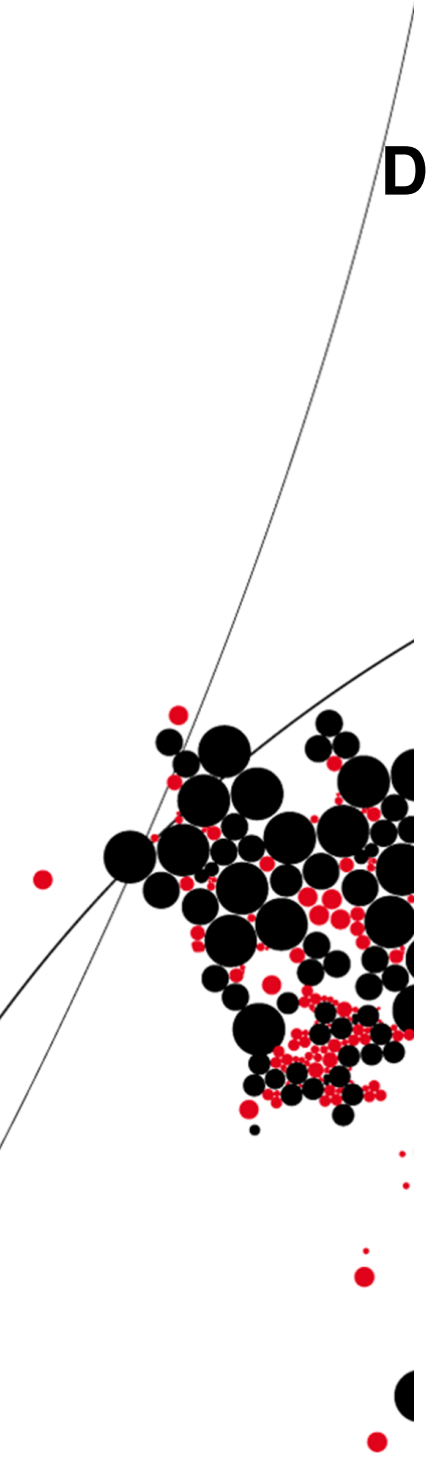
UNIVERSITY OF TWENTE.

**Faculty of Electrical Engineering,
Mathematics & Computer Science**

Diffuse more objects with fewer labels

Leander van den Heuvel

M.Sc. Thesis November 2023



Supervisors:
dr. ir. G.J. Burghouts - TNO,
msc. S.B. van Rooij - TNO,
dr.ing G. Englebienne,
dr. E. Mocanu

Telecommunication Engineering Group
Faculty of Electrical Engineering,
Mathematics and Computer Science
University of Twente
P.O. Box 217
7500 AE Enschede
The Netherlands

Preface

Saying goodbye is never easy. I suppose this also holds for the end of my time as a student at the university. Actually, the saying manifests itself in multiple ways. First, my time as a student is something I am very grateful for and I will probably look back a little bit nostalgically. The countless enjoyable moments during activities/training/events/board year I had during my time at the Student Rowing Association Phocas, is something I will remember. It brought me friendships that still last, and I will continue to last for a long time. The student time is something I thoroughly enjoyed, but all good things must come to an end. As I started to do my masters, the late nights came less often, study had more priority, and thoughts about a future career came to mind more often.

Which brings me to the second reason for the opening sentence. When the master thesis finally came into sight, I was very excited, but also a bit nervous. I wasn't sure if I would find the focus and motivation in such a long project. Would I find a subject that matched the long-anticipated expectations of a master's thesis? These worries quickly disappeared after my first weeks at TNO. As I was surrounded by people who had the same passion for Deep Learning and computer vision, I was inspired and encouraged to work and dig a little deeper into the subject of my thesis. The help and advice of colleagues in the department definitely made a difference in both the experience and the end result. Also, the weekly meetings with David, who provided me with ideas and theoretical feedback in the field of Diffusion Models, have brought my understanding of Diffusion models and this thesis to a higher level.

However, I would like to express my greatest gratitude to my supervisors Gertjan and Sabina. Their regular advice, ideas and constructive feedback was the defining help that really encouraged and inspired me throughout the project. Under their supervision, I received many opportunities to learn, experience and improve my thesis. One of the opportunities was a submission (an adapted version and improved method) of this thesis at the Workshop on Diffusion Models at NeurIPS, which was eventually accepted! I am very grateful for the nice collaboration we had (and happily, it didn't end there as I started working as an employee at TNO!). I will definitely cherish the nice memories and my time as an intern at TNO.

Summary

0.1 Introduction

In the field of computer vision, accurate and efficient object detection is crucial for a multitude of applications ranging from autonomous vehicles to surveillance systems. Most object detection models are trained in a supervised fashion, requiring a large corpus of images with objects of interest annotated. Labelling objects in images typically requires a human annotator that annotates every object of interest in an image. Therefore, creating large datasets for object detection models is often cumbersome and requires a lot of human effort. To reduce human effort, new strategies for training object detection models are required.

This work iterates on the domain adaptation concept, this field leverages an existing dataset(source domain) and uses a different but similar dataset (target domain) to detect objects in the images of interest. Current domain adaptation approaches are still outperformed by fully supervised approaches, where the target domain contains a sufficient amount of labelled images to train the model without any source domain. To improve the performance in the target domain with as few as labelled samples possible, this work proposes to finetune an object detection model using pseudo-labels from the target domain. However, this requires useful pseudolabels that do not have too many errors, which is challenging since the model is not trained on the target domain. This leads to the following two questions: can we enhance the recall of an object detection model without any fine-tuning in a target domain (1) and can we leverage the enhanced recall for more efficient domain adaptation with lower human annotation effort using human-verified pseudo-labels(2)?

To improve the detections without any labelled data, an ideal model would scale in performance in trade for compute time. Since the pseudo-labels only need to be generated once, computation time is less costly. Most object detection models have a deterministic output, this means that for an image x, output y will always be the same. This deterministic property makes performance scaling hard since the output cannot be changed. Instead, this work uses DiffusionDet, an object detection model that takes the image and a set of random bounding boxes and iteratively

aligns the location, aspect ratio and size with the objects in the image. Due to the random input, the output can vary between predictions. This work aims to leverage this stochastic property to improve the detections. Two methods are proposed: (1) Add more random bounding boxes as input, to increase the likeliness of detection of an object and (2) aggregate the output over multiple runs to increase the recall. To answer the second research question, a human-in-the-loop is proposed to verify the improved pseudolabels and these labels are used to fine-tune the object detection model. The assumption is that training with human-verified pseudo-labels requires relatively less human effort compared to training labels with human-generated labels.

To verify the experimental settings without any fine-tuning, the following domain adaptation task was proposed: a trained model on the source domain MS-COCO and evaluated on the VisDrone dataset. The shift between the domains involved a different viewing angle of the objects of interest. Generally, images from MS-COCO have a frontal view, whereas the images from VisDrone have an aerial perspective and were taken from a drone. In practice, images in VisDrone contained smaller objects with a different viewing angle compared to MS-COCO. In the first experiment, the detection performance of detections with more runs and more random bounding boxes was evaluated on VisDrone. The results revealed that adding more runs increased the recall and precision, a similar effect was also found for adding more random bounding boxes.

To reveal whether the improved detections could be utilised as pseudo-labels with a human in the loop, another experiment was conducted. Pseudo-labels were generated using the more runs setting with a human in the loop that would verify the labels by creating crops in the images containing correct predictions. Finally, the model was fine-tuned using the human-verified crops combined with either 10 or 50 ground-truth (GT) labelled images. This fine-tune setting was compared against a baseline model that was trained on ground-truth images only (either 10 or 50 GT samples). To reveal the effect of the improved detections, another model was trained on the same human-verified image crops containing pseudo-labels generated from a single run. Finally, an upper bound model was created that used the same human-verified crops containing the ground truth labels. Evaluation on VisDrone revealed that our proposed method outperformed the baseline model and also the model trained on the human-verified crops from a single run in both 10 and 50 GT samples. By calculating the human effort in hours per experimental method, this work revealed that the proposed method requires the least human effort to improve detection performance.

Contents

Chapter 1

Introduction

1.1 Motivation

In the realm of computer vision, the accurate and efficient detection of objects within images and videos serves as the pillar upon which a multitude of applications, from autonomous vehicles to surveillance systems, rely for their success and impact. The field of object detection is concerned with locating and identifying instances of interest in an image, a task that is often trivial for humans but can be challenging for a machine. Throughout the years, object detection models have received more applications [1]. For example, some applications are autonomous driving, robot vision, and video surveillance [1], [2]. The ability to automatically identify and locate objects makes these models useful for real-world applications.

Object detection in deep learning is generally considered a supervised learning problem [2]. Supervised learning for object detection typially involves a deep learning model that is exposed to pairs of data samples and corresponding labels. Using the datasamples, the model creates predictions that are corrected by the corresponding labels from the data samples. The labels serve as corrections to refine the model's predictions by updating the model's parameters. In this way, the model is tuned to produce object labels for a designated object detection task [3]. Consequently, the development of an effective object detection model requires a comprehensive dataset, comprising data samples along with their associated labels, customised for a specific application.

Current state-of-the-art deep learning object detection algorithms rely on a large corpus of labelled data for training, such as YOLO [4], EfficientNet [5] or Faster R-CNN [6]. These models are trained on large datasets that have a high number of labelled objects, ranging from 860k to 1.4 million labelled objects [7], [8]. The high number of data samples is required because the models need to learn to recognise objects in various settings such as different lighting conditions, varying viewing angles, and many backgrounds.

However, annotating large datasets is time-consuming, and since most work annotation is done by human labellers, it requires a lot of human effort. In object detection, human labellers are required to put bounding boxes with labels around each object in an image. This repetitive process takes a human annotator on average 50 seconds per object in an image that likely contains multiple objects. For the large datasets of up to 1.4 million objects currently used for training object detection models, the required human effort for creating the labels is enormous.

However, in some scenarios where the source and target domain do resemble, labelling a novel large dataset is not required for training an object detection model. For example, a source domain containing drawn objects such as cars and pedestrians can be used to pre-train a model for detecting real-world similar objects such as bicycles or different types of cars. After training with a larger dataset, the model is fine-tuned with the target dataset containing a few samples of the target domain to leverage the trained features for the target task. This process of adapting a model from one domain to another while having the same task is called domain adaptation. Domain adaptation allows one to train a model with few data samples of the target domain. During the fine-tuning process, the model adapts to the shift in the underlying distribution of the data samples.

Although domain adaptation enables improvements in the target domain with fewer labelled samples, performance often does not come close to fully supervised approaches [9] and offers room for improvement. One approach to improve domain adaptation is to combine existing strategies from other fields that aim to train with fewer labelled samples with domain adaptation. One interesting field that enables training on both unlabelled and labelled data is semi-supervised training. Within this field, a promising approach uses generated labels from the object detection model as training data to fine-tune the model. In [10], pseudo-labels are used to enhance the performance of the model. The authors introduce a teacher model, that generates pseudo-labels and a student-model that learns from the pseudo-labels. Using an exponential moving average, the fine-tuned weights of the student model are transferred to the teacher model. By combining the ground-truth (GT) labels and pseudo-labels, the teacher and student model converge together and the authors report that their method outperforms the baseline with GT labels only. By substituting GT labels with pseudo-labels, the field of semi-supervised learning aims at training with less labelled data.

1.2 Problem statement

This raises the question of whether domain adaptation can be improved with fewer labelled data samples using pseudo-labels from the target domain. Domain adap-

tation poses a strategy for training with fewer labelled samples for a target domain. However, for a sufficiently large domain gap, the model may still require a significant amount of labelled data to fine-tune the outputs for the target domain. Consider the difference between detecting objects from a frontal view compared to detecting objects from the air (aerial view). In this gap, similar objects can have different visual features since the different angles of view can expose new visual object properties. Another gap from the source to the target domain is that the object size will be smaller on average, since the distance from the capturing device to the object is larger. The illustrated domain gap will likely pose a problem for an object detector that has been trained with only objects from a frontal view. Therefore, an efficient training strategy is required to effectively adapt the model from one domain to another. More precisely, we consider the following domain adaptation problem:

- We have a model that is trained on the source domain, that involves detecting images from a frontal view. This domain has a sufficient number of data samples for training.
- We want to fine-tune the model for the target domain which has different properties. Objects in images are observed from an aerial view. Consequently, this results in detecting objects that are, in general, smaller and have different viewing angles. In this domain, labelled data samples are sparse, so effective fine-tuning is desirable.

1.3 Research questions

Having a large domain gap reduces the quality of the detections in the target domain. Initially, this makes fine-tuning for the target domain with pseudo-labels less attractive. Training with pseudo-labels works under the condition that the labels provide the model with sufficient qualitative information [11]. With a large domain gap, the quality of the pseudo-labels might not be sufficient for pseudo-labelling, which can result in a drop in accuracy after fine-tuning with pseudo-labels. However, if the recall and precision of the predictions can be improved, training with pseudo-labels might be beneficial. In this scenario, domain adaptation with fewer labelled samples would require two steps. First, improve the labels generated without any data from the target domain. Second, fine-tune the model with the enhanced detections to fine-tune with the pseudo-labels. Following these steps, we aim to answer the following research questions for a large domain gap:

- Can we enhance the recall of an object detection model without any fine-tuning in a target domain?

- Can we leverage the enhanced recall for more efficient domain adaptation with lower human annotation effort using human-verified pseudo-labels?

Chapter 2

Related work

This section covers the related work that aims to reduce the human labelling effort by training with fewer labelled data samples. To obtain qualitative pseudo-labels, several methods for detecting objects are discussed. By analysing work in the field of object detection, we explore methods that might enable us to find qualitative pseudo-labels and also explore methods that enable us to improve recall and precision without any fine-tuning in the target domain. Training with fewer labelled samples can occur in various scenarios, for example in Active Learning, Semi-supervised Learning and Domain Adaptation. In the second part of this section, several fields that are concerned with training with fewer labelled samples are discussed that can be leveraged to reduce the human labelling effort.

2.1 Object detection

The domain of object detection pertains to the tasks of both localizing and categorizing objects within an image. These objects belong to predefined classes and necessitate localization through axis-aligned bounding boxes. As mentioned in the research questions, the aim is to find pseudo-labels in the target domain that improve the performance in fine-tuning. Therefore the goal of this section is to identify a method that detects correct labels in the target domain as much as possible without any fine-tuning yet. Since these labels only need to be generated once for fine-tuning, inference speed is less important and can be traded for accuracy. Ideally, a model that improves accuracy while trading off inference speed would benefit this scenario.

An exciting field that concerns scaling detection performance across devices is once-for-all (OFA). Once-for-all refers to training the model once and using the trained parameters for all levels of computing. In [12] an OFA network is proposed that can scale with different compute requirements. This model has to be trained

once, and the same network parameters can be used to scale the performance for all different levels of compute power. The authors report that their OFA model in terms of latency outperformed EfficientNet [13] while maintaining the same accuracy. Unfortunately, the trained model in the author's work only scales downwards for different levels of computing. Specifically, subnetworks can be derived from the original model, indicating that its accuracy can only decrease and not improve. For leveraging the source domain as effectively as possible, a method that can scale upwards is required.

More recent work from [14] introduces DiffusionDet that has a similar once-for-all property but might also scale upward in accuracy after training. This model takes as input a set of proposal bounding boxes with random aspect ratio and scale and gradually adjusts the size, aspect ratio and location of the boxes to match the alignments of the objects. The boxes are iteratively adjusted in steps that can be set as a hyperparameter. Low-confidence boxes are removed, and Non-Maximum Suppression (NMS) is applied afterwards to remove class-specific duplicate predictions. The authors report that they can adjust the output by using more proposal boxes and/or using more sample steps. They report that their method can achieve higher performance at the cost of requiring more computing power. The characteristics of this model can be leveraged to bridge the domain gap with improved samples without fine-tuning which addresses the first research question. The network parameters obtained by training in the source domain can be leveraged to change the output at inference time. This could lead to improved detection in the target domain. Therefore, DiffusionDet will be used to investigate whether it is possible to improve the detections.

To validate possible performance improvements with DiffusionDet, a baseline comparison method is required. Currently, there are roughly two types of object detectors, single-stage and two-stage detectors. Single-stage detectors localise and classify objects in a single shot using dense sampling. These models detect objects in a single stage by using predefined boxes/keypoints of various scales and locations. Two-stage detectors have a separate module to generate region-based proposals. In the second stage, the region-based proposals are classified, low-confidence boxes are removed, and the coordinates of the proposals are refined. Since two-staged detectors use two separate stages to detect objects, they are generally slower than single-stage detectors.

Starting with single-stage detectors, Single Shot MultiBox Detector (SSD) by [15] extends VGG-16 [16], using additional auxiliary feature layers. This extension enabled the model to predict boxes of different scales and aspect ratios and their corresponding sizes. The first version of SSD struggled to predict small objects that are clustered together with substantial mutual occlusion. These were largely resolved in

later versions of SSD with more effective Backbones, such as ResNet [17].

The model from [4] named You Only Look Once (YOLO), is one of the most well-known object detectors. After the introduction of the original work, many iterations have been made to improve the model in terms of inference speed and accuracy [18], [19], [20]. This approach has been characterised by the division of the input image by a $S \times S$ grid. Each grid cell is responsible for predicting the bounding box in which the centre is located in the grid cell. The localization of the object is treated as a regression problem by using the grid cell's pixel information to predict the centre, size and confidence of the box. At the same time, the class probability of each grid cell is calculated and combined in a single tensor with the box coordinates and confidence. Finally, low-confidence predictions are removed and NMS is applied. The first YOLO model had similar issues as SSD with predicting small clustered objects, and the number of objects per grid cell was limited. Most of these drawbacks were resolved in later iterations. Although SSD and YOLO received architectural improvements and optimised training strategies, eventually YOLO outperforms SSD in most scenarios [21], [22], [23].

Continuing with two-stage detectors, the Region-Based Convolutional Neural Network (R-CNN) [24] was the first architecture proposed for the R-CNN family. This network has a region proposal module that produces 2000 object candidates. Using Selective Search [25], candidates with the highest probability of having an object are identified. These candidates are propagated through a CNN network that calculates a feature vector. The feature vector is passed through a class-specific Support Vector Machine (SVM) to obtain the confidence scores of the candidates. NMS is applied afterwards to remove class duplicates.

Several improvements after the introduction of the original R-CNN model have been proposed, part of which is described here. The Fast R-CNN model [26], enabled end-to-end training and shorter inference times. Faster R-CNN [6] further improved inference speed and detection performance. Another proposed improvement, Mask R-CNN [27] further enhanced the detection performance by adding a segmentation branch along with the detection and classifying heads. Although YOLO is a single-stage detector, most comparisons between Mask R-CNN and YOLO reported that YOLO outperformed Mask R-CNN in several computer vision tasks [28], [29], [30], [31], [32]. Therefore, YOLO seems the best option to compare DiffusionDet's performance against. A comparison in [33] revealed that from YOLOv5 onwards, newer YOLO models trade between speed and accuracy and make fundamental gains in both metrics. Since YOLOv5 [34] has an implementation in PyTorch [35] this work uses YOLOv5 as a baseline.

2.2 Domain adaptation for object detection

Often object detectors are pre-trained on a large-scale dataset. When utilizing it for a target domain, it needs to be fine-tuned with labelled data from the target domain. As mentioned before, in some scenarios, there are few available labelled data samples. Domain adaptation leverages the labelled data from the source domain to reduce the required labelled samples in the target domain. Some publications make different assumptions about the properties of the source and target domains, therefore, there are different methods to apply domain adaptation. One distinction within the field is the unsupervised [36], [9] and the supervised training procedures [37], [38], both are used in domain adaptation. Most of the work in domain adaptation is in the field of image classification and uses unsupervised training for Domain adaptation [38]. The field of image classification is concerned with assigning a label or category to an entire image. Image classification models take an image as input and output a class label that corresponds to the image. In contrast to supervised learning, unsupervised learning does not require labelled data, the model can identify patterns in the data without any labels. The most widespread objective of unsupervised learning in domain adaptation is to maximise domain invariance [36]. This means that a model needs to find domain-invariant representations throughout training [36], [39]. Theoretically, finding domain-invariant representations would lead to improved classification performance in both domains.

In the work from [40] an unsupervised method is shown using an adversarial model to train the image classification model. The adversarial model aims to identify whether an image is from the source or target domain using the latent space feature vector from the image classification model. Joint training of these networks enables the generator network to create domain-invariant latent feature vectors. The authors report that their work outperforms state-of-the-art results from [41] which uses subspace alignment for domain adaptation. Another approach of maximisation of domain invariance is from [42]. The authors introduce a novel CNN architecture that uses an additional domain adaptation layer. The output of this layer is used to calculate the Maximum Mean Discrepancy (MMD) between the source and target data. Using the loss of the classification error and MMD, the authors claim to obtain discriminative but domain-invariant features in their model. They report that their work outperforms previous state-of-the-art work on several domain adaptation benchmarks for image classification. Although both publications show notable target-domain performance improvements, their work is limited to image classification.

Although there is empirical evidence that domain adaptation benefits from maximising domain invariance, in [43] the objective for domain-invariant features in Deep

Learning models has been criticised. The authors show that there is a fundamental trade-off between learning invariant representations and achieving a small joint error among the source and target domains. They argue that successful deep domain adaptation methods require better alignment of domain conditional distributions. Another downside of these approaches is that they do not translate to object detection tasks. Object detection poses an additional challenge of locating the objects, therefore there are multiple objects which makes domain invariance maximization more challenging. This makes exploring maximizing domain invariance for domain adaptation in object detection not fit in the scope of this work.

Although less work has been done in domain adaptation for object detection, in [44] an intermediate dataset is proposed to bridge the domain gap in an Object Detection task. The model is fine-tuned using an intermediate synthetic dataset to bridge the domain shift. Using a CycleGAN [45], the authors create a synthetic training set to ease the domain gap from source to target. The authors report that their methods outperform other methods in most of their metrics. However, the creation of an intermediate dataset using an additional model introduces more complexity to the domain adaptation task. The CycleGAN as reported in the original work should be tuned well enough to create representative samples that bridge the source and target domain. This makes domain adaptation using this method unsuitable for the time available in this work.

Overall, leveraging the source domain for training with fewer labelled samples from the target domain seems like a good starting approach. However, the provided methods for domain adaptation provide additional challenges for object detection that do not fit in the scope of this work. Moreover, domain adaptation methods do not provide the same performance as supervised methods [9] trained in the target domain. This performance deficit leaves space for improvement for more effective fine-tuning in the target domain. Therefore, we want to explore whether the adaptation process can be further improved using other methods that enable training with fewer labels.

2.3 Active Learning

Another method that enables training with fewer labelled samples is Active Learning. This field aims to find a method that selects a subset of the training data that is most effective for learning [46], [47]. The field assumes that a set of samples in the target domain has yet to be labelled by an oracle. Therefore, Active Learning requires a strategy that selects the best images for training without any knowledge of the labels. The human labelling effort is reduced by finding an optimal subset of unlabelled training data. This field aims to find representative samples that allow the

model to account for the shift in the data sample distribution between the domains.

Most of the work in the visual field of Active Learning concerns image classification [46]. However, [48] proposes a margin sampling approach from [49] to use Active Learning for object detection. This approach calculates the class confusion of an object between the two classes with the highest confidence. Conceptually, this measure indicates the uncertainty of a detection in an image. The authors explore several methods for translating individual object uncertainties to image uncertainty. They report that on average the summation of the object uncertainties provides the best proxy for sampling images. The authors report that their sampling strategy outperforms the random sampling baseline. Although these results show that the class-confusion can pose a sampling strategy for Active Learning, the benefit compared to a random baseline is relatively small. Moreover, this method solely focuses on uncertainty during selection without taking the notion of the informativeness of the sample in the target domain. For domain adaptation, Active Learning does not seem to provide methods that fits to the scope of this work to reduce the human labelling effort for object detection. Therefore, this work does not utilize Active Learning methods and further exploration of fields that are concerned with training with fewer labels is required.

2.4 Semi-supervised training

Another field that is concerned with training with fewer labels is semi-supervised learning. Instead of training solely on labelled data, semi-supervised training uses both labelled and unlabelled data [50]. One possible approach within semi-supervised training is the use of pseudo-labels that are used for training with unlabelled data. In [51] an approach named Self-Training and the Augmentation driven Consistency regularization (STAC) is proposed. The authors train a teacher model with labelled data until convergence is reached. This object detection model is used to generate pseudo-labels from the unlabelled dataset. In the second step, another object detection model is fine-tuned together with GT labels and pseudo-labels to further enhance the performance. In addition to the pseudo-labelling strategy, data augmentations are used to further improve the performance of the model. The authors report that their training method outperforms the baseline training model that has only been trained on a random subset of labelled samples.

Although STAC outperforms the baseline, the authors do note that class imbalance of the labelled data could lead to reduced pseudo-label quality. To address this issue, Unbiased Teacher is introduced by [10]. The authors propose a teacher and student model that jointly learn from the pseudo-labels. The teacher model provides the student with pseudo-labels for fine-tuning and the teacher model is updated by

the student model via an exponential moving average (EMA). This allows the model to gradually improve the pseudo-labels. To address the potential class imbalance, Focal loss [52] is used to down-weigh overly confident pseudo-labels. The authors report that their method outperforms STAC on all their metrics. STAC and Unbiased Teacher show impressive results for training with fewer labels. However, both papers focus on in-domain performance improvements where the pseudo-labels from unlabelled data from the source domain. The proposed methods are not used for domain adaptation, where the quality of pseudo-labels might deteriorate because of the domain gap. This raises the question of whether pseudo-labelling can be used for domain adaptation.

The work of [53] provides insight into whether pseudo-labels can be used for domain adaptation of object detection. The authors propose an unsupervised approach by only fine-tuning with pseudo-labels in the target domain. The work introduces a confidence metric, which is a combination of classifier confidence and the variance between bounding box locations. Using the confidence of the predictions in the target domain, the authors identify crops containing pseudo-labels suitable for fine-tuning. During training, the confident regions with pseudo-labels are mixed with images and labels from the source domain. The loss of the pseudo-labels is weighted by the ratio of pseudo-labels in the crop that have confidence higher than a predefined threshold. Image crops with low-confidence pseudo-labels are penalized with lower losses. The authors report that their work achieves state-of-the-art performance and approaches supervised performance. They also show that their dynamic weighting of the pseudo-label loss outperforms other approaches, such as applying constants for weighting the pseudo-label loss. Their confidence metric is sufficiently able to assess the quality of the pseudo-labels such that the loss of incorrect pseudo-labels get downweighted.

Although their work shows that domain adaptation using pseudo-labelling is possible, it is limited to a domain gap with objects having the same object size and viewing angle. The authors demonstrate a method that can filter low-quality pseudo-labels, but they do not show how to improve the pseudo-labels for a domain gap where the objects in the target domain have different sizes and different visual features. It remains uncertain if their method is still able to provide suitable pseudo-labels for fine-tuning in a more substantial domain gap as shown in this work. Therefore, we chose to use a method with a human in the loop and want to investigate how we can effectively and efficiently leverage the human effort for domain adaptation.

Chapter 3

Methods

The first step in reducing the human effort for object detection is obtaining as many correct labels as possible in the target domain using an object detector. The relative number of correct labels generated by a model reduces the need for human intervention in the labelling process, therefore obtaining as much correct labels as possible is important for reducing the human effort. The once-for-all property of DiffusionDet might enable us to improve the accuracy and find qualitative labels in the target domain.

Conventional object detection systems, as mentioned earlier, use object candidates to generate bounding boxes around objects. Object candidates are commonly generated from empirical object priors [4] [26], [15], [54] or learned queries [6], [55], [56], [57]. These candidates are generated deterministically from the model's input. Therefore, for an input x, the model's output y is fixed. Instead, the model used in this work has a stochastic output. Hence, variations among predictions are observable. DiffusionDet [14] uses random bounding boxes as candidates for object detection. These boxes are randomly assigned across an image with varying sizes. This concept is inspired by classical diffusion models [58] that use Gaussian noise to generate images.

3.1 Stochastic Object Detection using DiffusionDet

In DiffusionDet, removing noise from object boxes in images involves a step-by-step approach. This paradigm is analogous to conventional diffusion models that adhere to a noise-to-image pattern. For DiffusionDet, this translates to a noise-to-detected-objects pattern, where the model denoises random bounding boxes instead of pixel noise [14]. In Figure 3.1 an overview of these patterns is visualized. In the middle section of Figure 3.1, a noise-to-image pattern is shown as used in most conventional diffusion models. In the lower part of Figure 3.1 the noise-to-object pattern is

shown, where the noise is represented by bounding boxes of random locations, aspects ratios and sizes. This stochastic approach has an appealing advantage: the outcome of DiffusionDet can be changed at inference time by adjusting the noise and denoising process. The hypothesis is that this feature can be leveraged to enhance the model's performance without fine-tuning.

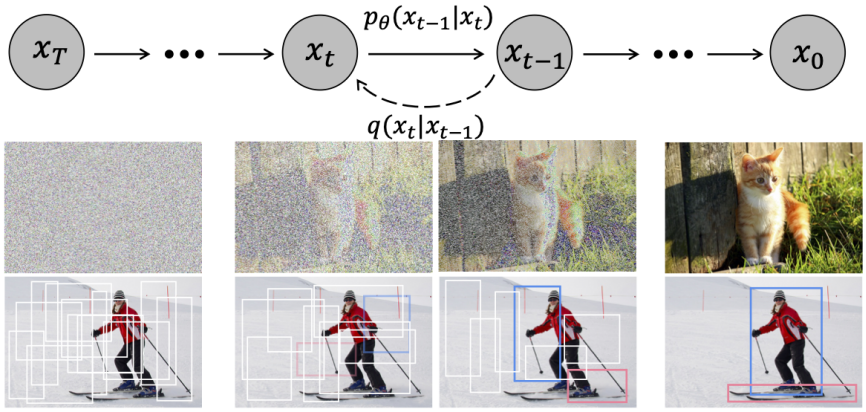


Figure 3.1: In the figure used from [14] the Diffusion process is visualized and compared against a conventional diffusion model for image generation. In the middle section, image noise is gradually removed, while at the bottom bounding boxes are stepwise adjusted and/or removed.

3.1.1 Changing the output at inference time

The authors report two major hyperparameters that affect the outcome at inference time [14]. First is the number of sampling steps for box denoising that adjusts the denoising process. As with conventional diffusion models, denoising takes place in multiple timesteps. Bounding boxes are stepwise adjusted in size and location during the diffusion process. The number of denoising steps can be set as a hyperparameter to adjust performance. The authors reported that more denoising steps improve performance, but increase inference time [14].

The second hyperparameter is the number of random bounding boxes to change the stochastic input of the model. An experiment by the authors revealed that using more random bounding boxes improves accuracy [14]. Models with fewer random bounding boxes seem to benefit more from adding more sampling steps compared to models with a relatively high number of random bounding boxes. This indicates that more random bounding boxes diminish the effect of more sampling steps and vice versa. The runtime analysis revealed that more sampling steps add relatively more latency compared to using more random bounding boxes. Therefore, we choose to experiment with adding more proposal boxes for domain adaptation.

An additional parameter is adjusting the input noise of the Diffusion model for the properties of the target domain. This could (partially) bridge the gap from the source to the target domain without fine-tuning. For DiffusionDet the noise is a set of bounding boxes with various sizes, aspect ratios and locations. In a domain gap where the objects in the target domain have a different size, the size of the random bounding boxes can be reduced.

3.2 More random proposal boxes

In the original work of DifussionDet [14], adding more bounding boxes seems to improve the in-domain performance. In this experiment, we want to validate whether the same effect applies out-domain. More specifically, we want to investigate whether this effect is also observable for a substantial domain gap. By adding more random proposal boxes for denoising, the hypothesis is that the probability of denoising a box to an object increases in the out-domain. Figure 3.2 visualizes the differences between using more random proposal boxes and the standard configuration. The aim is to detect the false negatives (FN) from the baseline model after using more random proposal boxes. The hypothesis is that the probability of detecting an object increases when using more boxes as shown in the example in the figure where the small person is detected with more random boxes as input. As a result, the expectation is that the recall should rise in proportion to the number of boxes being increased in the target domain.

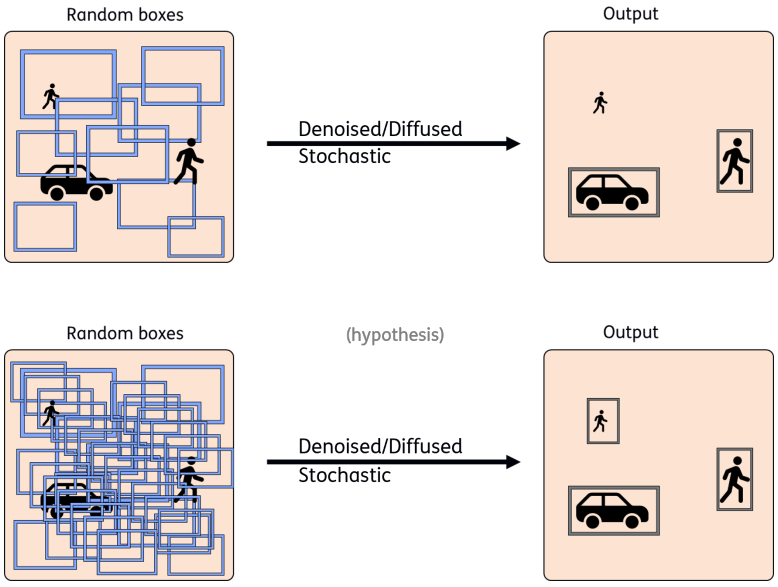


Figure 3.2: In this figure a comparison between the standard configuration and the experimental setting with more random proposal boxes is shown. In the lower configuration, more random proposal boxes are projected on top of the images to illustrate the effect of using more random box proposals.

3.3 Multiple runs

The goal of this section is to further explore the properties of the denoising process with respect to the number of bounding boxes. Instead of denoising all the boxes at once, this section proposes an iterative approach. By repeatedly predicting objects of the same image, the varying output across runs can be aggregated to supplement the final set of bounding boxes. By comparing this approach with more random bounding boxes, one can observe whether there is a dependence between boxes during the denoising process. If bounding boxes would compete during denoising, an iterative approach would be less affected by the competition. Since individual runs are independent of each other, boxes from one run do not affect the denoising process of another run. Therefore, if there exists a dependence between boxes in the diffusion process, a difference in performance should be observable between more runs vs. more boxes.

As with the previous method using more random proposal boxes, the expectation is that more runs can improve the recall since the variations among predictions are leveraged to improve the final prediction. In Figure 3.3 a schematic overview of this approach is visualized. After gathering all boxes, NMS is applied to remove duplicate detections. Algorithm 1 shows a high-level overview of the inference approach as

discussed in this section.

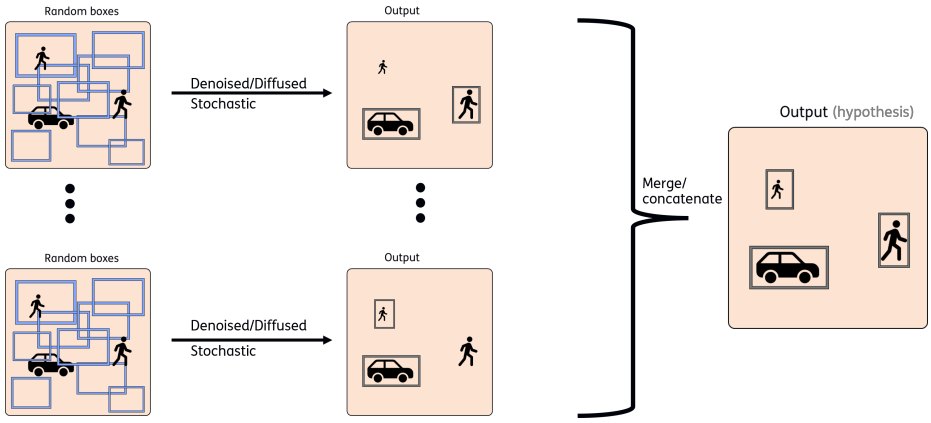


Figure 3.3: A visualization of combining multiple runs using DiffusionDet is shown in this figure. By leveraging the stochastic output of the model, the aim is to collect more objects by aggregating the boxes over multiple runs.

Algorithm 1 Multiple runs with DiffusionDet

procedure MULTIPLE RUNS DIFFUSIONDET(num_runs) ▷ Aggregation of more detections
    $img \leftarrow load\_img(path)$
    $predictions \leftarrow list$
    for i in range(num_runs) **do**
        $predictions.append($`DiffusionDet(img)`$)$
    end for
    $predictions\_final \leftarrow nms\_boxes(predictions, 0.5)$ ▷ Filter duplicate boxes
    return $predictions\_final$ ▷ Final set of predicted bounding boxes
end procedure

3.4 Smaller random proposal boxes

It is possible that there is a difference in the average box size between the source and the target domains. Given this knowledge of the target domain, we might exploit this knowledge to optimise the model for the properties of the objects in the target domain. Given that the model is trained with random proposal boxes on the source domain with an average area size of μ. Variable μ can be adjusted to align with the domain gap. For target domains where the average object sizes are relatively smaller, μ can be set smaller accordingly. In Figure 3.4 a visualization of this example is shown. The opposite adjustment can be made for target domains with larger

box sizes. By optimizing the stochastic input with the output of the target domain, the expectation is that this method can increase the probability of detecting an object in the target domain.

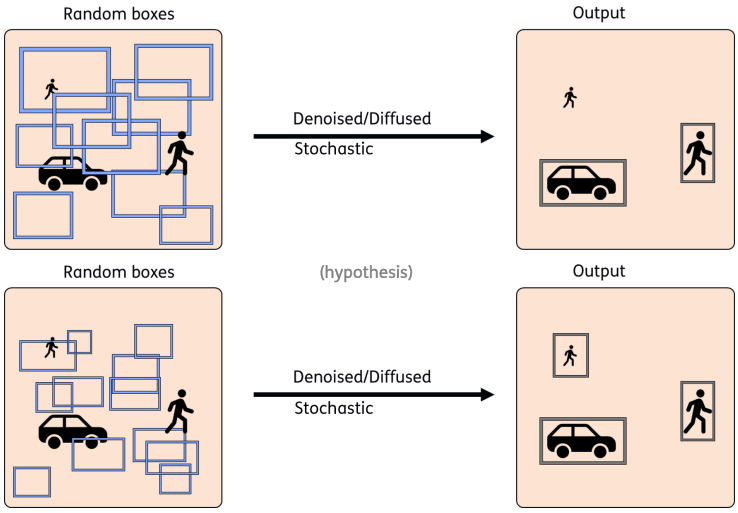


Figure 3.4: The upper configuration depicts the standard configuration whereas the lower configuration shows the experimental setting with smaller random proposal boxes.

3.5 Semi-supervised training using pseudo-labels

By leveraging DiffusionDet's ability to change the output at inference time, the suggested changes described in the previous sections can lead to improved detections. These detections can be utilized for pseudo-labelling to enhance the model's performance in the target domain using semi-supervised training. It is important to have improved detections because the pseudo-labels need to be of good quality [59]. Common mistakes such as FNs, where objects are not detected, and false positives (FP), where objects are mistakenly detected or misclassified objects in object detection should be minimised to prevent the model from learning from incorrect labels. Iterating from the improved recall, a human in the loop is used as an additional measure to guarantee the pseudo-label quality. Using a human in the loop for pseudo-label verification, we want to filter out low-quality pseudo-labels and only use better pseudo-labels for fine-tuning.

3.5.1 Obtaining pseudo-labels

Due to the domain gap, the expectation is that the pseudo-labels generated by DiffusionDet will not be good enough. Too many FNs or FPs in the pseudo-labels

can decrease the model's performance after fine-tuning. Therefore, the generated pseudo-labels should be verified whether the detections are correct. This task will involve a human in the loop that will be only concerned with verifying pseudo-labels. In practice, a human labeller should be able to identify regions containing correct pseudo-labels and locate them. After all labels in the fine-tuned dataset have been verified, the model is fine-tuned on the crops of the areas containing the correct pseudo-labels. More precisely, the proposed pseudo-label generation from this work involves three stages:

1. Generate pseudo-labels with DiffusionDet using the proposed enhancement from Sections 3.2, 3.3 and/or 3.4.
2. Annotate the regions with mostly correct labels by a human.
3. Crop and adjust the image and box coordinates to keep only the correct labels.

We assume that after human verification, there are still a sufficient amount of pseudo-labels left for fine-tuning. In Figure 3.5 an overview of the pseudo-label generation with a human-in-the-loop process is shown. To keep pseudo-labels consistent, the following guidelines were set up for the human-in-the-loop task:

- Identify and annotate all regions in the image containing correctly labelled objects.
- Include as much surrounding region of the object as possible to provide the model with as much image information as possible.
- Limit the task to verification, it is not allowed to add new labels.

Since the focus is on reducing the human labelling effort, the annotated areas are not checked by a second human annotator. Therefore it could be possible that the crop still includes incorrect labels. Hence, in the next section, we propose a method to combat potential issues with fine-tuning using incorrect labels.

3.5.2 Fine-tuning with crops containing pseudo-labels

Assuming that there are existing GT labels of the target domain, pseudo-labels are used to further improve the performance during fine-tuning. The expectation is that the human-verified pseudo-label process requires significantly less human effort. Therefore, the dataset size of pseudo-labels can be much larger compared to the size of the GT dataset.

The pseudo-labels and GT labels are sampled independently from each other and put together in one batch. Using a batch size of 2, each batch contains one

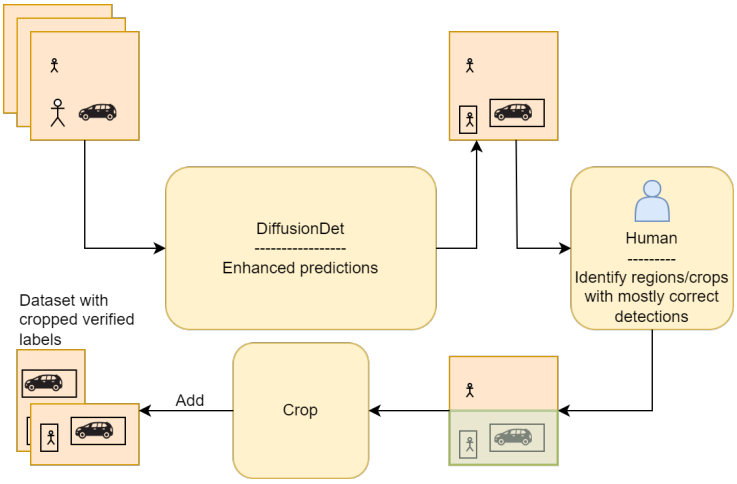


Figure 3.5: The process of generating pseudo-labels involves three stages. First, DifussionDet creates predictions over a set of images in the target domain. After labelling the images, a human identifies regions containing the correct labels. Lastly, the marked regions with the corresponding labels are cropped out of the images.

pseudo-labelled sample and one GT sample. Figure 3.6 provides an overview of the sampling procedure. When there are more pseudo-labels than GT labels, GT labels are again concurrently trained with different pseudo-labels in a new epoch. The proposed semi-supervised training procedure is shown in Algorithm 2. With the use of these experimental methods, the expectation is that the human labelling effort can be significantly reduced for fine-tuning DiffusionDet in the target domain.

Algorithm 2 The following pseudo-code depicts the training process utilized in this work, which involves the use of pseudo-labels.

procedure PSEUDO TRAIN($epochs$) ▷ train with pseudo-labels
    $data\_gt \leftarrow dataloader\_gt()$ ▷ Dataloader of GT-labels
    $data\_pseudo \leftarrow dataloader\_pseudo()$ ▷ Dataloader of Pseudo-labels
    $model \leftarrow DiffusionDet()$
    for i in range($epochs$) **do**
        $batch \leftarrow [data\_gt.get\_sample(), data\_pseudo.get\_sample()]$
        $loss \leftarrow model(\texttt{batch})$
        $loss.backward()$ ▷ Backpropagate loss
    end for
    return $model$ ▷ Trained model
end procedure

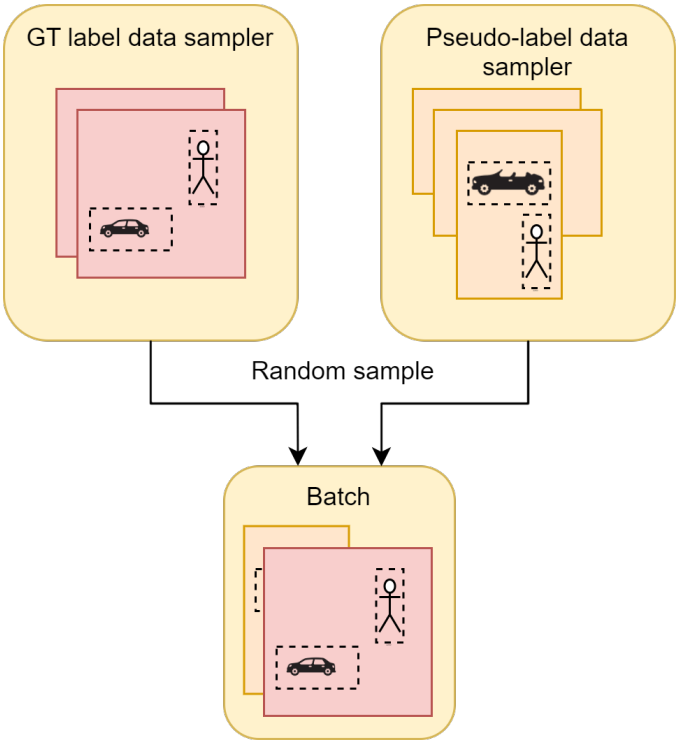


Figure 3.6: Images from the pseudo-label dataset and GT label dataset are drawn independently, using two dataloaders and put together in one batch for training.

Chapter 4

Experiments

This section evaluates the proposed experimental methods introduced in Section 3. This includes the used data, evaluation metrics and results.

4.1 Domain adaptation task

As mentioned in the problem statement in Section 1.2, the aim is to bridge substantial domain gap between the source and target domain. Therefore, the categories of interest between the two domains require sufficient visual differences to substantiate the gap. This section will describe and to some extent, quantify the domain gap.

4.1.1 Source domain: MS-COCO

Starting with the source domain, the MS-COCO (Microsoft Common Objects in Context) dataset [7] is used to pre-train the models. DiffusionDet and YOLO (baseline) are trained on the MS-COCO training set. The MS-COCO data set is a large-scale data set for object detection, keypoint estimation, and image segmentation. Throughout the years, the authors have made adaptations to the dataset, such as additional annotated images and adaptations in the test/train split sizes. In this work, the latest version from 2017 is used. The dataset consists of 118k static images for training and 5k images for validation. The data set has 80 object categories for object detection. As the name of the dataset suggests, the object categories include all types of common objects such as pets, food and care products.

4.1.2 Target domain: VisDrone

Continuing with the target domain, The VisDrone dataset [60] contains objects from an unmanned aerial vehicle's (UAV) perspective in urban areas. Images are taken from an aerial perspective. This results in some additional challenges with respect to

object detection. First, images were captured from a greater distance, which could lead to smaller objects, although the object size also depends on the camera lens that has been used. Also, some images contain motion blur, which is attributed to the drone's moving point of view. The dataset consists of 10k images containing on average 54k labels per category. There are 10 categories of objects, which are various types of road users such as cars, vans, or pedestrians. From the 10 categories, only a subset of the GT labels is used during evaluation; car, person, pedestrian, bus and truck. The dataset is divided in a train, test and validation set, containing 6471, 1610 and 548 images respectively.

VisDrone	**MS-COCO**	**This work**
Car	Car	Car
Person	Person	Person
Pedestrian		
Bus	Bus	Bus
Truck	Truck	Truck

Table 4.1: An overview of the used labels is shown in this table. Note that the categories "Person" and "Pedestrian" are merged in this work. Since the model is pre-trained on MS-COCO, there is no distinction made between pedestrians and persons. For convenience, both categories have been merged.

These categories were used since they overlap with the labels from MS-COCO and can be used for the domain adaptation task. Also, *person* and *pedestrian* are merged since the labels in MS-COCO do not make a distinction between the two categories from VisDrone. Therefore, the models in this work will classify both *person* and *pedestrian* as *person* in VisDrone. In Table 4.1 an overview of the categories is shown.

4.1.3 Characterizing the domain gap

Although both datasets have a small intersection in their types of categories, there is still a domain gap in terms of visual features. As mentioned before, the main difference between the source and target dataset in this work is caused by the different viewpoints. This results in smaller objects and different object features in VisDrone compared to MS-COCO. Figure 4.1 shows the differences between the two domains. The figure illustrates the differences in the viewing angle and distance. These have a considerable effect on the visual appearance of the objects. Consequently, a different viewing angle requires the object model to detect different visual features. For

example, in a frontal view of a person, the legs, torso and head are clearly visually distinguishable. However, from an aerial view, these elements can be represented by only a few pixels and the person's head and torso are possibly the main contributing factors for obtaining a detection. In a frontal view of cars, the bumper, bonnet and front window are the main visual features. In an aerial view, the roof and bonnet of the car are the main features. Unfortunately, without any metadata, it is hard to quantify the differences in viewing angles and distance to objects.

(a) MS-COCO (b) VisDrone

Figure 4.1: In column a), pictures from MS-COCO are shown. Both images contain frontal captures of humans and cars. In b), two images from VisDrone are shown. The UAV's aerial view results in smaller objects on average. The aerial view also enables to capture of more objects of interest.

Quantification of the domain gap

Although the visual features cannot be quantified, the object sizes can be measured to illustrate the domain gap. Visual inspection in Section 4.1.3 shows that the difference in object size is likely a result of the extended viewing distance from the capturing device towards the object. Although this does not fully comprehend the gap in terms of different object features between the domains, this measure relates to the difference in view distance and viewpoint and consequently the viewpoint affects the difference between object features. Hence, an analysis of the object sizes in MS-COCO and VisDrone is performed to quantify the domain gap. The aim of the analysis is to reveal two insights. First, reveal the size of the domain gap in terms of object size. Second, how DiffusionDet in the standard configuration performs under the domain gap. A pre-trained DiffusionDet model trained on MS-COCO with

300 random box proposals from [14] was used and only the labels *car*,*person*, *bus*, *truck* as mentioned in Section 4.1.2 were used for the analysis. These insights were respectively obtained using a frequency analysis of GT labels and predicted labels in both source and target domains. Figure 4.2 shows two histograms that depict the frequency of object sizes of the GT labels and predictions for MS-COCO and VisDrone. For convenience, the x-axis of the histogram of MS-COCO is cut off at 5000 pixels to make it comparable to the frequencies of VisDrone. First, comparing the frequencies of the GT labels shown in blue, VisDrone has significantly more smaller-sized objects. This is attributed to the increased viewing distance of the UAV. Second, the in-domain graph shows a correspondence between the frequency of GT labels and predictions. However, the out-domain predictions do not align with the GT object sizes. For the GT labels, the frequencies for small objects are much higher compared to the predicted labels' frequencies of small objects. It appears that DiffusionDet has difficulty detecting small objects within the target domain.

To further quantify the detection performance w.r.t. the object size, three categories of object size were introduced: *small*, *medium* and *large*. The ranges of the object sizes are created by dividing all objects from the GT labels of VisDrone into three equally sized groups. The resulting ranges of the categories are $[0, 104)$, $[104, 528)$ and $[528, \infty)$ respectively. The same ranges of box sizes were applied to the labels of MS-COCO for equal comparison. As a result of the domain gap, MS-COCO will likely have fewer objects of the category *small*, but the object category *large* will likely contain more objects. These categories provide additional insight during evaluation into the domain gap in terms of the object size in the target domain. It enables to visualisation of the effect of possible enhancements to the model with respect to the domain gap.

The expectation is that the analysis of the object sizes will reveal the domain gap. The model will likely perform worse on smaller objects compared to medium-sized and large objects. The reason is twofold: generally detecting smaller objects is considered more challenging since there is less pixel information and the model is not tuned to detect small objects since it was trained on the source domain, MS-COCO. As shown in Figure 4.2, the target domain contains significantly smaller objects. During the evaluation of the experimental methods, these categories will be used to provide additional insight into the object detection performance.

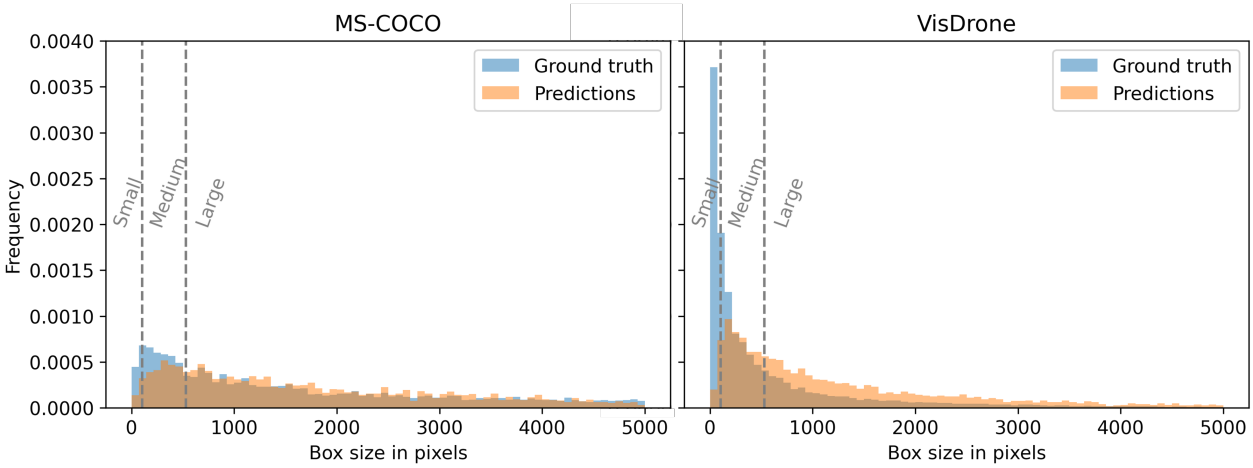


Figure 4.2: The histograms show the discrepancy between the source and target domain in terms of object size. Generally, the target domain contains smaller objects. Also, the model struggles with smaller objects since the number of smaller objects does not align with the number of ground truth small objects in the target domain.

Impact of the domain gap

After determining three categories of box sizes (*small*, *medium* and *large*), the baseline model is used to determine the impact of the domain gap. The baseline model is evaluated on the validation set of MS-COCO and the test set of VisDrone. The model is trained on MS-COCO and not fine-tuned on the target domain. The model's performance is evaluated with the mean average precision (mAP) and mean average recall (mAR) [61]. The mAP is the ratio of correctly detected objects to the total number of detected objects. Objects are considered detected when the intersection over union (IoU) of a bounding box is equal or larger than 0.5 with the ground truth. The mAR is the ratio of correctly detected objects to the total correct number of objects in the dataset.

This evaluation provides insight into the domain gap in practice. In Figure 4.3 a comparison between in-domain and out-domain performance is shown. When comparing the overall in-domain performance with the out-domain performance, the figure reveals a significant decrease in performance for all three categories. The performance per object category reveals that for both domains, the object size seems to correlate with the mAR and mAP values. Smaller boxes result in lower mAP and mAR values compared to medium and large boxes. In addition, the steeper line between object sizes in VisDrone compared to MS-COCO seems to indicate that smaller objects are more challenging in VisDrone. One possible explanation is that the combination of fewer pixel information and a different viewing angle leads to a

relatively higher drop in performance.

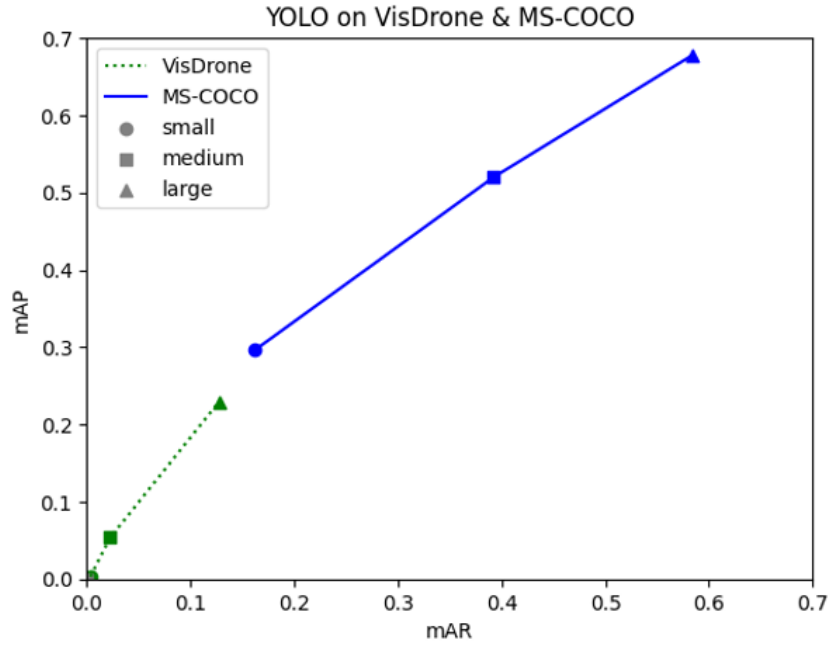


Figure 4.3: The solid blue line shows the in-domain performance on MS-COCO per box size. The dotted green line shows the box-size performance on VisDrone. In general, the model performs worse on smaller boxes. On average the performance on VisDrone is significantly worse on VisDrone compared to MS-COCO.

4.1.4 Baseline comparison with DiffusionDet

To verify whether DiffusionDet performs on par with the baseline model, YOLO on VisDrone, an additional comparison was made. In Figure 4.4 the performance of both models is shown. DiffusionDet's backbone for feature extraction is a ResNet-101 [17], this backbone is used across all experiments. To improve visibility, the graph is zoomed in on the right side of the figure. One can observe that DiffusionDet performs slightly worse for large objects. In this experiment, DiffusionDet's inference hyperparameters were set to the default setting for this work. The number of sampling steps was set to 10 and 300 random proposal boxes for denoising were used. Overall, the graph shows that DiffusionDet performs on par with YOLO using the default settings.

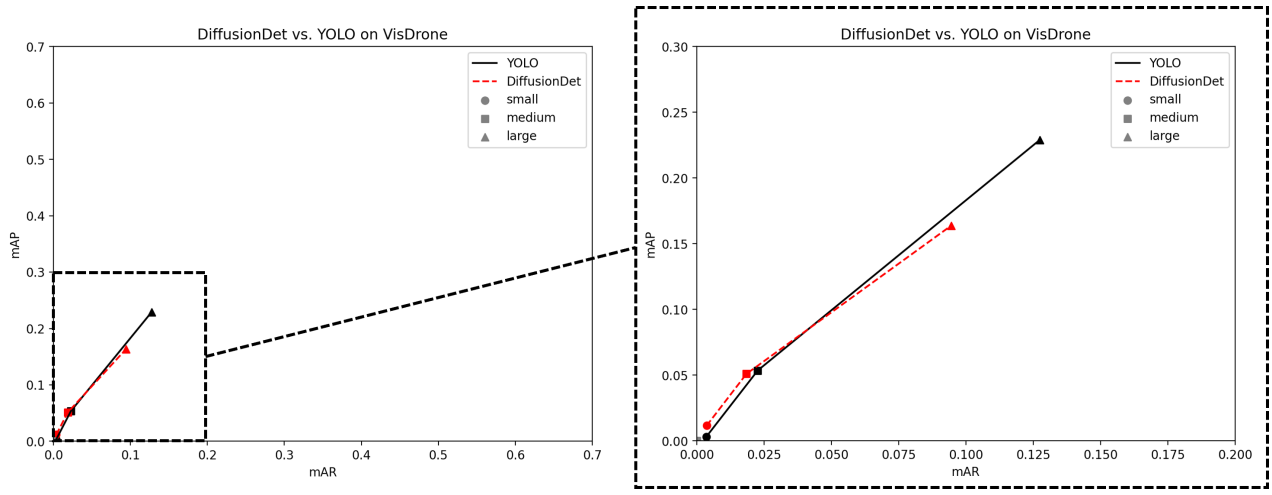


Figure 4.4: The evaluation shown above is an out-domain evaluation. The solid black line depicts the YOLO performance across the three object size categories. The red dashed line shows DiffusionDet's performance. Both models perform on par, except for the large object size category. In this category, YOLO performs slightly better.

4.2 Improving detections without finetuning

After establishing that DiffusionDet's default configuration performs on par with the baseline model YOLO on the target domain data, this section investigates whether DiffusionDet's performance can be further enhanced without any fine-tuning. This improves the accuracy of the detections compared to generated labels with DiffusionDet's standard configuration. Using more bounding boxes will probably affect the performance across the experimental methods. More boxes can be utilized by performing multiple inference runs, or use more random bounding box proposals at once. In order to ensure a fair comparison between each experimental method, the same number of random boxes were used in each configuration. Also, for the evaluation of the experiments in this section, the VisDrone validation set is used. For predictions with more runs, the total number of used random bounding boxes is calculated by multiplying the number of runs times the number of random bounding boxes per run. Also, the number of sampling steps to denoise the bounding boxes was set to 10 in all experiments. In Table 4.2 an overview is shown of all three different hyperparameters that have been used.

More boxes		**More runs**		**Smaller boxes**		
Boxes	Runs	Boxes	Runs	Boxes	Runs	f_{box_size}
300	1	300	1	300	1	1 (default)
2700	1	300	9	300	1	0.75
5400	1	300	18	300	1	0.5

Table 4.2: An overview of the model configurations for the experiments during inference time. As mentioned before, experiments with more runs and more random proposal boxes use an equal number of boxes in total. f_{box_size} is only mentioned in the column of Smaller boxes, since in all other experiments the default random box size of 1 is used.

In addition to using more boxes, an experiment with smaller random box proposals is introduced as well. Since the target domain contains smaller objects as shown in Figure 4.2, smaller random box proposals may ease the diffusion process and increase the likelihood of detecting an object. For each experiment shown in Table 4.2, 10 sample steps were used for the denoising process.

4.2.1 More random proposal boxes

Using the hyperparameters from Table 4.2, the model is evaluated for different numbers of random proposal boxes. As can be seen in Figure 4.5, using more boxes increases the mAR and mAP values across all categories of box size. Adding more boxes improves mAP and mAR mainly for *medium* and *large* boxes. Especially, the mAR improves for *large* boxes when using more random box proposals. Overall, the greatest gain can be observed when moving from 300 to 2700 boxes. For the category *large* the mAP increases from 0.18 to 0.34. Adding more random proposal boxes after 2700 random proposal boxes still improves the mAR and mAP values but only poses a small improvement in mAP from 0.34 to 0.36.

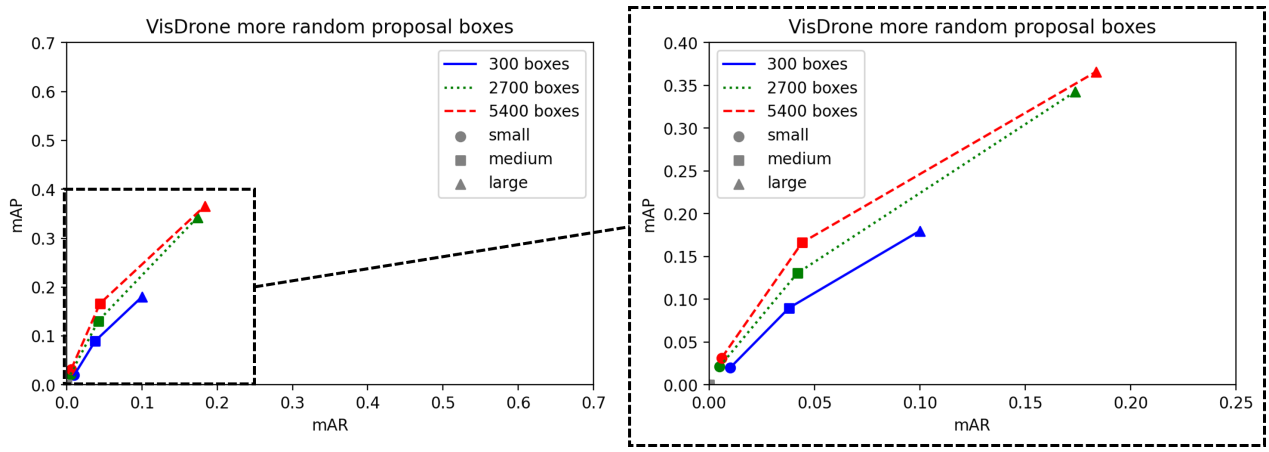


Figure 4.5: The results of using 300, 2700 and 5400 boxes are shown in blue, green and red respectively. More random box proposals improve the performance across all box categories. However, the largest improvements can be observed for the box category *large*.

4.2.2 More runs

To evaluate the effect of more runs of DiffusionDet on the performance of VisDrone, the mAR and mAP were calculated for each box size category as mentioned in Section 4.1.3. This identifies whether there is a dependence between boxes during the diffusion process. As can be shown in Figure 4.6 more runs have an equal effect on the performance of VisDrone as more random proposal boxes. Again, the largest gain is made from 1 to 9 runs in both mAR and mAP values across all box categories, but the most significant gain is visible for the *large* category.

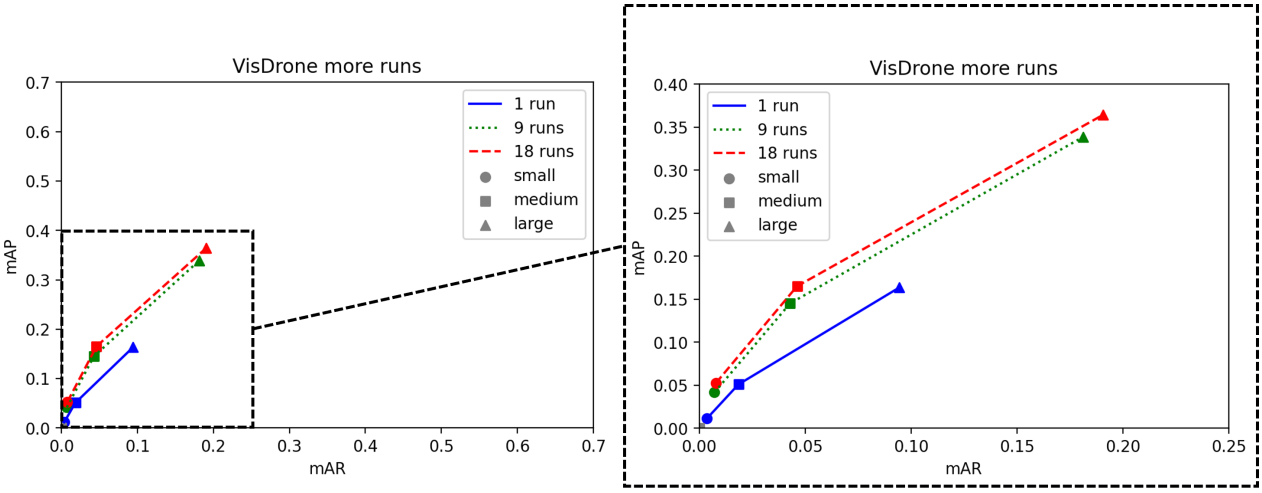


Figure 4.6: This figure shows the mAP and mAR performance per box size category across multiple run configurations. A similar colour coding is used as in the previous experiment: the blue, green and red line show the performance for 1, 9 and 18 runs respectively.

Although there are some minor performance differences observable between using more runs and more random proposal boxes, these can be likely attributed to the stochastic output of DiffusionDet. These results seem to show that there is no dependence between boxes in the Diffusion process and more runs or more random proposal boxes can be used to scale the performance of DiffusionDet. Therefore, more runs or more random box proposals can be used to scale the performance of DiffusionDet. Depending on the hardware configuration, the user can choose faster inference in trade for more memory using more random proposal boxes instead of more runs. In Table 4.3 an overview of GPU memory cost and runtime performance is shown. The GPU memory measurements were performed using nvidia-smi tool. Overall, the runtime speed is higher for using more boxes compared to more using equally more boxes. However, using more boxes requires more GPU memory. When there are memory constraints, using more runs can equally improve performance with less memory but greater inference time.

In Figure 4.7, the improvements of aggregating detections over multiple runs are visualised. Initially, only one car and one person in the zoomed image fragment are detected. This number gradually increases when detections from other runs are added. Even the occluded car at the top left is detected after the 18th run.

Boxes	Runs	GPU Memory	Inference time
300	1	3.429GB	0.481s/image
300	9	3.109GB	2.495s/image
300	18	3.109GB	4.667s/image
2700	1	3.617GB	1.775s/image
5400	1	5.541GB	3.633s/image

Table 4.3: This table conveys the runtime performance and GPU memory costs of DiffusionDet across different configurations.

Figure 4.7: In this figure the improved detections using more runs are visualized in an image of VisDrone.

4.2.3 Smaller random proposal boxes

The final experiment of this section evaluates the performance when using smaller random proposal boxes. The results per box category are shown in Figure 4.8. The figure reveals that smaller random box proposals result in a decrease in performance across all box size categories. The hypothesis of the experiments was that smaller random proposal boxes would ease the diffusion process since the random proposal boxes would better align with the objects in the target domain. In this case, the target domain contained smaller objects and smaller random proposal boxes would likely reduce the required denoising to the boxes.

However, smaller random box proposals seem to disturb the diffusion process and result in a decrease in performance. For the box category *small*, the mAP dropped from 0.02 to 0.0041 when changing $f_{box\_size} = 1$ to $f_{box\_size} = 0.75$. The mAP for the box category also dropped for $f_{box\_size} = 0.5$ from 0.02 to 0.0085. Also, the mAR and mAP levels dropped when using smaller box proposals in the box size category *medium* compared to the standard random box proposal size. One

explanation for this observation is that the diffusion process is fitted for the default box size. Adjusting the average random box proposal size can affect the regression task of the detection decoder which was fitted during training on the default random box size. As a result, the smaller random box proposals change the outcome of the regression and the boxes are not properly aligned with the object anymore. Hence, the random box proposal size will not be adjusted when generating pseudo-labels.

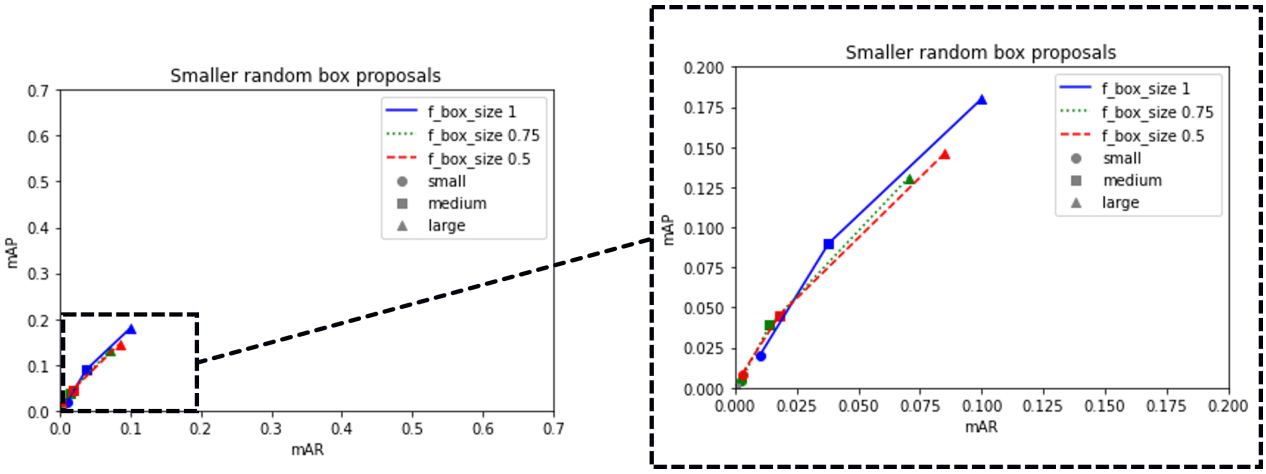


Figure 4.8: The blue line depicts the standard configuration where the random box size is multiplied by 1. The green and red line shows random proposal boxes that are multiplied by 0.75 and 0.5. Smaller random proposal boxes do not seem to provide any improvement across the box categories.

4.3 Reducing the human labelling effort

In the previous sections, several methods were analyzed to improve the performance without fine-tuning. More random proposal boxes or more runs can be used to improve recall and precision. To verify whether the improved detections can be leveraged for pseudo-labelling, two baseline models are introduced. First is a lower-bound model that only uses GT labels for fine-tuning in the target domain. The second baseline model is an upper-bound model that uses the same image crops from the human in the loop as mentioned in Section 3.5.1 that selects the correct pseudo-labels but uses the GT labels within the crop. Theoretically, this model should perform best but requires much more human labelling effort. In addition, to quantify the benefit of the improved recall and precision for pseudo-labelling, two experimental models are used. One model uses image crops containing pseudo-labels generated from 1 run and another model uses the same crops, containing

labels generated from 18 runs. The crops were made with labels generated from 18 runs. In Table 4.4 an overview of the performance and the respective required human labelling effort is shown.

For the pseudo-label generation, a random subset of 334 images of the validation training set was selected. The selected images with the corresponding pseudo-labels were uploaded to CVAT [62]. A human in the loop verified the labels and selected the areas that contained correctly labelled objects by DiffusionDet with 18 runs (better quality). Only predictions with a confidence level higher than 0.5 were considered for pseudo-labelling. The image parts that contained correctly labelled objects were cropped. This resulted in 706 crops that contained valuable labels. During pseudo-label verification, the annotation time was measured. On average, it took a human annotator 12.92 seconds per image to identify the areas containing the correct labels. In the literature, it was shown that the average annotation time of a bounding box is 50.8 seconds [63]. Using this measurement an estimation of the total human effort per training configuration was made, shown in Table 4.4. The training sets of 10 and 50 GT samples contained 488 and 2195 objects respectively.

When analyzing the values from Table 4.4, the labelling times reveal that using pseudo-labels adds relatively little extra annotation time compared to using GT labels. This is visible when comparing the labelling times between crops with pseudo-labels and crops with GT-labels. In both scenarios, the same image crops were used but either contained pseudo-labels or GT labels. The human-verified pseudo-labels do require significantly less human effort compared to the GT labels. Interestingly, the upper bound model with 10 GT samples performs has a lower mAP compared to the pseudo-labels from 18 runs. This is unexpected since the upperbound model uses GT labels in the image crops, therefore the labels in the crops should not contain any errors or missing labels. For 50 GT-samples the upperbound model outperforms the other semi-supervised settings as expected.

Semi-supervised overview	mAP_{50}	$Labelling\ time$	mAP_{50}	$Labelling\ time$
$\#samples$	10		50	
GT-labels	0.1661	6.89 h	0.1818	30.97 h
GT-labels + crops 1 run	0.1725	8.05 h	0.1925	32.17 h
GT-labels + crops 18 runs	**0.1792**	8.05 h	0.2017	32.17 h
GT-labels + crops GT-labels	0.1766	62.21 h	**0.2035**	86.30 h

Table 4.4: An overview of the performance and the human effort per fine-tune setting. The highest mAP values are highlighted in bold font. Interestingly, the upper bound model did not outperform our experimental method with GT labels and crops 18 runs.

In addition to Table 4.4, Figure 4.9 provides a visual comparison between the performances across different semi-supervised settings. Compared to the baseline model that has been trained only on GT labels, training the model with additional pseudo-labels improves the performance. As expected, generally training with crops containing pseudo-labels from DiffusionDet with 1 run has a lower performance than using pseudo-labels generated with 18 runs. This shows that the improved recall and precision benefit the pseudo-label generation process. For 50 samples, there is a gradual increase in performance when using more qualitative labels for fine-tuning. The upper bound model is almost equal in performance to the method proposed in this work: pseudo-labels from 18 runs. This graph shows that fine-tuning using pseudo-labels improves performance compared to using only GT labels. Using pseudo-labels with 18 runs and 10 GT samples almost equals fine-tuning on only 50 GT samples. This shows that pseudo-labelling can reduce the required number of GT labels for fine-tuning.

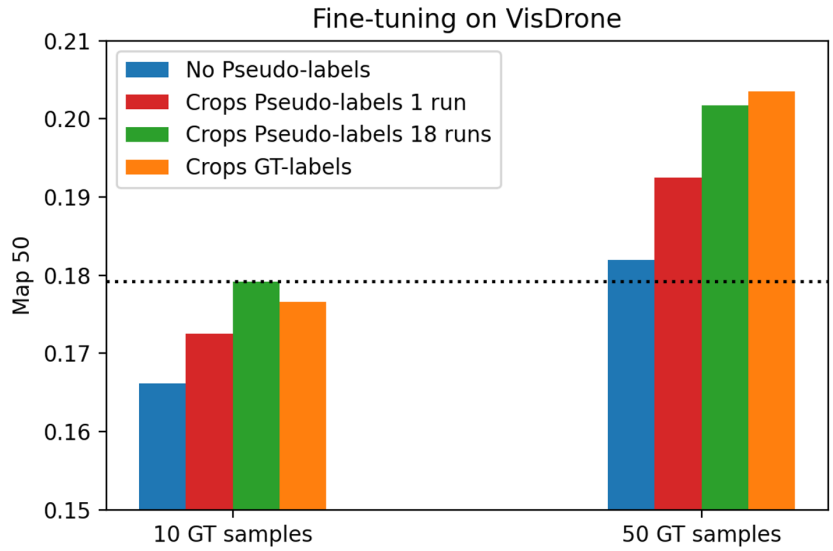


Figure 4.9: Using pseudo-labels improves the performance compared to models trained on GT labels only. Using pseudo-labels generated with 18 runs almost equals the performance of using 50 GT data samples as shown by the horizontal dotted line.

When adding human-verified pseudo-labels to 10 GT samples in the training set, the mAP increases from 0.1661 to 0.1792. When replacing the pseudo-labels in the image crops with GT samples, the performance slightly decreases for 10 GT samples. Adding more GT labels to the training set improves the performance. A model trained with 50 GT samples outperforms a model trained with 10 GT samples by 0.0158 in mAP. Using pseudo-labels generated from 18 runs with 50 GT samples improves the mAP with 0.020 with respect to the model trained on 50 GT

samples. Finally, as expected the model trained on 50 GT samples and crops containing pseudo-labels outperforms the model trained on 50 GT samples and crops containing pseudo-labels generated from 18 runs. With only a slight margin of 0.002 in mAP, the upperbound model outperforms our suggested approach with pseudo-labels.

After establishing performance improvements in the target domain using several fine-tuning strategies, the human effort for the relative improvement is analyzed. In a conventional fine-tuning setting only GT labels are used for training. Therefore, the relative improvement is measured with respect to the model trained on 10 GT samples. This model is considered a baseline to measure the relative improvement. From 10 GT samples onwards, the improvement and respective human effort are used to indicate how much human effort is required to improve the mAP from a model trained on 10 GT samples.

Without considering any pseudo-labels, improving the mAP using 40 additional GT samples requires 24.08 hours of labelling time and improves the performance with 0.016 in mAP. Therefore, increasing the performance $0.6520\mathrm{e}{-3}$ in mAP per hour of human effort. Improving the baseline by adding pseudo-labels generated with only 1 run using an additional 1.16 hours of human verification time effort improves the model's performance with 0.0064 in mAP, resulting in a relative improvement of $5.5172\mathrm{e}{-3}$ in mAP per hour of human effort. This value increases when using pseudo-labels generated with 18 runs to $11.2931\mathrm{e}{-3}$. Using human-verified pseudo-labels improves the efficiency and effectiveness of the human labelling effort in terms of performance using fine-tuning.

When repeating the same analysis for a model trained on 50 GT samples as a baseline, a similar pattern in the relative human effort efficiency is visible. The performance can be improved using pseudo-labels by adding 1.16 hours of human effort for pseudo-label verification. For pseudo-labels generated from 1 run, this results in a relative improvement of $9.2241\mathrm{e}{-3}$ in mAP per hour of human effort. The efficiency increases when pseudo-labels from 18 runs are used, and the resulting relative effort efficiency becomes $17.1552\mathrm{e}{-3}$. The upperbound model had unsurprisingly the best performance but comes at a high human effort cost. Therefore the effectiveness and efficiency of the human effort are relatively low with only $0.3922\mathrm{e}{-3}$ in mAP improvement per hour of human effort.

This analysis shows that fine-tuning with human-verified pseudo-labels as proposed in this work is the most effective and efficient method to reduce the human labelling effort to improve the model's performance. Using more GT samples to improve performance is the least effective method in terms of human effort. The effectiveness of the human verification process increases when using improved detections using more runs. For both 10 and 50 GT samples, adding pseudo-labels

is the relatively most effective and efficient training strategy in terms of human effort. The effectiveness of the pseudo-labelling method increases when the model is trained with more GT samples concurrently.

Chapter 5

Conclusions

This research aimed to reduce the human labelling effort for training an object detection model. The first step towards reducing the human effort was to improve the generated labels in the target domain. Quantitative analysis showed that using DiffusionDet the recall and precision in the target domain could be improved without fine-tuning. Using more random proposal boxes or more runs the recall and precision can be improved in the target domain. Using more runs to improve performance can be beneficial when one has to work with memory constraints and run the model with fewer boxes multiple times. By utilizing the stochastic input of DiffusionDet, it is possible to enhance the performance out domain. Surprisingly, smaller random proposal boxes did not improve the performance. Probably because the regression process is affected by the smaller box sizes. DiffusionDet can outperform the baseline model in the target domain by adjusting the hyperparameters for inference.

After improving recall and precision, this research aimed to reduce the human annotation effort using human-verified pseudo-labels. Using human-verified pseudo-labels the performance across different numbers of GT labels could be improved compared to the baseline that was trained with only GT samples. An analysis of the mAP improvement per hour of human effort showed that the proposed pseudo-labelling strategy is the most effective and efficient strategy for improving the model's performance in terms of human effort. The effectiveness of the method increased when the pseudo-labels were trained with 50 GT samples. This work shows that the human labelling effort can be reduced with labels generated from DiffusionDet for a substantial domain gap.

5.1 Recommendations for future work

This work does not address any possible class imbalance issues of the target domain. This could lead to model bias towards frequently occurring objects. Therefore

the pseudo-label quality will be reduced and fine-tuning will be less effective. Future work could improve the performance when the class imbalance issues in the pseudo-labels are addressed.

Another possible improvement might be to weigh the loss of individual bounding boxes based on the certainty of the box. This feature would reduce the loss of incorrect boxes and might enable lowering the threshold of the pseudo-labels. As a result, lowering the threshold of the pseudo-labels can improve the recall in the pseudo-labels, which is beneficial for training. This would require a metric that can approximate the possibility of the box being a correct prediction. The confidence level and/or another proxy for approximating box correctness could be used to weigh the loss of individual bounding boxes.

Finally, in the current approach, there is a teacher model that generates the pseudo-labels and a student model that learns from the pseudo-labels. This work implementation provides a one-directional training strategy; from teacher to student. A possible improvement would be the implementation of a strategy to train the student and teacher model gradually and in a beneficial manner. This could extend the ceiling of performance improvements using pseudo-labels since the trained weights can be used to further enhance the performance.

Bibliography

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

[2] S. S. A. Zaidi, M. S. Ansari, A. Aslam, N. Kanwal, M. Asghar, and B. Lee, "A Survey of Modern Deep Learning based Object Detection Models," *Digital Signal Processing: A Review Journal*, vol. 126, 4 2021. [Online]. Available: https://arxiv.org/abs/2104.11892v2

[3] G. Cheng and J. Han, "A survey on object detection in optical remote sensing images," pp. 11–28, 2016. [Online]. Available: http://dx.doi.org/10.1016/j.isprsjprs.2016.03.014

[4] J. Redmon, S. Divvala, R. Girshick, and A. Farhadi, "You Only Look Once: Unified, Real-Time Object Detection," in *2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2016. [Online]. Available: http://pjreddie.com/yolo/

[5] M. Tan, R. Pang, and Q. V. Le, "EfficientDet: Scalable and efficient object detection," *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 10 778–10 787, 2020.

[6] S. Ren, K. He, R. Girshick, and J. Sun, "Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks," in *Proceedings of the 28th International Conference on Neural Information Processing Systems - Volume 1*, ser. NIPS'15, vol. 39, no. 6. Cambridge, MA, USA: MIT Press, 6 2015, p. 91–99. [Online]. Available: https://arxiv.org/abs/1506.01497v3

[7] T. Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick, "Microsoft COCO: Common objects in context," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 8693 LNCS, no. PART 5, pp. 740–755, 2014.

[8] A. Kuznetsova, H. Rom, N. Alldrin, J. Uijlings, I. Krasin, J. Pont-Tuset, S. Kamali, S. Popov, M. Malloci, A. Kolesnikov, T. Duerig, and V. Ferrari, "The Open Images Dataset V4: Unified Image Classification, Object Detection, and Visual Relationship Detection at Scale," *International Journal of Computer Vision*, vol. 128, no. 7, pp. 1956–1981, 2020. [Online]. Available: https://storage.googleapis.com/openimages/web/2018-

[9] P. Oza, V. A. Sindagi, V. M. Patel, and S. Member, "Unsupervised Domain Adaptation of Object Detectors: A Survey," 2021.

[10] Y.-C. Liu, C.-Y. Ma, Z. He, C.-W. Kuo, K. Chen, P. Zhang, B. Wu, Z. Kira, and P. Vajda, "Unbiased Teacher for Semi-Supervised Object Detection," *ICLR 2021*, pp. 1–17, 2 2021. [Online]. Available: https://github.com/facebookresearch/unbiased-teacher.https://arxiv.org/abs/2102.09480v1

[11] Y. Wang, Z. Liu, and S. Lian, "Semi-supervised Object Detection : A Survey on Recent Research and Progress," vol. 11, pp. 1–10, 2023.

[12] H. Cai, C. Gan, T. Wang, Z. Zhang, and S. Han, "Once-for-All: Train One Network and Specialize it for Efficient Deployment," *8th International Conference on Learning Representations, ICLR 2020*, 8 2019. [Online]. Available: https://arxiv.org/abs/1908.09791v5

[13] M. Tan and Q. V. Le, "EfficientNet: Rethinking Model Scaling for Convolutional Neural Networks," *36th International Conference on Machine Learning, ICML 2019*, vol. 2019-June, pp. 10 691–10 700, 5 2019. [Online]. Available: https://arxiv.org/abs/1905.11946v5

[14] S. Chen, P. Sun, Y. Song, and P. Luo, "DiffusionDet Diffusion Model for Object Detection," *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV2023)*, 2022. [Online]. Available: https://github.com/ShoufaChen/

[15] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg, "SSD: Single Shot MultiBox Detector," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9905 LNCS, pp. 21–37, 12 2015. [Online]. Available: http://arxiv.org/abs/1512.02325http://dx.doi.org/10.1007/978-3-319-46448-0_2

[16] K. Simonyan and A. Zisserman, "Very Deep Convolutional Networks for Large-Scale Image Recognition," *3rd International Conference on Learning Representations, ICLR 2015 - Conference Track Proceedings*, 9 2014. [Online]. Available: https://arxiv.org/abs/1409.1556v6

[17] K. He, X. Zhang, S. Ren, and J. Sun, “Deep Residual Learning for Image Recognition,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, vol. 2016-December, pp. 770–778, 12 2015. [Online]. Available: https://arxiv.org/abs/1512.03385v1

[18] J. Redmon and A. Farhadi, “YOLO9000: Better, Faster, Stronger,” *Proceedings - 30th IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017*, vol. 2017-January, pp. 6517–6525, 12 2016. [Online]. Available: https://arxiv.org/abs/1612.08242v1

[19] J. Redmon and A. Farhari, “YOLOv3: An Incremental Improvement,” *arXiv preprint arXiv:1804.02767*, 4 2018. [Online]. Available: https://arxiv.org/abs/1804.02767v1http://arxiv.org/abs/1804.02767

[20] A. Bochkovskiy, C.-Y. Wang, and H.-Y. M. Liao, “YOLOv4: Optimal Speed and Accuracy of Object Detection,” 4 2020. [Online]. Available: https://arxiv.org/abs/2004.10934v1http://arxiv.org/abs/2004.10934

[21] J. A. Kim, J. Y. Sung, and S. H. Park, “Comparison of Faster-RCNN, YOLO, and SSD for Real-Time Vehicle Type Recognition,” *2020 IEEE International Conference on Consumer Electronics - Asia, ICCE-Asia 2020*, 11 2020.

[22] L. Tan, T. Huangfu, L. Wu, and W. Chen, “Comparison of RetinaNet, SSD, and YOLO v3 for real-time pill identification,” *BMC Medical Informatics and Decision Making*, vol. 21, no. 1, pp. 1–11, 12 2021. [Online]. Available: https://link.springer.com/articles/10.1186/s12911-021-01691-8https://link.springer.com/article/10.1186/s12911-021-01691-8

[23] A. Pansare, N. Sabu, H. Kushwaha, V. Srivastava, N. Thakur, K. Jamgaonkar, and M. Z. Faiz, “Drone Detection using YOLO and SSD A Comparative Study,” *2022 International Conference on Signal and Information Processing, IConSIP 2022*, 2022.

[24] R. Girshick, J. Donahue, T. Darrell, and J. Malik, “Rich feature hierarchies for accurate object detection and semantic segmentation,” *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, pp. 580–587, 11 2013. [Online]. Available: https://arxiv.org/abs/1311.2524v5

[25] J. R. Uijlings, K. E. Van De Sande, T. Gevers, and A. W. Smeulders, “Selective search for object recognition,” *International Journal of Computer Vision*, vol. 104, no. 2, pp. 154–171, 2013. [Online]. Available: http://disi.

[26] R. B. Girshick, "Fast {R-CNN}," *CoRR*, vol. abs/1504.0, 2015. [Online]. Available: http://arxiv.org/abs/1504.08083

[27] K. He, G. Gkioxari, P. Dollár, and R. Girshick, "Mask R-CNN," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 2, pp. 386–397, 3 2017. [Online]. Available: https://arxiv.org/abs/1703.06870v3

[28] M. Buric, M. Pobar, and M. Ivasic-Kos, "Ball detection using yolo and mask R-CNN," *Proceedings - 2018 International Conference on Computational Science and Computational Intelligence, CSCI 2018*, pp. 319–323, 12 2018.

[29] E. Prasetyo, N. Suciati, and C. Fatichah, "A Comparison of YOLO and Mask R-CNN for Segmenting Head and Tail of Fish," *ICICoS 2020 - Proceeding: 4th International Conference on Informatics and Computational Sciences*, 11 2020.

[30] I. W. A. S. Darma, N. Suciati, and D. Siahaan, "A Performance Comparison of Balinese Carving Motif Detection and Recognition using YOLOv5 and Mask R-CNN," *Proceedings - International Conference on Informatics and Computational Sciences*, vol. 2021-November, pp. 52–57, 2021.

[31] M. Ghafari, D. Mailman, P. Hatami, T. Peyton, L. Yang, W. Dang, and H. Qin, "A Comparison of YOLO and Mask-RCNN for Detecting Cells from Microfluidic Images," *4th International Conference on Artificial Intelligence in Information and Communication, ICAIIC 2022 - Proceedings*, pp. 204–209, 2022.

[32] C. Fatichah, N. Suciati, T. Mustaqim, and A. R. Revanda, "Detection of Acute Lymphoblastic Leukemia Subtypes using YOLO and Mask R-CNN," *2022 5th International Seminar on Research of Information Technology and Intelligent Systems, ISRITI 2022*, pp. 413–418, 2022.

[33] J. R. Terven and D. M. Cordova-Esparza, "A Comprehensive Review of YOLO: From YOLOv1 and Beyond," 4 2023. [Online]. Available: https://arxiv.org/abs/2304.00501v4

[34] G. Jocher, "ultralytics/yolov5: v7.0 - YOLOv5 SOTA Realtime Instance Segmentation (v7.0)," *Https://Github.Com/Ultralytics/Yolov5/Tree/V7.0*, p. 10, 11 2022. [Online]. Available: https://zenodo.org/record/7347926https://ultralytics.com/

[35] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, A. Desmaison, A. Köpf, E. Yang, Z. DeVito, M. Raison, A. Tejani, S. Chilamkurthy, B. Steiner, L. Fang, J. Bai, and S. Chintala, "PyTorch: An Imperative Style, High-Performance Deep Learning Library," *Advances in Neural Information Processing Systems*, vol. 32, 12 2019. [Online]. Available: https://arxiv.org/abs/1912.01703v1

[36] X. Liu, C. Yoo, F. Xing, H. Oh, G. E. Fakhri, J.-W. Kang, J. Woo, and C. C. J. Kuo, "Deep Unsupervised Domain Adaptation: A Review of Recent Advances and Perspectives," *APSIPA Transactions on Signal and Information Processing*, vol. 1, pp. 1–48, 2022.

[37] S. Motiian, M. Piccirilli, D. A. Adjeroh, and G. Doretto, "Unified Deep Supervised Domain Adaptation and Generalization," in *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, 2017, pp. 5716–5726.

[38] M. Wang and W. Deng, "Deep visual domain adaptation: A survey," *Neurocomputing*, vol. 312, pp. 135–153, 10 2018.

[39] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, V. Lempitsky, U. Dogan, M. Kloft, F. Orabona, and T. Tommasi, "Domain-Adversarial Training of Neural Networks," *Journal of Machine Learning Research*, vol. 17, pp. 1–35, 2016.

[40] Y. Ganin and V. Lempitsky, "Unsupervised Domain Adaptation by Backpropagation," *32nd International Conference on Machine Learning, ICML 2015*, vol. 2, pp. 1180–1189, 9 2014. [Online]. Available: https://arxiv.org/abs/1409.7495v2

[41] B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars, "Unsupervised Visual Domain Adaptation Using Subspace Alignment," 2013.

[42] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell, "Deep Domain Confusion: Maximizing for Domain Invariance," *arXiv preprint arXiv:1412.3474*, 2014.

[43] H. Zhao, R. Tachet Des Combes, K. Zhang, and G. J. Gordon, "On Learning Invariant Representations for Domain Adaptation," in *International conference on machine learning*, 2019, pp. 7523–7532.

[44] H. K. Hsu, C. H. Yao, Y. H. Tsai, W. C. Hung, H. Y. Tseng, M. Singh, and M. H. Yang, "Progressive domain adaptation for object detection," in *Proceedings - 2020 IEEE Winter Conference on Applications of Computer Vision, WACV 2020*, 2020, pp. 738–746.

[45] J. Y. Zhu, T. Park, P. Isola, and A. A. Efros, "Unpaired Image-to-Image Translation using Cycle-Consistent Adversarial Networks," *Proceedings of the IEEE International Conference on Computer Vision*, vol. 2017-October, pp. 2242–2251, 3 2017. [Online]. Available: https://arxiv.org/abs/1703.10593v7

[46] P. Ren, Y. Xiao, X. Chang, X. Chen, X. Wang, P.-Y. Huang, Z. Li, and B. B. Gupta, "A Survey of Deep Active Learning," *ACM Comput. Surv.*, vol. 54, no. 9, 2021. [Online]. Available: https://doi.org/10.1145/nnnnnnn.nnnnnnn

[47] S. Budd, E. C. Robinson, and B. Kainz, "Survey paper A survey on active learning and human-in-the-loop deep learning for medical image analysis," *Medical Image Analysis*, vol. 71, p. 102062, 2021. [Online]. Available: https://doi.org/10.1016/j.media.2021.102062

[48] C. A. Brust, C. Käding, and J. Denzler, "Active learning for deep object detection," in *VISIGRAPP 2019 - Proceedings of the 14th International Joint Conference on Computer Vision, Imaging and Computer Graphics Theory and Applications*, vol. 5, 2019, pp. 181–190.

[49] B. Settles, "Computer Sciences Department Active Learning Literature Survey," 2009.

[50] X. Yang, Z. Song, I. King, and Z. Xu, "A Survey on Deep Semi-Supervised Learning," *IEEE Transactions on Knowledge and Data Engineering*, pp. 1–20, 2022.

[51] K. Sohn, Z. Zhang, C.-L. Li, H. Zhang, C.-Y. Lee, and T. Pfister, "A Simple Semi-Supervised Learning Framework for Object Detection," 5 2020. [Online]. Available: https://arxiv.org/abs/2005.04757v2

[52] T. Y. Lin, P. Goyal, R. Girshick, K. He, and P. Dollar, "Focal Loss for Dense Object Detection," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 42, no. 2, pp. 318–327, 8 2017. [Online]. Available: https://arxiv.org/abs/1708.02002v2

[53] G. Mattolin, L. Zanella, E. Ricci, and Y. Wang, "ConfMix: Unsupervised Domain Adaptation for Object Detection via Confidence-based Mixing," in *Proceedings - 2023 IEEE Winter Conference on Applications of Computer Vision, WACV 2023*, 2023, pp. 423–433. [Online]. Available: https://github.com/giuliomattolin/ConfMix.

[54] C. Lu, R. Krishna, M. Bernstein, and L. Fei-Fei, "Visual relationship detection with language priors," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 9905 LNCS, pp. 852–869, 7 2016. [Online]. Available: https://arxiv.org/abs/1608.00187v1

[55] N. Carion, F. Massa, G. Synnaeve, N. Usunier, A. Kirillov, and S. Zagoruyko, "End-to-End Object Detection with Transformers," *Lecture Notes in Computer Science (including subseries Lecture Notes in Artificial Intelligence and Lecture Notes in Bioinformatics)*, vol. 12346 LNCS, pp. 213–229, 5 2020. [Online]. Available: https://arxiv.org/abs/2005.12872v3

[56] P. Sun, R. Zhang, Y. Jiang, T. Kong, C. Xu, W. Zhan, M. Tomizuka, L. Li, Z. Yuan, C. Wang, and P. Luo, "Sparse R-CNN: End-to-end object detection with learnable proposals," in *Proceedings of the IEEE Computer Society Conference on Computer Vision and Pattern Recognition*, 2021, pp. 14 449–14 458. [Online]. Available: https://github.com/PeizeSun/SparseR-CNN

[57] X. Zhu, W. Su, L. Lu, B. Li, X. Wang, and J. Dai, "Deformable DETR: Deformable Transformers for End-to-End Object Detection," *ICLR 2021 - 9th International Conference on Learning Representations*, 10 2020. [Online]. Available: https://arxiv.org/abs/2010.04159v4

[58] J. Ho, A. Jain, and P. Abbeel, "Denoising Diffusion Probabilistic Models," *Advances in Neural Information Processing Systems*, vol. 2020-December, 6 2020. [Online]. Available: https://arxiv.org/abs/2006.11239v2

[59] K. Wang, Y. Nie, C. Fang, C. Han, X. Wu, X. W. Wang, L. Lin, F. Zhou, and G. Li, "Double-Check Soft Teacher for Semi-Supervised Object Detection," *IJCAI International Joint Conference on Artificial Intelligence*, vol. 2, pp. 1430–1436, 7 2022. [Online]. Available: https://www.ijcai.org/proceedings/2022/199

[60] P. Zhu, L. Wen, D. Du, X. Bian, H. Fan, Q. Hu, and H. Ling, "Detection and Tracking Meet Drones Challenge," *IEEE Transactions on Pattern Analysis and Machine Intelligence*, vol. 44, no. 11, pp. 7380–7399, 1 2020. [Online]. Available: https://arxiv.org/abs/2001.06303v3

[61] M. Everingham, L. Van Gool, C. K. I Williams, J. Winn, A. Zisserman, M. Everingham, L. K. Van Gool Leuven, B. CKI Williams, J. Winn, A. Zisserman, C. K. Williams, J. Winn, and A. Zisserman, "The PASCAL Visual Object Classes (VOC) Challenge," *International Journal of Computer Vision*, vol. 88, no. 2, pp. 303–338, 6 2010. [Online]. Available: https://link.springer.com/article/10.1007/s11263-009-0275-4http://www.flickr.com/

[62] B. Sekachev, N. Manovich, M. Zhiltsov, A. Zhavoronkov, D. Kalinin, B. Hoff, TOsmanov, D. Kruchinin, A. Zankevich, DmitriySidnev, M. Markelov, Johannes222, M. Chenuet, a-andre, telenachos, A. Melnikov, J. Kim, L. Ilouz, N. Glazov, Priya4607, R. Tehrani, S. Jeong, V. Skubriev, S. Yonekura,

V. Truong, zliang7, lizhming, and T. Truong, “opencv/cvat: v1.1.0,” 8 2020. [Online]. Available: https://zenodo.org/record/4009388

[63] H. Su, J. Deng, and L. Fei-Fei, “Crowdsourcing annotations for visual object detection,” in *AAAI Workshop - Technical Report*, vol. WS-12-08, 2012, pp. 40–46. [Online]. Available: www.aaai.org