Robust Object Detection with Pseudo Labels from VLMs using Per-Object Co-teaching

Anonymous authors
Paper under double-blind review

Abstract

Foundation models, especially vision-language models (VLMs), offer compelling zero-shot object detection for applications like autonomous driving, a domain where manual labelling is prohibitively expensive. However, their detection latency and tendency to hallucinate predictions render them unsuitable for direct deployment. This work introduces a novel pipeline that addresses this challenge by leveraging VLMs to automatically generate pseudo-labels for training efficient, real-time object detectors. Our key innovation is a per-object co-teaching-based training strategy that mitigates the inherent noise in VLM-generated labels. The proposed per-object coteaching approach filters noisy bounding boxes from training instead of filtering the entire image. Specifically, two YOLO models learn collaboratively, filtering out unreliable boxes from each mini-batch based on their peers' per-object loss values. Overall, our pipeline provides an efficient, robust, and scalable approach to train high-performance object detectors for autonomous driving, significantly reducing reliance on costly human annotation. Experimental results on the KITTI dataset demonstrate that our method outperforms a baseline YOLOv5m model, achieving a significant mAP@0.5 boost (31.12% to 46.61%) while maintaining real-time detection latency. Furthermore, we show that supplementing our pseudo-labelled data with a small fraction of ground truth labels (10%) leads to further performance gains, reaching 57.97% mAP@0.5 on the KITTI dataset. We observe similar performance improvements for the ACDC and BDD100k datasets.

1 Introduction

Real-time object detection is paramount for safe navigation in autonomous driving systems, demanding rapid and accurate environmental perception Saleh et al. (2021). Traditional object detection methods, while effective, rely on extensive and precise human-annotated data, which is both labour and capital intensive Redmon et al. (2016). Vision-Language Models (VLMs) have emerged as a promising alternative, demonstrating remarkable zero-shot detection capabilities for a broad range of objects described through natural language prompts Li et al. (2023). This enables a potentially scalable paradigm where detector performance is no longer limited by the availability of human-labelled data Tang et al. (2021).

However, deploying large-scale VLMs directly in real-time autonomous driving scenarios faces significant hurdles. First, the pseudo-labels generated by VLMs are often noisy and imprecise, particularly in challenging edge cases like occlusions or adverse weather conditions, making them unreliable for safety-critical applications Gao et al. (2024); Han et al. (2018). Second, VLMs are computationally expensive, rendering them impractical for real-time inference on resource-constrained automotive platforms Gupta et al. (2024); Chadwick & Newman (2019). Simply training fixed-task object detectors on these pseudo-labels can lead to significant performance degradation due to inaccurate bounding boxes and misclassified objects Singh et al. (2024). Thus, a key challenge lies in developing methods to mitigate the noise inherent in VLM-generated labels and extract a reliable training signal Li et al. (2020b).

To address these challenges, we propose a novel pipeline that combines the benefits of VLM-based pseudo-annotation with a robust per-object-based co-teaching training strategy. Our approach leverages the zero-shot knowledge of VLMs to generate pseudo-labels and then trains two randomly initialised YOLOv5 models simultaneously. Each model selectively filters out potentially noisy samples from each mini-batch based on the other model's loss values. This allows us to leverage the scalability of VLMs while effectively mitigating the impact of inaccurate pseudo-labels. Notably, our approach is designed to outperform vanilla model distillation, which is negatively influenced by noisy teacher labels Li et al. (2020b), and benefits significantly from the inclusion of even a small percentage of ground truth data Tang et al. (2021).

We make the following key contributions in this paper.

- A novel per-object based coteaching framework to mitigate the impact of noisy annotations introduced by VLMs.
- Our per-object coteaching-based approach outperforms a baseline YOLOv5 model trained on raw pseudo-labels. Detailed experimental analysis to demonstrate a significant rise in the detection performance on KITTI, ACDC and BDD100K datasets. More specifically, we observe that the mAP@0.5 score improves by 15.49% on the KITTI dataset, 7.19% on the ACDC dataset and 11.07% on the BDD100K dataset.
- We perform an ablation study by (i) varying unlabeled data from 60% to 100%, mAP@0.5 rises from 38.34% to 46.61%, and (ii) mixing in 0–25% ground-truth annotations. The best result, 77.80% mAP@0.5 with 25% GT, represents a 31.19-point gain over the all-pseudo (100%) case.

Additionally, the proposed approach is computationally efficient compared to direct VLM inference and suitable for real-time object detection. It leverages unlabeled data without reliance on human annotation, which makes it more scalable.

2 Related Work

2.1 Object Detection

Open Vocabulary Detectors Zero-shot object detection addresses the challenge of detecting objects from categories not seen during training. Open-vocabulary object detection Gu et al. (2022); Minderer et al. (2022) expands this concept by allowing detection models to identify objects based on natural language descriptions without explicitly being trained on these classes.

Foundation models like OWL Minderer et al. (2022) and OWLv2 Minderer et al. (2024) leverage pre-trained vision-language models to enable zero-shot detection capabilities. These models align visual and textual embeddings in a shared semantic space, allowing the detection of objects described by arbitrary text prompts without category-specific training data.

OWLv2 Minderer et al. (2024) builds upon the original OWL architecture with improved training strategies and a more efficient design. It uses a vision transformer (ViT) backbone combined with a text encoder to process image regions and textual descriptions, computing similarity scores between them. This makes OWLv2 particularly valuable as an auto-labeller for domains with limited labelled data or novel object categories—a common scenario in autonomous driving environments.

Although foundation models like OWLv2 offer powerful zero-shot capabilities, they typically have substantial computational requirements that make them impractical for direct deployment on autonomous vehicles with limited hardware resources and real-time processing constraints Minderer et al. (2024); Zhu et al. (2022).

Single-Stage Detectors Single-Stage Detection methods, particularly the YOLO family of models, are popular for real-time applications. YOLO ("You Only Look Once") Redmon et al. (2016) pioneered a one-pass detection architecture that predicts bounding boxes and classes in a single network forward pass. This was followed by multiple updates to (v2 Redmon & Farhadi (2017), v3 Redmon & Farhadi (2018), v4 Bochkovskiy et al. (2020), v5 Jocher (2020), v7 Wang et al. (2022), v8 Jocher et al. (2023), etc.) which focused on improving performance while maintaining or improving latency. Recently Cheng et al. (2024) combined YOLO's efficiency with open-vocabulary capabilities using vision–language pre-training and a region-text contrastive loss to detect a wide range of object classes in a zero-shot manner.

2.2 Learning with Noisy Labels

Training neural networks with noisy labels is a challenging task because the networks can eventually fit the noise. Methods like MentorNet Jiang et al. (2018) proposed learning a curriculum model to down-weight or discard examples suspected to have wrong labels. Coteaching Han et al. (2018) is a training paradigm proposed to mitigate label noise where two identical models with random initialisation are trained in parallel, selecting a subset of small-loss (likely clean) examples from each mini-batch for the other network to learn from. It was further extended with some improvements in the classification setting Yu et al. (2019). Numerous extensions and alternatives have since been explored. Overall, the literature shows that tolerating or filtering noise during training (through co-teaching, mentor models, robust loss functions, etc.) is vital for maintaining performance when learning with noisy labels. We build on these insights to handle errors in pseudo-box annotations.

2.3 Pseudo-Labelling Strategies for Object Detection

Using pseudo-labels (model-predicted labels on unlabeled data) is a key technique in semi-supervised object detection. In self-training, a teacher model's detections on unlabeled images are treated as ground truth to train a student model. Radosavovic et al. (2018) is an early approach towards omni-supervised learning for object detection. It generates pseudo-bounding boxes by ensembling a model's predictions under multiple image transformations and then retraining the detector on this augmented pseudo-labelled set. A critical factor in this direction is filtering out poor predictions to avoid overfitting to bad data. Recent semi-supervised frameworks address this by using confidence thresholds or teacher-student mutual learning. Liu et al. (2021) is a notable method that mitigates bias toward easy classes in pseudo-labels. Gao et al. (2022) used pseudo-labels produced by an open vocabulary object detection model for training R-CNN and proposed it as an approach towards a universal object detector. However, pseudo-labels generated by open vocabulary models contain noisy labels, hallucinated boxes and inaccurate box coordinates.

2.4 Robust Object Detection

Training robust object detectors on noisy data goes beyond noisy labels. These methods have to deal with label noise, missing annotations, inaccurate bounding box coordinates, or out-of-distribution inputs. Chadwick & Newman (2019) systematically analysed how different noise types (classification errors, localisation errors, etc.) affect object detection. They proposed a per-object co-teaching strategy to mitigate label noise while training an R-CNN. Our approach differs in filtering strategy, which is based on the YOLO loss function. Li et al. (2020a) proposed an alternating optimisation scheme that iterates between correcting noisy labels and updating the detector. This handles noise in both class labels and box coordinates. Wan et al. (2019) introduced a meta learning solution using a small set of trusted, clean samples. Liu et al. (2022) proposed an R-CNN framework focusing on training with inaccurate bounding box coordinates.

3 Preliminaries

This section provides an overview of the key concepts and prior work foundational to our approach: zero-shot object detection, YOLO based single stage detection methods and coteaching-based methods to train robust models on noisy data.

3.1 Object Detection with YOLO

Object detection is a fundamental computer vision task that involves localising and classifying objects within an image. For autonomous driving applications, object detection models must balance accuracy with real-time performance to ensure safe navigation. The You Only Look Once (YOLO) Redmon et al. (2016); Redmon & Farhadi (2018) family of models have emerged as a leading approach for real-time object detection by framing detection as a regression problem.

YOLOv5 Jocher et al. (2021) represents a significant advancement in the YOLO architecture, offering various model sizes (from nano to extra-large) that provide different efficiency-accuracy trade-offs. The architecture divides an input image into a grid and predicts bounding boxes and class probabilities directly from full images in a single evaluation. The loss function in YOLOv5 is composed of three primary components:

$$\mathcal{L} = \lambda_{coord}\ell^{\text{box}} + \lambda_{obj}\ell^{\text{obj}} + \lambda_{cls}\ell^{\text{cls}} \tag{1}$$

Where ℓ^{box} represents the bounding box regression loss (typically a combination of CIoU loss Zheng et al. (2020)), ℓ^{obj} is the objectness confidence loss, and ℓ^{cls} is the classification loss. The λ terms are weighting factors that balance the contributions of each component.

While YOLO models provide efficient inference, they typically require extensive labelled training data and struggle with novel or rare object categories—a significant limitation for autonomous driving in complex and unpredictable real-world environments.

3.2 Learning with Noisy Labels using Coteaching

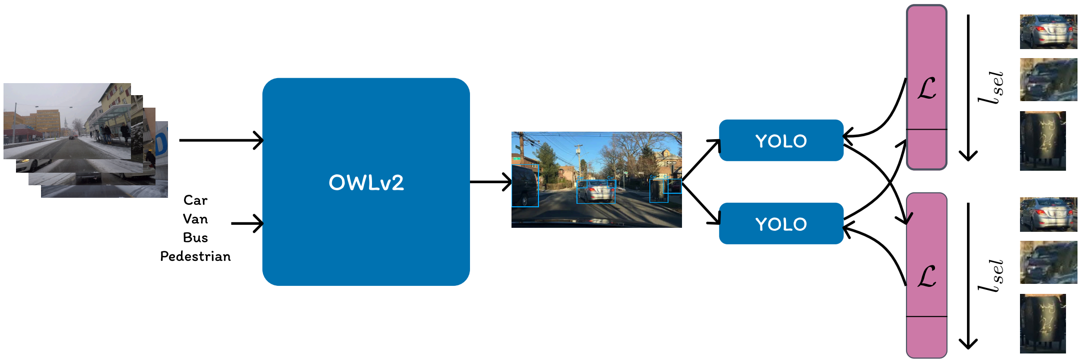


Figure 1: Pipeline for training robust, open vocabulary, real-time object detectors

Co-teaching. Co-teaching Han et al. (2018) is a robust training strategy designed to mitigate the impact of noisy labels. It employs two networks trained simultaneously, with each network learning from a different subset of the data. At every training iteration, each network selects samples from the mini-batch that produce the smallest losses (presumably clean labels) and uses these samples to update its peer network. The key intuition behind co-teaching is that low-loss samples are more likely to be correctly labelled, allowing models to mutually reduce the influence of noisy data. Formally, given two networks f_1 and f_2 with parameters θ_1 and θ_2, each network selects a proportion $R(T)$ of samples with the smallest losses. The updates are performed as follows:

$$\theta_1^{t+1} = \theta_1^t - \eta\nabla\mathcal{L}(f_1(X^{(2)}_{\text{small}}), Y^{(2)}_{\text{small}}) \tag{2}$$

$$\theta_2^{t+1} = \theta_2^t - \eta\nabla\mathcal{L}(f_2(X^{(1)}_{\text{small}}), Y^{(1)}_{\text{small}}) \tag{3}$$

where $X^{(i)}_{\text{small}}$ and $Y^{(i)}_{\text{small}}$ represent samples selected by network i based on their lowest losses, and η is the learning rate.

4 Methodology

In this section, we describe our methodology for building efficient and robust object detection models for autonomous driving. Our approach combines training on foundation model outputs with per-object coteaching

Algorithm 1: Per-Object Co-Teaching for Robust YOLO Training with Pseudo-Labels

```
Input: Training images $\mathcal{X}_{\mathrm{tr}}$, prompts $\mathcal{P}$; Test set $(\mathcal{X}_{\mathrm{te}}, \mathcal{Y}_{\mathrm{te}})$;
Estimated noise rate $\hat{r}$; total epochs $T$; ramp-up epochs $T_k$
Pre-processing: Obtain pseudo-labels using an open-vocabulary detector (OVD): $\tilde{\mathcal{Y}}_{\mathrm{tr}} \leftarrow \mathrm{OVD}(\mathcal{X}_{\mathrm{tr}}, \mathcal{P})$;
From the noisy training set $\tilde{\mathcal{D}}_{\mathrm{tr}} = \{(x_i, \tilde{y}_i)\}$;
Initialise two YOLO models $f_\theta$ and $g_\phi$ with random weights;
for $e = 1$ to $T$ do
    $r_e \leftarrow \hat{r} \cdot \min\big(e/T_k,\, 1\big)$;
    foreach mini-batch $\mathcal{B} = \{(x_b, \tilde{y}_b)\}_{b=1}^{B} \subset \tilde{\mathcal{D}}_{\mathrm{tr}}$ do
        Forward;
        $P_f \leftarrow f_\theta(\mathcal{B})$;
        $P_g \leftarrow g_\phi(\mathcal{B})$;
        Anchor-level selection loss (per positive anchor $j$);
        $\ell^{\mathrm{sel}}_{f,j} = \lambda_{\mathrm{box}} \ell^{\mathrm{box}}_{f,j} + \lambda_{\mathrm{cls}} \ell^{\mathrm{cls}}_{f,j}$;
        $\ell^{\mathrm{sel}}_{g,j} = \lambda_{\mathrm{box}} \ell^{\mathrm{box}}_{g,j} + \lambda_{\mathrm{cls}} \ell^{\mathrm{cls}}_{g,j}$;
        $N_{\mathrm{pos}} \leftarrow$ # positive anchors in $\mathcal{B}$;   $k \leftarrow \big\lceil (1 - r_e) N_{\mathrm{pos}} \big\rceil$;
        Co-teaching filter;
        $\mathcal{K}_f \leftarrow$ indices of the $k$ smallest $\{\ell^{\mathrm{sel}}_{g,j}\}$;   $\mathcal{K}_g \leftarrow$ indices of the $k$ smallest $\{\ell^{\mathrm{sel}}_{f,j}\}$;
        Masked YOLO loss;
        $\mathcal{L}_f = \sum_{j \in \mathcal{K}_f} \big(\lambda_{\mathrm{box}} \ell^{\mathrm{box}}_{f,j} + \lambda_{\mathrm{cls}} \ell^{\mathrm{cls}}_{f,j} + \lambda_{\mathrm{obj}} \ell^{\mathrm{obj}}_{f,j}\big)$;
        $\mathcal{L}_g = \sum_{j \in \mathcal{K}_g} \big(\lambda_{\mathrm{box}} \ell^{\mathrm{box}}_{g,j} + \lambda_{\mathrm{cls}} \ell^{\mathrm{cls}}_{g,j} + \lambda_{\mathrm{obj}} \ell^{\mathrm{obj}}_{g,j}\big)$;
        Back-propagation;
        $\theta \leftarrow \theta - \eta \nabla_\theta \mathcal{L}_f$;
        $\phi \leftarrow \phi - \eta \nabla_\phi \mathcal{L}_g$;
Output: Trained parameters $(\theta, \phi)$ of two YOLO detectors $f_\theta$ and $g_\phi$
Inference: $\widehat{\mathcal{Y}}_f \leftarrow f_\theta(\mathcal{X}_{\mathrm{te}})$;
  $\widehat{\mathcal{Y}}_g \leftarrow g_\phi(\mathcal{X}_{\mathrm{te}})$;
```

to create lightweight and robust detectors that can operate in challenging real-world conditions and scale with an increasing stream of unlabelled data.

Our pipeline consists of three main components:

1. A foundation model (OWLv2) that serves as an auto-labeller for an open set of classes.
2. Two student YOLO models that train on the outputs from the teacher.
3. A per-object coteaching mechanism that enables the student models to discard the noisy objects and train on clean samples.

Figure 1 illustrates the overall architecture of our approach.

4.1 Generating Pseudo Labels Using Foundation Model (VLM)

We employ OWLv2 Minderer et al. (2024) as our foundation model teacher. OWLv2 is a vision-language model that excels at open-vocabulary object detection, allowing it to identify and localise a wide range of objects beyond those seen during training. This capability is crucial for autonomous driving, where vehicles must recognise and respond to unusual or rare objects.

While OWLv2 provides high-quality detections, it has two limitations that we address:

- Computational overhead: OWLv2 is too large and slow for real-time inference on automotive hardware.
- Label noise: Foundation models can produce hallucinated or inaccurate detections, especially in edge cases.

We use OWLv2 offline to generate pseudo-labels on a large, diverse dataset of driving scenarios. These pseudo-labels serve as the foundation for training our student models.

4.2 Per-Object Co-teaching of YOLO Models Using Pseudo Labels

We choose YOLOv5 Jocher et al. (2021) for its excellent speed-accuracy trade-off for downstream tasks. YOLOv5 builds upon the one-stage detection paradigm introduced in the original YOLO Redmon et al. (2016) and incorporates architectural improvements from subsequent versions Redmon & Farhadi (2017; 2018); Bochkovskiy et al. (2020); Jocher et al. (2021); Wang et al. (2022).

We train two separate YOLOv5 models with identical architectures but different initialisations to serve as our co-teaching pair. Both models are optimised for automotive hardware, with particular attention to inference speed and memory footprint.

The core of our approach is a co-teaching mechanism, adapted for object detection, that enables our two student models to learn collaboratively while being robust to the label noise inherent in the teacher's pseudo-labels. Co-teaching was originally proposed for image-level classification tasks with noisy labels Han et al. (2018). However, in object detection, a single image can contain a mix of correctly and incorrectly labelled objects. A simple image-level selection would be suboptimal, as it would discard valuable clean labels within an otherwise "noisy" image.

To address this, we introduce a granular, **anchor-level co-teaching filter**. Coteaching strategy is based on the insight that different network initialisations will cause the two models to learn clean, simple patterns before fitting to the noise in the pseudo labels Arpit et al. (2017). We leverage this by having each model select high-confidence anchor boxes for its peer to train on.

The standard YOLO loss consists of bounding box regression loss (ℓ^{box}), classification loss (ℓ^{cls}), and objectness loss (ℓ^{obj}). We first define a specialised selection loss (ℓ^{sel}) to identify clean anchors. We explain this choice in 7.1.

$$\ell_j^{\text{sel}} = \lambda_{\text{box}} \ell_j^{\text{box}} + \lambda_{\text{cls}} \ell_j^{\text{cls}} \tag{4}$$

The filtering process for each mini-batch proceeds as follows:

1. Both student models, f_θ and g_ϕ, perform a forward pass on the batch and compute the selection loss ℓ_j^{sel} for every positive anchor.
2. A forget-rate, r_e, determines the proportion of anchors to be discarded. We calculate the number of clean anchors to keep, $k = \lceil (1 - r_e) N_{\text{pos}} \rceil$, where N_{pos} is the total number of positive anchors in the batch.
3. To train model f_θ, we identify the set of anchor indices $\mathcal{K}_f$ that correspond to the k smallest selection losses calculated by its peer, model g_ϕ.
4. Symmetrically, model g_ϕ is given the indices $\mathcal{K}_g$ corresponding to the k smallest selection losses from model f_θ.

Each model is then updated using a masked YOLO loss, where the full detection loss—including the objectness term—is computed only on the clean set of anchors selected by its peer. The final loss for model f_θ is:

$$\mathcal{L}_f = \sum_{j \in \mathcal{K}_f} \left(\lambda_{\text{box}} \ell_{f,j}^{\text{box}} + \lambda_{\text{cls}} \ell_{f,j}^{\text{cls}} + \lambda_{\text{obj}} \ell_{f,j}^{\text{obj}} \right) \tag{5}$$

This cross-selection and update strategy creates a robust training loop where each network benefits from the high-confidence selections of the other, effectively filtering out noise and preventing error accumulation.

To further stabilise training, we employ a curriculum learning strategy by gradually adjusting the forget-rate r_e over time. Early in training, the models are still learning basic features, so we use a small forget-rate, meaning we trust a larger portion of the pseudo-labels. As training progresses, the models become more discerning, and we can increase the forget-rate to filter more aggressively. We implement this with a linear ramp-up schedule for the forget-rate:

$$r_e = \hat{r} \cdot \min\left(1, \frac{e}{T_k}\right) \tag{6}$$

Here, e is the current epoch, T_k is a ramp-up period, and $\hat{r}$ is the estimated noise rate of the pseudo-labels. This curriculum allows the models to first learn from a wide distribution of samples and then gradually focus on the cleanest examples, enhancing final model robustness. After training two models in this co-teaching framework, we can use either model for inference or use any ensemble of outputs from both models if required. Complete details of the training methodology are given in Algorithm 1.

5 Experimental Setup

Psuedo Labels For autolabelling the images in each dataset, we use OWLv2 with the pre-trained checkpoint provided by the authors Minderer et al. (2024). It is the current state-of-the-art model for zero-shot object detection and is widely used for the pseudo-labelling task. GT represents the Ground Truth Labels of the dataset. Auto Labels represent the output from OWLv2 without any post-processing. Pseudo labels represent the Hard labels processed from Auto labels using NMS and confidence-based thresholding with a default threshold of 0.3.

YOLOv5 as Student Model for Per-Object Coteaching For training an efficient downstream model using pseudo labels, we use the YOLOv5 Jocher (2020) architecture. It is a well-studied and widely used model for single-stage object detection. We use YOLOv5 because a full, active and from scratch implementation is publicly available Jocher (2020) with easy access to tuning the model internals. Newer versions of YOLO are not usually published and released as a training API with low-level control Jocher & Qiu (2024). Although we use YOLOv5 for the reasons above, our proposed training procedure is YOLO version-agnostic and can be transferred to other YOLO variants directly. We did not use recent Transformer-based Object Detection method like DeTR Carion et al. (2020) due to higher latency and compute requirements, which defeats our purpose of an efficient real-time detection model. We use the YOLOv5m variant throughout our study. We also note that the coteaching framework will require two models to fit inside a GPU simultaneously. This will mean we will utilise double the GPU memory to fit the same size model and the same batch size during training. We use the per-object coteaching approach proposed in Algorithm 1.

Baselines We compare the performance of our method with the following baselines in a similar setting. In this comparison, we use a different set of labels from training the model.

- **OWLv2** Minderer et al. (2024): Auto labels generated by the VLM with prompts of each class.
- **Base**: YOLO model is trained on Pseudo Labels.
- **Soft Distillation** Hinton et al. (2015): YOLO model is trained using Soft Distillation.
- **Data Distillation** Radosavovic et al. (2018): Generate multiple pseudo labels for each sample with multiple independent transformations.
- **Coteaching**: Han et al. (2018) Model is trained using vanilla Coteaching by sorting per image loss and discarding a few samples from each mini-batch.

The benchmark performance for our method is a YOLO model trained on the ground truth dataset. In our setting, we are assuming we don't have access to ground truth labels and using VLMs to generate pseudo labels.

			KITTI		**ACDC**		**BDD100K**	
Model	*Method*	*Labels*	mAP@0.5	0.5:0.95	mAP@0.5	0.5:0.95	mAP@0.5	0.5:0.95
OWLv2	Base	None	32.34	16.52	18.82	8.92	30.81	15.86
YOLOv5m	Base	Pseudo	31.12	16.18	20.12	9.27	32.14	16.2
YOLOv5m	Soft Distillation	Auto	34.12	17.21	21.75	10.1	35.12	16.9
YOLOv5m	Data Distillation	Psuedo	37.15	18.51	25.41	12.88	37.42	17.61
YOLOv5m	Standard Coteaching	Pseudo	39.35	20.01	23.13	11.96	36.86	17.48
YOLOv5m	Per Object Coteaching	Pseudo	**46.61**	**22.05**	**27.31**	**14.28**	**43.21**	**20.81**
YOLOv5m	Base	GT	90.3	68.5	29.57	15.09	51.91	28.24

Table 1: Comparison of zero shot object-detection results (mAP, %).

Datasets Used We perform experiments on Autonomous driving datasets KITTI Geiger et al. (2012), ACDC Sakaridis et al. (2021), and BDD100k Yu et al. (2020). In all datasets, we used the task of 2D Object detection. KITTI has a training dataset of 7.5k images, which we split into a train and a validation set with an 80:20 ratio. The ACDC dataset contains images from adverse conditions like fog, rain, etc., which are difficult for an autolabeller to label. BDD100k has a total of 70k training set, 10k validation and 20k test set images. We removed labels like 'misc' from the training set of all datasets, as an autolabeller cannot detect these vague terms without specific training data for the label.

Hyperparameters For the Base YOLO method trained on pseudo labels, we used the default hyperparameter from Ultralytics Jocher (2020) ($\lambda_{\text{box}} = 0.05$, $\lambda_{\text{cls}} = 0.3$ and $\lambda_{\text{obj}} = 0.7$) and trained a YOLOv5 from scratch for 200 epochs. For our coteaching approach, we trained the model from scratch for 300 epochs with a noise rate warm-up for 150 epochs, where it increases linearly from 0 to 0.2.

Performance Metrics Used For comparing the performance on Object Detection, we compare the mAP@0.5 and mAP@0.5:0.95 metrics of all the methods for images in the validation set. We also compare the inference efficiency of OWLv2 and YOLOv5m to analyse how viable it is for real-world self-driving deployment.

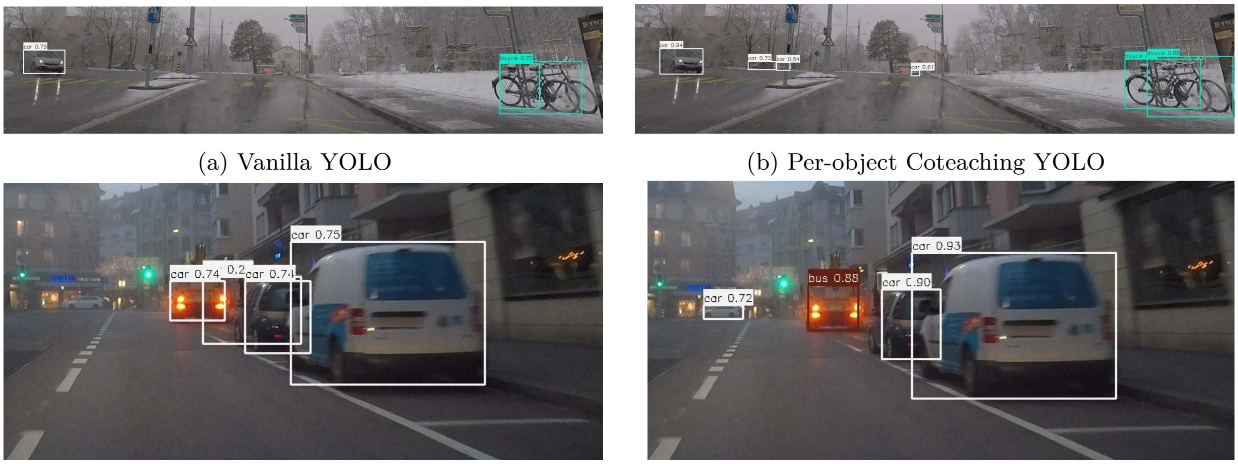
6 Results

In this section, we present detailed experimental evaluations demonstrating the effectiveness of our proposed pipeline.

6.1 Detection Performance

We present the performance comparison of all methods on the validation sets of the KITTI, ACDC and BDD100k datasets in Table 1. We observe clear improvements in detection performance when using our proposed pipeline compared to the baseline distillation method. On the KITTI dataset, our method achieves an mAP@0.5 of 46.61%, substantially higher than the baseline's 31.2%. Similarly, the stricter mAP@0.5:0.95 metric increases from 16.18% in the baseline to 22.05% using our pipeline. We also notice that per-object coteaching outperforms general coteaching performed per-image. The trend is consistent across all three datasets, with comparable relative improvements observed.

We also notice that when the dataset is easier for the model to learn in the presence of ground truth, our pipeline underperforms due to a lack of clean labels. However, in a similar setting in the presence of adverse images (ACDC contains data in adverse conditions like rain, fog, etc.), our pipeline reaches its full potential and performs close to the model trained on ground truth. For example, in a relatively easier dataset like KITTI, the difference in mAP@0.5 with our pipeline and a model trained on ground truth is a massive 43.7%; however, in ACDC, this difference is only 2.26%.



(a) Vanilla YOLO

(b) Per-object Coteaching YOLO

(c) Vanilla YOLO

(d) Per-object Coteaching YOLO

Figure 2: Comparison of predictions made with vanilla YOLO trained and YOLO trained with our method.

6.2 Efficiency Comparison for Real-Time Object Detection

For real-time object detection applications in autonomous vehicles, computational efficiency and real-time latency are important. We conducted a comprehensive benchmarking of YOLOv5m and OWLv2 (OwlViT-base-patch32) to evaluate their suitability for deployment. We perform our profiling experiments on a single NVIDIA GeForce RTX 2080.

Inference Performance YOLOv5m demonstrates significantly superior inference speed with a mean inference time of 12.65ms compared to OWLv2's 38.98ms. This translates to theoretical frame rates of 79.0 FPS for YOLOv5m versus 25.7 FPS for OWLv2. Autonomous driving applications require 30+ FPS for real-time perception. YOLOv5m meets this threshold with substantial headroom, while OWLv2 falls short of real-time requirements.

Resource Utilization YOLOv5m contains 21.2M parameters (80.8 MB) compared to OWLv2's 153.2M parameters (584.5 MB). At its peak, YOLOv5m utilises 153.6 MB while OWLv2 utilises 657.3 MB of GPU memory. For edge deployment scenarios with limited GPU memory, YOLOv5m's 4.3× reduction in GPU memory usage represents a critical advantage.

6.3 Predictions on Sample Images

In Figure 2, subfigures a and b contrast the baseline YOLO detector with our per-object co-teaching model on the snowy-weather scene: whereas the baseline a fails to detect three distant cars under low visibility and produces loose, misaligned bounding boxes on the bicycles, our method b recovers all of the cars and fits the bicycle boxes tightly and accurately. Likewise, comparing c and d, the baseline c yields a spurious car detection and misclassifies a bus, but our approach d eliminates that false positive and correctly labels the bus with a tight bounding box.

7 Ablation study

7.1 Why $\ell^{\text{sel}} = \ell^{\text{box}} + \ell^{\text{cls}}$

We found the objectness score (ℓ^{obj}) to be an unreliable signal for selection due to its sensitivity to background in images. Therefore, our selection loss for each positive anchor j deliberately excludes the objectness term. This allows us to identify anchors that are both well-localised and correctly classified, which is a more stable indicator of a clean label. In table 2, we present results of a controlled study with different choices of ℓ^{sel} and that excluding ℓ^{obj} results in better overall performance of the model.

ℓ^{sel}	mAP@0.5 (%)
ℓ^{box}	31.45
ℓ^{cls}	4.5
ℓ^{obj}	36.42
$\ell^{\text{box}} + \ell^{\text{cls}}$	**46.61**
$\ell^{\text{cls}} + \ell^{\text{obj}}$	34.16
$\ell^{\text{box}} + \ell^{\text{obj}}$	31.96
$\ell^{\text{box}} + \ell^{\text{cls}} + \ell^{\text{obj}}$	43.82

Table 2: Comparision with choice of ℓ^{sel}

7.2 Increasing unlabeled data

We further analyse the scalability of our proposed pipeline by incrementally varying the amount of unlabeled data used for training the detector. Table 3 illustrates the results of this controlled experiment conducted on the KITTI dataset. We observe a consistent increase in mAP@0.5 performance as we progressively scale up the training set size from 60% to 100% of available unlabeled data.

Specifically, the mAP@0.5 increases from approximately 38% when trained on just 60% of the data, to over 46% with the entire unlabeled dataset. This validates our hypothesis that the co-teaching mechanism effectively filters label noise and allows the model to scale gracefully with additional unlabeled training data. This can be a promising approach, as collecting a lot of unlabeled data is significantly easier compared to labelling existing data precisely.

Pseudo Labels (%)	Ground Truth (%)	mAP@0.5 (%)
60	0	38.34
70	0	39.49
80	0	41.98
90	0	44.2
100	0	46.61

Table 3: Impact of increasing the unlabeled images in the pipeline in the KITTI Dataset.

7.3 Semi Supervised Setting

We mix some percentage of ground truth labels in the training data and analyse how this affects our performance. As we increase the GT data, especially in datasets with huge differences in performance when trained on ground truth, the performance increases significantly. This is mainly due to some labels that the VLM couldn't pick up during the labelling process, but a few samples from the ground truth significantly improved the model's ability to identify and detect these classes. We present the results in table 4. We show that incorporating just 10% of precisely labelled ground truth data improves the performance from 46.61% to 57.97%.

8 Conclusion

Our comprehensive evaluations demonstrate the clear advantages of our proposed pipeline. Our per-object co-teaching mechanism robustly addresses pseudo-label noise, significantly improving accuracy across multiple datasets and evaluation metrics compared to baseline distillation. Additionally, our pipeline maintains efficient real-time inference, which is vital for practical autonomous driving applications. We also illustrate

Pseudo Labels (%)	Ground Truth (%)	mAP@0.5 (%)
100	0	46.61
95	5	49.42
90	10	57.97
85	15	65.13
80	20	72.42
75	25	77.80

Table 4: Impact of incorporating a small percentage of ground truth annotations during training on the KITTI Dataset.

that the judicious use of even minimal ground truth labels or increased unlabeled data can both substantially boost performance, highlighting our method's practical viability in real-world autonomous driving scenarios.

References

Devansh Arpit, Stanisław Jastrzebski, Nicolas Ballas, David Krueger, Emmanuel Bengio, Maxinder S. Kanwal, Tegan Maharaj, Asja Fischer, Aaron Courville, Yoshua Bengio, and Simon Lacoste-Julien. A closer look at memorization in deep networks, 2017. URL `https://arxiv.org/abs/1706.05394`.

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

Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *European conference on computer vision*, pp. 213–229. Springer, 2020.

Samuel Chadwick and Paul Newman. Training object detectors with noisy data, 2019.

Tianheng Cheng, Lin Song, Yixiao Ge, Wenyu Liu, Xinggang Wang, and Ying Shan. Yolo-world: Real-time open-vocabulary object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 16901–16911, 2024.

Mingfei Gao, Chen Xing, Juan Carlos Niebles, Junnan Li, Ran Xu, Wenhao Liu, and Caiming Xiong. Open vocabulary object detection with pseudo bounding-box labels. In *European Conference on Computer Vision*, pp. 266–282. Springer, 2022.

Wenbin Gao, Shuo Yang, Xiaolong Wang, Jianmin Ji, and Jian Yang. Vlm-pl: Advanced pseudo labeling approach for class incremental object detection via vision-language model, 2024.

Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *Conference on Computer Vision and Pattern Recognition (CVPR)*, 2012.

Xiuye Gu, Tsung-Yi Lin, Weicheng Kuo, and Yin Cui. Open-vocabulary object detection via vision and language knowledge distillation. In *ICLR*, 2022.

Aniruddha Gupta, Sujoy Paul, and Aniruddha Sinha. An introduction to vision-language modeling, 2024.

Bo Han, Jiayuan Yao, Gang Niu, Mingming Shan, Hui Niu, Mingli Xu, and Ivor W Tsang. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *Advances in Neural Information Processing Systems*, volume 31, pp. 8536–8546, 2018.

Geoffrey Hinton, Oriol Vinyals, and Jeff Dean. Distilling the knowledge in a neural network. *arXiv preprint arXiv:1503.02531*, 2015. URL `https://arxiv.org/abs/1503.02531`.

Lu Jiang, Zhengyuan Zhou, Thomas Leung, Li-Jia Li, and Li Fei-Fei. Mentornet: Learning data-driven curriculum for very deep neural networks on corrupted labels. In *International conference on machine learning*, pp. 2304–2313. PMLR, 2018.

Glenn Jocher. Ultralytics yolov5, 2020. URL https://github.com/ultralytics/yolov5.

Glenn Jocher and Jing Qiu. Ultralytics yolo11, 2024. URL https://github.com/ultralytics/ultralytics.

Glenn Jocher et al. Yolov5 by ultralytics. https://github.com/ultralytics/yolov5, 2021. Accessed: 2025-03-01.

Glenn Jocher et al. Yolov8 by ultralytics. https://github.com/ultralytics/ultralytics, 2023. Accessed: 2025-03-01.

Junnan Li, Caiming Xiong, Richard Socher, and Steven Hoi. Towards noise-resistant object detection with noisy annotations. *arXiv preprint arXiv:2003.01285*, 2020a.

Kang Li, Yuhong Wang, Qiang Wang, Liang Qing, and Mingkui Tan. Distilling knowledge from noisy labels. In *Advances in Neural Information Processing Systems*, volume 33, 2020b.

Yuming Li, Yiming Li, Yuxin Chen, Yujie Li, Yuxin Li, Yuxin Peng, Zhaopeng Li, Zhaoyang Li, et al. Vision language models in autonomous driving: A survey and outlook, 2023.

Chengxin Liu, Kewei Wang, Hao Lu, Zhiguo Cao, and Ziming Zhang. Robust object detection with inaccurate bounding boxes. In *European Conference on Computer Vision*, pp. 53–69. Springer, 2022.

Yen-Cheng Liu, Chih-Yao Ma, Zijian He, Chia-Wen Kuo, Kan Chen, Peizhao Zhang, Bichen Wu, Zsolt Kira, and Peter Vajda. Unbiased teacher for semi-supervised object detection. *arXiv preprint arXiv:2102.09480*, 2021.

Matthias Minderer, Alexey Gritsenko, Austin Stone, Maxim Neumann, Dirk Weissenborn, Alexey Dosovitskiy, Aravindh Mahendran, Anurag Arnab, Mostafa Dehghani, Zhenzhen Shen, et al. Simple open-vocabulary object detection with vision transformers. In *ECCV*, pp. 679–697, 2022.

Matthias Minderer, Alexey Gritsenko, and Neil Houlsby. Scaling open-vocabulary object detection, 2024. URL https://arxiv.org/abs/2306.09683.

Ilija Radosavovic, Piotr Dollár, Ross Girshick, Georgia Gkioxari, and Kaiming He. Data distillation: Towards omni-supervised learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pp. 4119–4128, 2018.

Joseph Redmon and Ali Farhadi. Yolo9000: Better, faster, stronger. In *CVPR*, pp. 7263–7271, 2017.

Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. *arXiv preprint arXiv:1804.02767*, 2018.

Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *CVPR*, pp. 779–788, 2016.

Christos Sakaridis, Dengxin Dai, and Luc Van Gool. ACDC: The adverse conditions dataset with correspondences for semantic driving scene understanding. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2021.

Ahmed Saleh, Mohammad Siam, Mostafa Elkerdawy, Mohammed Bennamoun, Sreenatha Anavatti, and John Raygosa. Deep learning for object detection and scene perception in self-driving cars: Survey, challenges, and open issues. *Complex & Intelligent Systems*, 7(1):81–105, 2021.

Karan Singh, Abhishek Kumar, Manish Singh, Satish Kumar Singh, Raghavendra Singh, Ajay Kumar Singh, Sanjay Kumar Singh, Ashutosh Kumar Singh, Anoop Kumar Singh, et al. Beyond clean data: Exploring the effects of label noise on object detection performance. *Knowledge-Based Systems*, 285:111261, 2024.

Kaihua Tang, Liangzhe Niu, Jianyu Wang, Hooman Ghasemzadeh, and J Zico Kolter. Semi-supervision with noisy labels for object detection. In *CVPR*, 2021.

Fang Wan, Chang Liu, Wei Ke, Xiangyang Ji, Jianbin Jiao, and Qixiang Ye. C-mil: Continuation multiple instance learning for weakly supervised object detection. In *Proceedings of the IEEE/CVF conference on computer vision and pattern recognition*, pp. 2199–2208, 2019.

Chien-Yao Wang, Alexey Bochkovskiy, and Hong-Yuan Mark Liao. Yolov7: Trainable bag-of-freebies sets new state-of-the-art for real-time object detectors. *arXiv preprint arXiv:2207.02696*, 2022.

Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning, 2020. URL `https://arxiv.org/abs/1805.04687`.

Xingrui Yu, Bo Han, Jiangchao Yao, Gang Niu, Ivor Tsang, and Masashi Sugiyama. How does disagreement help generalization against label corruption? In *International Conference on Machine Learning*, pp. 7164–7173, 2019.

Zhaohui Zheng, Ping Wang, Wei Liu, Jinze Li, Rongguang Ye, and Dongwei Ren. Distance-iou loss: Faster and better learning for bounding box regression. *arXiv preprint arXiv:1911.08287*, 2020.

Jiawei Zhu, Haogang Feng, Shida Zhong, and Tao Yuan. Performance analysis of real-time object detection on jetson device. In *Proceedings of the 22nd IEEE/ACIS International Conference on Computer and Information Science (ICIS)*, pp. 156–161, Zhuhai, China, June 2022. IEEE. doi: 10.1109/ICIS54925.2022.9882480.