CHALMERS
UNIVERSITY OF TECHNOLOGY

Exploring Open World Object Detection on Autonomous Driving Image Data

Evaluating and Enhancing the Performance of a Transformer Based OWOD Method on the Zenseact Open Dataset

Master's thesis in Electrical Engineering

Hanna Olsson
Lukas Johansson

Department of Electrical Engineering
CHALMERS UNIVERSITY OF TECHNOLOGY
Gothenburg, Sweden 2024

Master's thesis 2024

Exploring Open World Object Detection on Autonomous Driving Image Data

Evaluating and Enhancing the Performance of a Transformer Based OWOD Method on the Zenseact Open Dataset

Hanna Olsson
Lukas Johansson

CHALMERS
UNIVERSITY OF TECHNOLOGY

Department of Computer Science and Engineering
Chalmers University of Technology
Gothenburg, Sweden 2024

Evaluating and Enhancing the Performance of a Transformer Based OWOD Method on the Zenseact Open Dataset
Hanna Olsson
Lukas Johansson

Supervisor: Fredrik Kahl, Electrical Engineering
Advisor: Jens Henriksson, Semcon
Examiner: Fredrik Kahl, Electrical Engineering

Master's Thesis 2024
Department of Electrical Engineering
Chalmers University of Technology
SE-412 96 Gothenburg
Telephone +46 31 772 1000

Cover: Description of the picture on the cover page (if applicable)

Typeset in LaTeX
Gothenburg, Sweden 2024

Evaluating and Enhancing the Performance of a Transformer Based OWOD Method on the Zenseact Open Dataset
Hanna Olsson
Lukas Johansson
Department of Electrical Engineering
Chalmers University of Technology

Abstract

Open world object detection (OWOD) enhances traditional object detection by not only recognizing classes it was trained on but also identifying novel classes as 'unknown', while also incrementally learning these new classes. Since OWOD was introduced in 2021, various methods have been developed, typically trained and evaluated using benchmark datasets like MS-COCO. In this thesis, we examine the performance of one of the state-of-the-art OWOD methods, PROB, in a new context by applying it to the autonomous driving dataset, Zenseact Open Dataset (ZOD), and explore various strategies to enhance its performance. To evaluate the performance, we apply a standard framework in OWOD, looking at wilderness impact (WI), absolute open set error (A-OSE) and unknown recall (U-recall) for the unknown classes and mean average precision (mAP) for the known classes. Our results demonstrate that PROB exhibits inferior performance across all metrics on ZOD compared to benchmark datasets. Modifications to the initial method revealed that tuning the objectness temperature was unnecessary, while adjusting the class distributions for more even representation improved performance for less common classes. The most significant performance improvement was observed when incorporating curriculum learning, which involves changing the training structure by starting with easier training examples and gradually progressing to more difficult ones. However, neither of these improved methods reach the performance of PROB when applied to benchmark datasets, which can primarily be attributed to ZOD being a very different and challenging dataset. These findings underscore the difficulty of applying OWOD methods to diverse real-world datasets and highlight the need for further research to develop more robust and adaptable detection models.

Keywords: Open Word Object Detection, Object Detection, Machine Learning, PROB, Zenseact Open Dataset, Curriculum Learning.

Acknowledgements

We would like to express our sincere gratitude to everyone who has supported us throughout the process of completing this master's thesis.

First, we would like to thank Semcon for facilitating this thesis. Special thanks go to our supervisors, Jens Henriksson, PhD, and Sabrina Pereira, for their exceptional guidance.

We are also grateful for the assistance from our supervisor and examiner at Chalmers, Professor Fredrik Kahl, whose deep knowledge in object detection was invaluable.

Additionally, we would like to acknowledge the support from AI Sweden. Without your computational power, this thesis would not have been possible.

Here, you can say thank you to your supervisor(s), company advisors and other people that supported you during your project.

Hanna Olsson, Lukas Johansson, Gothenburg, 2024-06-19

Contents

List of Figures

List of Tables

1

Introduction

As machine learning advances, new applications emerge for object detection, demanding models that can adapt and learn in a manner akin to how humans learn, where new information is learned iteratively as new information is presented rather than learning all at once. Traditionally, object detection models have been constrained by the class distribution of their training sets, only able to identify objects they were trained on. This limitation prompts a need for models that can not only recognize unfamiliar objects during inference but also incrementally learn the new object classes. The concept of open world object detection (OWOD) has recently been introduced to address this need, with various methods showing promising results across diverse tasks. However, since most methods are trained and evaluated using standard, general-purpose object detection datasets like MS-COCO [1] and Pascal VOC [2], it is of interest to explore how these methods perform in more specialized application areas. Autonomous Driving is a suitable application of OWOD as it is crucial that AD-systems adapts to and accurately identifies unexpected or new objects on the road to ensure safety and reliability, particularly since the traffic scene is continually evolving with the introduction of new vehicles. Datasets used for autonomous driving differ from standard datasets as they include a wide variety of object sizes and a broad range of weather conditions. This thesis aims to evaluate how well an Open World Object Detection method performs on such a dataset and explores the extent to which its performance can be enhanced in this demanding application.

1.1 Background

Traditional object detection operates under the assumption that classes encountered during inference are the same as the ones introduced during training, which is known as closed set object detection. A problem with closed set object detection, other than its limitations in displaying human-like behaviour, is that it often classifies unknown objects into one of the known classes with high precision, decreasing the models reliability [3]. This becomes a problem when applying the object detection models in real world scenarios, such as autonomous driving scenes, where accuracy is important.

Open world object detection, first introduced in [3], is a new paradigm of object detection

that offers a more flexible method for recognizing and learningobjects. Unlike traditional models that only identify objects they have been trained on, OWOD aims to also classify new, unseen objects as 'unknown'. Additionally, it aims to learn these new classes over time without the need to retrain the entire model from scratch, also known as incremental learning. The latter leads to the issue of catastrophic forgetting, where the model loses its ability to recognize previously learned classes after being retrained exclusively on new ones. Therefore, in OWOD frameworks, an essential part is incorporating techniques to reduce the risk of this phenomenon. Following the introduction of OWOD, numerous methods have been developed for this purpose and have mainly branched in two segments; methods that utilize transformers and methods that build on the structure of the first introduced OWOD method, the Open World Object Detector (ORE) [3]. These methods are predominately evaluated on benchmark datasets such as MS-COCO [1]. However, in [4] the ORE method is evaluated on two standard autonomous driving datasets, the Indiana Driving Dataset and the Berkeley Deep Dive dataset, showing lacking performance. Thus far, someone has yet to explore the performance of a transformer based OWOD method on an autonomous driving dataset.

State-of-the art object detection systems for autonomous vehicles (AVs) utilize deep learning models that operate under closed-world assumptions, i.e., only recognizing objects that are present in the training data [3]. However, in the complex and unpredictable real-world environment, autonomous driving systems are likely to encounter many objects that fall outside of their training distribution. This limitation leads to issues where these systems incorrectly classify unfamiliar objects as either irrelevant background elements or incorrectly associate them with one of the pre-trained, known categories [5]. DiBiase *et al.* [5] highlights an instance where an autonomous vehicle's object detection model misidentifies a novel object on the road as part of the road itself, causing a crash. Given that OWOD models have the potential to significantly reduce the misclassification of novel objects [3], it becomes interesting to further explore how these models perform on open road data.

In autonomous driving scenarios it is essential to detect and differentiate between different moving classes, such as cars, pedestrians, bikes etc. The differentiation is especially important since different dynamic classes will act and move in vastly different ways. One scenario in which this was exemplified was when electric scooters appeared on the road scene. Since AD object detectors at the time were not trained on data containing electric scooters they were often falsely classified as pedestrians. This caused safety hazards as electric scooters can move at much greater speeds, with some being able to reach up to 45 km/h [6]. In this scenario a better detection of unknown objects could have greatly reduced the hazard, as the electric scooters could have been recognised as an unknown class, signaling to the driving system to take greater caution around the object. Additionally, quick learning of the new class is important in order to improve the AD-decision making capabilities, which can be improved using incremental learning. Since unknown object detection and incremental learning are the central elements of open world object detection, it is naturally interesting to test if existing OWOD methods can be adapted to aid the task to better handle changes in the traffic scene.

1.2 Aim

The aim of this project is to find out to what degree a state-of-the-art transformer based OWOD method can be used to detect novel, never before seen (in training data) moving objects in a diverse autonomous driving image dataset, and update the model on newly annotated objects without retraining from scratch, in order to reduce the compute requirements of updating the model. This will be evaluated on the Zenseact Open Dataset (ZOD) [7] which is one of the most well annotated and varied open-source autonomous driving datasets available today. Our aim can be summarized in the following research questions:

- How well does a state-of-the-art transformer based OWOD method perform on perform on ZOD in regards to learning novel dynamic objects?

- What modifications can be made to the method or its training process to enhance performance on ZOD, with regards to both detecting unknown and known objects?

1.3 Limitations

In this project we will limit our performance assessment to an evaluation framework that is commonly used in the research field of OWOD. The framework looks at both the ability to detect unknown and known objects with regards to the metrics wilderness impact (WI), absolute open set error (A-OSE), unkown recall (U-recall) and mean average precision (mAP).

Due to limited computational resources available, it was not feasible to use the entire Zenseact Open Dataset for training purposes. Consequently, the dataset was restricted to images captured in clear or partly clear weather conditions during daytime. This limitation may affect the robustness and generalizability of the model, as it has not been trained on or exposed to a full range of diverse scenarios, particularly challenging conditions such as night-time driving or adverse weather. Such conditions could introduce different kinds of novel objects or alter the appearance of known objects, potentially impacting the model's ability to accurately detect and learn novel objects in these environments. Future studies with access to enhanced computational resources could address this limitation by incorporating a more diverse set of images, thereby improving the model's robustness and applicability in real-world autonomous driving situations.

For the second part of our thesis, which focuses on improving the performance of the chosen method, we will not alter the architecture of the model itself. The decision to avoid architectural modifications is driven by the substantial time investment such changes typically require, which is beyond the scope of this project. Instead, our efforts will be concentrated on adjusting how the model is trained and on tuning the hyperparameters. This approach allows us to explore optimization strategies within the existing structural

framework of the model, aiming to enhance performance through more efficient training methods and optimal parameter settings.

1.4 Thesis Outline

This thesis begins with an in-depth exploration of Open World Object Detection, detailed in Section 2, 'Theory'. Here, we delve into the main challenges of OWOD, review various methods that have emerged since the concept's introduction in 2021, and discuss common performance evaluation techniques. Special attention is given to the specific OWOD method selected for this study, describing its origin and architecture.

In Section 3, 'Methodology', we describe our criteria for selecting the OWOD method and how we applied the method to ZOD. We begin by detailing how the dataset was reduced to comply with time limitations and the preprocessing steps undertaken. The method can be considered in two main parts: the initial strategy for applying the method to ZOD, and the subsequent modifications made to improve performance. These modifications include hyperparameter tuning, curriculum learning, and focused cropping.

The results of our experiments, including both the initial strategy and the modified approaches, are presented in Section 4. These results are evaluated using the standard benchmark metrics described in Section 2. In Section 5, we interpret these findings, exploring the reasons behind the performance for both known and unknown objects, discussing the implications for future applications of OWOD in autonomous driving, and suggesting potential extensions for future work. Finally, the thesis concludes with Section 6, summarizing the findings and addressing our research questions.

2

Theory

This section will cover a more in depth explanation of the Open World Object Detection problem and give a brief introduction to the work that this thesis builds on. The architecture of the method that will be investigated (PROB) will also be described, as well as the Zenceact Open Dataset which we will use to train the model. Finally it will be presented how Open World Object Detection methods most commonly are evaluated.

2.1 The Open World Object Detection Problem Formulation

For standard object detection tasks, a model M is trained on a dataset $D = \{\mathbf{X}, \mathbf{Y}\}$, where $\mathbf{X}$ represents the input images and $\mathbf{Y}$ represents the labels. The set $\mathbf{X}$ consists of M images $\mathbf{X} = \{\boldsymbol{I}_1, \ldots, \boldsymbol{I}_M\}$, and each image $\boldsymbol{I}_m$ is associated with a set of labels $\mathbf{Y}_m = \{\boldsymbol{y}_1, \ldots, \boldsymbol{y}_K\}$. Each label $\boldsymbol{y}_k$ corresponds to an object in the image $\boldsymbol{I}_m$, containing the class label and the associated bounding box, denoted by $\boldsymbol{y}_k = [l_k, x_k, y_k, w_k, h_k]$, where l_k is the label, and x_k, y_k, w_k, h_k represent the center coordinates, width, and height of the bounding box, respectively. The objective is to accurately predict the label and bounding box for each object in each image.

For OWOD, the problem formulation is broadened, as detailed by [3]. This approach introduces the concept of unknown classes alongside a temporal dimension. At any given time t, there exists a set of known classes $K^t = \{1, 2, \ldots, C\}$, complemented by an undefined set of unknown classes $U = \{C + 1, \ldots\}$. The model M_C is initially trained on these known classes. However, during inference, it may also detect objects from the unknown classes. The dataset for known classes at time t is denoted as $D^t = \{\mathbf{X}^t, \mathbf{Y}^t\}$, like the standard object detection framework. The model's task extends beyond merely identifying instances of known classes; it must also recognize and categorize novel objects as 'unknown'. Subsequent to inference, there will be a large set of unidentified objects $\mathbf{U}^t$. An oracle, typically a human, then selects n new classes of interest from these unknown objects, supplying the model with fresh training images featuring these newly identified classes. The model is then trained on a new dataset containing these new classes, without re-training the model scratch on the entire previous dataset, resulting in a model M_{C+n}.

Consequently, the set of known classes is updated to $K_{t+1} = K_t + \{C+1, \ldots, C+n\}$, and the cycle continues. In Figure 2.1, we illustrate the cyclical process of the OWOD system, highlighting the steps involved in model training and updating with new classes.

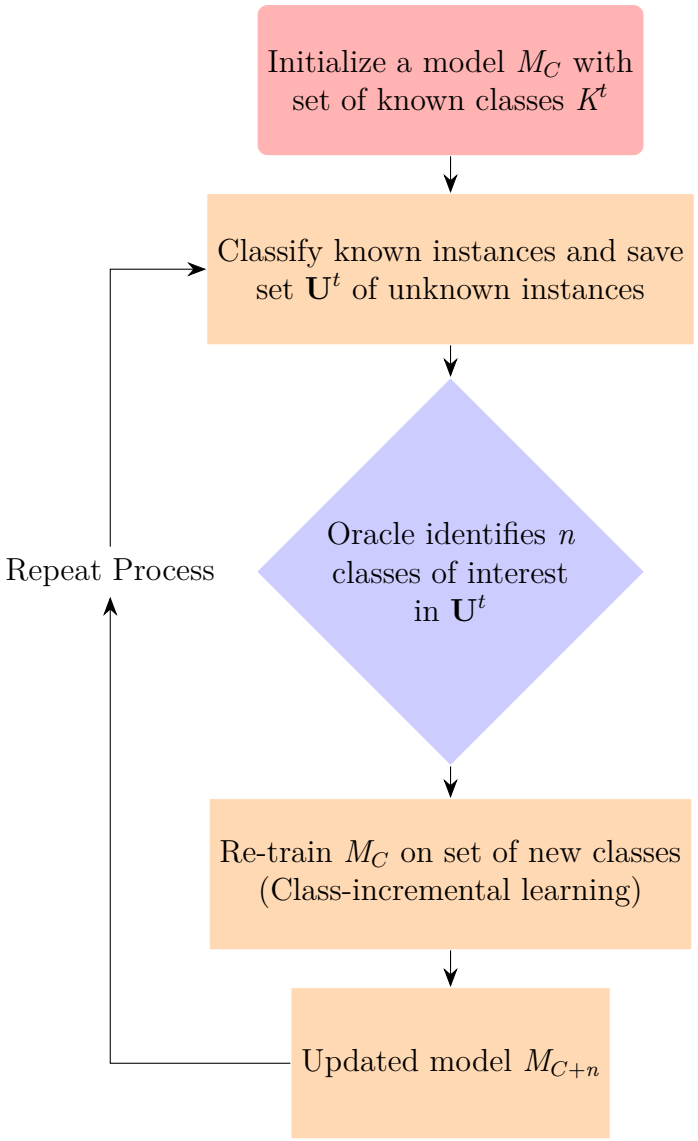


Figure 2.1: The Open World Object Detection Loop.

One of the pivotal stages within the OWOD loop is the re-training phase, also called class-incremental learning. There exist a variety of methodologies for executing this stage, and it is underscored in [8] that the re-training dataset should exclusively comprise images of the new classes, avoiding any representation of classes previously learned. This phase, however, introduces the risk of catastrophic forgetting, where the model may lose its grasp on earlier learned classes when re-trained solely on new ones (See Section 2.1.1) . To counter this issue, most approaches incorporate a fine-tuning step, where the model is fine-tuned on a carefully selected small portion of the earlier dataset. The strategy for choosing this subset can also vary between methods.

2.1.1 Class-incremental Learning and Catastrophic Forgetting

One of the pivotal stages within the OWOD loop is the re-training phase, also called class-incremental learning [9]. As discussed, class-incremental learning involves the model's ability to progressively learn new classes from supplementary data, this should be carried out without losing knowledge of the classes that were introduced previously. There exist a variety of methodologies for executing this part of the method, and it is underscored in [8] that the re-training dataset should exclusively comprise images of the new classes, avoiding any representation of classes previously learned. This phase, however, introduces the risk of catastrophic forgetting, where the model may actually lose its grasp on earlier learned classes when re-trained solely on new ones. To mitigate the challenge of forgetting previously learned classes in class-incremental learning, it is standard practice to retain a small subset of data from these prior classes for use in re-training. Typically, many OWOD methods initially re-train the model exclusively with images of new objects, and then fine-tune it using the preserved subset of the older classes once the primary training is complete. However, the strategy for choosing this subset can vary between OWOD methods.

2.1.2 Background-Unknown Distinction

An OWOD method has two main parts to its ability to classify unknown objects. First, its ability to distinguish unknowns from already known objects, and second, its ability to distinguish unknowns from background. The second aspect often presents challenges due to the fact that, in many datasets, not all objects of interest are labeled. This issue is explored in depth in [8]. OWOD methods train to identify and categorize new objects as unknown. However, if an object is identified as unknown without being labeled in the dataset, it leads to a false positive, as the model's true classification for that object would be 'background'. For instance, if the 'bird' category is unlabeled in the dataset, and the model identifies a bird as an unknown, it results in a false positive. This is problematic as it unjustly penalizes the model for recognizing unknown objects.

Traditionally, the Pascal VOC dataset has been the go-to choice for training OWOD models, despite its limitation of incomplete annotations. For a comprehensive and fair assessment of various OWOD approaches, it is crucial to use a dataset that is fully annotated. Alternatively, narrowing the focus to a segment of the dataset that is completely annotated can also serve the purpose.

2.2 Related Work

The concept of the open set problem, which challenges the traditional model training limitations by distinguishing between known and unknown objects, was first formalized in 2013 [10]. This foundational work opened the door to exploring how models could be designed to not just rely on the training distribution. Subsequent research, including a significant study in 2020 [11], further advanced the understanding and approaches to this problem. Deep neural networks, as explored in 2016 [12], have shown promise in efficiently addressing the open set problem by leveraging their capacity for complex pattern recognition. Building on these developments, Joseph *et al.* in 2021 [3] merged open set

learning with incremental learning to introduce the innovative concept of Open World Object Detection, marking a pivotal shift towards more adaptable and robust detection models.

The framework for OWOD in [3], named Open World Object Detector (ORE), leverages contrastive clustering in the latent space for clear discrimination between classes - it forces instances of the same class closer together and instances of different classes further apart. It also uses a Regional Proposal Network to pseudo-label unknown instances, generating regions that likely contain objects. The method incorporates an energy function which, due to the previous contrastive clustering, provides a clear separation in energy level between known and unknown objects. This makes discrimination between the two possible. In their approach to incremental learning, the authors draw upon prior studies [13] [14] that highlight the effectiveness of exemplar replay. This technique involves preserving a small subset of examples from previously learned classes for use during the process of incremental learning. Like numerous future approaches, ORE undergoes training and evaluation on the MS-COCO and Pascal VOC datasets. Additionally, it introduces an evaluative framework, employing metrics such as Wilderness Impact (WI) and Absolute Open Set Error (A-OSE) for assessing the classification of unknown objects, alongside the conventional mean Average Precision (mAP) for evaluating the classification of known objects (see Section 2.3). This evaluation is conducted across both previously known and currently known classes. Given these metrics ORE demonstrates superior performance compared to the proposed base line of Faster R-CNN, which is a state-of-the-art object detection framework.

Following the introduction of OWOD, many new methods have emerged that move the paradigm forward. As shown inf Figure 2.1 the landscape of OWOD methods can be broadly categorized into two branches - transformer based methods [15] [16] [17] and models that enhance the ORE framework in varying ways [8] [4] [18] [19]. Just like ORE, the latter group leverage two-stage detection where the models generates region proposals and then classifies the objects in those regions, building on the Faster R-CNN structure. Transformer based methods on the other hand build on the deformable transformer (DETR) [20] which creates a more integrated model that performs the background/unknown-distinction and the unknown/known classification in a single learning structure or network. The recent transformer based models, such as PROB and CAT, have shown state-of-the-art performance for both unknown and known object detection when evaluated on the M-OWODB benchmark (see Section 2.3.2), as seen in Table 2.2.

Efforts have been made to adapt the Open World Object Detection framework to road scene datasets. In [4] they find that the method presented in [3] transfers badly to road scenes. They argue that this is in part because of big differences in intra-class sizes of objects in road scene data. Other general datasets such as MS-COCO [1], which is a commonly used dataset to train and evaluate OWOD models, tend to have objects at the same distance from the camera, and therefore does not have the same problem. They present their own OWOD method that builds on ORE where they introduce a new loss function called focal regression loss, and utilize curriculum learning where the

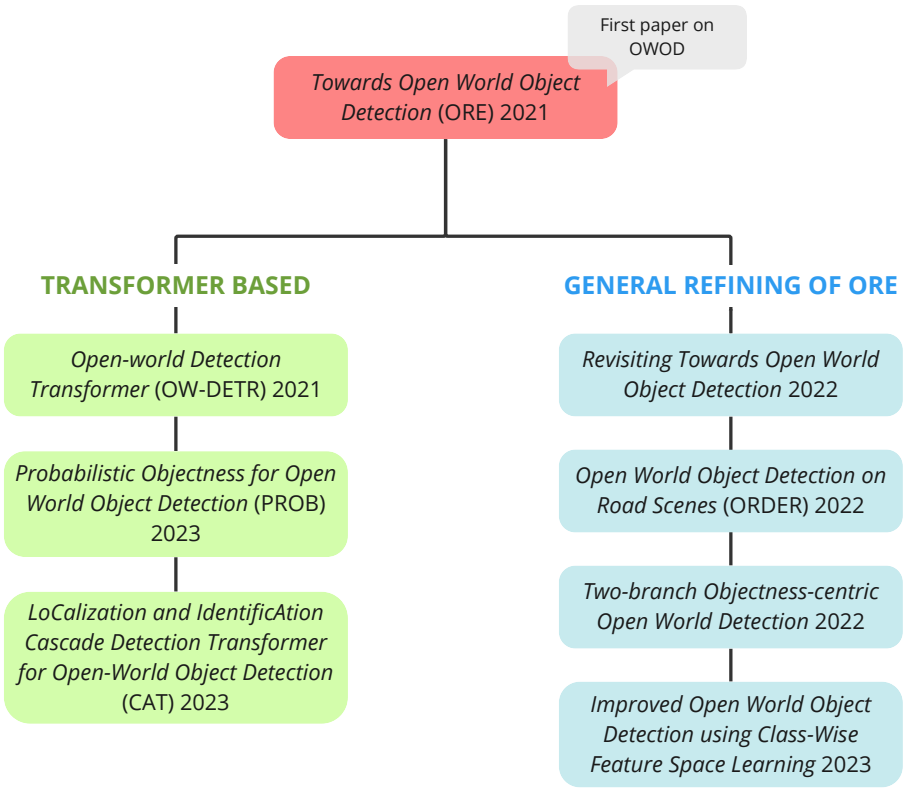


Table 2.1: Mapping of OWOD methods, grouped by transformer methods and refining of ORE (none exhaustive). The order is chronological and the connections do not translate to paper references.

Task IDs (→)	Task 1		Task 2				Task 3				Task 4		
	U-Recall	mAP (↑)	U-Recall	mAP (↑)			U-Recall	mAP (↑)			mAP (↑)		
	(↑)	Current	(↑)	Previously	Current	Both	(↑)	Previously	Current	Both	Previously	Current	Both
ORE	4.9	56.0	2.9	52.7	26.0	39.4	3.9	38.2	12.7	29.7	29.6	12.4	25.3
OW-DETER	7.5	59.2	6.2	53.6	33.5	42.9	5.7	38.3	15.8	30.8	31.4	17.1	27.8
PROB	19.4	59.5	17.4	55.7	32.2	44.0	19.6	43.0	22.2	36.0	35.7	18.9	31.5
CAT	23.7	60.0	19.1	55.5	32.2	44.1	24.4	42.8	18.7	34.8	34.4	16.6	29.9

Table 2.2: State-of-the-art comparison on MS-COCO split (M-OWODB split) [21]. The comparison is shown in terms of U-Recall and mAP. Each task introduces an additional 20 object classes.

learning process is initialized with larger objects and successively introduced to smaller, more difficult instances. They train and evaluate their model on Berkeley Deep Dive (BDD) dataset and the Indiana Driving Dataset (IDD) which encompass road scene training frames from the Us and India, showing superior performance to ORE on the same evaluation framework.

To the best of our understanding, there has yet to be an exploration into how transformer-based Open World Object Detection (OWOD) methods perform on road scene data,

nor an examination of potential adaptations to optimize their effectiveness for such applications. Furthermore, the datasets employed thus far lack European representation and present limitations regarding the volume of labeled dynamic objects. Table 2.3 provides a comparative analysis of labeled dynamic objects across the BDD, IDD, and ZOD datasets.

Category	**ZOD**	**IDD**	**BDD**
Car	✓	✓	✓
Pedestrian	✓	✓	✓
Truck	✓	✓	✓
Animal	✓	✓	✓
Bicycle	✓	✓	✓
Bus	✓	✓	✓
Motorcycle	✓	✓	✓
Van	✓	✓	
Tram/Train	✓	✓	
Trailer	✓		
HeavyEquip	✓		
Wheelchair	✓		
Stroller	✓		
Personal Transporter	✓		

Table 2.3: Comparison of annotated dynamic objects in ZOD, IDD, and BDD

2.3 OWOD Evaluation

To quantify the performance of an OWOD method there are several aspects that need to be taken into account. A benchmark needs to both measure the ability to of a method to incrementally learn new classes as well as retain knowledge of passed learned classes. Additionally, the set of metrics used need to quantify the models ability to distinguish between background/objects, known/unknowns, as well as the classification ability among the known classes. In this section we will explain how these benchmarks are structured and what what metrics are used.

2.3.1 Evaluation Method

To assess the models ability to introduce and efficiently learn new classes, the set of metrics also need to be measured at incremental expansions of the set of known classes. To do this the generally used benchmarking method for the OWOD setting is divided into a set of tasks. The chosen metrics are calculated for each task $t = 1, 2, \ldots, m$. There is no limit to the amount of tasks to divide the dataset into, however, the standard benchmarks have four tasks. For the second task and each task after, mAP is presented separately for the set of previously known classes, the newly introduced classes, and the combined set of known classes. This is to distinguish between the learning of new classes and the remembering of old classes. In the last task all the classes are known, which means that the unknown metrics can not be calculated for that task. An example of what such a table could look like can be seen in Figure 4.1.

2.3.2 Standard Benchmark

On of the first and most popular benchmarks for OWOD methods is the “superclass-mixed OWOD benchmark" (M-OWODB). M-OWODB was introduced in [3] and includes images from MS-COCO [1] and PASCAL VOC2007 [2], and PASCAL VOC2012. The images are grouped into four sets of non overlapping Tasks s.t. classes in a task T_t are not introduced until t is reached. Each task adds an additional 20 known classes and all the labels of the unknown classes are removed from the dataset during training.

Task IDs (→)	Task 1				Task 2						Task 3						Task 4		
	U-Recall	WI	A-OSE	mAP (↑)	U-Recall	WI	A-OSE	mAP (↑)			U-Recall	WI	A-OSE	mAP (↑)			mAP (↑)		
	(↑)	(↓)	(↓)	Current	(↑)	(↓)	(↓)	Previously	Current	Both	(↑)	(↓)	(↓)	Previously	Current	Both	Previously	Current	Both
Method 1	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-
Method 2	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-

Table 2.4: Format of benchmark results table. Task 1 only show results for new classes because there are no previous classes. Task 4 show no unknown based metrics as there are no unknown classes left in the dataset.

2.3.3 mAP

Mean Average Precision (mAP) is a measure of the accuracy among the known classes. The average precision for one class is the area under the recall-precision graph, with recall on the x-axis and precision on the y-axis. This gives a summarized measure of the precision over a range of recall thresholds which gives a very general measure. For a specific use case where the required recall threshold is known it might be better to use

the recall value at just that specific threshold or a smaller range of thresholds, but for the purposes of benchmarking and generality, we choose to use the average precision metric. In the OWOD framework, mAP is calculated for previously known classes (i.e., classes introduced in the previous task), currently known classes (i.e., classes introduced in the specific current task), and for both groups collectively.

2.3.4 A-OSE

The **Absolute Open-Set Error** (A-OSE) metric is used to measure the ability to distinguish between knowns and unknowns. It is the total number of unkown objects are wrongly classified as knowns. As it is not a mean or average, this metric can only be used for comparing models evaluated on the same dataset. Therefore, this metric will not be used to compare the performance of PROB on our dataset to the performance of PROB on say the M-OWODB benchmark, but we will be able to use it for hyperparameter tuning for our dataset.

2.3.5 WI

Wilderness Impact (WI), first introduced in [11], is a measure showing how much worse the precision of the known objects becomes after adding unknown objects. It is defines as

$$WI = \frac{P_{\mathcal{K}}}{P_{\mathcal{K}\cup\mathcal{U}}} - 1, \tag{2.1}$$

where $P_{\mathcal{K}}$ is the precision when evaluated on only known classes and $P_{\mathcal{K}\cup\mathcal{U}}$ is the precision when evaluated on known and unknown classes. WI can be measured at different recall levels, but the standard for OWOD benchmarks set by Joseph *et al.* in [3] is a recall level of 0,8.

2.3.6 U-Recall

To measure the ability to find unknown objects, we use the **Unknown Recall** (U-Recall) metric. We emphasize that unknown recall is more relevant that unknown precision because the consequences of labeling an unknown object as background and missing out on that information is more costly than having background be predicted as unknowns. This is because the oracle has the ability to sort out the interesting objects among unknowns, but it will not know about the objects not predicted as unknown. The way that U-recall is calculated in PROB and other OWOD methods originating from DETR [20] is not entirely intuitive. In these methods there are always a fixed number of predictions (commonly 100) being made due to the fixed number of query embeddings generated by the transformer decoder. The U-recall is calculated using all those detections without a threshold on the detection confidence. What this means is that the U-recall is a measure of how many unknowns it can find given a constant number of guesses, but not of the confidence scoring ability on the unknowns.

2.4 Detection Transformers

The detection transformer (DETR) architecture is an integral part of the PROB-method that we use. Detection transformers, first introduced in 2020 [20], presents a novel way to use transformers for end-to-end object detection. In the paper they state: "the main features of DETR are the conjunction of the bipartite matching loss and transformers with (non-autoregressive) parallel decoding". This means that all object predictions are generated in one single pass through the decoder of the transformer in way that is invariant to the permutation of the detected objects. This is in contrast to language processing transformers, where the output is dependent on previously generated words. The output of the DETR decoder is then propagated to separate classification- and regression heads. During training, the predicted labels and bounding boxes are matched with ground truth objects through bipartite matching. This means that each ground truth box is optimally matched with at most one prediction to minimize the global matching cost. Let y denote the ground truth set of objects, and $\hat{y} = \{\hat{y}_i\}_{i=1}^{N}$ the set of N predictions. Assuming N is larger than the number of objects in the image, let y be a set of size N padded with $\emptyset$ (no object). Formally, to find the optimal bipartite matching between these two sets, the permutation of N elements $\sigma \in S_N$ with the lowest cost is found:

$$\hat{\sigma} = \arg\min_{\sigma \in S_N} \sum_{i}^{N} \mathcal{L}_{match}(y_i, \hat{y}_{\sigma(i)}), \tag{2.2}$$

where $\mathcal{L}_{match}(y_i, \hat{y}_{\sigma(i)})$ is a pair-wise matching cost detailed in [20]. The optimization of this problem is performed with an efficient algorithm called Hungarian matching detailed in [22].

The bipartite matching ensures that only the most accurate prediction is matched to each ground truth object, and discards the other predictions. This has a similar function to non-maximum suppression which is a common method in other object detection methods, but it differs in that it is not a post processing step, and is instead done as a part of the loss function. It also differs in that it is done for all the predicted objects simultaneously, as opposed to one by one. This enables the use of a global loss function and an end-to-end architecture without the use of "proposals" or "anchors", which are commonly used methods in other state of the art object detection methods.

The method utilizes a ResNet backbone that embed the images to a compact feature representation, which, together with a 3D positional encoding, is used as input to the transformer. The dimensions of the activation map is generally C=2048 x h0/32 x w0/32 which normally results in 7x7x2048 dimensionality of the output when using input images of dimension 224x224 **resnet**. Examples of features represented by the ResNet output could be lines, simple shapes and colors. The expectation is that these features are general enough that they can be used to embed images in most image-domains, so that it can be used without having to be fine tuned for a specific use case.

One of the advantages of DETR is that the self-attention mechanism of the transformer architecture enables the model to find relations between all the features of the whole image in an efficient way. One downside of it is that it can have difficulties finding small objects. This has been further improved with deformable detection transformer (D-DETR) architecture [23], where the attention modules only attend to a small set of key sampling points around a reference, resulting in drastically lower convergence speeds (x10) and improved performance especially on small objects [23].

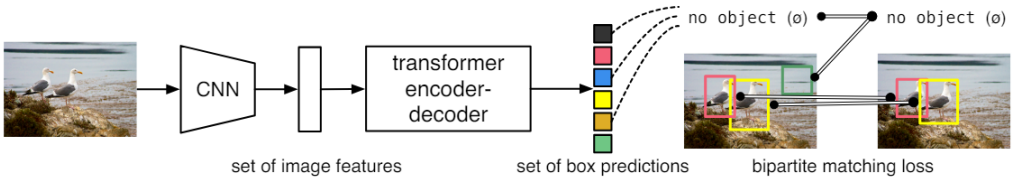


Figure 2.2: Architecture of the detection transformer [20].

2.5 The PROB-method

The paper "Probabilistic Objectness for Open World Object Detection" outlines a novel approach to the OWOD problem that is based on the D-DETR method. The main addition of PROB is a class agnostic probibalistic objectness head with the purpose to learn an objectness probability distribution in the object query feature space. The objectness probability is the probability that an object query is in fact an object, and in this case, since unknown labels are not available during training, it is learned only from the known objects. The assumption here is that the unknowns will be closer to the knowns than the background in the object query feature space. The objectness can then be used to predict unknowns directly, without the need for unknown object proposals or pseudo-labeling schemes. This enables a global loss function, which reduces the complexity of the model [16]. A high level architecture of the method can be seen in Figure 2.3.

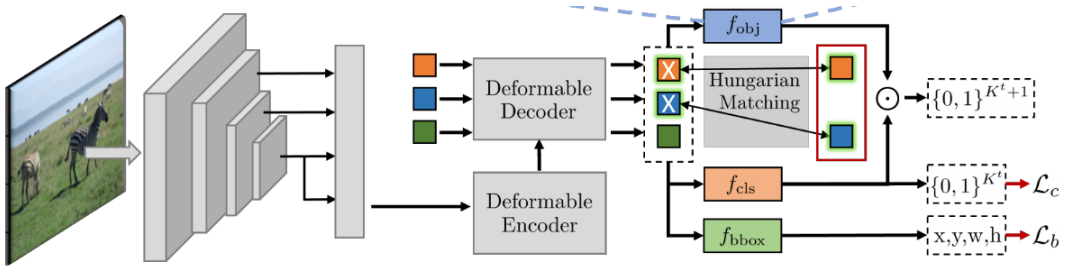


Figure 2.3: Architecture of the PROB method [16].

2.5.1 Detection Head

PROB has the same structure as D-DETR up untill the output of the Deformable decoder. Here the N query embeddings $\{\boldsymbol{q}_i\}_{i=1,\ldots,N}$ generated by the deformable decoder are not just propagated to a classification head and a regression head, but also to a probibalistic

objectness head that estimates the probability $p(o|\boldsymbol{q})$ of a query being an object. The objectness probability is parameterized by a multivariate Gaussian distribution in the query embedding space, i.e., $p(o|\boldsymbol{q}) \sim \mathcal{N}(\mu, \Sigma)$.

2.5.2 Prediction

By separating the class prediction $p(\boldsymbol{l}|o, \boldsymbol{q})$ from the objectness estimation, the classification head can work under the assumption that the query is an object (known or unknown), while the objectness head can learn a class agnostic objectness. This means that, in contrast to many other OWOD methods outputting $K_t + 2$ logits, with the +1 logit representing unknown objects and the +2 logit representing background, the classification head of PROB outputs only $K_t + 1$ with the last logit representing a general "not known" class representing both unknown objects and background. This is made possible by "outsourcing" the objectness estimation to the objectness head. The final prediction confidence is calculated as the product of the objectness head and the classification head:

$$p(\boldsymbol{l}|\boldsymbol{q}) = p(\boldsymbol{l}|o = 1, \boldsymbol{q}) \cdot p(o = 1|\boldsymbol{q}). \tag{2.3}$$

2.5.3 Learning

Because the method uses a global loss function, the backpropagation can propagate the whole model in single sweeps unlike other OWOD methods. The backbone is effectively finetuned in the process as it receives training but has its BatchNorm layers frozen. For the objectness head, PROB introduces a novel way to train it. The training consists of alternating between (i) embeddings distribution estimation and (ii) likelihood maximization of embeddings that represent known objects [16]. The likelihood maximization is achieved by first calculating the Mahalanobis distance, which is a measure of how far a point is from a distribution, between the matched query embeddings and the objectness distribution. Then the squared Mahalanobis distance is penalised in the global loss function, causing queries corresponding to known objects to move closer to the objectness distribution during training. A visualization of the training can be seen in Figure 2.4.

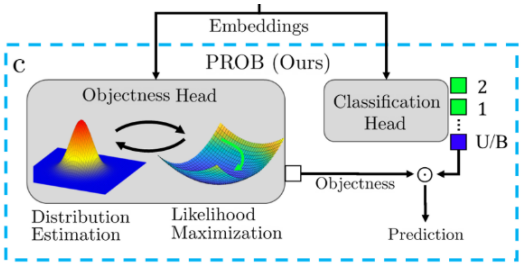


Figure 2.4: Visualization of the training of the objectness head [16].

2.5.4 Hyperparameters

The objectness temperature is a hyperparameter that controls the degree of confidence in the objectness estimation and it is defined as

$$f_{obj}^{t}(\boldsymbol{q}) = \exp\left(-\tau \cdot d_M(\boldsymbol{q})^2\right) = \left[\exp\left(-d_M(\boldsymbol{q})^2\right)\right]^{\tau}, \tag{2.4}$$

where τ is the objectness temperature, $\boldsymbol{q}$ is the query embedding, f_{obj}^{t} is the objectness prediction. The objectness temperature parameter effectively creates an exponential relationship between the learned objectness estimation and the objectness output during inference, where higher temperature lowers the confidence of the objectness prediction and vice versa. This parameter does not affect the training of the model, and can therefore be finetuned during validation without retraining.

2.5.5 Exemplar Replay

PROB, and most other OWOD-methods, utilizes an exemplar replay step at the end of each incremental task. The idea of exemplar replay is to store a few key examples of each of the known classes. These exemplars should represent their whole class in a good enough way that training on only the the exemplars as a finetuning step after the training on the new classes will mitigate catastrophic forgetting. For PROB this is the case, where exemplar replay is a crucial step to preserve known class accuracy. The exemplar datapoints are chosen from the examples encountered during training. While most methods use a random selection of the objects encountered, PROB selects the exemplars conditioned on their objectness score, specifically, those with extra high or low score.

2.6 The Zenceact Open Dataset

The Zenseact Open Dataset (ZOD) [7], launched in October 2023, serves as a detailed resource for autonomous driving research. It overcomes past constraints found in similar datasets by providing a wider variety of data and long-range annotations, alongside images of very high resolution.

The dataset consists of frames, sequences and drives. For this thesis, the focus will be on the "frames" component of the dataset. ZOD features 100,000 varied images from 14 European countries, captured through a forward-facing RGB car camera with a 120-degree field of view and a high resolution of 3848x2168 pixels. The images also have accompanying lidar data and two seconds of car sensor data for every image. Similar to other object detection datasets, all frames have 2D bounding boxes and labels for the annotated objects.

Compared to benchmark datasets for the OWOD task, such as MS-COCO [1], ZOD differs to a high degree. MS-COCO features a wide array of everyday scenes, from living room snapshots to images of a sleeping dog, presenting a significant number of frames with static objects. In contrast, ZOD captures images from a moving vehicle's perspective, resulting in a collection with many more dynamic scenes. Since the ZOD frames capture road data, there is also a high representation of distant (thus small) objects compared to

MS-COCO. As discussed in Section 2.2, ZOD also differs from other road scene datasets have been used and applied to OWOD models. Not only does it cover European roads but also spans nine times the geographical area covered by the next largest dataset, Waymo Open [24]. Moreover, it features annotations for distances up to 245 meters, significantly surpassing the 150-meter annotation range of most comparable datasets.

2.7 Curriculum Learning

Similar to the concept of open world object detection, curriculum learning (CL) also connects to mimicking a more human-like behaviour in machine learning models. The term 'curriculum learning' was first introduced by Bengio *et al.* in [25] and builds in the previous work of Elman in [26] where his experiments demonstrated that starting with a simpler, more restricted architecture and gradually increasing its complexity facilitated successful learning. Thus, CL is a training strategy in machine learning where examples are presented to the model in a meaningful order, starting with simpler concepts and gradually progressing to more complex ones. This approach mimics the human and animal learning process, which benefits from structured and incremental exposure to new information. It is similar to how teachers design curricula, starting with simple concepts and gradually introducing more complex ones to facilitate effective learning for students.

Bengio *et al.* experimented with curriculum learning for various tasks, including vision and language modeling and demonstrated he efficacy of curriculum learning in accelerating training and enhancing model performance by help guiding training to better local minima in non-convex optimization problems. The experiment on vision modeling involved classifying geometric shapes as either rectangles, ellipses and triangles. The experiment used two different datasets where the first one included images of basic shapes with low variability in shape and the second one included more complex shapes with greater variability in shape. The experiment showed that training a multi-layer neural network first on the dataset with basic shapes for a certain number of epochs, then switching to the dataset with complex shapes, resulted in a lower generalization error.

One of the challenges with curriculum learning is ranking the training examples by difficulty. Hacohen *et al.* explored using transfer learning and bootstrapping for this purpose. In the transfer learning approach, a pre-trained neural network on a larger dataset ranks the difficulty of examples in the target dataset. In the bootstrapping approach, the model is first trained on the target dataset without curriculum learning, and the resulting classifier is then used to rank the training examples. Their experiments showed that both transfer learning from a pre-trained 'teacher' network and bootstrapping methods provide similar benefits in terms of faster learning and improved final test performance. Another, simpler approach to using curriculum learning for object detection was demonstrated by [4], where they leveraged the fact that larger objects are easier to learn than smaller ones [27]. Thus, they ranked the training examples by difficulty based on the area of the bounding box

around each instance.

3

Methods

The following chapter details the methodologies used for carrying out this project. We start by describing the rationale behind choosing PROB as the OWOD method to investigate, to later describe how ZOD was reduced to enable training during the restricted time frame. Our method consists of two main components: applying PROB to the reduced version of ZOD, and modifying the initial strategy to enhance performance. In the first part, we outline the image preprocessing, the division of the dataset into tasks, and provide a summary of the training framework. In the second component, we explore three strategies to boost performance: adjusting the object temperature, implementing curriculum learning, and recropping the images to emphasize underrepresented classes. Finally, we explain our methods for visualizing the ground truth annotations and predictions to analyze and gain deeper insights into our final results.

3.1 Choosing an OWOD Method

There are several OWOD methods we could have chosen to use in this project. To decide on a method we first conducted a comprehensive literature study into existing OWOD methods. We found that, as of writing, the two methods showing best performance are PROB [16] and CAT [17], and they both incorporate the transformer based Deformable-DETR architecture as part of the model. Table 2.2 shows the performance of both models compared with OW-DETR [15], the first OWOD method based on the DETR architecture. We can see that both PROB and CAT have very similar results on the M-OWODB split, also referred to as the M-OWODB benchmark.

Ultimately we chose to go with the PROB method. The reason for this is that the PROB method has a significantly more simple architecture, which, unlike CAT, does not contain an object proposal step which allows for a global loss and a simpler training procedure. All else equal, a simpler model is better.

3.2 Dataset Reduction

When the project began, the intention was to utilize the entire Zenseact Open Dataset, which includes 100,000 frames. However, during the initial training phase, processing one epoch took two hours. Consequently, training for the first task was projected to take approximately 80 hours, with the complete training across all tasks estimated to span around 13 days, given the constraints of our 12GB GPU. Due to the time constraints of our thesis, we needed to reduce the dataset to shorten the training time.

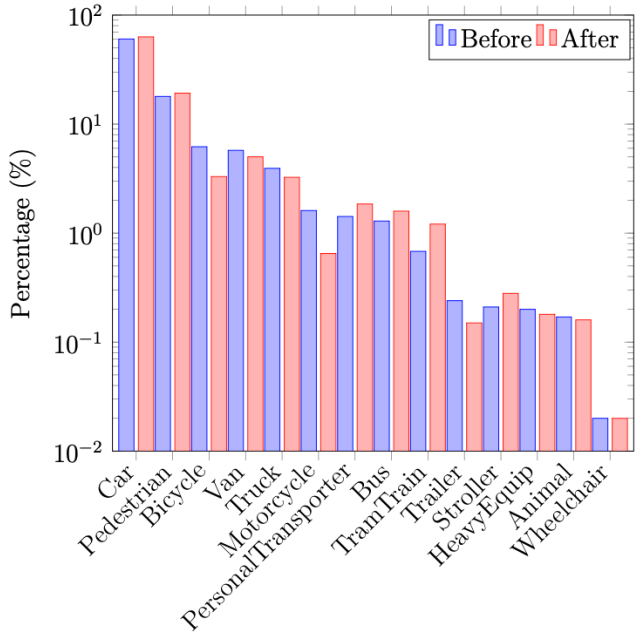
Since ZOD is well annotated with extensive metadata, including geography, time, and weather, we were able to easily select one metadata category and limit our scope accordingly. We decided to reduce the dataset to only include frames where the weather was clear or partly clear and during daytime. This approach aligns with our objective of assessing how well the PROB method applies to road scene data. Opting for data depicting good weather conditions during daylight offers the simplest scenario. If the method struggles in these ideal conditions, it is unlikely to perform better under more challenging circumstances. Therefore, the reduced dataset serves as an effective starting point for our evaluation.

The reduction was carried out using the ZOD Python library. We processed each frame individually, filtering out those where the `"scraped_weather"` variable was annotated as either `"clear-day"` or `"partly-clear-day"`. To validate the filtering we used visual inspection of a few images in the reduced dataset. The quality of the reduction was validated by comparing the distribution of the dynamic objects in the original dataset with the reduced dataset. As seen in Figure 3.1 the distribution is roughly the same for the reduced dataset as for the original dataset. Thus, the reduced dataset is representative of the original dataset in regards to class distribution.

After filtering the dataset to include only images captured during clear days, about 17% of the original dataset remained. As discussed in further detail in Section 3.3.1, additional cropping of the frames was performed which led to some images lacking dynamic objects, which are crucial for training. These non-contributory frames were thus removed, reducing the dataset by an additional half. Consequently, the dataset was reduced down to approximately 8200 frames. This substantial reduction significantly decreased the training time required across all tasks.

3.3 Applying PROB to the Reduced Version of ZOD

When training PROB on the reduced Zenseact Open Dataset (ZOD) there are four main steps that need to be considered, which are outlined in Figure 3.1. The initial step involves preprocessing the images in the dataset if needed, which in this instance included cropping the images to reduce their size for improved compatibility with the backbone. The next critical step involves splitting the data into tasks, as the OWOD framework relies on incremental learning, introducing the model to new class sets progressively. The second step comprises two main components: selecting the classes for the different tasks and



Dynamic Class	Total Nr of Instances
Car	43178
Pedestrian	13146
Van	3424
Bicycle	2256
Truck	2220
Motorcycle	446
Bus	1087
Personal Transporter	1263
Tram/Train	827
Trailer	105
Heavy Equip	120
Stroller	194
Animal	111
Wheelchair	14

Figure 3.1: Comparison of the distribution of the dynamic objects before and after the reduction of the dataset. The original distribution is kept for the reduced dataset.

allocating the corresponding training images to each task. Once the first two steps are completed, it's time to train the PROB method using ZOD, followed by analyzing the results to explore if any hyperparameters could be adjusted to enhance performance. The following section will give an in-depth description of our method for each of these four steps.

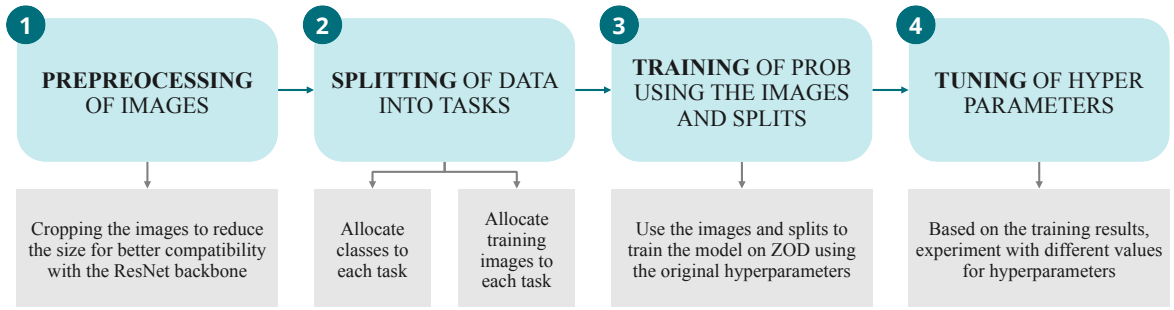


Table 3.1: Flowchart over the four main steps to train the PROB method on ZOD

3.3.1 Data Preprocessing

The ResNet architecture is used as a backbone in our model, which means that we do not include the fully connected "reasoning" part of the model. Instead we only use the feature extraction-base. Since the base of the ResNet is a strict CNN without any fully connected layers there is no theoretical limit to how big an input image can be. But because the convolution and pooling layers of the base downscales the width and height with a factor of 32, the only requirement is that both dimensions be divisible by 32.

Even though there is no theoretical size boundary to the input images, the model risks performing worse on inputs that are much larger that what the model has been trained on. The ResNet50, referring to 50 residual layers, is trained on the ImageNet dataset, which has an average image resolution of 469x387px. Since the images in our dataset are in 4K resolution (3848x2168px), it is important to downscale the dimensions of the input to roughly match the dimensions of the data that the ResNet50-model has been trained on.

Reducing the input dimension can be done through either lowering resolution or cropping. We settled cropping the images to a size of 800x600 pixels. The reason for choosing cropping over resolution downscaling is that ZOD includes a lot of very small annotated objects that are far away in the road scene, which is enabled by the high resolution of the ZOD frames. Downscaling the images would reduce the resolution on the already small objects, which in a lot of cases could make them much harder or practically impossible to detect. For example, it is easy to imagine how the most distant vehicle in the bottom left image of Figure 3.2 would be very hard to detect if the resolution had been reduced meaningfully. To create the cropped dataset we cropped a random part of each image, excluding the areas of the top 800 and bottom 400 pixels since these areas very rarely includes important road scene information. An example of the cropping of an image can be seen in image 3.2.

Finally we use a preprocessing step during training that is introduced in DETR and also used in PROB, which consists of cropping a random rectangular part of the image and resizing it again to 800x800 pixels. According to [20] this improves AP with about one unit if done during training. We are not entirely sure why this is but our guess is that it trains the model to better detect big objects as objects will take up a greater are after cropping and upsizing. Since we use a cropping step before this we are not certain that this is needed in our case, and a comprehensive ablation study should include ablation of this step.

3.3.2 Dataset Splits

One of the key steps in applying the PROB method on ZOD involves partitioning the dataset into distinct tasks. Consistent with the standard evaluation framework for Open World Object Detection (as discussed in Section 2.3), we have segmented the data into four tasks. Each task is defined by a specific set of classes on which the model is trained.

The strategy for choosing which classes should be included in which task was a combination of introducing more common classes in the beginning and trying to keep a balanced number of instances across the tasks. We prioritized more common classes in the initial tasks to simulate real-world applications, where a model typically starts training on the most common classes and later incorporates rarer classes encountered during inference. To make the splits we thus had to analyze the instance distribution of dynamic classes in ZOD, which is visualized in Figure 3.1. As expected the most common object class in ZOD is cars, followed by pedestrians, which made them clear candidates for the first task. We also included the next two most common classes in the first task, van and bicycle.

Figure 3.2: Example of one datapoint in ZOD. Full 4K image (top) with ignored image regions visualized in red and example cropping region in blue. Cropped image used as model input (bottom left) and extra zoomed in image (bottom right) showcasing small objects, whith ground truth annotations in light blue.

Since we selected the four most common classes for Task 1, it naturally contained the most instances, but this would also be true for a real-world scenario. However, for the subsequent tasks, we aimed to maintain a balance in the number of instances. To achieve this, we paired more common classes, such as motorcycles and trains, with rarer ones like strollers. The final distribution of classes across tasks is shown in Table 3.2.

Task	Classes Included
Task 1	Car, Pedestrian, Bicycle, Van
Task 2	Truck, Trailer
Task 3	Motorcycle, Tram/Train, Stroller, Heavy Equip
Task 4	Personal Transporter, Bus, Animal, Wheelchair

Table 3.2: Classes included in each task.

The second step in the process involves appropriately dividing the images to reflect these class-based splits. To facilitate efficient training we chose to append images to a task where at least one object in the image belonged to one of the classes in that task. We also chose not to include any of the succeeding classes in a task, as the real-world application would involve classes introduced in subsequent tasks being novel objects that the model has not previously encountered. The procedure is repeated for each task, starting with task 1. We remove images used in the previous task before selecting images for the next, ensuring there is no image overlap between tasks. Although the latter aligns with the standard practices for data splitting in Open World Object Detection, our method deviates from the norm by including classes from previous tasks in the current task. The standard, as outlined in Section 2.1, dictates that images for succeeding tasks should only include instances of classes specific to each task. However, as shown in Table 3.3, we deviate from this standard by for example including instances of car in tasks 2, 3 and 4 even though the car class belongs to task 1. We acknowledge that following the standard might be feasible for general purpose datasets such as MS-COCO, but when using a a more specific dataset such as ZOD where all images generally display the same setting (road scenes) it is very hard to not have some classes being present in most of the images. In our case the car class occurs naturally in almost every image, making it infeasible to not include that class in succeeding tasks.

The final distribution of images and instances across the tasks is detailed in Table 3.5. Notably, we reserved roughly 10% of the reduced dataset for validation and an equal fraction for testing. The validation is conducted a few times during training of each task as well as after the training is done. As indicated in the table, each task is validated using the same validation set and tested using the same test set. This consistent approach allows us to directly compare performance across tasks, ensuring that improvements in model accuracy are due to enhanced training rather than variations in the validation data.

3.3.3 Training

When training the PROB model on ZOD we used the original framework and hyperparameters presented in the PROB code base, with a few modifications to accommodate the high volume of objects in the ZOD frames.

Dynamic class	Task 1	Task 2	Task 3	Task 4
Car	17687	6179	3843	7160
Pedestrian	5274	1039	1423	2896
Bicycle	1147	176	133	349
Van	1266	579	321	564
Truck	0	1308	207	269
Trailer	0	68	7	13
Motorcycle	0	0	277	77
Tram/Train	0	0	492	137
Stroller	0	0	111	40
Heavy Equipment	0	0	88	11
Personal Transporter	0	0	0	986
Bus	0	0	0	882
Animal	0	0	0	88
Wheelchair	0	0	0	7

Table 3.3: Number of instances of all dynamic classes in each task.

	Task 1	Task 2	Task 3	Task 4
# training images	3883	877	642	1189
# validation images	803	803	803	803
# test images	817	817	817	817
# train instances	25374	9349	6902	13479
# validation instances	6558	6558	6558	6558
# test instances	6729	6729	6729	6729

Table 3.5: The chosen task composition and the number of images and instances across all tasks, the validation set and the test set. Note that all tasks are validated on the same validation set and evaluated on the same test set.

The training process is segmented into seven steps, beginning with training on task 1. After evaluating the model, a subset of images from the task is selected for exemplar replay to aid in future training. A fixed number of images are selected as exemplars for each known class individually so that catastrophic forgetting is mitigated for all classes. For all subsequent tasks, initial training is followed by a fine-tuning step. This fine-tuning, which utilizes the previous task's exemplar replay file, helps mitigate catastrophic forgetting. Figure A.1, illustrates the structure of initially training on a task followed by fine-tuning in the subsequent step.

As described, we initiate each task with the same number of epochs as specified in the original script. However, we now use the validation set to evaluate performance every five epochs. We compute all designated metrics, including A-OSE, WI, unknown recall, and mAP. Our primary focus is on mAP for the combination of previously and currently known classes, together with unknown recall. This approach helps us determine whether training should be halted earlier or extended beyond the initially planned range of epochs.

3.3.4 Chosen Metrics

We need to choose a set of metrics that measure all the important aspects of the OWOD setting, i.e., ability to distinguish between background/objects, ability to distinguish between unknown/known and ability to classify known objects. They also need to allow us to compare to the benchmark performance of PROB which is presented with mAP and U-recall. Ultimately this lead us to choose the metrics mAP, A-OSE, WI and U-recall. mAP for the classification abilty, A-OSE and WI to distinguish between knowns/unknowns, and U-recall for distingusihing unknowns/background. Further explanation of these metrics can be found in Section 2.3.

3.4 Modifications for Increased Performance

The second objective of this thesis is to investigate strategies to enhance the performance of PROB on ZOD. While there are several possible approaches, we focus on the straightforward methods of minor hyperparameter optimization, employing curriculum learning to specifically address the issue of poor performance on small objects and redoing the cropping of the images to focus more on low represented classes. To circle back to the flowchart presented in Figure 3.1 describing the main steps in applying PROB on ZOD, these modifications collectively address each of the four steps. As visualized in Figure 3.6, by adjusting hyperparamters we make changes to step 4, and by re-doing the image crops we change the method for preprocessing. The inclusion of curriculum learning involves changes to both steps 2 and 3, as the dataset needs to additionally be split into difficulty levels and the training structure is altered.

3.4.1 Hyperparameter Optimization: Object Temperature

The initial step to enhance PROB's performance on ZOD was to examine and adjust appropriate hyperparameters, which in our case came down to adjusting the objectness temperature parameter. The objectness temperature parameter, as described in section 2.5.4, has an effect on several different metrics. When evaluating PROB, Zohar *et al.*

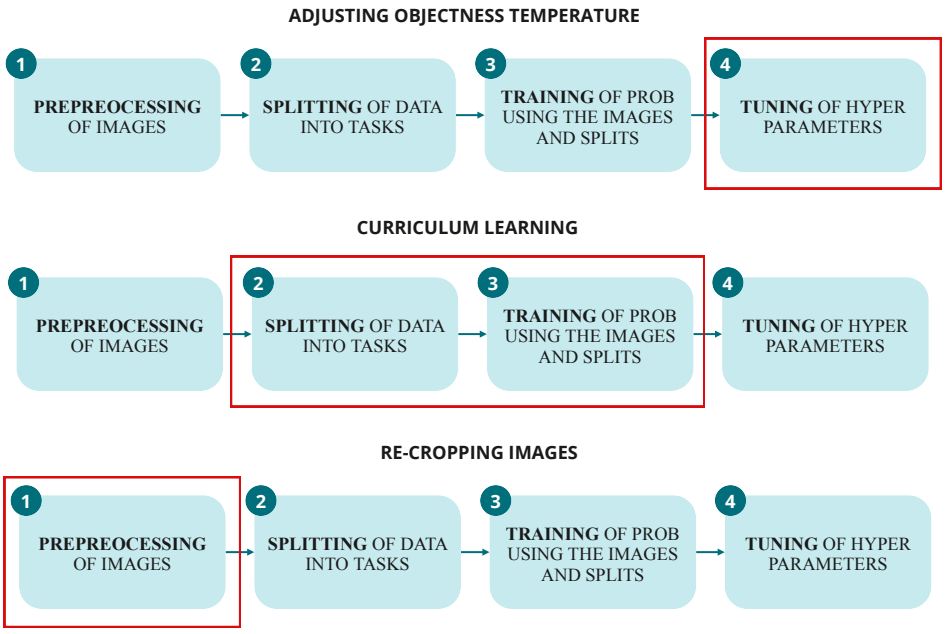


Table 3.6: Highlighting the parts (marked by red rectangles) of the four main steps to applying PROB on ZOD that are altered when modifying our initial approach for better performance

found that the choice of this parameter is a trade off between known precision and unknown-known confusion, which makes the tuning a matter of not only optimization, but also of judgement about the correct trade off for the application. If better distinction between knowns and unknowns is important, then bias toward a higher value, but if a high AP is important, then bias toward a lower value. For the benchmarks in PROB they settled for a value of 1.3. As described in Section 2.5.4 this parameter can be tuned during validation, which means that the model does not need to be retrained for different values. This hyperparameter needs to be finetuned to the specific dataset, which is why we conducted a sweep experiment similar to the one conducted by Zohar. et. al in the supplementary material of [16]. For the sweep experiment, we used the trained model from the initial setup described in Section 3.3. We suspected that the optimal value for our data could be close to the value used in the PROB-paper, and therefore conducted the experiment around the value 1.3, on values between 0 and 3, where the results of tuning the hyperparameter are detailed in Section 4.2.1.

3.4.2 Curriculum Learning

When looking at the initial performance of PROB on ZOD, we observed a significant variation in performance depending on the size of the instances. Using the validation set, we computed the recall for different bounding box size groups, as illustrated in Table 3.7. This analysis revealed inferior performance on smaller instances in contrast to larger ones.

Looking at the distribution of bounding box areas in ZOD displayed in Figure 3.3, we see that the relative performance of small objects is significant. Thus, for increased performance of PROB on ZOD one could focus on increasing the performance on smaller

Bounding Box Area	Objects	Detected Objects	Recall
0 - 100	104	1	0.0096
100 - 250	378	68	0.1799
250 - 500	672	292	0.4345
500 - 1k	873	537	0.6151
1000 - 10k	2858	2153	0.7533
10 000 - 100k	900	782	0.8689

Table 3.7: The performance of PROB on instances of different sizes in ZOD.

instances. As previously explored by [4], this can be done using curriculum learning where you start by feeding the model simpler instances and then gradually expanding the training set with harder instances.

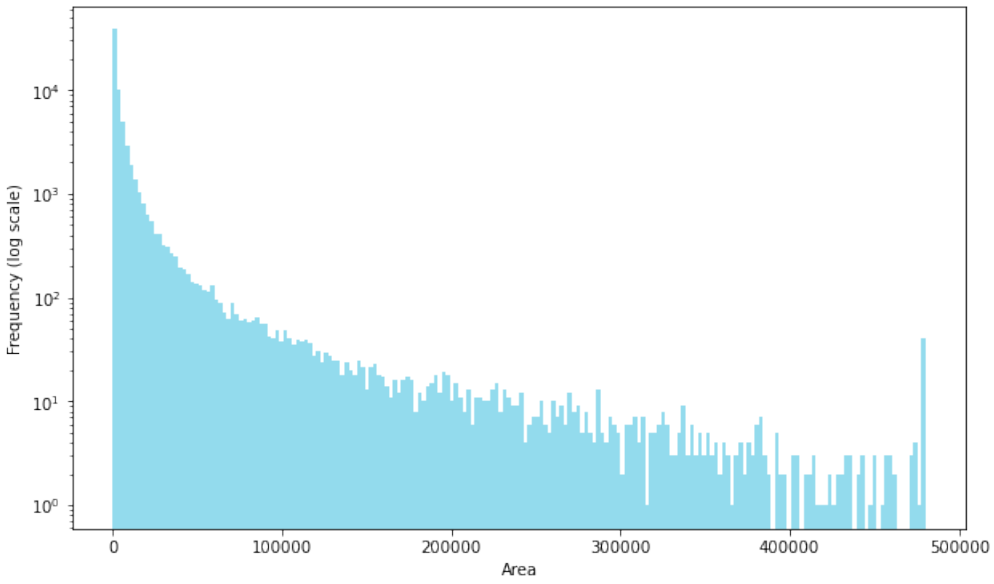


Figure 3.3: Visualization of the bounding box area distribution of the dynamic instances in the reduced dataset, showing that the relative proportion of small objects is significant.

Following the methodology outlined in [4], we categorize the dataset for each task T_i , $i \in \{1, 2, 3, 4\}$, into three difficulty levels: D_{easy}, D_{medium}, and D_{hard}. These categories are based on the area of the bounding boxes, and the data is incrementally introduced to the model as described in equation 3.1. To maintain a uniform distribution of object classes across these difficulty levels, we perform this categorization for each class individually. Specifically, for every task, we examine the distribution of bounding box areas within each class to set class-specific area thresholds. These thresholds are determined by the 33rd and 67th percentiles, ensuring that each difficulty group contains an equal number of instances.

$$T_i = \begin{cases} D_{\text{easy}} & I_1;\ \text{if } Ar_{\text{bbox}} < Ar_{\text{easy}} \\ D_{\text{easy}} + D_{\text{medium}} & I_2;\ \text{if } Ar_{\text{bbox}} < Ar_{\text{medium}} \\ D_{\text{easy}} + D_{\text{medium}} + D_{\text{hard}} & I_3 \end{cases} \tag{3.1}$$

The lower and upper thresholds are outlined in Table 3.8. Using these thresholds, we generate new annotations for each task and iteration. We process the original XML annotation files to calculate the bounding box area for each object. Each object is then categorized as 'easy', 'medium', or 'hard', according to the thresholds set for its class and task. These categorized objects are stored in new annotation files, ensuring retention of all original metadata, which is necessary for subsequent processing in the script. In the final step, we exclude annotations that do not contain any classes relevant to the specific task. This approach means that the distribution of classes across the 'easy', 'medium', and 'hard' categories does not always divide neatly into thirds for the previously known classes, as some annotations are omitted.

Object	Task 1		Task 2		Task 3		Task 4	
	Lower	Upper	Lower	Upper	Lower	Upper	Lower	Upper
Car	1425	6163	744	2997	924	3772	840	3247
Pedestrian	546	1836	374	956	397	1298	434	1221
Bicycle	1485	4488	871	2048	1115	3348	1020	2904
Van	1981	8528	1223	4764	1382	6335	1299	4606
Truck	–	–	1352	7625	1714	7505	1350	5873
Trailer	–	–	3111	7959	5343	15385	1383	6156
TramTrain	–	–	–	–	4472	16161	2557	8330
Motorcycle	–	–	–	–	1248	4488	1040	2880
Stroller	–	–	–	–	953	3086	807	1837
HeavyEquip	–	–	–	–	2117	15235	4368	11626
Bus	–	–	–	–	–	–	2105	9945
PersonalTransporter	–	–	–	–	–	–	1088	3311
Animal	–	–	–	–	–	–	334	1396
Wheelchair	–	–	–	–	–	–	1423	3913

Table 3.8: Thresholds for categorizing object detection difficulty based on bounding box areas: objects with bounding box areas below the lower threshold are classified as 'hard', those within the range between the lower and upper thresholds as 'medium', and those exceeding the upper threshold as 'easy'.

Equation 3.1 shows that the number of epochs for training in the three iterations are set as I_1, I_2, and I_3. Initially, we planned to evenly distribute the original total number of epochs across these three iterations to maintain consistent training durations with curriculum learning. However, all objects are only present in the last iteration, leading to a reduction of the training time on all objects to one-third of what was previously allocated, which seemed insufficient. Consequently, we adjusted our strategy by dividing the total epochs into thirds only for the first two iterations (easy objects and easy + medium objects). For the final phase, which includes all objects (easy, medium, and hard), we allocated the same number of epochs as the initial total training for each task. For instance, as task

1 originally had 40 epochs, in the curriculum learning setup, we assigned 14 epochs to each of the first two iterations and 40 epochs to the final phase. Just like for the initial training we use the validation set to do tests each 5 epochs, and based on the known average precision ('both') and the unknown recall ('U-recall') we choose the best model in that epoch interval. For the fine-tuning steps, we did not change the number of epochs from the original training.

In order to see the effects of CL for different object size groups we decided to calculate metrics for different size groups individually. When doing this there are a couple things that need to be taken into consideration. Calculating true positives and false negatives of an individual size group is straight forward as we have the ground truth of the objects in the size group and can simply check if a prediction matches. Calculating false positives is not as straight forward because we do not necessarily have a ground truth of the falsely detected object, and the size of a detection can not be used to determine the size of the object that it is detecting. And even if the false positive does detect an object of another class with a ground truth label, the detection and the ground truth can still be in different size groups, which makes it unclear which size group the false positive belongs to. For these reasons we decided not to calculate false positives for individual size groups and will therefore only present class recall and not precision.

3.4.3 Altering the dataset: Focused Cropping

We also created a second cropped training dataset in which we increase the number of instances of the less common object classes in order to achieve better learning outcomes specifically for these classes. The reason for this is that, as we can see in Table 3.3, there are some classes such as wheelchair, heavy equipment and trailer that have a very low representation in our first dataset, and we found a clear correlation between the number of instances and the AP achieved for the different classes in the results from the training with the first dataset. In order to achieve a higher prevalence of the less common classes in our dataset we decided to create a new cropped dataset where the cropping is done in a different way. Instead of cropping a random part of the full image we locate all instances of the less common classes and crop the each image around these objects so that they are contained in the new cropped images. We will hereafter refer to this method as Focused Cropping.

There are a few steps to the focused cropping method. First we count the number of instances for each class in the full image dataset. Then, for each class, in ascending order of number of class instances, we go through all appearances of that class. If it appears in an image that has not yet been added to the dataset, we crop around it and save it to the new dataset. The cropping of the image is done such that the whole bounding box of the object is contained within the cropped region, but otherwise randomly so that the the new object is as likely to appear in any part of the cropped image. If the object does not fit within the bounds of the cropping dimensions we instead crop the image such that the top-left corner of the cropping area aligns with the top-left corner of the objects bounding box. The resulting class instances per task with the FC dataset can be seen in Table 3.9, and the percentage increase of each class can be seen in Figure 3.4.

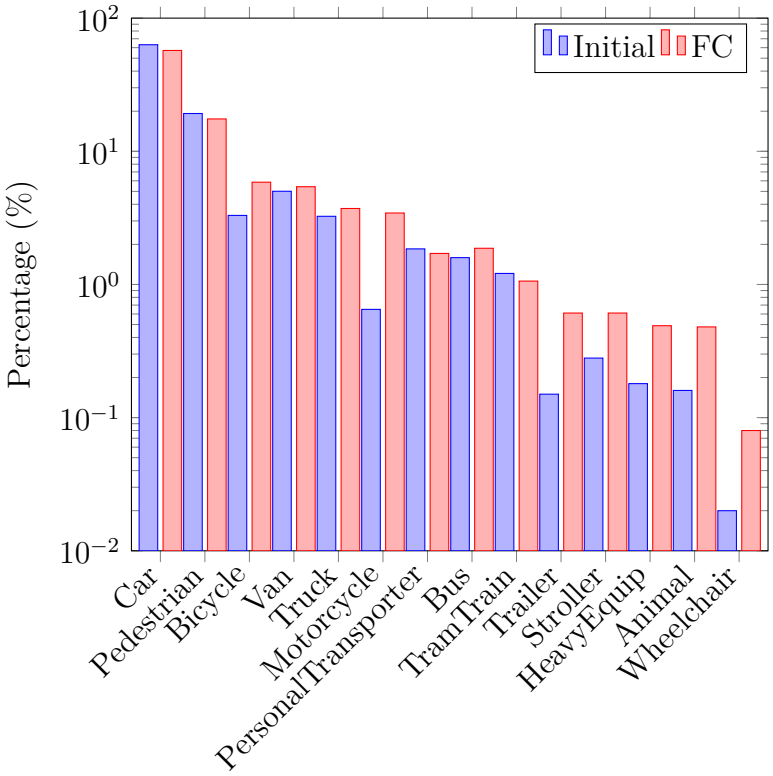


Figure 3.4: Comparison of the distribution of the dynamic objects in the initial train set and in the train set generated with focused cropping.

Note that the focused cropping is done only for the training set and not for the testing and validation sets. The first reason for this is that we want to use the same test and validation sets in all our training and testing in order to be able to compare the results of our different modifications in a fair manner. The second reason is that we want the test and validation set to be representative of the class distribution in the original dataset which is the class distribution naturally occurring in real world traffic. This would not be the case if the test and validation sets were created using focused cropping as the less common classes would be disproportionally occurring in the datasets.

3.5 Visual Comparison of Annotations and Predictions

In order to get a good visual understanding of what the annotations look like and how the detections performs we wanted to be able to efficiently view both the ground truth annotations and the detections for a large number of images. To achieve this we added functionality to save all the detections done during training into a json file. Then, with just the json file of detections and the annotation files for the testset we used the python library FiftyOne to display the images. When loading a dataset into FiftyOne it creates an unstructured database of all the images, labels and detections in the dataset, which allows for a great deal of efficiency and flexibility with functionality such as finding images with certain object labels or detections, and highlighting certain labels or detections of interest. Another hugely helpful feature of FiftyOne is the ability to dynamically change

Dynamic class	Task 1	Task 2	Task 3	Task 4
Car	13253	9930	19302	15313
Pedestrian	3662	1498	5807	6754
Bicycle	2330	548	1305	1629
Van	1603	1017	1900	1556
Truck	0	2011	1182	668
Trailer	0	521	101	63
Motorcycle	0	0	3133	752
Tram/Train	0	0	835	190
Stroller	0	0	445	77
Heavy Equipment	0	0	609	70
Personal Transporter	0	0	0	1883
Bus	0	0	0	1770
Animal	0	0	0	530
Wheelchair	0	0	0	89

Table 3.9: Number of instances of all dynamic classes in each task after centering cropped images around low frequency instances.

the confidence threshold and exclusively display detections passing the threshold. This allowed for an efficient way to visualize the effects of different confidence thresholds on the detections.

4

Results

The following section will detail the results of our different approaches for applying PROB to ZOD. We will start by presenting the results from our initial approach and continue with the results from tuning the objectness temperature, incorporating curriculum learning and using focused cropping.

4.1 Results of Initial Training of PROB on ZOD

The results from training the PROB method on the reduced version of ZOD, as shown in Table 4.1, indicate a general decline in performance across almost all key metrics when compared to training on the standard OWOD benchmark dataset, M-OWODB. Specifically, when looking at tasks 1 through 4, there is a noticeable decrease in U-Recall and mAP, suggesting that PROB is less effective at both unknown and known object detection in the ZOD environment. The observed results reveal that A-OSE is significantly lower for ZOD compared to MS-COCO. It is important to note that A-OSE depends on the number and distribution of objects across tasks, which vary between these two datasets. Therefore, direct comparisons of A-OSE may not provide a fair assessment due to these inherent differences in dataset composition. Despite this, the overall lower performance across most metrics on ZOD highlights challenges in applying the PROB method to varied datasets. This observation emphasizes the necessity for adaptations or specific optimizations of PROB to enhance its effectiveness across datasets with unique characteristics, such as ZOD compared to the more standardized MS-COCO.

Task IDs (→)	Task 1				Task 2						Task 3						Task 4		
	U-Recall (↑)	WI (↓)	A-OSE (↓)	mAP (↑) Current	U-Recall (↑)	WI (↓)	A-OSE (↓)	mAP (↑) Previously	mAP (↑) Current	mAP (↑) Both	U-Recall (↑)	WI (↓)	A-OSE (↓)	mAP (↑) Previously	mAP (↑) Current	mAP (↑) Both	mAP (↑) Previously	mAP (↑) Current	mAP (↑) Both
PROB (MS-COCO)	19,4	0,056	5195	59,5	17,4	0,034	6452	55,7	32,2	44	19,6	0,015	2541	43	22,2	36	35,7	18,9	31,5
PROB (ZOD)	4	0,048	874	41,2	2,8	0,039	686	38,4	8,4	28,4	2,58	0,034	564	28,7	8,1	20,5	22,9	9,1	19
Change	-79,4%	-14,3%	N/A	-30,8%	-83,9%	14,7%	N/A	-31,1%	-73,9%	-35,5%	-86,8%	126,7%	N/A	-33,3%	-63,5%	-43,1%	-35,9%	-51,9%	-39,7%

Table 4.1: The results of training PROB on the reduced version of ZOD compared to training PROB on MS-COCO. Across all metrics and tasks except for WI on task 1, PROB performs worse on ZOD compared to MS-COCO.

4.2 Results of Modifications of PROB on ZOD

To improve the performance of PROB on ZOD, we implemented multiple strategies, including adjusting the object temperature, using curriculum learning and modifying the distribution of object instances within the dataset by focusing the image crops on low represented classes. The upcoming section will detail how these adaptations have impacted PROB's effectiveness when trained on ZOD.

4.2.1 Object Temperature

The results of the object temperature sweep in Figure 4.1 clearly show a jump in performance for mAP and U-recall in the range between 0.5 and 1, with little to no improvement after. Naturally, the object temperature should be set above this threshold. A-OSE, on the other hand, is steadily increasing in this range and beyond, which makes the temperature choice a compromise between mAP/U-recall and A-OSE. We found that a temperature of 1.3 assures that mAP and U-recall are kept above the improvement threshold while simultaneously minimizing A-OSE as much as possible. 1.3 is also the value used by Zohar *et al.* [16], which further assures us that it is a reasonable value. Consequently we used an object temperature of 1.3 in all our testing.

Although WI sees improvement at higher temperatures, the fluctuations are fairly small in comparison, and we therefore decided to let the other metrics dictate our choice. Also note that we did not assume the optimal temperature to be the same as in [16], as Zohar *et al.* has pointed out that this is a hyperpaprameter that needs to be tuned to the dataset [28].

4.2.2 Curriculum Learning

When incorporating curriculum learning into the training of the PROB method on ZOD, the results, illustrated in Table 4.2, indicate a minimal or even negative effect on the metrics related to the detection of unknown objects. In contrast, there is a notable improvement in the metrics assessing the detection of known objects. Table 4.2 show an improvement in mAP of 8-33% for all class groups (previously known, currently known and both) across all tasks. Figure 4.2 illustrates an example of the detections from the initial training compared to those using curriculum learning. It clearly shows that the latter approach is more successful in identifying smaller objects, as it manages to identify the small pedestrian while the initial training fails to do so.

In Table 4.3 we compare the recall for objects of different bounding box areas for the initial training and after using curriculum learning. It is clear that using curriculum learning has increased the recall across almost all sizes and tasks. However, the observed increase in recall predominantly affects objects with smaller bounding box areas. This outcome aligns with the intended goal of curriculum learning, which is to specifically enhance the model's ability to more effectively recognize smaller objects. This focus on smaller objects suggests that the curriculum learning approach is successfully fine-tuning the model's sensitivity to features at reduced scales.

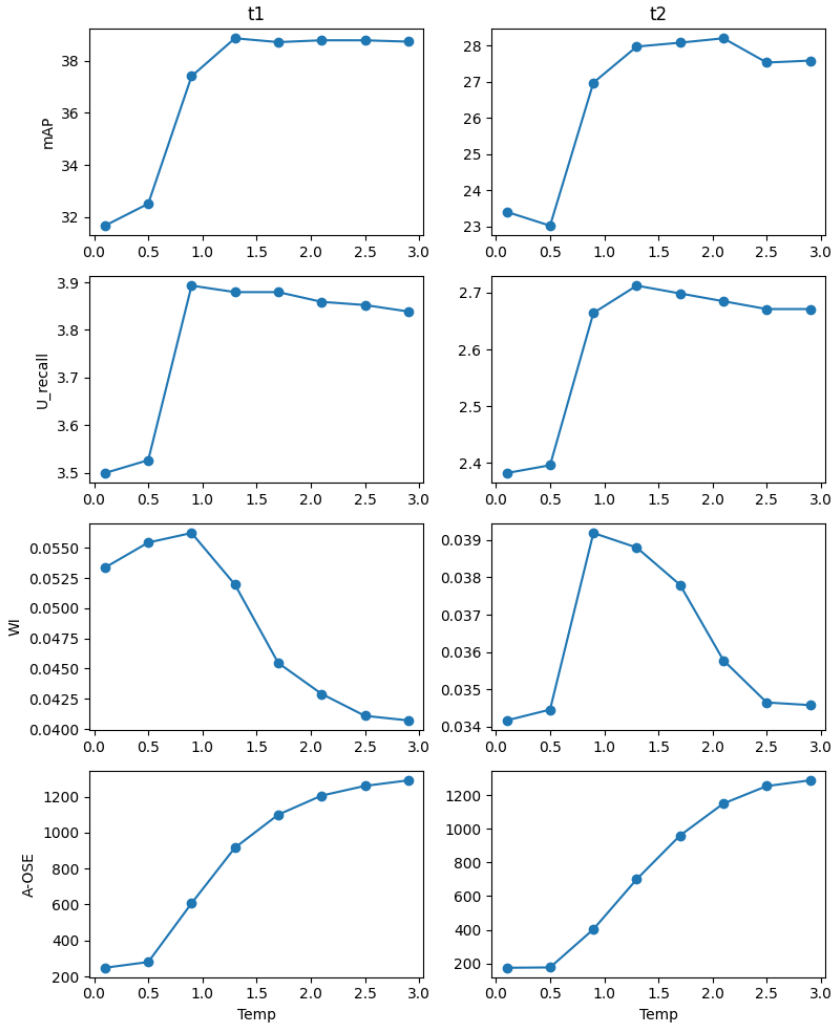


Figure 4.1: Results of object temperature sweep for task 1 and 2.

Task IDs (→)	Task 1				Task 2						Task 3						Task 4		
	U-Recall (↑)	WI (↓)	A-OSE (↓)	mAP (↑) Current	U-Recall (↑)	WI (↓)	A-OSE (↓)	mAP (↑) Previously	mAP (↑) Current	mAP (↑) Both	U-Recall (↑)	WI (↓)	A-OSE (↓)	mAP (↑) Previously	mAP (↑) Current	mAP (↑) Both	mAP (↑) Previously	mAP (↑) Current	mAP (↑) Both
PROB (ZOD)	**4**	**0,048**	**874**	41,2	2,8	**0,039**	**686**	38,4	8,4	28,4	2,58	**0,034**	**564**	28,7	8,1	20,5	22,9	9,1	19
PROB + CL (ZOD)	3,9	0,049	900	**45,7**	**3,1**	0,04	854	**46**	**10,5**	**34,1**	**2,75**	0,035	964	**31,2**	**10,3**	**22,8**	**26**	**12,1**	**22**
Change	-2,5%	2,1%	3,0%	10,9%	10,7%	2,6%	24,5%	19,8%	25,0%	20,1%	6,6%	2,9%	70,9%	8,7%	27,2%	11,2%	13,5%	33,0%	15,8%

Table 4.2: The results of training PROB on the reduced version of ZOD using curriculum learning compared to the initial training (not using curriculum learning). Adding curriculum learning seems to have little to negative effect on unknown detection while consequently increasing the known detection across all tasks.

4.2.3 Focused Cropping

When revising our initial approach for applying PROB to ZOD by focusing the cropped images on low represented classes to facilitate more training examples of these classes, there is little to no effect on the mAP for task 1 and 2, as seen in Table 4.4. On the contrary, the focused cropping seems to have the biggest impact on the mAP for the currently known classes in task 3 and 4. An example of this is shown in Table 4.3, where the stroller (a currently known class in task 3) is predicted more accurately and

Task 1

Bounding box area (→)	<100	100-250	250-500	500-1k	1k-10k	10k-100k
# Objects in dataset	104	378	672	873	2858	900
Nr. of detetcted objects (Before)	1	68	292	573	2153	782
Nr. of detetcted objects (CL)	4	99	319	582	2240	812
Recall (Before)	0,010	0,180	0,435	0,615	0,753	0,869
Recall (CL)	0,039	0,262	0,475	0,667	0,784	0,902
Change in recall with CL	301%	46%	9%	8%	4%	4%

Task 2

Bounding box area (→)	<100	100-250	250-500	500-1k	1k-10k	10k-100k
# Objects in dataset	104	387	689	895	2975	947
Nr. of detetcted objects (Before)	1	74	278	510	2168	821
Nr. of detetcted objects (CL)	6	91	306	567	2240	823
Recall (Before)	0,010	0,191	0,404	0,570	0,729	0,867
Recall (CL)	0,058	0,235	0,444	0,634	0,753	0,869
Change in recall with CL	501%	23%	10%	11%	3%	0%

Task 3

Bounding box area (→)	<100	100-250	250-500	500-1k	1k-10k	10k-100k
# Objects in dataset	105	388	701	905	3057	1006
Nr. of detetcted objects (Before)	1	80	262	528	2148	843
Nr. of detetcted objects (CL)	4	103	324	558	2264	868
Recall (Before)	0,010	0,206	0,374	0,583	0,703	0,838
Recall (CL)	0,038	0,266	0,462	0,617	0,741	0,943
Change in recall with CL	301%	29%	24%	6%	5%	13%

Task 4

Bounding box area (→)	<100	100-250	250-500	500-1k	1k-10k	10k-100k
# Objects in dataset	108	392	720	930	3188	1058
Nr. of detetcted objects (Before)	5	85	294	533	2219	875
Nr. of detetcted objects (CL)	6	111	320	562	2267	862
Recall (Before)	0,046	0,217	0,408	0,573	0,696	0,827
Recall (CL)	0,056	0,283	0,444	0,604	0,711	0,815
Change in recall with CL	20%	31%	9%	5%	2%	-1%

Table 4.3: The recall for objects of different bounding box areas for the initial training and after using curriculum learning, across all four tasks. Adding curriculum learning has increased the recall for almost all size groups across all tasks, showing most effective on smaller objects.

confidently after using focused cropping compared to the initial training. Generally there is no significant change in the unknown metrics, expect for A-OSE which is reduced across all tasks.

Task IDs (→)	Task 1				Task 2						Task 3						Task 4		
	U-Recall (↑)	WI (↓)	A-OSE (↓)	mAP (↑) Current	U-Recall (↑)	WI (↓)	A-OSE (↓)	mAP (↑) Previously	mAP (↑) Current	mAP (↑) Both	U-Recall (↑)	WI (↓)	A-OSE (↓)	mAP (↑) Previously	mAP (↑) Current	mAP (↑) Both	mAP (↑) Previously	mAP (↑) Current	mAP (↑) Both
PROB (ZOD)	**4**	0,048	874	**41,2**	2,8	0,039	686	**38,4**	**8,4**	**28,4**	**2,58**	0,034	564	**28,7**	8,1	20,5	22,9	9,1	19
PROB (ZOD_FC)	3,7	**0,041**	**745**	41	**2,9**	**0,038**	**488**	37	7,9	27,3	2,51	0,034	**448**	28,4	**24,5**	**26,8**	**24,7**	**10,9**	**20,8**
Change	-7,5%	-14,6%	-14,8%	-0,5%	3,6%	-2,6%	-28,9%	-3,6%	-6,0%	-3,9%	-2,7%	0,0%	-20,6%	-1,0%	202,5%	30,7%	7,9%	19,8%	9,5%

Table 4.4: The results of training PROB on ZOD with focused cropping (denotes as ZOD_FC in the figure) compared to training on the initial image crops. The results show better performance for A-OSE across all tasks. The larges effect can be seen for mAP for currently known classes in task 3 and 4.

Figure 4.2: Zoomed in example of how performance on small objects is improved with curriculum learning (CL). The images show ground truth and predictions without CL (left) and with CL (right). An improvement in both detecting small objects and scoring them can be seen with CL.

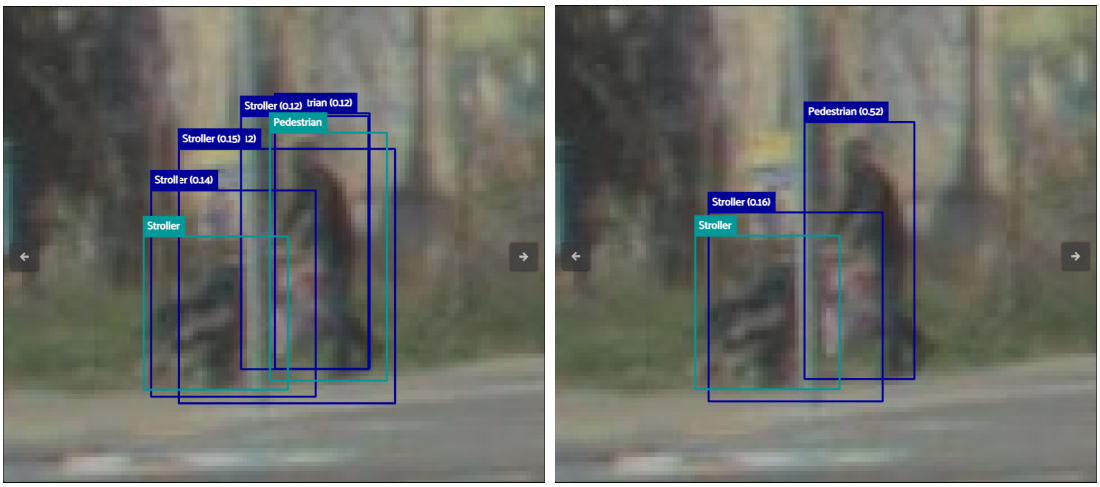


Figure 4.3: Zoomed in example of how performance on a less common class such as stroller is improved with focused cropping. The images show ground truth and predictions without focused cropping (left) and with focused cropping (right). An improvement in both precision and scoring for the stroller class can be seen with focused cropping.

5

Discussion

In this chapter, we delve into the analysis of our experimental results, starting with an exploration of both known and unknown metrics. We aim to contextualize these findings within the broader landscape of object detection, particularly in challenging scenarios such as those presented by the ZOD dataset. Our discussion covers the initial approach, the effects of curriculum learning, and the impact of focused cropping on detection performance, as well as a brief discussion on the objectness temperature. Additionally, we consider the implications of our findings for real-world applications in autonomous driving and suggest directions for future research.

5.1 Known Metrics

In this section, we will discuss the known metrics, including mean Average Precision (mAP) for previously known classes, currently known classes, and both combined. We will evaluate these metrics for the initial strategy of applying PROB to ZOD, the incorporation of curriculum learning, and the focus on cropping images to emphasize classes with low representation in the initial reduced dataset.

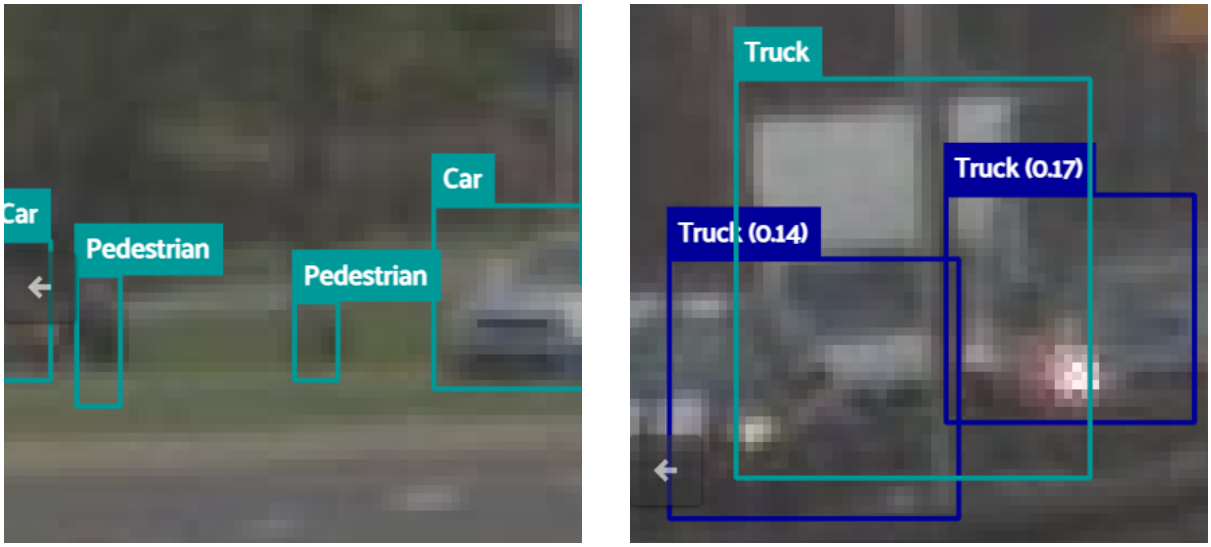
5.1.1 Known Metrics for the Initial Training

As seen in the results of the initial training, PROB consequently performed worse when trained and evaluated on ZOD than on the benchmark dataset MS-COCO. As previously stated there is a big difference between the two datasets, thus the inferior performance was expected. As ZOD is annotated at a distance of up to 245 meters, the dataset contains many distant and small objects, naturally making the the dataset harder. As seen in Table 3.7, PROB performs poorly on these small objects, partially explaining the poor performance.

When visually inspecting the ground truth annotations and model predictions, we noticed that the challenging nature of the dataset primarily makes it difficult for the model to predict bounding boxes of the correct size. An example of this can be seen in Figure 5.1, where the predictions for the truck do not sufficiently overlap with the ground truth

bounding box to exceed the IoU threshold of 0.5. As a result, these predictions will not be considered true positives.

Examining only the ground truth annotations, we find further evidence of the challenges posed by distant annotations. In Figure 5.1 (a), which shows an example image from the test set, there are two pedestrian annotations that are barely recognizable due to their distance and the resulting very few pixels. Since there is a very small difference between an annotated pedestrian in this case, and what should be considered background, its likely that the model also learns to predict dynamic objects where there is actually is just background - resulting in a negative effect on the overall mAP.



(a) (b)

Figure 5.1: Zoomed in example of how instances in the dataset can be very hard to detect (a). Zoomed in example of how the model identifies the right class but where the IoU is under 0.5 (b). The images show ground truth and predictions.

5.1.2 Known Metrics for Curriculum Learning

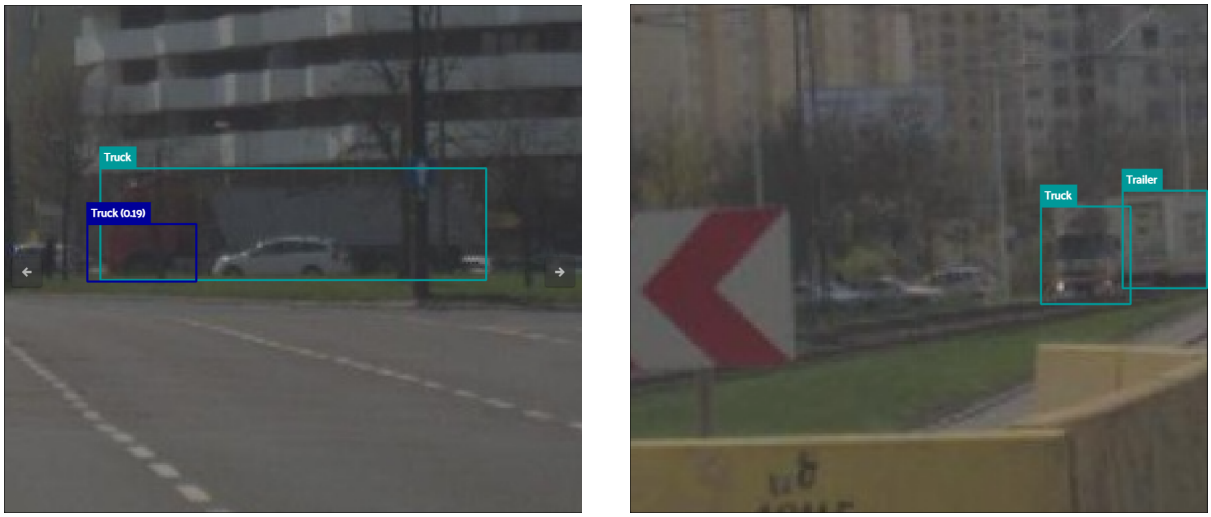
When comparing the results of PROB with curriculum learning to the normal PROB method in Section 4.2 we can see that the mAP is meaningfully increased. Notably, it is evident in Figure 4.3 that the known recall is improved across all the the different object size classes. The greatest improvement percentage wise can be seen among the smaller size classes, which is the result we hoped to achieve with CL. But more surprisingly, the recall saw positive improvement across all size classes, with the greatest improvement in terms of the number of true positives among the bigger size classes. This means that there are no drawbacks with including CL in terms of known performance, except for potentially longer time to convergence.

5.1.3 Known Metrics for Focused Cropping

The results from changing the preprocessing step to use focused cropping instead of random cropping show an insignificant change in mAP for tasks 1 and 2, but a significant increase for tasks 3 and 4, particularly for the currently known classes which are increased

by 202% and 19.8% respectively. Since we selected the classes for task 1 by allocating the most common ones to mimic a real-life scenario, they naturally had a very high representation from the beginning. This explains why they are not significantly affected by the focused cropping.

However, for task 2 we expected a increase of the mAP for the currently known classes, as number for trailer instances in the training set had been increased from 68 to 521. When examining the average precision for trailers in task 2, as shown in Table A.1, we see that both the initial method and the addition of focused cropping result in very low AP for trailers (0.3 and 0.4, respectively). On the contrary we see that the increase of stroller instances in task 3 from 88 initially to 445 when using focused cropping increased the AP from 3.4 to 28.7. This indicates that an increase in AP is not solely dependent on the number of training examples provided. We speculate that an additional factor is that some classes are inherently more difficult to learn, while others are easier. Looking at the dynamic classes in ZOD, the stroller class differentiates from other classes hence making it easier to identify. Trailer on the other hand have more features in common with other classes , such as car, van and truck. Additionally, by visually inspecting the ground truth annotations and predictions, we notice that the labeling of trailers in ZOD is inconsistent, making it harder for the model to learn this class effectively. This is exemplified in Figure 5.2, where in image (b), the trailer is annotated as the rear end of a truck, while in image (a), the entire vehicle is labeled as a truck. This also has effect on the truck predictions as the model in image (a) only predicts the front as truck.



(a) (b)

Figure 5.2: An example of inconsistencies in trailer annotations. In image (b) the vehicle is annotated as truck in the front and trailer in the back, while the same type of vehicle is annotated as truck entirely in image (a). The images show ground truth and predictions

In task 4, we observe an increase in the mAP for the currently known classes. However, when inspecting the AP per class, it is evident that the performance for the wheelchair class remains very poor as Table A.1 shows that the AP remains at 0. We expected a increase for this class as well since the training instances were increased from 7 to 89. Even though the percentage increase is substantial, we believe the poor performance is

due to the training examples still being very low relative to other classes in the training set, where many classes have over 1,000 instances. Additionally, the test set only contains three wheelchairs, giving the model very few opportunities to correctly predict them.

5.2 Unknown metrics

In this section we will discuss our results in unknown metrics. First we will cover why our U-recall is so low across all our tests compared to the benchmark on MS-COCO while WI sees slight improvement in the initial strategy. Then we look at the changes in A-OSE with CL when adding CL and focused cropping, and discuss what these results tells us about the the model and the dataset.

5.2.1 Low U-recall

One thing that we can see across all our results is that the U-recall is very low, especially when compared to PROB's performance on MS-COCO. Comparing the results on MS-COCO and our results on ZOD in Table 4.1, we see that we achieve a U-recall of 4% whereas Zohar *et al.* reaches 19,4%. Since U-recall is not affected by false positive unknowns, there are two possible areas of failure when it comes to U-recall. The first is that an unknown object is simply not detected. The second is that the object is detected, but for some reason the detection does not pass the IoU threshold of 0,5. Again we believe that both of these problems are greatly elevated due to the difficulty of the ZOD dataset. Similarly to our discussion of the performance on the mAP metric, we see that the the thorough annotation of small objects in ZOD makes it very hard to find the objects, and when they are found, it is hard to reach a IoU above 0,5.

It is logical to expect that these problems are even more pronounced when detecting a generic group of unknown objects in a single "Unknown" class rather than detecting a known class from the features specific to that class. For example, it is easier to discern that a tall dark figure with something round on top is probably a human than it is to discern whether a randomly shaped dark figure is an object or not. Given that the background in traffic images is often very filled with nature landscapes or city buildings there is no shortage of randomly shaped dark figures in the background. Therefore a lot of the unknown detections will be objects in the background that will not be labeled in our dataset. This increases the number of true positives which we have argued is not as important as false negatives, but since the detection D-DETR makes a fixed number of predictions (100 in our implementation) it is possible for enough false positives to "use up" enough queries that some unknown labeled objects in the road scene does not get picked up. We believe that this phenomenon is elevated due to the prevalence of small labeled objects in the dataset such as the pedestrians if example (a) of Figure 5.1 because the prevalence of these very small and obscure annotations during the training will teach the objectness head that small obscure shapes are indeed objects worth detecting.

Our second guess to why U-recall is so much worse for ZOD than for MS-COCO is that we have a far lower number of classes (14 instead of 100), which likely makes the objectness head less generalised. One of the things we wanted to examine it this project was whether

dynamic road scene objects are similar enough that the objectness head can learn a good representation of them specifically by learning only from other dynamic road scene objects, and our results show that this is not the case. Instead it seems that even within dynamic road scene objects there is enough variance between the classes that it might be more beneficiary to have a more generalised objectness distribution learned from a wider range of classes that can find a wider range of unknown objects including all unknown road scene objects.

U-recall is not improved by either focussed cropping or CL. This is expected because these additions are focused on improving mAP of the known classes, and have therefore minor to no impact on improving disctinction between unknown objects and background. These additions are aimd at improve training, and since there are no labels for unknown objects during training these additions cannot materially improve U-recall.

5.2.2 Improved Wilderness Impact

In our initial results, the only metric that improved with ZOD compared to MS-COCO is WI. As explained in Section 2.3.5 it measures the reduction in precision for the known classes when unknown objects are introduced to the dataset. This means that with the same performance in open set, a lower WI means a worse performance on closed set. The conclusion is that the performance on ZOD compared to MS-COCO in closed set is comparatively worse than on open set. This is evidence that the lower performance in mAP on ZOD compared to MS-COCO is not due to the combination of open set evaluation on ZOD, but rather something about ZOD itself, which further strengthens our assessment that ZOD is a difficult dataset to detect.

5.2.3 Unknown metric changes with CL

From the CL results in Table 4.2 we can see among the unknown metrics that there are no significant changes in U-recall or WI. A-OSE, however, is worse for task 2 and 3 which means that more unknowns are being classified as known. Why is hard to say for sure, but the fact that mAP simultaneously improves suggests that the improvement in known recall positively outweighs the effects of A-OSE on known precision. Given these results our opinion is that the positives of CL outweighs the downsides.

5.2.4 A-OSE improvement with focused cropping

The only unknown metric that sees substantial improvement with focused cropping is A-OSE with a reduction ranging from 14 to 28 percent across task one through three. This means that there are less unknowns being falsely detected as known objects. Despite this, mAP for task 1 and 2 is not improved, which leads us to believe that there are instead more knowns being labeled as unknown. This would mean that the classification head becomes more biased toward the unknown class with focused cropping. Further research could investigate if suppression of the unknown class logit in the classification head would lead to improved mAP in this case. The other possible reason for why A-OSE is improved but not mAP would be an increased confusion between the known classes, which we do not think is the case because the use of focused cropping increases the number of known

objects in the training data, which, if anything, should reduce the confusion among the known classes.

5.3 Class specific performance

Our results for the initial training show a clear discrepancy between the mean average precision (mAP) for previously known classes (classes in earlier tasks) and currently known classes (classes introduced in the current task). To better understand the underlying factors of this behavior, we reviewed class-specific performance for each task, which can be seen in Table A.1. The data reveals that classes in task 1 consistently outperform other classes across all tasks in terms of average precision (AP). In contrast, several classes, including Heavy Equipment, Trailer, Animal, and Stroller, have significantly lower performance, with mAP scores close to zero.

Figure 5.3 illustrates the relationship between the average precision and the number of instances for each class in each task. Classes are arranged in descending order by instance count. Notably, in the first task, the car class, which has the highest number of instances, also shows the highest AP. As the number of instances decreases, there is a corresponding decline in AP. Although less pronounced, this pattern is seen in subsequent task as well. Thus, the classes with very low representation also have a very low AP. This is likely due to insufficient training examples to effectively learn the distinguishing features of these classes. Limited data can result in a model that struggles to generalize from its training environment to real-world scenarios, leading to poor performance on less common objects or scenarios.

As discussed in Section 3.3.2, we opted to include the most common classes in the first task to simulate a real-world scenario. These classes, being so common, maintain high representation across all tasks, which correlates with a consistently high AP. For all subsequent tasks, the classes from task 1 are considered 'previously known classes. As a result, the mAP for this group remains high. When looking at the 'currently known classes' there are fewer classes to average over and since the mAP is not weighted a low AP for a class has a very high impact. For example, there are only two currently introduced classes in task two - truck and trailer. Truck is represented 1308 times in task 2 while trailer only is only present in 68 instances (due to low general representation in the reduced dataset as seen in Figure 3.1) resulting in a AP of 20.5 and 0.3 respectively. Since the mAP is calculated by just taking the average of the AP for the corresponding classes, the low AP for trailer reduces the currently known mAP to 11.1, which is a huge difference form the previously known mAP of 36.4.

This differentiation helps explain the observed discrepancies in mAP between previously known and currently known classes. To address this imbalance and provide a more accurate reflection of model performance across classes with varying instance counts, adopting a weighted mAP could be beneficial. By calculating mAP based on the number of instances for each class, this method would give more weight to classes with higher

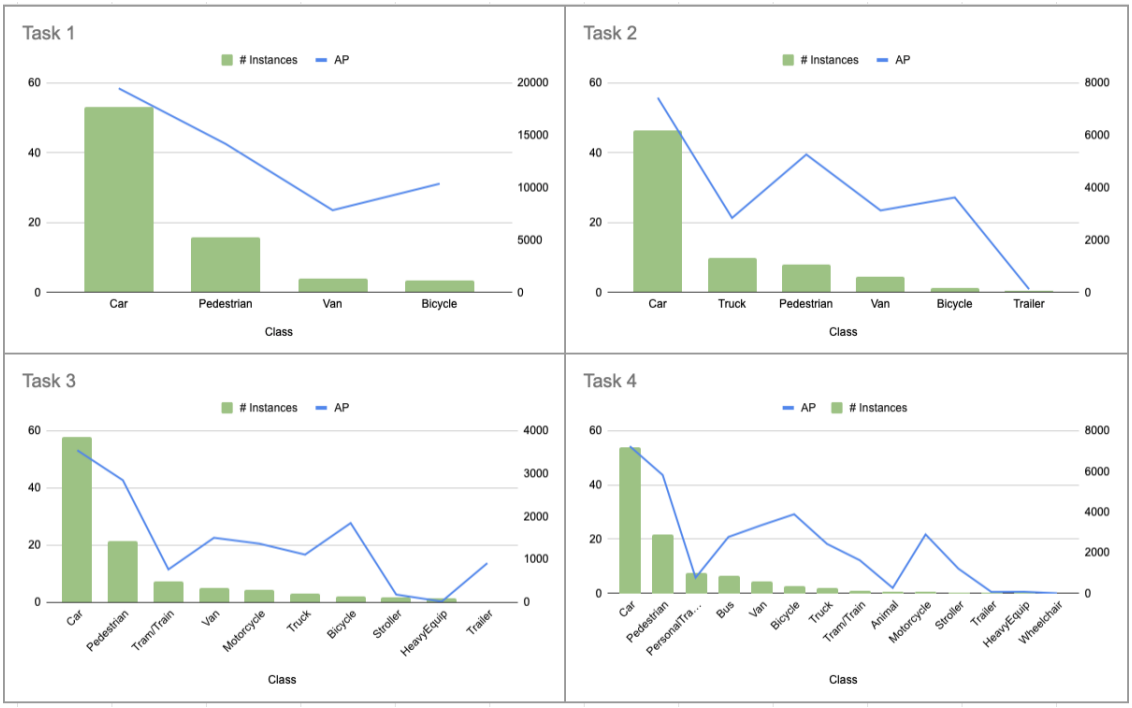


Figure 5.3: Visualization of the number of instances (right axis) and average precision (AP) (left axis) per class and per task. Overall, a decline in AP can be seen as the number of instances decrease.

representation, ensuring that their performance has a proportionate impact on the overall metric.

By implementing this approach, we obtain the revised mAP for each task and class group, as shown in Table 5.1. We can observe that the mAP for the currently known classes have increased compared to the non-weighted mAP for all tasks, but it is still low compared to the mAP for the previously known classes. This is still expected since the classes introduced in task 2,3 and 4 still have few instances compared to the classes in task 1, and thus the model is not provided with enough training examples to learn these classes correctly. The most significant change in mAP when comparing non-weighted and weighted is observed for the combination of the previously and currently known classes (both), as the high mAP and instance count of classes from task 1 heavily influence the overall score.

Task IDs (→)	Task 1	Task 2			Task 3			Task 4		
	mAP (↑)	mAP (↑)			mAP (↑)			mAP (↑)		
	Current	Previously	Current	Both	Previously	Current	Both	Previously	Current	Both
Non-weighted	38,9	36,4	11,1	28	29,4	8,7	21,1	21,7	6,9	17,4
Weighted	53,6	51,4	15,7	46,2	47,5	9,3	42,1	49,9	16,6	45,1

Table 5.1: Comparison of non-weighted and weighted mAP scores Across all tasks. The results are for the initial appraoch to applying PROB to ZOD.

5.4 Difference between Benchmark and Inference

One thing that we would like to clarify is that the way the PROB is implemented by [16] and by us is really not optimized for inference. What we mean by this is that there is no thresholding of the final confidence of the detections and there are no postprocessing methods like non maximum suppression (NMS) in place to assure that there is only one detection per object. One might wonder how we can measure the performance of of the model without postprocessing in place. The answer is that the metrics that we use do not require postprocessing, or have a postprocessing step indirectly build in to them. For example, mAP is measured such that the highest scoring prediction with an IoU above 0.5 with the annotation will be used as the prediction. This means that if you make the right prediction with higher confidence, you won't be penalized by making "wrong" predictions with lower confidence for the same object, which can be seen as an indirect type of postprocessing, only that it includes the use of annotations. This type of matching can be done during testing but not during inference. No postprocessing is needed to calculate U-recall because it is always calculated from the 100 unknown detections with highest confidence, which means that no confidence threshold is involved in the calculation.

In order to remove double detections during inference something like NMS would need to be added. NMS makes sure that whenever the IoU of two detections goes over a threshold (usually between 0.3 and 0.5), the detection with the lower confidence is removed. As explained in Section 2.4, NMS is replaced by Hungarian Matching during training, but Hungarian Matching can not be done during inference because there are no ground truths to match with. In our case, we believe that NMS should be implemented with the adjustment that known object detections always have priority over unknown detections regardless of confidence. This is because we have seen that it is much more common with unknown detections on known objects than known detections on unknown objects, and the confidence of unknowns can often be high. After this step there needs to be thresholding to remove detections with low confidence, which would need to be tuned for the specific use case. The threshold could be applied directly to the logits of the classification head or to the objectness score, or to the product of both which is the confidence used in testing.

Especially, it could be a good idea to have different thresholds for knowns and unknowns individually. We found that correct unknown detections tend to have higher confidence than correct known detections, and we also found that there are a lot of erroneous unknown detections with low confidence, which was much rarer for the known detections. Thus a higher threshold for the unknowns is likely a good idea.

5.5 Objectness temperature

Why a value of 1,3 happened to be a good value for object temperature both for ZOD and for MS-COCO we are not entirely sure. One likely reason is that both datasets have a similar amount of labeled objects per image, 8.5 for our reduced version of ZOD and 7.7 for for MS-COCO [1]. These values being close would increase the chance that the feature embedding distributions for both datasets are similar, thus requiring a similar

object temperature. While this is true, the big difference in number of classes (14 in our version of ZOD and 100 in MS-COCO) also suggests that the distributions should be different, which leads us to believe that 1.3 is not necessearily a value that generalises well between different datasets, but rather that it was a concidence that they both had a similar optimal value. This is further supported by the fact that the shape of the graphs for mAP, U-recall and WI differ substantially from the shapes found in the sweep study for MS-COCO [16].

5.6 Real-World Implications for Autonomous Driving

The integration of open world object detection in autonomous driving systems aims to enhance the vehicle's ability to recognize both known and unknown objects, thus improving overall safety and adaptability. However, our study on the PROB method, applied to ZOD, underscores a significant trade-off: the inclusion of OWOD techniques results in a lower mean Average Precision (mAP) for known objects, while still achieving very low unknown recall (U-recall). This trade-off raises important questions about the practicality and benefits of OWOD in real-world AD applications.

One of the most notable impacts of implementing OWOD is the compromised mAP for known objects. In our study, the focus on detecting unknown objects and incrementally learning these, led to a decrease in the system's ability to accurately recognize and classify known objects. This reduction in mAP has several implications. High mAP for known objects is crucial for the safe operation of AD systems. Known objects, such as other vehicles and pedestrians, constitute the majority of interactions an autonomous vehicle encounters. A drop in detection accuracy for these objects would reduce the safety, potentially leading to situations where the vehicle fails to respond appropriately to common hazards.

Given these challenges, it would not be feasible to replace the current AD object detection systems with OWOD techniques. A more feasible approach could be to use OWOD techniques in parallel with current OD systems rather than replacing them. This parallel approach could leverage the strengths of both methods, ensuring high mAP for known objects while enhancing the system's ability to detect unknown objects. While detecting unknown objects is important, the current methods need significant improvement to make the trade-off worthwhile. Future research should aim to enhance U-recall without sacrificing mAP for known objects.

5.7 Future Work

In light of our findings, several avenues for future research and improvements present themselves. These proposed directions aim to address the limitations encountered in our study and to explore new methods that could enhance the performance and applicability of PROB, particularly in challenging scenarios like those presented by ZOD. By expanding the dataset, refining training techniques, and leveraging new techniques for curriculum learning, we can push the boundaries of the performance of PROB on both ZOD and other

datasets. Below, we outline key areas for potential development and experimentation.

5.7.1 Training with more data

There are some things that should be done differently pertaining to the dataset with more time and compute resources available for training. The training could be done with the whole ZOD dataset instead of a reduced version. This includes using all images (not just the clear/partly clear) and tiling each image fully with crops so that all objects in the dataset are included. A higher amount of data would likely improve the performance for all classes, but especially for the less common classes. It would also show how the model adapts to different weather and light conditions, which is ultimately an important aspect when it comes to applying PROB in real world scenarios. It would also remove the need for focused cropping since all available objects are already in the dataset.

5.7.2 Distance threshold during training

As explained in the discussion of mAP, we believe that the extremely distant and small annotated objects in our dataset are not necessarily conducive to better learning results during training. Therefore, since ZOD includes lidar data with distance annotations, it would be interesting to use the object distance annotations to filter out the smallest/most distant objects during training and see if the performance improves. A distance threshold like this could essentially be tuned as a hyperparameter.

5.7.3 Adapt PROB to Sequential Data

Another interesting area of future work is to adapt PROB to time series data. This would be more realistic for a model deployed in a moving car where objects have continuous presence through a series of frames. It would also allow movement of an object to affect the predictions, which is especially useful in the domain of dynamic objects. For example, if a distant object is at a constant distance from the camera as the camera moves at a speed, chances are that it is also a vehicle on the same road. This could verly likely improve the detections at far distances where visibility is limited.

5.7.4 Alter Unknown Confidence Calculation

Finally, we had an idea about how the scoring of unknown objects could potentially be improved by changing how the final result confidence is calculated. Our proposal is that the final confidence of the unknown objects should be based solely on the objectness score from the objectness head without multiplying the value with the "known/background" logit in the classification head. The reasoning is this: The objectness head will score higher for objects similar to known classes, but the C+1 logit representing "unknown/background" might score higher the less similar an object is to any known object as it is essentially a score of "not known". So when detecting unknowns, there could be a risk that objects that are less similar to knowns actually achieve higher scores simply because the C+1 logit is high, which has the opposite effect of the objectness estimation. In that case maybe it would be reasonable to let the classification head sort unknowns from knowns, but then let the final confidence of the unknowns be just the score from the objectness head as this should favor objects more similar to the known object classes.

5.7.5 Bootstrapping Technique for Curriculum Learning

As mentioned in Section 2.7, one of the main challenges with curriculum learning is how to rank the difficulty of the training examples. In our approach, we follow the setup demonstrated in [4], where we simply rank object instance difficulty by the size of the bounding box area—large objects are easy while small objects are hard. However, another interesting approach would be to try the bootstrapping method presented in [29], where you first train the model on the dataset and then use that model to rank the training examples by difficulty.

As we have already trained PROB without using curriculum learning (the initial approach), one could extend our work by using that model to rank the training examples by difficulty and then apply curriculum learning based on the results. To implement this bootstrapping method, you would start by using our pre-trained model and calculating the loss for each training example based on the model's predictions. The calculated loss for each training example serves as a proxy for its difficulty: a lower loss corresponds to an easy training example, while a high loss corresponds to a hard one. Re-train the model using the ranked training examples, beginning with the easiest examples and gradually introducing more difficult ones.

6

Conclusion

The first objective of this thesis was to find out how a state-of-the-art transformer based OWOD method performs on ZOD in detecting and learning both known and unknown dynamic objects, and we evaluate this using the PROB method [16]. In order to stay within the time and compute constraints of this project we reduced ZOD to a subset of images with good lighting and visibility in order to maximize training speed. Even though this reduces the generality of the trained model, it serves as a test of whether PROB has the ability to be effective on the whole ZOD dataset.

The second objective was to find ways to improve this performance by altering the method itself or the way it is trained. We examined three different approaches to this: tuning the objectness temperature hyperparameter of the PROB model, improving the class diversity of the dataset using focused cropping, and using curriculum learning during training to improve the performance on smaller objects in the dataset. These three modifications, together with the initial application of PROB, resulted in four different experiments.

Consistent across all our experiments on ZOD is that the performance is lower both in metrics for known objects detection such as mAP and metrics for unknown object detection such as U-recall. We believe that this reduction in performance is mainly due to the difficulty of the ZOD dataset, but also partly due to the limited amount of data that was used in our training. The metric that saw the greatest decrease in performance was U-recall with around 80% reduction across all tasks. We argue that the negative effects of the difficult dataset are amplified for unknown objects due to the generality of the "unknown" class. Considering this, the conclusion is that PROB does not perform well on ZOD. This tells us that the domain of road scene image data is challenging for PROB, and perhaps OWOD methods in general, but also that the results could likely could likely be improved with another approach to the annotations in the dataset.

The addition of curriculum learning to the training resulted in a substantial increase in mAP ranging between 10-30% across all tasks. As anticipated, we saw the greatest improvement in recall for the smallest objects in the dataset, but surprisingly it increased

the recall for all objects sizes including the biggest ones, which was an unexpected positive. The only metric that performs notably worse is A-OSE, which increases significantly in the second and third task. Despite the reduced performance in A-OSE, we believe that the positive impact of curriculum learning considerably outweigh the negative in this case. Note that while the results are positive, they still are not close to the metrics on M-OWODB, but it is possible that curriculum learning could be a positive addition when training PROB for M-OWODB as well.

The addition of focused cropping resulted in substantial improvement in mAP for task 3 and 4, which was in line with our expectations considering that the later tasks introduces the more rarely occurring classes. However, we also saw slight deterioration of mAP in task 2 which was unexpected and did not match our expectations. From this we conclude that focused cropping is useful in the case that it is important to maximize the performance on uncommon classes in the dataset, but not necessarily an improvement across all tasks.

More research is needed to better understand how OWOD methods perform in the AD domain as well as other distinct domains of image data. We believe that the most noteworthy result from this project is that we have shown how curriculum learning can be an important tool in the toolbox for OWOD research going forward.

Bibliography

[1] T.-Y. Lin, T.-Y. Lin, M. Maire, *et al.*, "Microsoft coco: Common objects in context," *European Conference on Computer Vision*, 2014. DOI: 10.1007/978-3-319-10602-1_48.

[2] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman, "The pascal visual object classes (voc) challenge," *International journal of computer vision*, vol. 88, pp. 303–338, 2010.

[3] K. J. Joseph, S. H. Khan, F. Khan, and V. Balasubramanian, "Towards open world object detection," *Computer Vision and Pattern Recognition*, 2021. DOI: 10.1109/cvpr46437.2021.00577.

[4] D. Singh, S. N. Rai, K. J. Joseph, *et al.*, "Order: Open world object detection on road scenes," *null*, 2021. DOI: null.

[5] G. D. Biase, H. Blum, R. Siegwart, and C. Cadena, "Pixel-wise anomaly detection in complex driving scenes," *Computer Vision and Pattern Recognition*, 2021. DOI: 10.1109/cvpr46437.2021.01664.

[6] S. Gilroy, D. Mullins, E. Jones, A. Parsi, and M. Glavin, "E-scooter rider detection and classification in dense urban environments," *Results in Engineering*, vol. 16, p. 100 677, 2022.

[7] M. Alibeigi, W. Ljungbergh, A. Tonderski, *et al.*, "Zenseact open dataset: A large-scale and diverse multimodal dataset for autonomous driving," *IEEE International Conference on Computer Vision*, 2023. DOI: 10.1109/iccv51070.2023.01846.

[8] X. Zhao, Y. Ma, D. Wang, Y. Shen, Y. Qiao, and X. Liu, "Revisiting open world object detection," *IEEE transactions on circuits and systems for video technology (Print)*, 2023. DOI: 10.1109/tcsvt.2023.3326279.

[9] S.-A. Rebuffi, S.-A. Rebuffi, A. Kolesnikov, *et al.*, "Icarl: Incremental classifier and representation learning," *Computer Vision and Pattern Recognition*, 2017. DOI: 10.1109/cvpr.2017.587.

[10] W. J. Scheirer, W. J. Scheirer, A. Rocha, *et al.*, "Toward open set recognition," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2013. DOI: 10.1109/tpami.2012.256.

[11] A. R. Dhamija, A. R. Dhamija, M. G^{o}nther, *et al.*, "The overlooked elephant of object detection: Open set," *IEEE Workshop/Winter Conference on Applications of Computer Vision*, 2020. DOI: 10.1109/wacv45572.2020.9093355.

[12] A. Bendale, A. Bendale, T. E. Boult, and T. E. Boult, "Towards open set deep networks," *Computer Vision and Pattern Recognition*, 2016. DOI: 10.1109/cvpr.2016.173.

[13] J. Knoblauch, J. Knoblauch, H. Husain, *et al.*, "Optimal continual learning has perfect memory and is np-hard," *International Conference on Machine Learning*, 2020. DOI: null.

[14] A. Prabhu, A. Prabhu, P. H. S. Torr, P. H. S. Torr, P. K. Dokania, and P. K. Dokania, "Gdumb: A simple approach that questions our progress in continual learning," *European Conference on Computer Vision*, 2020. DOI: 10.1007/978-3-030-58536-5_31.

[15] A. Gupta, S. Narayan, K. J. Joseph, S. H. Khan, F. Khan, and M. Shah, "Ow-detr: Open-world detection transformer," *Computer Vision and Pattern Recognition*, 2021. DOI: 10.1109/cvpr52688.2022.00902.

[16] O. Zohar, K.-C. Wang, and S. Yeung, "Prob: Probabilistic objectness for open world object detection," *Computer Vision and Pattern Recognition*, 2023. DOI: 10.1109/cvpr52729.2023.01101.

[17] S. Ma, Y. Wang, Y. Wei, *et al.*, "Cat: Localization and identification cascade detection transformer for open-world object detection," *Computer Vision and Pattern Recognition*, 2023. DOI: 10.1109/cvpr52729.2023.01885.

[18] M. A. Iqbal, Y. Yoon, M. U. S. Khan, and S. K. Kim, "Improved open world object detection using class-wise feature space learning," *IEEE Access*, 2023. DOI: 10.1109/access.2023.3335602.

[19] R. Fang, G. Pang, L. Zhou, X. Bai, and Q. Xu, "Unsupervised recognition of unknown objects for open-world object detection," *arXiv.org*, 2023. DOI: 10.48550/arxiv.2308.16527.

[20] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-end object detection with transformers," in *European conference on computer vision*, Springer, 2020, pp. 213–229.

[21] R. Fang, G. Pang, L. Zhou, X. Bai, and J. Zheng, "Unsupervised recognition of unknown objects for open-world object detection," *arXiv preprint arXiv:2308.16527*, 2023.

[22] R. Stewart, M. Andriluka, and A. Y. Ng, "End-to-end people detection in crowded scenes," in *Proceedings of the IEEE conference on computer vision and pattern recognition*, 2016, pp. 2325–2333.

[23] X. Zhu, X. Zhu, W. Su, *et al.*, "Deformable detr: Deformable transformers for end-to-end object detection," *arXiv: Computer Vision and Pattern Recognition*, 2020. DOI: null.

[24] P. Sun, P. Sun, H. Kretzschmar, *et al.*, "Scalability in perception for autonomous driving: Waymo open dataset," *Computer Vision and Pattern Recognition*, 2020. DOI: 10.1109/cvpr42600.2020.00252.

[25] Y. Bengio, Y. Bengio, J. Louradour, *et al.*, "Curriculum learning," *International Conference on Machine Learning*, 2009. DOI: 10.1145/1553374.1553380.

[26] J. L. Elman and J. L. Elman, "Learning and development in neural networks: The importance of starting small," *Cognition*, 1993. DOI: 10.1016/0010-0277(93)90058-4.

[27] L. Jiao, L. Jiao, F. Zhang, *et al.*, "A survey of deep learning-based object detection," *IEEE Access*, 2019. DOI: 10.1109/access.2019.2939201.

[28] O. Zohar, *Comment on issue #19: Too many unknown objects with high confidence*, `https://github.com/orrzohar/PROB/issues/19`, Accessed: 2024-05-18, GitHub, 2023.

[29] G. Hacohen, G. Hacohen, D. Weinshall, and D. Weinshall, "On the power of curriculum learning in training deep networks.," *International Conference on Machine Learning*, 2019. DOI: `null`.

A

Appendix 1

A.1 Training code

```
$PYTHON_CMD -u main_open_world.py \
    --output_dir "${EXP_DIR}/taks_B" --dataset ZOD --
   PREV_INTRODUCED_CLS 4 --CUR_INTRODUCED_CLS 2\
    --train_set 'task_B' --val_set 'zod_val' --epochs 51\
    --model_type 'prob' --obj_loss_coef 8e-4 --obj_temp 1.3 --
   freeze_prob_model\
    --exemplar_replay_selection --exemplar_replay_max_length 1743 --
   exemplar_replay_dir ZOD\
    --exemplar_replay_prev_file "learned_zod_taskA_ft.txt" --
   exemplar_replay_cur_file "learned_zod_taskB_ft.txt"\
    --pretrain "${EXP_DIR}/task_A/checkpoint0040.pth" --lr 2e-5\
     ${PY_ARGS}

$PYTHON_CMD -u main_open_world.py \
     --output_dir "${EXP_DIR}/task_B_ft" --dataset ZOD --
   PREV_INTRODUCED_CLS 4 --CUR_INTRODUCED_CLS 2 \
     --train_set 'learned_zod_t2_ft' --val_set 'zod_val' --epochs 111
   --lr_drop 40\
     --model_type 'prob' --obj_loss_coef 8e-4 --obj_temp 1.3\
     --pretrain "${EXP_DIR}/t2/checkpoint0050.pth"\
     ${PY_ARGS}
```

Figure A.1: An extract of the training code. For each task, except the first, there is initial training on the chosen dataset for that task, it is then followed by fine tuning using the exemplar replay file from the previous task.

A.2 Average Precision per Class

Task 1

	Car	Pedestrian	Bicycle	Van	Truck	Trailer	Motorcykle	Tram/Train	Stroller	HeavyEquip	nal Transj	Bus	Animal	Wheelchair
PROB (ZOD)	59.8	43.4	**42.3**	19.2	-	-	-	-	-	-	-	-	-	-
PROB + CL (ZOD)	**63.1**	**47.7**	32.9	27.3	-	-	-	-	-	-	-	-	-	-
PROB (ZOD_FC)	53.8	39.2	**42.3**	**28.8**	-	-	-	-	-	-	-	-	-	-

Task 2

	Car	Pedestrian	Bicycle	Van	Truck	Trailer	Motorcykle	Tram/Train	Stroller	HeavyEquip	nal Transj	Bus	Animal	Wheelchair
PROB (ZOD)	56.6	41.0	35.4	20.5	16.5	0.3	-	-	-	-	-	-	-	-
PROB + CL (ZOD)	**61.0**	**43.4**	**35.7**	**28.3**	**24.6**	**0.6**	-	-	-	-	-	-	-	-
PROB (ZOD_FC)	52.8	34.5	34.5	26.8	15.4	0.4	-	-	-	-	-	-	-	-

Task 3

	Car	Pedestrian	Bicycle	Van	Truck	Trailer	Motorcykle	Tram/Train	Stroller	HeavyEquip	nal Transj	Bus	Animal	Wheelchair
PROB (ZOD)	53.6	42.0	**36.2**	23.8	16.3	0.4	19.4	5.7	7.3	0.1	-	-	-	-
PROB + CL (ZOD)	**61.6**	**44.2**	33.3	**27.8**	**19.8**	0.4	**24.1**	12.4	3.4	**27.4**	-	-	-	-
PROB (ZOD_FC)	52.8	39.4	34.6	25.2	17.8	0.4	18.8	**23.0**	**28.7**	1.2	-	-	-	-

Task 4

	Car	Pedestrian	Bicycle	Van	Truck	Trailer	Motorcykle	Tram/Train	Stroller	HeavyEquip	nal Transj	Bus	Animal	Wheelchair
PROB (ZOD)	58.1	42.3	34.2	27.9	17.5	**1.7**	17.6	17.8	11.3	0.8	12.0	23.5	0.8	0
PROB + CL (ZOD)	**62.4**	**46.5**	33.4	**31.0**	18.1	1.6	**24.7**	**21.9**	**19.3**	0.2	9.7	**27.7**	**10.8**	0
PROB (ZOD_FC)	53.8	39.5	**35.6**	23.1	**22.7**	**1.7**	17.8	15.5	15.9	**21.6**	**14.8**	24.3	4.4	**3.0**

Table A.1: The average precision per class for all four task and all three variations of applying PROB to ZOD (the initial method (PROB (ZOD), the method of adding curriculum learning (PROB + CL (ZOD)), and the method of adding focused cropping (PROB (ZOD_FC)).