielded the best results, while for FCOS, the optimal values for mAP@50, mAP@75, and F1 scores were similar, with α = 0.25 being optimal for the harder metric, mAP@50–95. For Faster R-CNN and SSD, which followed a similar trend, α values between 0.5 and 0.75 delivered the best results. Finally, for SSDLite, α = 0.5 appeared to be the most effective configuration. That said, this architecture generally delivered weak results across all metrics in the litter detection experiments. However, it did show a modest yet meaningful improvement in the Pascal VOC 2012 evaluation.

Thus, the findings of this ablation study indicate that setting the α parameter between 0.25 and 0.5 for the proposed approach provides the most consistent and effective results across the detection metrics. Employing these values provides the teacher enough say to help without overwhelming the student. When the teacher's influence was set to the maximum (α = 1), performance dropped sharply, showing that too much reliance on the teacher can hold the student back. Although the results align with those presented in [102], tuning α based on the particular task is still necessary. Some models, for instance, performed better on the litter detection tasks when α was set to 0.75, while the same models reached their best performance on Pascal VOC 2012 with a value of 0.25. The appropriate value depends on the nature of the data and the specific problem being addressed.

4.10 Visual Comparison of Model Predictions

In addition to the evaluation outlined in the previous experiments, which is based on established object detection metrics, it is equally valuable to consider whether improvements can be observed visually through the integration of LUPI using the proposed methodology. This form of comparison helps to assess whether the predictions are qualitatively better. To address this, the following section presents visual results across the various experiments, comparing outputs from both within- and across-dataset models, as well as from those trained on the Pascal VOC 2012 dataset. The aim is to compare the visual predictions from the best-performing baseline and student models of the same architecture type, using sample images taken from the test sets of each dataset. The corresponding visual results are illustrated in Figures 4.19, 4.20, 4.21, and 4.22.



(a) (b)

(c) (d)

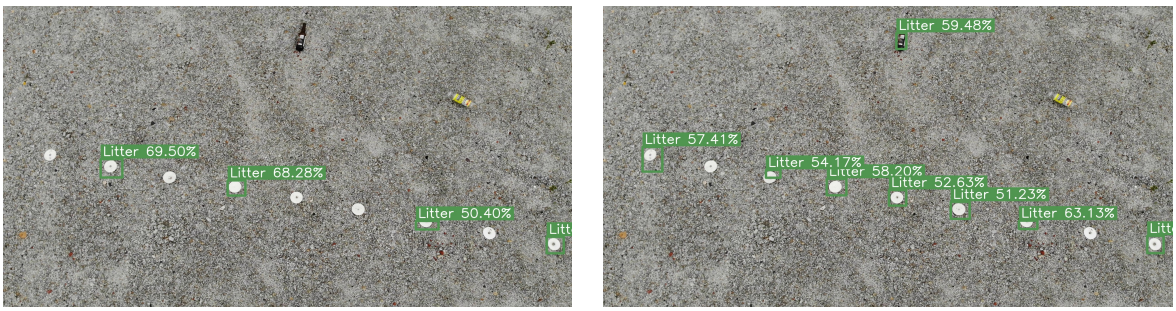
Figure 4.19 Sample predictions on test subset images from detection models trained on the SODA dataset. Subfigures (a) and (c) show outputs from the baseline models, while (b) and (d) present predictions from their corresponding student models. Subfigures (a) and (b) illustrate RetinaNet results at an altitude of 1 metre; (c) and (d) show outputs from Faster R-CNN.

Examining the visual results from the within-dataset experiments, as shown in Figure 4.19, it is evident that the integration of LUPI results in a noticeable improvement over the baseline model. In subfigures (a) and (b), this improvement is clear: although both RetinaNet models successfully detect all litter objects, the LUPI student model produces higher confidence scores, notably in the scenario tailored for detection at a 1-metre altitude. In subfigures (c) and (d), where detection becomes more challenging due to the significantly higher altitude, the improvement achieved by the Faster R-CNN student model is even more evident. It identifies more litter instances than the corresponding baseline and assigns higher confidence scores to its predictions. This further reinforces the practical value of incorporating the proposed approach.

In the cross-dataset evaluation, where models trained on the SODA dataset were tested on other binary litter detection datasets to assess how well the proposed approach generalises, the improvement is also clearly visible. This is illustrated in Figures 4.20 and 4.21. For the BDW dataset, using the best-performing model based on the FCOS architecture, the student model detected more litter instances than the baseline. Similarly, for UAVVaste, where the top-performing model was Faster R-CNN, the student model again identified more objects. In both cases, the continued trend of higher

(a) (b)

Figure 4.20 Sample predictions on test subset images from the FCOS detection model, trained on the SODA dataset and evaluated on the BDW dataset. Subfigure (a) shows outputs from the baseline model, while (b) presents predictions from the corresponding student model.



(a) (b)

Figure 4.21 Sample predictions on test subset images from the best-performing detection model, Faster R-CNN, trained on the SODA dataset and evaluated on the UAVVaste dataset. Subfigure (a) shows outputs from the baseline model, while (b) presents predictions from the corresponding student model.

confidence scores reinforces the notion that integrating LUPI contributes to more effective and robust learning.

A similar trend is evident in the general object detection task with multi-label detection, as demonstrated in the Pascal VOC 2012 experiment. The improvement is especially noticeable when using the best-performing architecture, Faster R-CNN, as shown in Figure 4.22. In the first example, the baseline prediction in (a) has a low detection score, and the bounding box does not fully capture the object of interest. In contrast, the student model produces a bounding box that accurately encloses the object, reflecting a more precise prediction. In the second example, the baseline prediction in (c) performs reasonably well but misses some important elements. It fails to detect the person inside the car and overlooks the car in the background, while also assigning relatively low confidence scores, which negatively impact the mAP. In comparison, the student prediction in (d) correctly identifies all the relevant objects except for the horse in the background. In the third example, the student model correctly predicted both

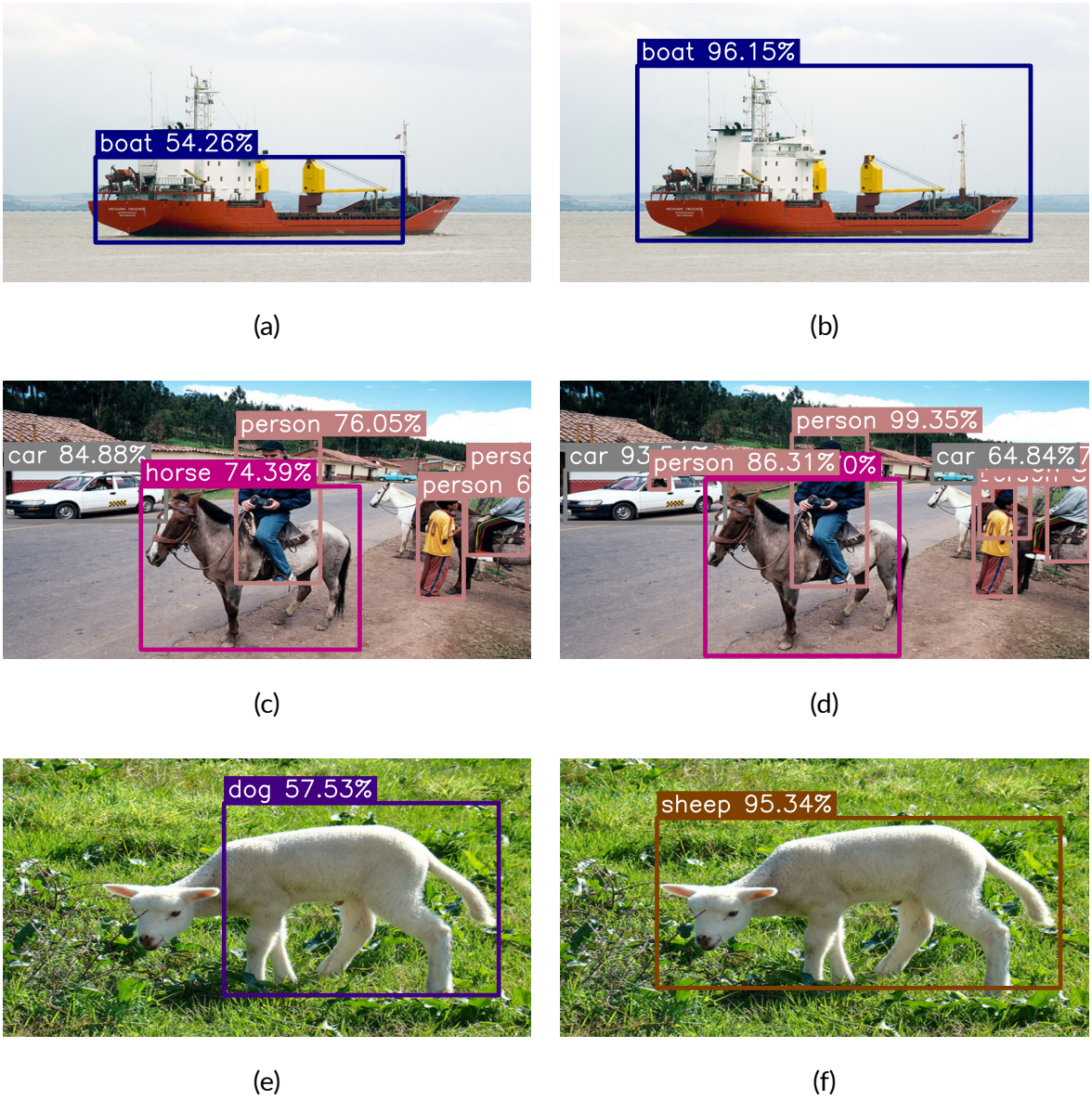


Figure 4.22 Sample predictions from the best-performing detection model, Faster R-CNN, trained on the Pascal VOC 2012 dataset. Subfigures (a), (c), and (e) show outputs from the baseline model, while (b), (d), and (f) present predictions from the corresponding student model.

the type and the bounding box position of (f), while the baseline model in (e) mistakenly labelled the sheep as a dog. This improvement in object coverage, classification and confidence underscores the overall efficacy of the proposed methodology, demonstrating its advantages in both litter detection and general object detection tasks.

4.11 Visual Interpretability of Model Predictions

Additional tests were conducted to evaluate the visual interpretability of how the models make predictions on specific images. To facilitate this, three commonly used class

activation mapping techniques were employed to assess any improvements between the baseline and student models from a feature-based perspective. These methods included Grad-CAM [156], Grad-CAM++ [157], and Eigen-CAM [158].

Grad-CAM and Grad-CAM++ both use a gradient-based approach, where the gradient of a target concept is propagated to the final convolutional layer, resulting in a coarse localisation map that highlights key regions in the image relevant for the prediction. Grad-CAM++ builds on this process by providing clearer visual explanations, improving object localisation and enabling better interpretation of multiple object instances within a single image[156, 157]. Meanwhile, Eigen-CAM utilises principal components derived from the learned features in the convolutional layers, offering a different approach to explaining model predictions [158].

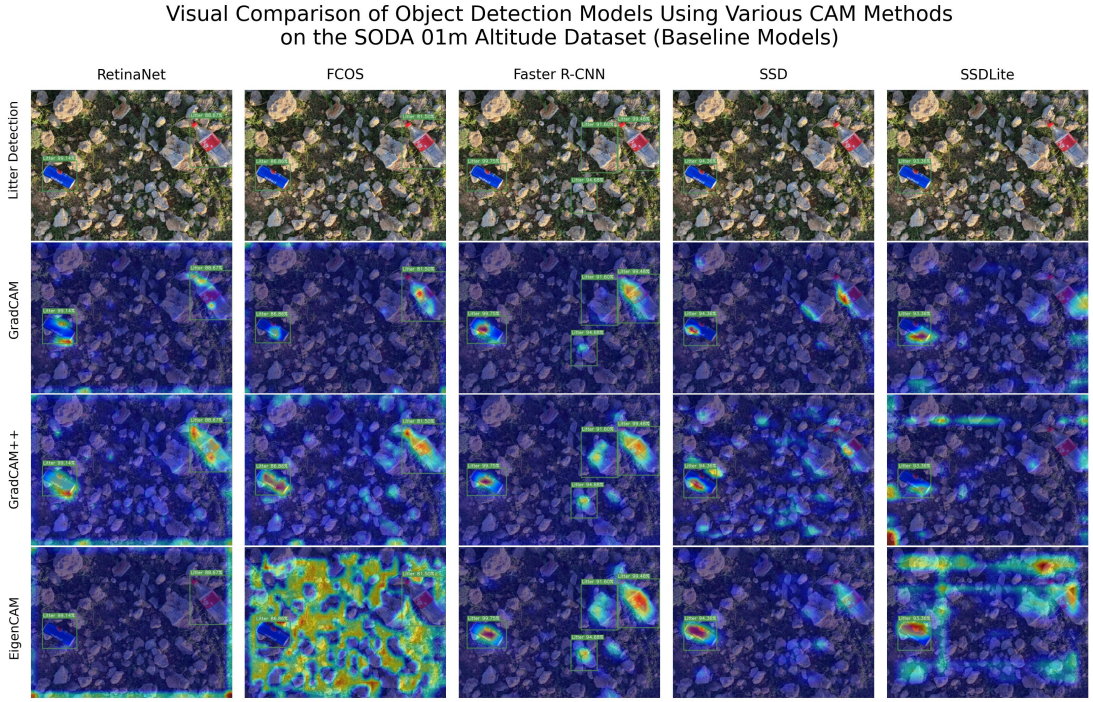
This test was carried out across all selected architecture types, using the specific target layer where the distillation process took place for each model (see Subsection 3.6.2) to examine whether there were visual differences in feature-based learning between the baseline and student models. Although this analysis was conducted on all trained models and across various images for consistency, only the results from models trained on the SODA 1-metre altitude dataset are presented here for brevity. This decision was made as the experiment yielded the most significant improvements under the proposed LUPI-based approach, with the other experiments showing relatively similar results. The visual outcomes of this test are shown in Figure 4.23.

A consistent trend observed in Section 4.10 is also evident here, with the student models generally outperforming the baselines. For example, the Faster R-CNN baseline mistakenly identified rocks as litter, an error that the student model corrected. Likewise, both the SSD and SSDLite baseline models missed certain litter objects, while the student models successfully identified them.

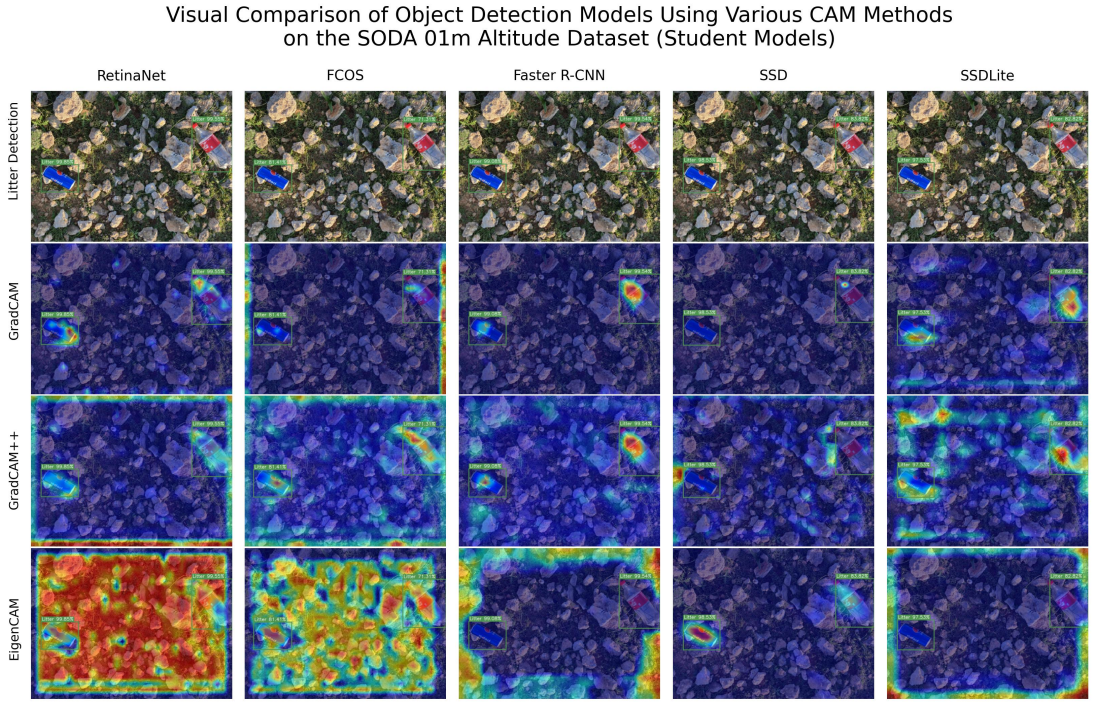
Shifting the focus to the class activation maps, the visual differences between baseline and student models are more subtle. While the activation maps from Grad-CAM, Grad-CAM++, and Eigen-CAM are relatively similar in overall appearance, some meaningful observations can still be made. In the Grad-CAM outputs, even for objects that were not detected by the SSD and SSDLite baselines, some degree of attention was still allocated to those regions. This suggests that although the models failed to classify the objects correctly, they were at least partially aware of their presence.

A consistent pattern across the baselines, particularly in scenes with rocky terrain, was the misallocation of attention. The models often focused on background elements such as rocks, which may have contributed to the misclassifications. In contrast, the student models appeared to suppress this background attention, instead concentrating more directly on the relevant object areas.

In the Grad-CAM++ results, the effect of the challenging backgrounds became even more evident. The baseline models directed substantial attention to the rocky



(a)



(b)

Figure 4.23 Visual comparison of object detection models using various CAM methods on the SODA dataset at a 1-metre altitude for binary litter detection. (a) shows predictions from baseline models; (b) presents predictions from the corresponding student models.

textures rather than the actual litter, which likely impaired their predictions. The student models, however, showed a shift in behaviour: their attention was more evenly spread across the scene but still retained a clear focus on the objects of interest, avoiding the distractive influence of the background.

Finally, the Eigen-CAM results also show a difference in attention. The student models spread their focus more evenly across the entire image, including the object regions, without being sidetracked by the background. In contrast, the baseline models focused more on the litter objects. However, in this case, the baseline models did give more attention to the specific litter, while the student models seemed to distribute their attention more widely, including the background.

The results across the remaining datasets followed consistent patterns, reinforcing the conclusion that the proposed LUPI-based method improves detection accuracy while helping the model focus more effectively. This ability to shift focus away from irrelevant background features and towards the objects of interest, while in cases of challenging backgrounds, dispersing attention across the whole image instead of fixating on specific background elements, appears to be a crucial factor in the improved performance of the student models.

4.12 Limitations in Privileged Information Generation

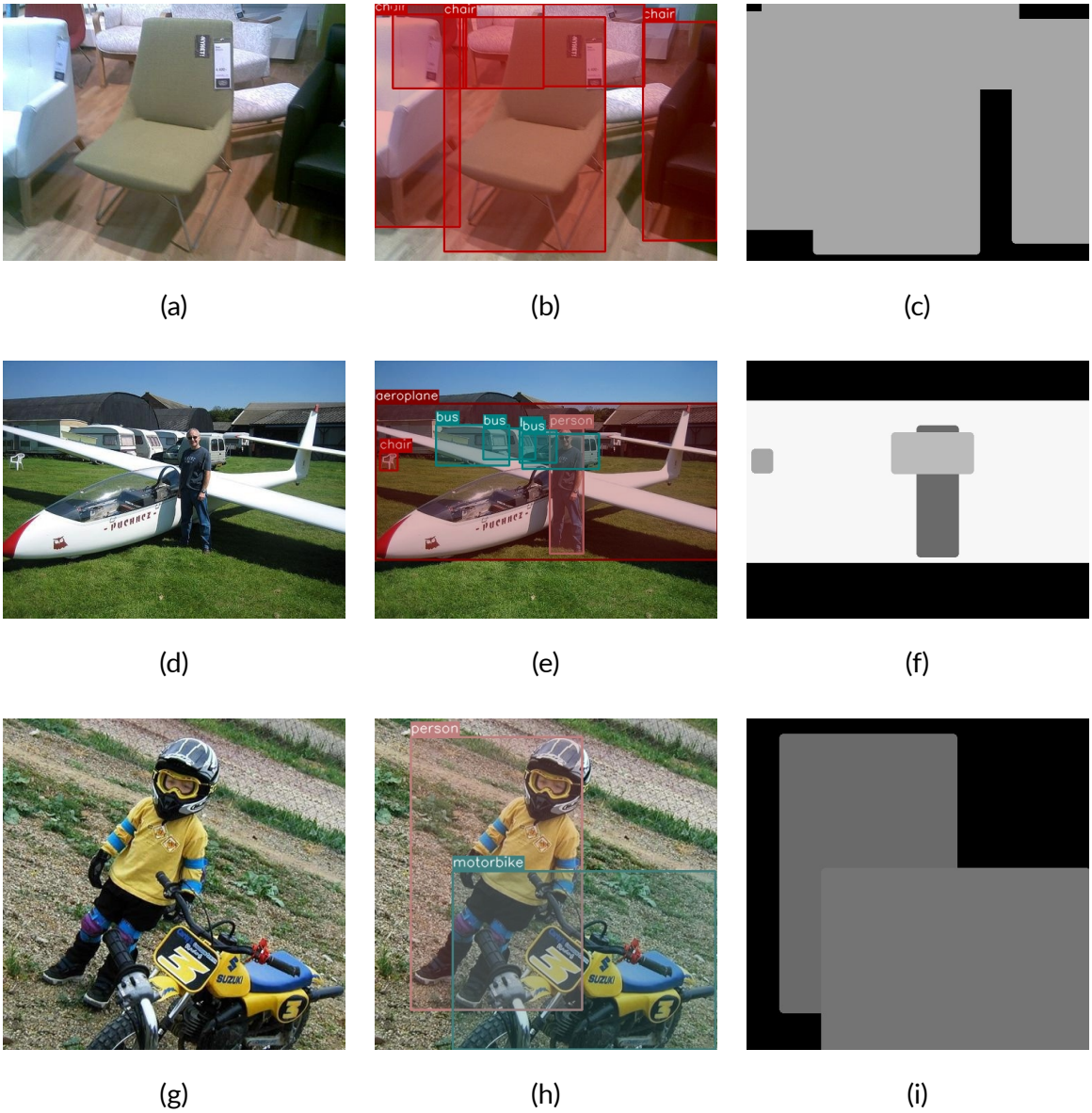
While it has been demonstrated that leveraging privileged information in object detection offers significant benefits, especially with simplified modifications and lightweight alternatives to complex architectures, it is also prudent to consider the limitations associated with this approach. One of the primary challenges lies in generating the privileged information itself. As previously acknowledged, the increased training time required for this method, compared to excluding privileged information, is a key drawback. However, there are still issues in effectively representing certain objects through this information.

These limitations can primarily be broken down into three key challenges: (i) overlapping objects of the same type, (ii) large objects obscuring smaller ones, and (iii) difficulties in distinguishing between object classes in datasets with a large number of categories. In the case of UAV-based litter detection, where litter objects are relatively small and spaced apart (as seen in Figure 3.1), this method performs well. However, in scenarios with overlapping objects, the challenge of encoding privileged information becomes more pronounced.

Figure 4.24 highlights several challenges in generating privileged information from the Pascal VOC 2012 dataset. In the first row, scene (a) shows an arrangement of several chairs placed in close proximity. The bounding box view (b) reveals significant overlaps between neighbouring chairs due to their similar positioning and orientation. When

translated into the privileged mask (c), these overlapping regions merge into a single connected shape, obscuring the boundaries between objects. This merging can lead the model to interpret separate chairs as a single instance, creating ambiguity in training and reducing its capacity to distinguish multiple objects of the same category.

The second row illustrates the impact of occlusion by large objects. Scene (d) depicts a man standing in front of an aeroplane, with several buses positioned behind it, none entirely obscured. The bounding boxes in (e) show that parts of these buses intersect with the aeroplane's bounding box. In the privileged mask (f), the bounding boxes are processed in descending order of size, with the largest first, as described in



(a) (b) (c)

(d) (e) (f)

(g) (h) (i)

Figure 4.24 Examples of failure cases in privileged information generation. Subfigures (a), (d), and (g) show original images from the Pascal VOC 2012 dataset, while (b), (e), and (h) display the corresponding images with overlayed ground truth annotations. Subfigures (c), (f), and (i) present the privileged information for these cases.

Algorithm 1; however, some of the buses are still partially obscured by the plane's mask in the privileged information. This limitation becomes even more pronounced in datasets such as COCO, where dense scenes frequently contain many small objects overlapping or clustered together, increasing the likelihood of important features being masked out.

The third row demonstrates limitations related to colour encoding. Scene (g) presents a boy standing beside a motorcycle, with bounding boxes in (h) that touch at their edges with some minor overlapping. The privileged mask (i) assigns each object a different colour; however, the chosen colours are so similar that they are difficult to distinguish. When a dataset contains many categories, the number of distinct colours available for a grayscale mask becomes insufficient, leading to such near-duplicate shades being reused. This can create ambiguity, especially where objects are adjacent, as their categories may be visually indistinguishable in the mask. While employing RGB masks could allow each category to be assigned a unique colour, this approach would substantially increase both training time and computational requirements due to the greater input dimensionality.

Conclusively, these examples highlight the limitations inherent in generating privileged information, particularly in encoding bounding box data for object detection networks. The difficulties become more pronounced in scenarios involving overlapping objects or complex scenes with numerous categories. While strategies such as sorting or expanding the colour pool may alleviate some of these issues, there remains a need for further investigation into how this method can be generalised for larger, more intricate use cases. Nevertheless, the advantages of employing this approach significantly outweigh its limitations, suggesting that continued research in this area is both necessary and valuable.

4.13 Discussion

After carefully reviewing the experimental results, it is appropriate to consider how they relate to the central research question. This study aimed to explore the feasible integration of the LUPI paradigm into object detection frameworks and to assess the practical viability of this approach. To support this investigation, four specific objectives were drawn up. The first objective (**O1**) focused on evaluating the real-world applicability of the method, using the challenging use case of litter detection as an example of small object detection. The second objective (**O2**) centred on implementing and testing the approach across a broad selection of established object detection models, to examine whether the proposed strategy could generalise across architectures with different design characteristics.

Objectives **O3** and **O4** broadened the scope of evaluation. These objectives fo-

cused on assessing the method within a structured experimental framework, considering both intra-dataset and cross-dataset performance. The intent was to examine its effectiveness under progressively challenging conditions and varying environmental factors. Particular emphasis was placed on the trade-offs between detection accuracy and computational efficiency, both of which are central to assessing practical viability. The results were consistent and encouraging. In three demanding within-dataset scenarios involving litter detection from UAV imagery, student models trained using the LUPI paradigm consistently outperformed baseline models, achieving clear improvements in detection accuracy.

Although the extent of improvement varied depending on the complexity of the task, with more modest increases observed in more challenging scenarios, the approach continued to demonstrate success, showing improvements ranging from 0.02 to 0.15 in the strict mAP@50–95 metric. Preliminary experiments also explored the selection of optimal tiling parameters, contributing to the broader aim of determining feasibility within UAV-based litter detection use cases. Further evaluation focused on the method's ability to generalise to external datasets, specifically BDW and UAVVaste. These experiments once again confirmed that incorporating privileged information resulted in meaningful performance benefits. This outcome remained consistent even in cross-dataset evaluations, a setting where object detection models frequently encounter difficulties in maintaining accuracy.

The Pascal VOC 2012 dataset was subsequently employed to assess the method across a wider variety of object classes. The results demonstrated that the proposed approach maintained robustness even in more general-purpose object detection scenarios. Throughout the full set of experiments, five commonly used architectures were selected: Faster R-CNN, SSD, RetinaNet, SSDLite, and FCOS. For each architecture, three model variants were trained: a baseline, a teacher, and four student models developed within the LUPI framework. Certain architectures, including RetinaNet and FCOS, yielded better results in UAV-based litter detection tasks, whereas SSDLite and SSD achieved better performance in the Pascal VOC evaluation. Faster R-CNN demonstrated consistent performance across both contexts, although its relative advantage varied depending on the dataset and detection challenge. This variation can be attributed to the presence of overlapping objects in the privileged information, which may have confused the RPN. Moreover, these outcomes suggest that differences in baseline performance are likely driven by the degree to which each architecture is suited to the characteristics of the target task, rather than by any inherent constraint of the proposed LUPI approach.

In total, *120* object detection models were trained and evaluated, which made it possible to conduct an ablation study on the α parameter to identify optimal conditions for student model learning. Although the proposed approach leads to longer training times and some minor limitations in the privileged information generation, it does not

impact the model size, the number of parameters, or the computational speed. As a result, lightweight models remain suitable, as they can deliver improved accuracy while maintaining efficiency during inference. Since inference is typically performed far more frequently than training in most deployment scenarios, this trade-off is both practical and worthwhile.

Finally, visual outputs and feature-based analyses indicate that models trained with the proposed method can detect a greater number of objects with higher confidence. These models also exhibit stronger feature representations, further supporting the robustness of the approach. Overall, the findings suggest that the research question has been answered with a reasonable degree of confidence. The method demonstrates practical feasibility, with quantifiable performance boosts in detection accuracy. These improvements are especially significant in scenarios where constraints on model size and inference speed are imperative, such as in UAV-based small object detection or other similar object detection applications.

4.14 Conclusion

This chapter began by outlining the hardware setup and evaluation strategy that underpinned the experiments. It then moved on to the optimal tiling experiments conducted on the SODA dataset, which were essential as a preliminary step in achieving the study's objectives. The subsequent section focused on the within-dataset evaluation of UAV-based litter detection, covering three experiments performed on the SODA dataset. The chapter then shifted to the cross-dataset evaluation, where experiments on the BDW and UAVVaste datasets were carried out to assess the generalisability of the trained models. Further analysis explored how detection performance varied with object size, concentrating on whether the proposed LUPI methodology offered improvements in recognising small-scale objects. Following this, the Pascal VOC 2012 evaluation tested the method's ability to handle multi-label detection across a wide range of categories. An ablation study on the alpha parameter was then presented, along with a visual comparison of model predictions and an interpretability analysis. The chapter also addressed the limitations encountered in generating the privileged information, concerning the bounding boxes used to augment the feature representations. The chapter concluded with a synthesis of the results, confirming that the research question has been addressed with a reasonable degree of confidence in light of the stated objectives.

Chapter 5 Conclusion

"The end of one journey is simply the beginning of another."

– A. A. Milne

5.1 Summary and Revisited Objectives

Object detection remains a central problem within computer vision, supporting applications ranging from medical diagnostics [2] and autonomous systems [4] to environmental monitoring [6], including the detection and classification of litter [13]. Recent advances have produced deeper and more complex models capable of identifying intricate visual patterns. However, improvements in detection accuracy are often accompanied by greater computational demands [9, 10], which raises concerns about scalability and energy efficiency, especially in resource-constrained settings.

Despite the abundance of object detection architectures made available through the success of deep learning, the prevailing trend of increasing accuracy via added model complexity remains dominant. Yet, the object detection task itself poses unique challenges: it requires the identification of objects at varied scales, across cluttered or occluded backgrounds, often under conditions of class imbalance and sparse data distribution. These complexities call for alternative strategies that do not rely solely on enlarging model capacity.

This dissertation explored the feasibility of adapting the LUPI paradigm to object detection models, focusing on its potential for improving accuracy without increasing model size or computational overhead. Such an approach addresses a common trade-off seen in many state-of-the-art models [67, 91]. Although LUPI has shown promise in other vision tasks, such as classification, its application to object detection has remained largely unexplored. This gap served as the foundation for the research question: can LUPI be effectively integrated into object detection models, and if so, how feasible and generalisable is such an approach?

To meet objective **O1**, this work proposed both a theoretical framework and a practical approach for incorporating privileged information, specifically in the form of bounding box masks, into the object detection pipeline. This was achieved through a teacher–student network design, where the teacher model receives additional information during training and distils this knowledge to the student network, which is then

used at inference.

Objective **O2** was addressed by applying the proposed methodology across five widely used object detection architectures: Faster R-CNN, SSD, RetinaNet, SSDLite, and FCOS. These architectures were chosen to cover both one-stage and two-stage detection paradigms, offering a comprehensive testbed for evaluating adaptability.

To evaluate objective **O3**, extensive experiments were conducted using both within-dataset and cross-dataset evaluation. The SODA dataset served as the primary benchmark in the context of UAV-based litter detection, while BDW and UAVVaste datasets were employed to assess generalisation performance. An additional optimal tiling experiment was also carried out as a prerequisite for achieving this objective, serving as a dataset pre-processing step to improve model robustness in cluttered or wide-area aerial imagery.

Objective **O4** involved assessing the trade-off between detection accuracy and computational cost, as well as examining broader applicability. This was accomplished through further evaluation on the Pascal VOC 2012 dataset, a general-purpose detection benchmark. The findings showed that student models trained using the LUPI-based approach consistently outperformed their respective baseline counterparts, achieving higher accuracy while maintaining the same model size. Inference time remained constant across both student and baseline models, although the addition of the teacher network increased training time.

Across all experiments, a total of *120* models were trained and evaluated. These included baseline and student variants across all selected architectures and datasets. Results confirmed that integrating LUPI into object detection workflows leads to performance improvements without inflating model complexity or inference latency. The baseline models remained unaltered in architecture and size, enabling a direct and fair comparison with the LUPI-trained student models. An ablation study on the distillation coefficient (α) was also conducted to analyse its influence on learning stability and performance. Additionally, visual comparisons of detection results were presented to illustrate qualitative improvements and highlight the interpretability of the student models.

In summary, all four objectives have been successfully met. The findings confirm that LUPI presents a viable approach for improving object detection performance in computationally constrained scenarios, without increasing model depth or parameter count, and without affecting inference time.

5.2 Applications

Whilst the proposed methodology is broadly applicable and can be integrated into virtually any object detection framework, its strength lies in improving detection accuracy

without increasing model size or extending inference time. This makes it highly suitable for deployment in contexts where lightweight models and efficient computation are essential. The following are prominent examples where such qualities are especially beneficial:

UAV-based Litter Detection and Geolocation Systems As demonstrated in this study, UAV-based litter detection systems rely on fast, accurate detection under tight hardware constraints. These applications demand real-time processing for the identification of small litter objects scattered across complex terrains. Within the Aerial Waste Identification and Geolocation System (AWIGS) Project [159], this requirement is exemplified by the need to process large volumes of UAV imagery through lightweight AI models, verify detections through human oversight, and generate precise geolocation coordinates to enable the rapid deployment of cleanup teams to inaccessible areas. Such operations must be completed within minutes of data acquisition, underscoring the necessity for models that combine high accuracy with low computational overhead. Inspired by the AWIGS Project, the proposed approach supports this need by increasing accuracy without imposing a heavier computational load.

Traffic Monitoring Systems Real-time traffic analysis systems must detect vehicles, pedestrians, and infrastructure under varying and dynamic conditions, including fluctuations in lighting and changes in weather. Resolution presents an additional challenge for object detection models, as the use of computationally expensive architectures combined with high-quality imagery does not always guarantee superior results. In certain cases, lightweight models have demonstrated comparatively better performance than more resource-intensive counterparts, as noted in [26]. Incorporating the LUPI approach can further improve detection accuracy while retaining compact model architectures, making it a viable choice for embedded systems that cannot accommodate large and complex networks [160].

Video Surveillance Monitoring Surveillance applications operate continuously and often on resource-constrained devices. Object detection must be responsive and consistent, especially in crowded scenes or poor lighting. The proposed LUPI-based method offers more reliable detection outcomes while preserving computational efficiency, supporting long-term deployment in such systems [161].

Underwater Exploration Systems Tasks such as underwater archaeology or marine ecosystem monitoring are frequently limited by low visibility and constrained onboard processing. By improving accuracy without adding complexity, the proposed method is well-suited for object detection in underwater environments, where it can be implemented using compact models on submersible platforms [162].

5.3 Limitations and Future Work

Limitations

While the proposed method offers clear benefits in detection performance, it also introduces specific challenges that may affect its usability in real-world scenarios. The points below highlight areas where practical constraints or methodological assumptions may limit adoption.

Training Overhead A major drawback of the proposed approach lies in the additional training complexity introduced by the teacher–student setup. The need to train an auxiliary model, alongside processing and integrating privileged information, results in significantly longer training times. While this is acceptable for offline applications, it presents a barrier for time-sensitive or resource-limited deployments.

Privileged Information Availability For many object detection tasks, privileged data is not naturally available. It often needs to be constructed or simulated, which can be time-consuming and labour-intensive. In some cases, such as with hyperspectral data, acquiring privileged information can be especially challenging. This limits the method's practical applicability to scenarios where such data is either readily available or justifiable due to high-performance demands.

Design Constraints on Privileged Inputs The effectiveness of privileged information largely depends on the creativity of the researcher. For datasets such as Pascal VOC and COCO, there is no predefined standard for how this supplementary data should be structured. Its design needs to be customised for the specific task, meaning the success of the approach often depends on the researcher's ability to create relevant and contextually appropriate representations.

Future Work

Although the current implementation shows promise, several directions remain open for refinement and expansion. The following considerations outline how the method could be developed further to broaden its impact and application.

Exploring Advanced Architectures Future studies could examine how this method performs when applied to more recent state-of-the-art object detection frameworks. Architectures such as YOLOv12 [67], DETR [9], and RT-DETR [10] offer diverse design philosophies that test current assumptions regarding which forms of knowledge can be transferred most effectively from teacher to student. Testing across

these frameworks may highlight new strengths or surface weaknesses in the LUPI formulation.

Improved Encoding Strategies Further refinement is needed in how privileged information is presented to the teacher. The bounding box mask used in this study is just one of many possible representations. Future work may explore semantic maps, attention hints, or other structured cues that better capture object relationships or contextual dependencies within the scene.

Expanded Knowledge Distillation Techniques One promising direction is modifying how the teacher's knowledge is transferred. Rather than focusing solely on intermediate feature maps, the distillation process could be expanded to influence directly both the classification and regression branches. This may improve the alignment between teacher guidance and student predictions, especially for harder-to-detect objects.

Architecture-specific Optimisation The current study used consistent training parameters across all models to ensure uniformity. However, fine-tuning hyperparameters for each architecture could lead to further improvements in performance. This includes adjusting learning rate schedules, exploring different data augmentation techniques, and modifying loss function weights, which could bolster training results.

Extension to Segmentation Tasks The method may also be adapted for dense prediction problems such as semantic or instance segmentation. Since segmentation requires high spatial precision [80], incorporating privileged information could support the network in learning subtle spatial cues without increasing its inference footprint.

References

[1] N. Micallef, C. J. Debono, D. Seychell, and C. Attard, "Automatic detection of covid-19 pneumonia in chest computed tomography scans using convolutional neural networks," in *2022 IEEE 21st Mediterranean Electrotechnical Conference (MELECON)*, 2022, pp. 1118–1123. DOI: 10.1109/MELECON53508.2022.9843100.

[2] N. Micallef, D. Seychell, and C. Bajada, "Exploring the u-net++ model for automatic brain tumor segmentation," *IEEE Access*, vol. PP, pp. 1–1, Sep. 2021. DOI: 10.1109/ACCESS.2021.3111131.

[3] S. Abdul-Khalil, S. Rahman, S. Mutalib, S. Kamarudin, and S. Kamaruddin, "A review on object detection for autonomous mobile robot," *IAES International Journal of Artificial Intelligence (IJ-AI)*, vol. 12, p. 1033, Sep. 2023. DOI: 10.11591/ijai.v12.i3.pp1033-1043.

[4] E. Martínez and A. P. del Pobil, "Object detection and recognition for assistive robots," *IEEE Robotics & Automation Magazine*, vol. PP, pp. 1–1, Mar. 2017. DOI: 10.1109/MRA.2016.2615329.

[5] S. Li, H. Zhang, and F. Xu, "Intelligent detection method for wildlife based on deep learning," *Sensors (Basel)*, vol. 23, no. 24, p. 9669, Dec. 2023. DOI: 10.3390/s23249669.

[6] A. M. Roy, J. Bhaduri, T. Kumar, and K. Raj, "Wildect-yolo: An efficient and robust computer vision-based accurate object localization model for automated endangered wildlife detection," *Ecological Informatics*, vol. 75, p. 101 919, 2023, ISSN: 1574-9541. DOI: https://doi.org/10.1016/j.ecoinf.2022.101919.

[7] P. F. Proença and P. Simões, "Taco: Trash annotations in context for litter detection," *arXiv preprint arXiv:2003.06975*, 2020.

[8] D. Bashkirova *et al.*, "Zerowaste dataset: Towards deformable object segmentation in cluttered scenes," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, Jun. 2022, pp. 21 147–21 157.

[9] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," *CoRR*, vol. abs/2005.12872, 2020. arXiv: 2005.12872.

[10] Y. Zhao *et al.*, *Detrs beat yolos on real-time object detection*, 2024. arXiv: 2304.08069 [cs.CV].

[11] Z. Li *et al.*, "Development and challenges of object detection: A survey," *Neurocomputing*, vol. 598, p. 128 102, 2024, ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2024.128102.

[12] Z. Li *et al.*, "Development and challenges of object detection: A survey," *Neurocomputing*, vol. 598, p. 128 102, 2024, ISSN: 0925-2312. DOI: https://doi.org/10.1016/j.neucom.2024.128102.

[13] D. Pisani, D. Seychell, C. J. Debono, and M. Schembri, "Soda: A dataset for small object detection in uav captured imagery," in *2024 IEEE International Conference on Image Processing (ICIP)*, 2024, pp. 151–157. DOI: 10.1109/ICIP51287.2024.10647335.

[14] V. Vapnik and A. Vashist, "A new learning paradigm: Learning using privileged information," *Neural networks : the official journal of the International Neural Network Society*, vol. 22, pp. 544–57, Jul. 2009. DOI: 10.1016/j.neunet.2009.06.042.

[15] S. Wang, S. Chen, T. Chen, and X. Shi, "Learning with privileged information for multi-label classification," *Pattern Recognition*, vol. 81, pp. 60–70, 2018, ISSN: 0031-3203. DOI: https://doi.org/10.1016/j.patcog.2018.03.033.

[16] V. Sharmanska, N. Quadrianto, and C. H. Lampert, "Learning to transfer privileged information," *CoRR*, vol. abs/1410.0389, 2014. arXiv: 1410.0389.

[17] J. Wang, W. Guo, T. Pan, H. Yu, L. Duan, and W. Yang, "Bottle detection in the wild using low-altitude unmanned aerial vehicles," in *2018 21st International Conference on Information Fusion (FUSION)*, 2018, pp. 439–444. DOI: 10.23919/ICIF.2018.8455565.

[18] M. Córdova *et al.*, "Litter detection with deep learning: A comparative study," *Sensors*, vol. 22, no. 2, 2022, ISSN: 1424-8220. DOI: 10.3390/s22020548.

[19] M. Kraft, M. Piechocki, B. Ptak, and K. Walas, "Autonomous, onboard vision-based trash and litter detection in low altitude aerial images collected by an unmanned aerial vehicle," *Remote Sensing*, vol. 13, no. 5, 2021, ISSN: 2072-4292. DOI: 10.3390/rs13050965.

[20] D. Pisani and D. Seychell, "Detecting litter from aerial imagery using the soda dataset," in *2024 IEEE 22nd Mediterranean Electrotechnical Conference (MELECON)*, 2024, pp. 897–902. DOI: 10.1109/MELECON56669.2024.10608507.

[21] Y. Liu, P. Sun, N. Wergeles, and Y. Shang, "A survey and performance evaluation of deep learning methods for small object detection," *Expert Systems with Applications*, vol. 172, p. 114 602, 2021, ISSN: 0957-4174. DOI: https://doi.org/10.1016/j.eswa.2021.114602.

[22] K. Tong, Y. Wu, and F. Zhou, "Recent advances in small object detection based on deep learning: A review," *Image and Vision Computing*, vol. 97, p. 103 910, 2020.

[23] United Nations, *Transforming our world: The 2030 agenda for sustainable development*, General Assembly resolution A/RES/70/1, 2015. [Online]. Available: https://sdgs.un.org/2030agenda.

[24] K. Pilz, L. Heim, and N. Brown, *Increased compute efficiency and the diffusion of ai capabilities*, 2024. arXiv: 2311.15377 [cs.CY].

[25] M. Everingham, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman, *The PASCAL Visual Object Classes Challenge 2012 (VOC2012) Results*, http://www.pascal-network.org/challenges/VOC/voc2012/workshop/index.html, 2012.

[26] M. Bugeja, M. Bartolo, M. Montebello, and D. Seychell, "Mrtmd: A multi-resolution dataset for evaluating object detection in traffic monitoring systems," *IEEE Access*, vol. 13, pp. 134 460–134 483, 2025. DOI: 10.1109/ACCESS.2025.3585986.

[27] M. Bartolo, K. Makantasis, and D. Seychell, "Learning using privileged information for litter detection," in *Proceedings of the 2025 13th European Workshop on Visual Information Processing (EUVIP)*, Valletta, Malta, 2025.

[28] H. Harzallah, F. Jurie, and C. Schmid, "Combining efficient object localization and image classification," in *2009 IEEE 12th International Conference on Computer Vision*, 2009, pp. 237–244. DOI: 10.1109/ICCV.2009.5459257.

[29] G. Chen *et al.*, "A survey of the four pillars for small object detection: Multiscale representation, contextual information, super-resolution, and region proposal," *IEEE Transactions on Systems, Man, and Cybernetics: Systems*, vol. 52, no. 2, pp. 936–953, 2022. DOI: 10.1109/TSMC.2020.3005231.

[30] Z.-Q. Zhao, P. Zheng, S.-T. Xu, and X. Wu, "Object detection with deep learning: A review," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 30, no. 11, pp. 3212–3232, 2019. DOI: 10.1109/TNNLS.2018.2876865.

[31] A. Alhardi and M. Afeef, "Object detection algorithms & techniques," Mar. 2024, pp. 391–399, ISBN: 978-625-6530-93-5.

[32] D. Prasad, "Survey of the problem of object detection in real images," *International Journal of Image Processing (IJIP)*, vol. 6, p. 441, Dec. 2012.

[33] Z. Zou, K. Chen, Z. Shi, Y. Guo, and J. Ye, "Object detection in 20 years: A survey," *Proceedings of the IEEE*, vol. 111, no. 3, pp. 257–276, 2023.

[34] K. Oksuz, B. C. Cam, S. Kalkan, and E. Akbas, "Imbalance problems in object detection: A review," *CoRR*, vol. abs/1909.00169, 2019. arXiv: 1909.00169.

[35] G.-G. Belmar, T. Bouwmans, and A. J. R. Silva, "Background subtraction in real applications: Challenges, current models and future directions," *Computer Science Review*, vol. 35, p. 100 204, 2020.

[36] T. Q. Vinh and N. T. N. Anh, "Real-time face mask detector using YOLOv3 algorithm and haar cascade classifier," in *2020 international conference on advanced computing and applications (ACOMP)*, IEEE, 2020, pp. 146–149.

[37] F. Javed Mehedi Shamrat, A. Majumder, P. R. Antu, S. K. Barmon, I. Nowrin, and R. Ranjan, "Human face recognition applying haar cascade classifier," in *Pervasive Computing and Social Networking: Proceedings of ICPCSN 2021*, Springer, 2022, pp. 143–157.

[38] N. Dalal and B. Triggs, "Histograms of oriented gradients for human detection," in *2005 IEEE computer society conference on computer vision and pattern recognition (CVPR'05)*, Ieee, vol. 1, 2005, pp. 886–893.

[39] B. Bhattarai, R. Subedi, R. R. Gaire, E. Vazquez, and D. Stoyanov, "Histogram of oriented gradients meet deep learning: A novel multi-task deep network for 2d surgical image semantic segmentation," *Medical Image Analysis*, vol. 85, p. 102 747, 2023.

[40] D. G. Lowe, "Distinctive image features from scale-invariant keypoints," *International journal of computer vision*, vol. 60, no. 2, pp. 91–110, 2004.

[41] H. Bay, T. Tuytelaars, and L. Van Gool, "SURF: Speeded Up Robust Features," *Proceedings IEEE European Conference on Computer Vision*, pp. 404–417, 2006.

[42] E. Rublee, V. Rabaud, K. Konolige, and G. Bradski, "Orb: An efficient alternative to sift or surf," in *2011 International Conference on Computer Vision*, 2011, pp. 2564–2571. DOI: 10.1109/ICCV.2011.6126544.

[43] B. D. Shivahare, S. Suman, S. S. N. Challapalli, P. Kaushik, A. D. Gupta, and V. Bibhu, "Survey paper: Comparative study of machine learning techniques and its recent applications," in *2022 2nd International Conference on Innovative Practices in Technology and Management (ICIPTM)*, vol. 2, 2022, pp. 449–454. DOI: 10.1109/ICIPTM54933.2022.9754206.

[44] M. A. Hearst, S. T. Dumais, E. Osuna, J. Platt, and B. Scholkopf, "Support vector machines," *IEEE Intelligent Systems and their applications*, vol. 13, no. 4, pp. 18–28, 1998.

[45] H. Bhatt, V. Shah, K. Shah, R. Shah, and M. Shah, "State-of-the-art machine learning techniques for melanoma skin cancer detection and classification: A comprehensive review," *Intelligent Medicine*, vol. 3, no. 3, pp. 180–190, 2023.

[46] P. Viola and M. Jones, "Rapid object detection using a boosted cascade of simple features," in *Proceedings of the 2001 IEEE computer society conference on computer vision and pattern recognition. CVPR 2001*, IEEE, vol. 1, 2001, pp. I–511.

[47] A. Saffari, C. Leistner, J. Santner, M. Godec, and H. Bischof, "On-line random forests," in *2009 ieee 12th international conference on computer vision workshops, iccv workshops*, IEEE, 2009, pp. 1393–1400.

[48] D. Seychell and C. J. Debono, "Efficient object selection using depth and texture information," in *2016 Visual Communications and Image Processing (VCIP)*, 2016, pp. 1–4. DOI: 10.1109/VCIP.2016.7805519.

[49] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan, "Object detection with discriminatively trained part-based models," *IEEE transactions on pattern analysis and machine intelligence*, vol. 32, no. 9, pp. 1627–1645, 2009.

[50] Z. Li, F. Liu, W. Yang, S. Peng, and J. Zhou, "A survey of convolutional neural networks: Analysis, applications, and prospects," *IEEE Transactions on Neural Networks and Learning Systems*, vol. 33, no. 12, pp. 6999–7019, 2022. DOI: 10.1109/TNNLS.2021.3084827.

[51] M. Jogin, M. ., M. Madhulika, G. Divya, R. Meghana, and S. Apoorva, "Feature extraction using convolution neural networks (cnn) and deep learning," May 2018, pp. 2319–2323. DOI: 10.1109/RTEICT42901.2018.9012507.

[52] Y. LeCun, Y. Bengio, and G. Hinton, "Deep learning," *Nature*, vol. 521, pp. 436–44, May 2015. DOI: 10.1038/nature14539.

[53] Y. Bengio and Y. Lecun, "Convolutional networks for images, speech, and time-series," Nov. 1997.

[54] X. Lu, Q. Li, B. Li, and J. Yan, "Mimicdet: Bridging the gap between one-stage and two-stage object detection," *CoRR*, vol. abs/2009.11528, 2020. arXiv: 2009.11528.

[55] A. Vaswani *et al.*, "Attention is all you need," *CoRR*, vol. abs/1706.03762, 2017. arXiv: 1706.03762.

[56] M. Bartolo and D. Seychell, *Correlation of object detection performance with visual saliency and depth estimation*, 2024. arXiv: 2411.02844 [cs.CV].

[57] E. M. Silva Machado, I. Carrillo, M. Collado, and L. Chen, "Visual attention-based object detection in cluttered environments," in *2019 IEEE SmartWorld, Ubiquitous Intelligence & Computing, Advanced & Trusted Computing, Scalable Computing & Communications, Cloud & Big Data Computing, Internet of People and Smart City Innovation (SmartWorld/SCALCOM/UIC/ATC/CBDCom/IOP/SCI)*, 2019, pp. 133–139. DOI: 10.1109/SmartWorld-UIC-ATC-SCALCOM-IOP-SCI.2019.00064.

[58] M. Bartolo, D. Seychell, and J. Bajada, *Integrating saliency ranking and reinforcement learning for enhanced object detection*, 2024. arXiv: 2408.06803 [cs.CV].

[59] J. C. Caicedo and S. Lazebnik, "Active object localization with deep reinforcement learning," in *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, Dec. 2015.

[60] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun, *Overfeat: Integrated recognition, localization and detection using convolutional networks*, cite arxiv:1312.6229, 2013.

[61] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, *You only look once: Unified, real-time object detection*, cite arxiv:1506.02640, 2016.

[62] J. Redmon and A. Farhadi, "Yolo9000: Better, faster, stronger," in *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2017, pp. 6517–6525. DOI: 10.1109/CVPR.2017.690.

[63] A. Bochkovskiy, C. Wang, and H. M. Liao, "Yolov4: Optimal speed and accuracy of object detection," *CoRR*, vol. abs/2004.10934, 2020. arXiv: 2004.10934.

[64] Z. Zhang, T. He, H. Zhang, Z. Zhang, J. Xie, and M. Li, "Bag of freebies for training object detection neural networks," *CoRR*, vol. abs/1902.04103, 2019. arXiv: 1902.04103.

[65] G. Jocher, *ultralytics/yolov5: v3.1 - Bug Fixes and Performance Improvements*, https://github.com/ultralytics/yolov5, version v3.1, Oct. 2020. DOI: 10.5281/zenodo.4154370. [Online]. Available: https://doi.org/10.5281/zenodo.4154370.

[66] Z. Ge, S. Liu, F. Wang, Z. Li, and J. Sun, "YOLOX: exceeding YOLO series in 2021," *CoRR*, vol. abs/2107.08430, 2021. arXiv: 2107.08430.

[67] Y. Tian, Q. Ye, and D. Doermann, *Yolov12: Attention-centric real-time object detectors*, 2025. arXiv: 2502.12524 [cs.CV].

[68] S. Aharon *et al.*, *Super-gradients*, 2021. DOI: 10.5281/ZENODO.7789328.

[69] T. Cheng, L. Song, Y. Ge, W. Liu, X. Wang, and Y. Shan, "Yolo-world: Real-time open-vocabulary object detection," in *Proc. IEEE Conf. Computer Vision and Pattern Recognition (CVPR)*, 2024.

[70] W. Liu *et al.*, "SSD: single shot multibox detector," in *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part I*, ser. Lecture Notes in Computer Science, B. Leibe, J. Matas, N. Sebe, and M. Welling, Eds., vol. 9905, Springer, 2016, pp. 21–37. DOI: 10.1007/978-3-319-46448-0_2.

[71] T. Lin, P. Goyal, R. B. Girshick, K. He, and P. Dollár, "Focal loss for dense object detection," in *2017 IEEE International Conference on Computer Vision (ICCV)*, Venice, Italy: IEEE Computer Society, Oct. 22–29, 2017, pp. 2999–3007, ISBN: 978-1-5386-1032-9. DOI: 10.1109/ICCV.2017.324.

[72] M. Sandler, A. Howard, M. Zhu, A. Zhmoginov, and L.-C. Chen, *Mobilenetv2: Inverted residuals and linear bottlenecks*, 2019. arXiv: 1801.04381 [cs.CV].

[73] C. Le, "Real-time hair and clothes segmentation on mobile devices," Ph.D. dissertation, Vietnam National University, Hanoi – University of Engineering and Technology, Jun. 2021. DOI: 10.13140/RG.2.2.16236.08325.

[74] A. Howard *et al.*, "Searching for mobilenetv3," *CoRR*, vol. abs/1905.02244, 2019. arXiv: 1905.02244.

[75] Z. Tian, C. Shen, H. Chen, and T. He, "FCOS: Fully Convolutional One-Stage Object Detection," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, Seoul, Korea (South): IEEE, Oct. 27–Nov. 2, 2019, pp. 9626–9635. DOI: 10.1109/ICCV.2019.00972.

[76] R. B. Girshick, J. Donahue, T. Darrell, and J. Malik, "Rich feature hierarchies for accurate object detection and semantic segmentation," *CoRR*, vol. abs/1311.2524, 2013. arXiv: 1311.2524.

[77] A. Krizhevsky, I. Sutskever, and G. E. Hinton, "Imagenet classification with deep convolutional neural networks," in *Advances in Neural Information Processing Systems 25: 26th Annual Conference on Neural Information Processing Systems 2012. Proceedings of a meeting held December 3-6, 2012, Lake Tahoe, Nevada, United States*, P. L. Bartlett, F. C. N. Pereira, C. J. C. Burges, L. Bottou, and K. Q. Weinberger, Eds., 2012, pp. 1106–1114.

[78] R. Girshick, "Fast r-cnn," *CoRR*, vol. abs/1504.08083, 2015.

[79] S. Ren, K. He, R. B. Girshick, and J. Sun, "Faster R-CNN: towards real-time object detection with region proposal networks," *CoRR*, vol. abs/1506.01497, 2015. arXiv: 1506.01497.

[80] K. He, G. Gkioxari, P. Dollár, and R. B. Girshick, "Mask R-CNN," in *2017 IEEE International Conference on Computer Vision (ICCV)*, Venice, Italy: IEEE Computer Society, Oct. 22–29, 2017, pp. 2980–2988, ISBN: 978-1-5386-1032-9. DOI: 10.1109/ICCV.2017.322.

[81] X. Wang, G. Wei, S. Chen, and J. Liu, "An efficient weakly semi-supervised method for object automated annotation," *Multimedia Tools and Applications*, vol. 83, pp. 1–24, Jun. 2023. DOI: 10.1007/s11042-023-15305-0.

[82] T. Lin, P. Dollár, R. B. Girshick, K. He, B. Hariharan, and S. J. Belongie, "Feature pyramid networks for object detection," *CoRR*, vol. abs/1612.03144, 2016. arXiv: 1612.03144.

[83] J. Dai, Y. Li, K. He, and J. Sun, "R-FCN: object detection via region-based fully convolutional networks," *CoRR*, vol. abs/1605.06409, 2016. arXiv: 1605.06409.

[84] M. Tan, R. Pang, and Q. V. Le, "Efficientdet: Scalable and efficient object detection," *CoRR*, vol. abs/1911.09070, 2019. arXiv: 1911.09070.

[85] M. Tan and Q. V. Le, "Efficientnet: Rethinking model scaling for convolutional neural networks," in *Proceedings of the 36th International Conference on Machine Learning (ICML)*, K. Chaudhuri and R. Salakhutdinov, Eds., ser. Proceedings of Machine Learning Research, vol. 97, Long Beach, California, USA: PMLR, Jun. 9–15, 2019, pp. 6105–6114. DOI: 10.5555/3305890.3306031.

[86] A. Dosovitskiy *et al.*, "An image is worth 16x16 words: Transformers for image recognition at scale," *CoRR*, vol. abs/2010.11929, 2020. arXiv: 2010.11929.

[87] H. Zhang *et al.*, *Dino: Detr with improved denoising anchor boxes for end-to-end object detection*, 2022. arXiv: 2203.03605 [cs.CV].

[88] F. Li, H. Zhang, S. Liu, J. Guo, L. M. Ni, and L. Zhang, "Dn-detr: Accelerate detr training by introducing query denoising," in *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, 2022, pp. 13 619–13 627.

[89] S. Liu *et al.*, "DAB-DETR: Dynamic anchor boxes are better queries for DETR," in *International Conference on Learning Representations*, 2022.

[90] S. Liu *et al.*, *Grounding dino: Marrying dino with grounded pre-training for open-set object detection*, 2024. arXiv: 2303.05499 [cs.CV].

[91] W. Lv, Y. Zhao, Q. Chang, K. Huang, G. Wang, and Y. Liu, *Rt-detrv2: Improved baseline with bag-of-freebies for real-time detection transformer*, 2024. arXiv: 2407.17140 [cs.CV].

[92] L. Beyer *et al.*, "Paligemma: A versatile 3b VLM for transfer," *CoRR*, 2024. DOI: 10.48550/ARXIV.2407.07726. arXiv: 2407.07726.

[93] A. Steiner *et al.*, "Paligemma 2: A family of versatile vlms for transfer," *CoRR*, vol. abs/2412.03555, 2024. DOI: 10.48550/ARXIV.2412.03555. arXiv: 2412.03555.

[94] B. Xiao *et al.*, *Florence-2: Advancing a unified representation for a variety of vision tasks*, 2023. arXiv: 2311.06242 [cs.CV].

[95] Y. Wang, L. Zhang, L. Wang, and Z. Wang, "Multitask learning for object localization with deep reinforcement learning," *IEEE Transactions on Cognitive and Developmental Systems*, vol. 11, no. 4, pp. 573–580, 2019. DOI: 10.1109/TCDS.2018.2885813.

[96] M. Zhou *et al.*, "Reinforcenet: A reinforcement learning embedded object detection framework with region selection network," *Neurocomputing*, vol. 443, pp. 369–379, 2021.

[97] M. Ayle, J. Tekli, J. El-Zini, B. El-Asmar, and M. Awad, "Bar — a reinforcement learning agent for bounding-box automated refinement," *Proceedings of the AAAI Conference on Artificial Intelligence*, vol. 34, no. 03, pp. 2561–2568, Apr. 2020.

[98] J. Dai *et al.*, "Deformable convolutional networks," in *2017 IEEE International Conference on Computer Vision (ICCV)*, Venice, Italy: IEEE Computer Society, Oct. 22–29, 2017, pp. 764–773, ISBN: 978-1-5386-1032-9. DOI: 10.1109/ICCV.2017.89.

[99] X. Zhou, D. Wang, and P. Krähenbühl, "Objects as points," *CoRR*, 2019. arXiv: 1904.07850.

[100] F. C. Akyon, O. Altinuc, and S. Temizel, "Slicing aided hyper inference and fine-tuning for small object detection," in *2022 IEEE International Conference on Image Processing (ICIP)*, Shanghai, China: IEEE, Oct. 2022. DOI: 10.1109/icip46576.2022.9897990.

[101] V. N. Vapnik and R. Izmailov, "Learning using privileged information: Similarity control and knowledge transfer," *J. Mach. Learn. Res.*, vol. 16, pp. 2023–2049, 2015.

[102] K. Makantasis, K. Pinitas, A. Liapis, and G. N. Yannakakis, "From the lab to the wild: Affect modeling via privileged information," *IEEE Transactions on Affective Computing*, vol. 15, no. 2, pp. 380–392, 2024. DOI: 10.1109/TAFFC.2023.3265072.

[103] D. Lopez-Paz, L. Bottou, B. Schölkopf, and V. Vapnik, "Unifying distillation and privileged information," in *4th International Conference on Learning Representations, ICLR 2016, San Juan, Puerto Rico, May 2-4, 2016, Conference Track Proceedings*, Y. Bengio and Y. LeCun, Eds., 2016.

[104] D. Pechyony and V. Vapnik, "On the theory of learning with privileged information," in *Advances in Neural Information Processing Systems*, J. Lafferty, C. Williams, J. Shawe-Taylor, R. Zemel, and A. Culotta, Eds., vol. 23, Curran Associates, Inc., 2010.

[105] G. E. Hinton, O. Vinyals, and J. Dean, "Distilling the knowledge in a neural network," *CoRR*, vol. abs/1503.02531, 2015. arXiv: 1503.02531.

[106] S. Sun, W. Ren, J. Li, R. Wang, and X. Cao, *Logit standardization in knowledge distillation*, 2024. arXiv: 2403.01427 [cs.CV].

[107] A. M. Mansourian, A. Jalali, R. Ahmadi, and S. Kasaei, *Attention-guided feature distillation for semantic segmentation*, 2025. arXiv: 2403.05451 [cs.CV].

[108] Z. Zheng *et al.*, *Localization distillation for object detection*, 2022. arXiv: 2204.05957 [cs.CV].

[109] G. Habib, T. jan Saleem, S. M. Kaleem, T. Rouf, and B. Lall, *A comprehensive review of knowledge distillation in computer vision*, 2024. arXiv: 2404.00936 [cs.CV].

[110] M. Schembri and D. Seychell, "Small object detection in highly variable backgrounds," in *2019 11th International Symposium on Image and Signal Processing and Analysis (ISPA)*, 2019, pp. 32–37. DOI: 10.1109/ISPA.2019.8868719.

[111] J. Ma *et al.*, "Arbitrary-oriented scene text detection via rotation proposals," *IEEE Transactions on Multimedia*, vol. 20, no. 11, pp. 3111–3122, 2018. DOI: 10.1109/TMM.2018.2818020.

[112] A. Deidun, A. Gauci, S. Lagorio, and F. Galgani, "Optimising beached litter monitoring protocols through aerial imagery," *Marine Pollution Bulletin*, vol. 131, pp. 212–217, 2018, ISSN: 0025-326X. DOI: https://doi.org/10.1016/j.marpolbul.2018.04.033.

[113] O. Authors, *Odm - a command line toolkit to generate maps, point clouds, 3d models and dems from drone, balloon or kite images*, 2017. [Online]. Available: https://github.com/OpenDroneMap/ODM.

[114] G. Niu, J. Li, S. Guo, M.-O. Pun, L. Hou, and L. Yang, "Superdock: A deep learning-based automated floating trash monitoring system," in *2019 IEEE International Conference on Robotics and Biomimetics (ROBIO)*, 2019, pp. 1035–1040. DOI: 10.1109/ROBIO49542.2019.8961509.

[115] S. Bak, D. Hwang, H. Kim, and H. Yoon, "Detection and monitoring of beach litter using uav image and deep neural network," *ISPRS - International Archives of the Photogrammetry, Remote Sensing and Spatial Information Sciences*, vol. XLII-3/W8, pp. 55–58, Aug. 2019. DOI: 10.5194/isprs-archives-XLII-3-W8-55-2019.

[116] V. Badrinarayanan, A. Kendall, and R. Cipolla, "Segnet: A deep convolutional encoder-decoder architecture for image segmentation," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 39, no. 12, pp. 2481–2495, 2017. DOI: 10.1109/TPAMI.2016.2644615.

[117] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei, "Imagenet: A large-scale hierarchical image database," in *2009 IEEE Conference on Computer Vision and Pattern Recognition*, 2009, pp. 248–255. DOI: 10.1109/CVPR.2009.5206848.

[118] T. Wang, Y. Cai, L. Liang, and D. Ye, "A multi-level approach to waste object segmentation," *CoRR*, vol. abs/2007.04259, 2020. arXiv: 2007.04259.

[119] E. Shelhamer, J. Long, and T. Darrell, "Fully convolutional networks for semantic segmentation," *2015 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 3431–3440, 2014.

[120] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, "Pyramid scene parsing network," *2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pp. 6230–6239, 2016.

[121] Z. Huang, X. Wang, L. Huang, C. Huang, Y. Wei, and W. Liu, "Ccnet: Criss-cross attention for semantic segmentation," in *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, Oct. 2019.

[122] L.-C. Chen, G. Papandreou, F. Schroff, and H. Adam, "Rethinking atrous convolution for semantic image segmentation," *arXiv:1706.05587*, 2017.

[123] Y. Li, Y. Chen, N. Wang, and Z.-X. Zhang, "Scale-aware trident networks for object detection," in *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, 2019, pp. 6053–6062. DOI: 10.1109/ICCV.2019.00615.

[124] Y.-H. Liao and J.-G. Juang, "Real-time uav trash monitoring system," *Applied Sciences*, vol. 12, no. 4, 2022, ISSN: 2076-3417. DOI: 10.3390/app12041838.

[125] D. Das, D. Kaushik, T. Sayeed, P. Dhar, and T. Shimamura, "Outdoor trash detection in natural environment using a deep learning model," *IEEE Access*, vol. VOLUME 11, pp. 97 549–97 566, Sep. 2023. DOI: 10.1109/access.2023.3313166.

[126] R. Pfeiffer *et al.*, "Use of uavs and deep learning for beach litter monitoring," *Electronics*, vol. 12, no. 1, 2023, ISSN: 2079-9292. DOI: 10.3390/electronics12010198.

[127] M. Yang and G. Thung, "Classification of trash for recyclability status," *CS229 project report*, vol. 2016, no. 1, p. 3, 2016.

[128] D. Seychell, M. Kenely, M. Bartolo, C. J. Debono, M. Bugeja, and M. Sacco, "Efficient automatic annotation of binary masks for enhanced training of computer vision models," in *2023 IEEE International Symposium on Multimedia (ISM)*, 2023, pp. 256–259. DOI: 10.1109/ISM59092.2023.00049.

[129] D. Pisani, "An investigation into small object detection from aerial imagery," M.S. thesis, University of Malta, 2023.

[130] G. Jocher, A. Chaurasia, and J. Qiu, *Ultralytics yolov8*, version 8.0.0, 2023. [Online]. Available: https://github.com/ultralytics/ultralytics.

[131] J. Feyereisl, S. Kwak, J. Son, and B. Han, "Object localization based on structural svm using privileged information," in *Advances in Neural Information Processing Systems*, Z. Ghahramani, M. Welling, C. Cortes, N. Lawrence, and K. Weinberger, Eds., vol. 27, Curran Associates, Inc., 2014.

[132] C. Wah, S. Branson, P. Welinder, P. Perona, and S. Belongie, "The caltech-ucsd birds-200-2011 dataset," California Institute of Technology, Tech. Rep. CNS-TR-2011-001, 2011.

[133] S. Sun, C. Zhang, and Y. Tian, "A new method for structured learning with privileged information," in *Computational Science – ICCS 2018*, Y. Shi *et al.*, Eds., Cham: Springer International Publishing, 2018, pp. 453–461, ISBN: 978-3-319-93701-4.

[134] C. H. Lampert and M. B. Blaschko, "Structured prediction by joint kernel support estimation," *Mach. Learn.*, vol. 77, no. 2–3, pp. 249–269, Dec. 2009, ISSN: 0885-6125. DOI: 10.1007/s10994-009-5111-0.

[135] V. Sharmanska, N. Quadrianto, and C. H. Lampert, "Learning to rank using privileged information," in *2013 IEEE International Conference on Computer Vision*, 2013, pp. 825–832. DOI: 10.1109/ICCV.2013.107.

[136] V. Sharmanska, N. Quadrianto, and C. H. Lampert, *Learning to transfer privileged information*, 2014. arXiv: 1410.0389 [cs.CV].

[137] T.-Y. Lin *et al.*, "Microsoft COCO: common objects in context," in *Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V*, D. J. Fleet, T. Pajdla, B. Schiele, and T. Tuytelaars, Eds., ser. Lecture Notes in Computer Science, vol. 8693, Cham, Switzerland: Springer, 2014, pp. 740–755. DOI: 10.1007/978-3-319-10602-1_48.

[138] T. Boger and T. Ullman, "What is "where": Physical reasoning informs object location," *Open Mind (Cambridge)*, vol. 7, pp. 130–140, May 2023. DOI: 10.1162/opmi_a_00075.

[139] L. Itti, C. Koch, and E. Niebur, "A model of saliency-based visual attention for rapid scene analysis," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 20, no. 11, pp. 1254–1259, Nov. 1998, ISSN: 0162-8828. DOI: 10.1109/34.730558.

[140] A. Linardos, M. Kümmerer, O. Press, and M. Bethge, "Calibrated prediction in and out-of-domain for state-of-the-art saliency modeling," *CoRR*, vol. abs/2105.12441, 2021. arXiv: 2105.12441.

[141] L. Yang, B. Kang, Z. Huang, X. Xu, J. Feng, and H. Zhao, *Depth anything: Unleashing the power of large-scale unlabeled data*, 2024. arXiv: 2401.10891 [cs.CV].

[142] R. Ranftl, A. Bochkovskiy, and V. Koltun, "Vision transformers for dense prediction," *CoRR*, vol. abs/2103.13413, 2021. arXiv: 2103.13413.

[143] L. Itti and C. Koch, "Computational modelling of visual attention," *Nature Reviews Neuroscience*, vol. 2, no. 3, pp. 194–203, 2001.

[144] S. A. McMains and D. C. Somers, "Multiple spotlights of attentional selection in human visual cortex," *Neuron*, vol. 42, no. 4, pp. 677–686, 2004, ISSN: 0896-6273. DOI: https://doi.org/10.1016/S0896-6273(04)00263-6.

[145] A. Kirillov *et al.*, *Segment anything*, 2023. arXiv: 2304.02643 [cs.CV].

[146] N. Ravi *et al.*, "Sam 2: Segment anything in images and videos," *arXiv preprint arXiv:2408.00714*, 2024.

[147] K. He, X. Zhang, S. Ren, and J. Sun, "Delving deep into rectifiers: Surpassing human-level performance on imagenet classification," *CoRR*, vol. abs/1502.01852, 2015. arXiv: 1502.01852.

[148] D. P. Kingma and J. Ba, *Adam: A method for stochastic optimization*, 2017. arXiv: 1412.6980 [cs.LG].

[149] S. Ruder, "An overview of gradient descent optimization algorithms," *CoRR*, 2016. arXiv: 1609.04747.

[150] A. Wang *et al.*, *Yolov10: Real-time end-to-end object detection*, 2024. arXiv: 2405.14458 [cs.CV].

[151] S. G. K. Patro and K. K. Sahu, "Normalization: A preprocessing stage," *CoRR*, 2015. arXiv: 1503.06462.

[152] J. H. Hosang, R. Benenson, and B. Schiele, "Learning non-maximum suppression," *CoRR*, vol. abs/1705.02950, 2017. arXiv: 1705.02950.

[153] O. Unel, B. Ozkalayci, and C. Cigla, "The power of tiling for small object detection," Jun. 2019. DOI: 10.1109/CVPRW.2019.00084.

[154] I. Herdiana, M. A. Kamal, Triyani, M. N. Estri, and Renny, *A more precise elbow method for optimum k-means clustering*, 2025. arXiv: 2502.00851 [stat.ME].

[155] I. Robinson, P. Robicheaux, and M. Popov, *Rf-detr*, https://github.com/roboflow/rf-detr, SOTA Real-Time Object Detection Model, 2025.

[156] R. R. Selvaraju, A. Das, R. Vedantam, M. Cogswell, D. Parikh, and D. Batra, "Grad-cam: Why did you say that? visual explanations from deep networks via gradient-based localization," *CoRR*, vol. abs/1610.02391, 2016. arXiv: 1610.02391.

[157] A. Chattopadhyay, A. Sarkar, P. Howlader, and V. N. Balasubramanian, "Grad-cam++: Generalized gradient-based visual explanations for deep convolutional networks," *CoRR*, vol. abs/1710.11063, 2017. arXiv: 1710.11063.

[158] M. B. Muhammad and M. Yeasin, "Eigen-cam: Class activation map using principal components," *CoRR*, vol. abs/2008.00299, 2020. arXiv: 2008.00299.

[159] D. Seychell *et al.*, *Awigs: Aerial waste identification and geolocation system*, https://xjenzamalta.mt/, Funded by Xjenza Malta, University of Malta, Malta, 2024.

[160] X. Pan, Y. Guo, and A. Men, "Traffic surveillance system for vehicle flow detection," in *2010 Second International Conference on Computer Modeling and Simulation*, vol. 1, 2010, pp. 314–318. DOI: 10.1109/ICCMS.2010.75.

[161] A. Raghunandan, Mohana, P. Raghav, and H. V. R. Aradhya, "Object detection algorithms for video surveillance applications," in *2018 International Conference on Communication and Signal Processing (ICCSP)*, 2018, pp. 0563–0568. DOI: 10.1109/ICCSP.2018.8524461.

[162] E. Zammit, D. Seychell, C. J. Debono, T. Gambin, and J. Wood, "Underwater archaeological object detection through bidirectional photogrammetric fusion," in *2024 IEEE International Conference on Image Processing Challenges and Workshops (ICIPCW)*, 2024, pp. 4122–4126. DOI: 10.1109/ICIPCW64161.2024.10769188.

Appendix A Preliminary Experiments

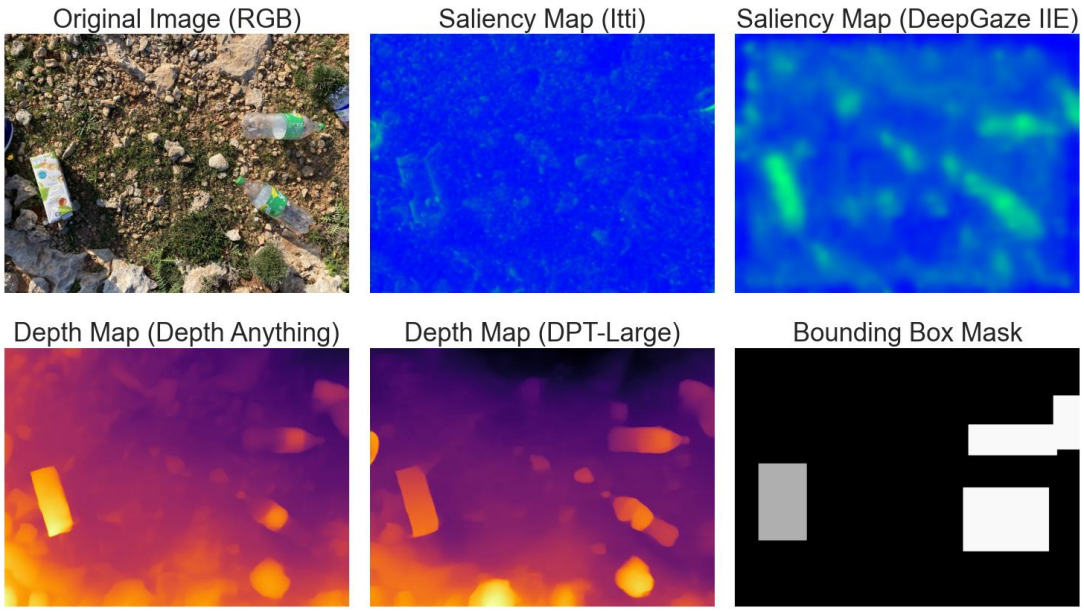


Figure A.1 Examples of generated privileged information derived from a selected image in the SODA dataset [13]. The original image (RGB, three-channel) is shown alongside corresponding single-channel representations, each depicting a distinct form of privileged information that may be considered during training.

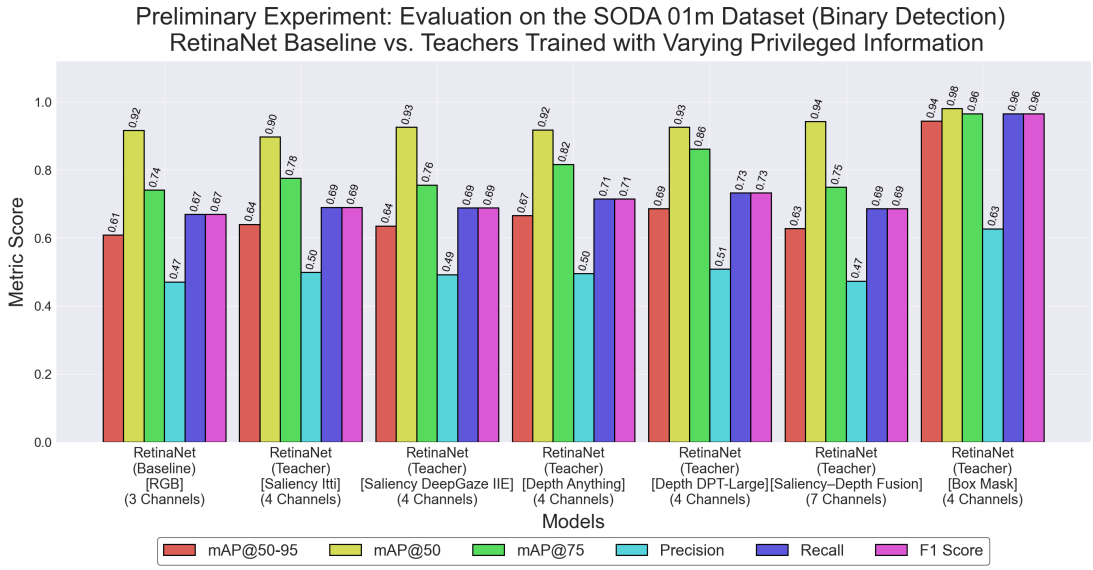


Figure A.2 Evaluation results on the SODA 1-metre altitude dataset for the binary litter detection task. The figure compares the RetinaNet baseline model with various teacher models trained using different forms of privileged information. Among the tested variants, the model utilising Bounding Box Mask privileged information yielded the most notable improvement, while other types of privileged information resulted in only marginal gains relative to the baseline.

Appendix B Implementation Overview

The complete implementation of this work, including all relevant code and resources, is available at the following GitHub repository: https://github.com/mbar0075/lupi-for-object-detection. The repository is thoroughly maintained, containing all the code utilised in this study, covering both the system requirements and model setup. Detailed instructions for setup and execution are provided within the repository.