electronics

MDPI

Article

Improving Object Detection Accuracy with Self-Training Based on Bi-Directional Pseudo Label Recovery

Shoaib Sajid, Zafar Aziz , Odilbek Urmonov and HyungWon Kim *

College of Electrical and Computer Engineering, Chungbuk National University, Cheongju 28644, Republic of Korea; shoaib@cbnu.ac.kr (S.S.); zafar343@cbnu.ac.kr (Z.A.); odilbek@cbnu.ac.kr (O.U.)

* Correspondence: hwkim@cbnu.ac.kr

Abstract: Semi-supervised training methods need reliable pseudo labels for unlabeled data. The current state-of-the-art methods based on pseudo labeling utilize only high-confidence predictions, whereas poor confidence predictions are discarded. This paper presents a novel approach to generate high-quality pseudo labels for unlabeled data. It utilizes predictions with high- and low-confidence levels to generate refined labels and then validates the accuracy of those predictions through bi-directional object tracking. The bi-directional object tracker leverages both past and future information to recover missing labels and increase the accuracy of the generated pseudo labels. This method can also substantially reduce the effort and time needed in label creation compared to the conventional manual labeling. The proposed method utilizes a buffer to accumulate detection labels (bounding boxes) predicted by the object detector. These labels are refined for accuracy though forward and backward tracking, ultimately constructing the final set of pseudo labels. The method is integrated in the YOLOv5 object detector and tested on the BDD100K dataset. Through the experiments, we demonstrate the effectiveness of the proposed scheme in automating the process of pseudo label generation with notably higher accuracy than the recent state-of-the-art pseudo label generation schemes. The results show that the proposed method outperforms previous methods in terms of mean average precision (mAP), label generation accuracy, and speed. Using the bi-directional recovery method, an increase in mAP@50 for the BDD100K dataset by 0.52% is achieved, and for the Waymo dataset, it provides an improvement of mAP@50 by 8.7% to 9.9% compared to 8.1% of the existing method when pre-training with 10% of the dataset. An improvement by 2.1% to 2.9% is achieved as compared to 1.7% of the existing method when pre-training with 20% of the dataset. Overall, the improved method leads to a significant enhancement in detection accuracy, achieving higher mAP scores across various datasets, thus demonstrating its robustness and effectiveness in diverse conditions.

Keywords: pseudo labels; semi-supervised training; self-training; object tracker

check for updates

Citation: Sajid, S.; Aziz, Z.; Urmonov, O.; Kim, H. Improving Object Detection Accuracy with Self-Training Based on Bi-Directional Pseudo Label Recovery. *Electronics* **2024**, *13*, 2230. https://doi.org/10.3390/electronics13122230

Academic Editors: Cheng Siong Chin and Namgi Kim

Received: 9 May 2024
Revised: 28 May 2024
Accepted: 4 June 2024
Published: 7 June 2024

Copyright: © 2024 by the authors. Licensee MDPI, Basel, Switzerland. This article is an open access article distributed under the terms and conditions of the Creative Commons Attribution (CC BY) license (https://creativecommons.org/licenses/by/4.0/).

1. Introduction

Identifying the moving objects on the road such as vehicles, pedestrians, and cyclists is a fundamental challenge for self-driving cars. Normally, the input data consist of images, and the desired output is a set of accurate bounding boxes that localize the detected objects. This task is known for its complexity, requiring high precision and reliability. Currently, most object detector CNN models [1–4] are trained using a training dataset that is created by manually adding a bounding box labeled to road image sequences [5,6].

In most machine learning and convolutional neural network (CNN) based detectors [7,8], the success of detection relies on collecting and labeling a large amount of training and testing data in a variety of road environments. If self-driving vehicles are limited to traveling within a specific geographic region, and their training data collection is limited in that region, such detectors may provide good detection accuracy when operated in the same

region or in similar environments. However, when a self-driving car, whose CNN detector is trained on the data from particular region, travels outside of that specified region, its detection performance often experiences significant degradation. The challenge stems from the uncertainty of the geographical regions that self-driving vehicles may travel to in the future. As a result, the conventional methods of training the detector of autonomous car on the pre-collected and labeled data of specific regions would not provide the best accuracy even at the high cost of creating a large hand-labeled dataset.

Consider a scenario where the self-driving system is trained on urban roads in Germany [5,6], but the car ends up being driven on the mountain roads in South Korea, where road conditions, vehicle sizes, weather, and the overall environment, including roadside objects such as trees and color of traffic signs roads, differ significantly. Prior research has shown that these differences can lead to a drop in detection accuracy of more than 35% [9]. This is one of the most significant challenges for consumer self-driving cars.

This problem can be solved with the Semi-Supervised Object Detection (SSOD) technique, which deals with the challenge of training the model using a combination of unlabeled and labeled data. As labeled data are usually not available for every specific area, human-annotated labeled data from a public dataset are used to supervise a learning process of detector. The unlabeled data collected from the local area through the onboard camera sensor of a car are utilized to improve the model's generalization on the target domain, making the model more robust on the target area.

Consider the scenario where a vehicle travels on the same route every day, commuting from home to the workplace, or vice versa. After reaching either home or the workplace, the car mostly stays idle, parked in the parking lot for extended periods. If the detector system of an autonomous car collects a sufficient amount of image data during this trip, it can subsequently retrain its detector offline while being parked every day. The new retrained model can gradually increase accuracy, as it is trained every day on the additional data collected. This adaptive process aims to enhance the detection performance of autonomous cars, particularly in the case when they travel from one particular region to another.

Moreover, driving in heavy rain is also a big problem for the detection systems of smart cars [10]. These systems use onboard camera sensors to identify the objects on the road in a timely manner. In rainy weather, however, the water on the windshield (if camera sensors are mounted on the dashboard) or on the camera lens (if camera sensors are mounted on the roof) makes the input images unclear, and thus objects are often missed. In this situation, the control system of the vehicle may use wipers to clean the surface. However, those wipers may sometimes block the view and cause extra difficulties in acquiring clear input images from the onboard cameras. Using those unclear images, a detector of an autonomous vehicle may identify the type and location of the objects incorrectly, and then its decision might lead to a dangerous accident. The same issue may happen during snowy weather, as the view of the camera also can be blocked shortly between two wipes. During this time, the detection and tracking identity of some objects can easily be lost.

We introduce a novel SSOD method for the recovery of detection labels or bounding boxes while they are distorted or occluded by rain, snow, or even by the windshield wiper. This approach requires vehicles to collect the data over time in the form of a video, as we need the sequential frames to identify an inter-frame relation of the object location using the tracker. The sequential frames (video format) also allow us to verify whether an object appearing in consecutive frames follows a motion model and can be linked in several frames based on the time. These two above conditions allow us to use objects with high confidence in consecutive frames, calculate their trajectory, predict their next state, and match the predicted state with the lower confidence objects in the next frame. We perform this in the forward direction (online) as well as in backward direction (offline) to recover the objects with poor confidence values as shown in Figure 1.

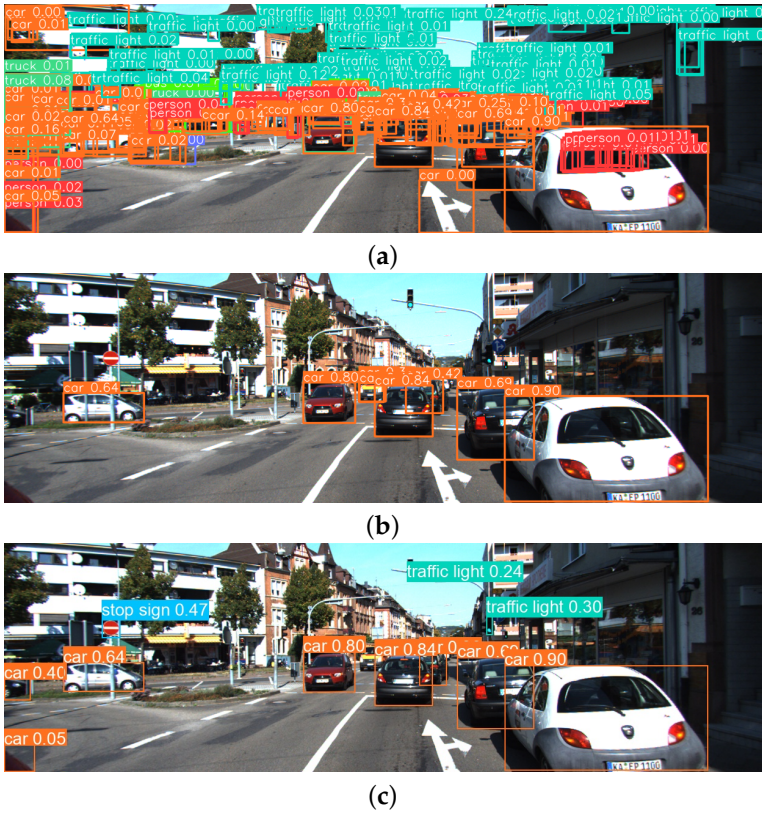


Figure 1. The object detector can produce a high number of predictions (**a**). The predictions below the threshold are filtered out, resulting in a reduced set of predictions. While this filtering process helps increase prediction certainty, it also leads to the loss of potentially crucial information (**b**). The recovery algorithm recovers the crucial information lost due to thresholding (**c**).

The proposed approach is evaluated on autonomous driving datasets, BDD [11], and Waymo [12]. The approach consistently demonstrates effectiveness across all datasets, showcasing that by following the outlined procedure, the model produces pseudo labels of higher quality and successfully adapts to the target domain. This adaptation significantly enhances the accuracy of the retrained model. Furthermore, the procedure is simple enough to implement, making it a noteworthy stride toward enabling the safe navigation of autonomous vehicles without geographical constraints. Some of the notable contributions of this paper are listed below.

- We explore the impact of recovered pseudo labels which become lost due to distortion or occlusion of the target objects, leading to poor confidence.
- We make a use of a bi-directional object tracker to be able to utilize past and future frames to recover the lost pseudo labels.
- The proposed method produces improved pseudo labels with a much simpler method as compared to [13,14], which are complex and require a lot of resources.

2. Related Work

Semi-Supervised Object Detection (SSOD) has gained significant attention in recent years owing to its ability to leverage both labeled and unlabeled data, thereby greatly expanding the applicability of object detection models. Most SSOD approaches adhere to consistency-based regularization [15–17] and pseudo labeling [18,19] strategies.

The Consistency Regularization [15–17] approach promotes consistency between predictions made by the model on different augmentations or transformations of the same unlabeled sample. By ensuring that the model's predictions remain stable across various perturbations of the data, it becomes more robust and capable of handling variations in object appearance and position. However, this approach may include a pseudo label of some

specific object that has been wrongly generated after applying a particular augmentation on the image. Then, those wrongly generated pseudo labels may cause a degradation in model accuracy after running the self-training process.

Self-Training techniques [18,19] start with training an initial model using the limited labeled data available. Then, this model is used to predict object bounding boxes and labels for the unlabeled data. The predicted bounding boxes with a high confidence are then considered pseudo labels and combined with the original labeled data to create an augmented training set. The model is then fine-tuned using this combined dataset. Other SSOD approaches include tri-training, which is an extension of self-training.

Tri-training [20,21] labeled data but with different initializations or architectures. Each model generates pseudo labels for the unlabeled data, and samples, for which at least two models agree on the labels, are added to the training set. Tri-training enhances the model's robustness and reduces the impact of noisy pseudo labels. Tri-training requires three base models, which is computationally very expensive and requires a lot of processing power. Tri-training also assumes that the initial models are diverse, which is not always true, resulting in negative training results.

In **co-training** [13,22], multiple detectors are trained independently on the same data, each using distinct feature representations or viewpoints of the data. The models then share their predictions on the unlabeled data, and samples for which there is consensus among the models are included in the training set. Co-training leverages diverse perspectives to improve performance with limited labeled data. It is not possible to always have multiple perspectives of the objects. Another assumption is that each view is conditionally independent. It is, however, only a speculation which may not hold always.

Multi-task learning [14,23,24] involves combining the object detection task with another related task, such as image segmentation or object tracking. By jointly training the model on multiple tasks, it learns shared representations that can enhance detection performance, even when labeled data are scarce. This method assumes that all tasks share the same underlying representation, which is not true for all cases. This has an adverse effect on the training, where the performance of one task can degrade due to the influence of other tasks.

Active learning techniques [25,26] select the most informative unlabeled samples for labeling by a human annotator. By strategically choosing which samples to label, the model can achieve better performance with fewer labeled examples. This allows the user to focus on only the important samples. It is most useful in reducing the training time if there is a scarce amount of data. However, that is rarely the case. Usually, the data are limited, making it impractical for use.

Data augmentation [27–29] involves creating synthetic samples by applying various transformations (e.g., rotation and scaling) to the labeled data. Augmented samples are then mixed with the original labeled data, expanding the training dataset, and improving model generalization. Data augmentation can be applied in offline (backward) pseudo label generation to create multiple different augmented views of the same image. Those augmented images later can be provided to the detector, which may generate detection labels for objects which were not clearly visible in the original image. However, it requires labeled data, and it is feasible for self-training with new data.

ASTOD [30] is an adaptive self-training method for object detection. It first generates pseudo labels and then refines them before running re-training the target detector. ASTOD enhances video object detection by integrating spatial and temporal features. It extracts spatial features from individual frames and temporal features from consecutive frames, fusing them together. An adaptive attention mechanism focuses on relevant spatial–temporal regions. This combined representation is fed into a detection head to generate bounding box proposals and class predictions. Training is performed end-to-end to optimize feature extraction, fusion, and detection tasks.

Utilizing videos for object detection has been explored in [31–35] to streamline labeling efforts by extracting additional 2D bounding boxes from videos. The main idea is to

leverage the temporal information to extend weakly labeled instances or potential object proposals across frames, which are then used as pseudo labels to retrain the detectors. Low-Confidence Samples Mining [36] utilizes the low-confidence predictions to generate more accurate bounding boxes. In [36], the authors developed multiple branches for high-confidence labels and low-confidence labels separately. The method also leverages poor confidence pseudo labels as demonstrated by [36]. Instead of employing the complicated and computationally intensive hanger branching technique proposed in [36], this method applies a lightweight and fast pseudo label generation scheme. The concept is also similar to [37,38], which improve the initial detections of the object detector in an offline manner and aggregate them across the whole sequence with a standard tracker [39]. However, the proposed approach introduces two different aspects. First, the end goal is to improve the object detector by fine-tuning it with improved pseudo labels, whereas the previous works solely focus on improving the pseudo labels. Secondly, the method uses bi-directional recovery while maintaining track history for several frames, which is crucial in recovering missing detections far away.

3. Proposed Method

The key to the successful self-training lies in the quality of the pseudo labels. The quality of pseudo labels is ensured by preventing false positive and false negative results in the detection. Whereas avoiding false positives warrants that the predicted labels are correct, avoiding false negatives guarantees the coverage of all objects with labels in the image.

One of the biggest challenges in self-training is to obtain high-quality pseudo labels for far objects which have a small size or appearance with partial occlusion by other objects. For such hard-to-detect scenarios, the detector fails to provide high-quality pseudo labels. We exploit tracking information to deal with this problem by leveraging two facts in the autonomous driving scenario. First, the available unlabeled data are in the form of sequences over time as video data. Second, the objects of interest and self-driving car with the data collection sensor travel in fairly constrained ways. We run the object detector on logged data. Once the CNN detector detects vehicles for a few frames, we can estimate the motion of the detected vehicles and categorize them as approaching or departing vehicles using their motion information. For a departing vehicle, the object detector can produce a high-confidence detection when they are fairly large in size or at a close distance to the onboard camera sensor. Once the departing vehicle starts departing away from the camera, its size becomes smaller and the detection confidence also becomes lower. Moreover, once the vehicles move away from the camera, they are more likely to be partially or fully occluded. The occlusion also lowers the confidence values. Similarly, an approaching vehicle is difficult to detect with high confidence, as it is small in size initially. Once an approaching vehicle moves closer to the camera and becomes larger in size, it can be detected more accurately.

Our method recovers labels for most approaching and departing vehicles using the bi-directional recovery algorithm and then adds them to the pseudo labels, removing false negatives.

The self-training is conducted by repeatedly re-training the pre-trained model as shown in Figure 2. The re-training is performed using the combination of the unlabeled video sequence with the recovered pseudo labels and the labeled data used in the pre-training step as shown in Figure 3. After each iteration of self-training, if the self-trained model is improved, it is adopted in the next iteration. Otherwise, the self-trained model is discarded, and the previous model from the last iteration is maintained.

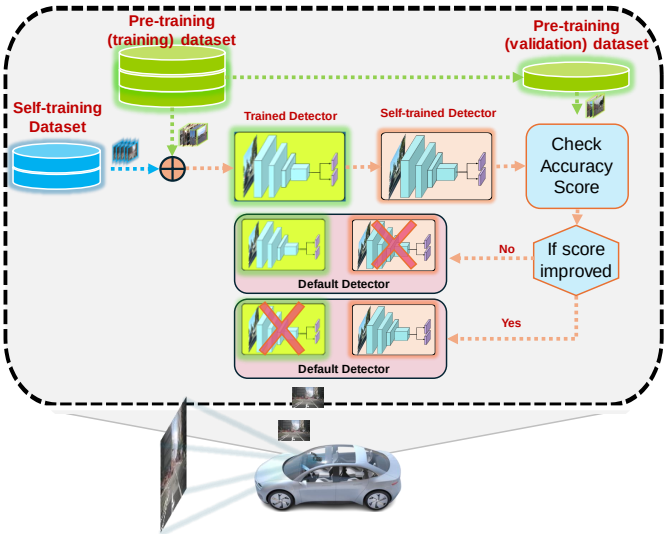
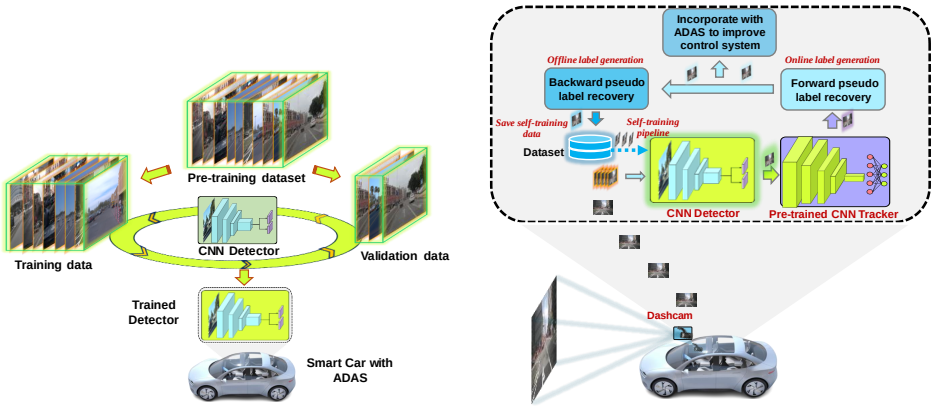


Figure 2. Bi-directional tracking generates and recovers pseudo labels to update the detector through the self-training process.



(**a**) (**b**)

Figure 3. The overall flow of the proposed method: (**a**) the initial training phase where the CNN detector is trained on an initial dataset; (**b**) the trained detector which leverages a vehicle ADAS and performs inference along with online (forward) recovery; when an object is recovered, then corresponding frame is stored in the self-training dataset.

3.1. Bi-Directional Pseudo Label Recovery

We pre-train an object detector on a limited labeled dataset as shown in Figure 3a. The pre-trained model is used to generate the labels for the unlabeled sequential videos. The labels are passed through the forward recovery scheme to improve the predictions. The improved predictions are given to the ADAS system to improve the performance. We implement a history buffer of n frames to initiate the backward recovery. Once n frames have been processed by the forward recovery algorithm, we run the backward recovery algorithm. The proposed bi-directional pseudo label recovery technique recovers the lost or missing bounding boxes. The improved pseudo labels are stored in the self-training dataset as shown in Figure 3b.

In the bi-directional recovery algorithm, the predictions are segregated into high-confidence and low-confidence labels as shown in Figure 4. The high-confidence labels are used to initiate forward and backward tracking. The low-confidence labels are the target labels to be recovered by matching them with the tracker's predictions. Upon a successful match between tracker's prediction and low-confidence label, if the Intersection over

Union (IoU) score between them is above the pre-defined threshold, the box is recovered. The recovered box is added to the pseudo labels list for the self-training.

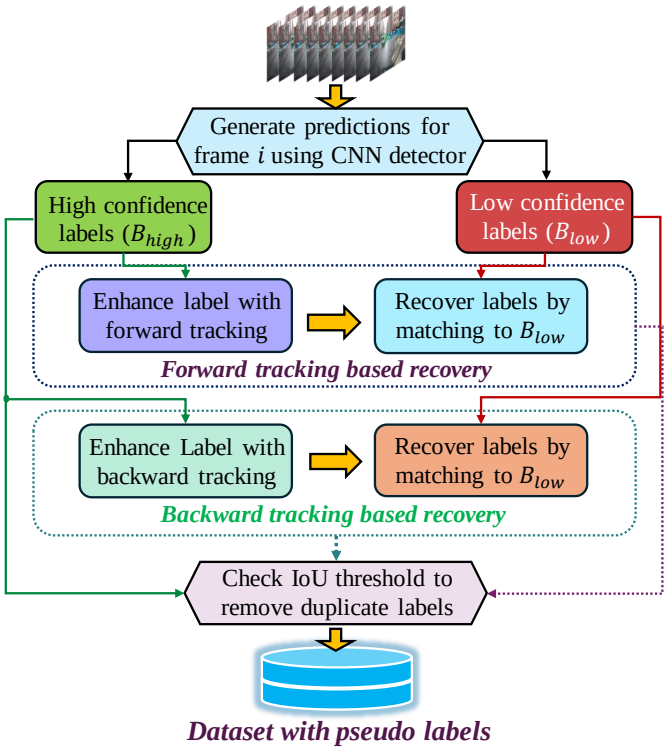


Figure 4. Overall bi-directional tracking-based pseudo label recovery method.

3.2. Improved Pseudo Labels with Tracking

3.2.1. Essence of Tracking in Pseudo Label Generation

The test accuracy is usually improved using frame-wise detection results with online tracking [40–42] . The detected objects are associated with current and past frames to make trajectories. Using the track information, it is possible to filter out false positives and false negatives, as well as to adjust the initial detection labels in the frames.

We explore this idea of online object tracking with a Kalman filter (KF) based tracker, which has shown promising results in the leaderboard of popular tracking benchmarks. Using KF provides a lightweight tracking solution in comparison to learning-based trackers [43], which requires a trained CNN model to perform tracking.

The Kalman filter is a recursive estimator that uses a series of measurements over time, incorporating a predicted state and the uncertainty associated with it, to produce an optimal estimate of the current state. We use the technique introduced in [39], which is based on eight-dimensional state space (x, y, λ, h, x', y', λ', h') that contains the bounding box center position (x, y), aspect ratio λ, height h, and their respective velocities (x', y', λ', h') in image coordinates.

To utilize the motion information, the Mahalanobis distance (squared) is used [44] between the predicted Kalman states and new bounding boxes in the current frame. The Mahalanobis distance measures how many standard deviations the detection is away from the mean track location. Using this information, we exclude unlikely associations by thresholding their Mahalanobis distance with a predefined threshold. For each detected label, denoted as b_j, we run a pre-trained CNN detector to compute the label appearance descriptor. Then, we keep a tracking history up to the last N_k associated appearance descriptors for each track label k. In the current experiment presented this work, we set N_k to 20. The Mahalanobis distance is measured as follows:

$$D_M(\mathbf{x}, \mathbf{y}) = \sqrt{(\mathbf{x} - \mathbf{y})^T \mathbf{S}^{-1} (\mathbf{x} - \mathbf{y})} \quad (1)$$

Moreover, the Mahalanobis distance offers insights into potential object locations based on the motion of the object and guarantees a meaningful result in short-term predictions. The appearance information is useful when re-establishing identities (Track IDs) after extended occlusion periods, where motion becomes less distinctive.

3.2.2. Track Association

For each track labeled as k, we keep track of the number of frames elapsed since the last successful tracking association, denoted as a_k. This counter is continuously updated during the KF prediction phase. a_k resets to zero whenever the track successfully associates with a detected label and the association cost is less than the threshold. Tracks that exceed a predefined maximum age limit, denoted as A_{max}, are considered tracks that have disappeared from the scene and are then consequently removed from the set of active tracks. When a detection cannot be linked to an existing track, we initiate new track hypotheses. Initially, these new tracks are considered tentative, and they undergo their first m frames with the expectation of establishing a successful tracking association at each step. However, any tracks that fail to associate with the track history within their first m frames are deleted. We empirically identify the best value of m as three frames. To confirm a track, we check the history in m previous frames. If an object appears consecutively, we change a tentative track to a confirmed track. Online tracking, which we call forward tracking, uses only past information to improve current detections. To further improve the quality of the pseudo labels, the proposed method utilizes both online and offline tracking. The offline tracking, called a backward tracking, is obtained by conducting the tracking algorithm backwards from the current frame to the prior frames. While bi-directional tracking is not applicable to real-time detection inference, we conduct it offline to acquire highly accurate pseudo labels for self-learning. The detailed procedure of bi-directional tracking is presented in Algorithm 1 later.

3.2.3. Example for Label Recovery

Figure 5 illustrates an example of forward tracking to improve the quality of pseudo labels for lost predictions. In step 1, it runs the detector on all m frames and then it separates the predicted labels in two categories: high-confidence and low-confidence labels using two thresholds (v_1 was detected with high confidence in earlier frames and then its confidence decreased, as it travels faster than the ego vehicle and reaches a long distance; thus, in the current frame f_i, the confidence of v_1 becomes low). In step 2, the high-confidence pseudo labels are passed to the tracker, where a tracklet for each object is properly initiated. In step 3, the confirmed tracklets produce the predicted state at the end of every track (so the tracker confirms that the low-confidence label of v_1 is true and it starts considering it as a potential pseudo label). In step 4, the predicted state of each tracklet is compared with the low-confidence detection labels in corresponding frames. If the IoU between the predicted tracklets location and the location of low-confidence labels is above the threshold, then the low-confidence label is considered a recovered label (as we see, the tracker output and low-confidence label of v_1 is overlapping with a high IoU ratio). In step 5, once a detection label is recovered, we assume that it has higher errors in terms of label position. We take an average of the detected label position (x, y) and the predicted state (x', y') of the matching tracklet and recover the label with the averaged states (in step 5, we may see that after the averaging concept, v_1 obtains a relatively accurate pseudo label shown with the blue bounding box). The averaged label location (x'', y'') is added to the pseudo labels:

$$x''_n = \frac{x_n + x'_n}{2} \quad \text{and} \quad y''_n = \frac{y_n + y'_n}{2} \tag{2}$$

The proposed pseudo label generation runs object detection inference only one time for each frame during the normal inference process of the self-driving vehicle. The detection results for $m + n + 1$ frames are stored in the history buffer and reused for forward and

backward recovery. In addition, it offers significantly higher accuracy of the generated pseudo labels.

Algorithm 1 Bi-directional recovery of lost labels.

Require:
Video Data $Data_{video}$
IOU threshold T_{IoU}
Number of previous n and future m frames
Ensure:
The lists of pseudo labels recovered by forward and backward tracking L_{fwd}, L_{bkw}
A list of the final recovered labels L_{recov}

▷ sort labels according to confidence value
for every frame f_i of $Data_{video}$ **do**
  $L_{Det} = CNNDetector(f_i)$
  Split and save L_{Det} into two sub-lists, L_{Det}^{Low} and L_{Det}^{High}
end for

▷ retrieve labels via forward recovery
for every frame i in range of n and $len(L_{Det})$ **do**
  $L_{curr} = L_{Det}^{high}[i]$ ▷ labels of current frame
  $L_{prev} = L_{Det}^{high}[i-n:i-1]$ ▷ labels of previous n frames
  $Track_{confm}, Track_{miss} \leftarrow Filter_{Kalman}(L_{curr}, L_{prev})$
  for each track in $Track_{miss}$ **do**
    Assign l_{pred} as predicted label of track
    for each label l in $L_{Det}^{low}[i]$ **do**
      $t_{IoU}^{l} \leftarrow CalculateIoU(l_{pred}, l)$
      if $t_{IoU}^{l} > T_{IoU}$ and $l[cls] == l_{pred}[cls]$ **then**
        $Avg(l) \leftarrow CalculateAvgLabel(l_{pred}, l)$
        Add $Avg(l)$ in L_{fwd}
      end if
    end for
  end for
end for

▷ retrieve labels via backward recovery
for each frame i in range of $len(L_{Det})$ and $len(L_{Det}) - m$ **do**
  $L_{curr} = L_{Det}^{high}[i]$ ▷ labels of current frame
  $L_{fut} = L_{Det}^{high}[i+1:i+m]$ ▷ labels of future m frames
  $Track_{confm}, Track_{miss} \leftarrow Filter_{Kalman}(L_{curr}, L_{fut})$
  for each track in $Track_{miss}$ **do**
    Assign l_{pred} as predicted label of track
    for each label l in $L_{Det}^{Low}[i]$ **do**
      $t_{IoU}^{l} \leftarrow CalculateIoU(l_{pred}, l)$
      if $t_{IoU}^{l} > T_{IoU}$ and $l[cls] == l_{pred}[cls]$ **then**
        $Avg(l) \leftarrow CalculateAvgLabel(l_{pred}, l)$
        Add $Avg(l)$ in L_{bkw}
      end if
    end for
  end for
end for
$L_{recov} = NMS(L_{Det}^{high}, L_{fwd}, L_{bkw})$

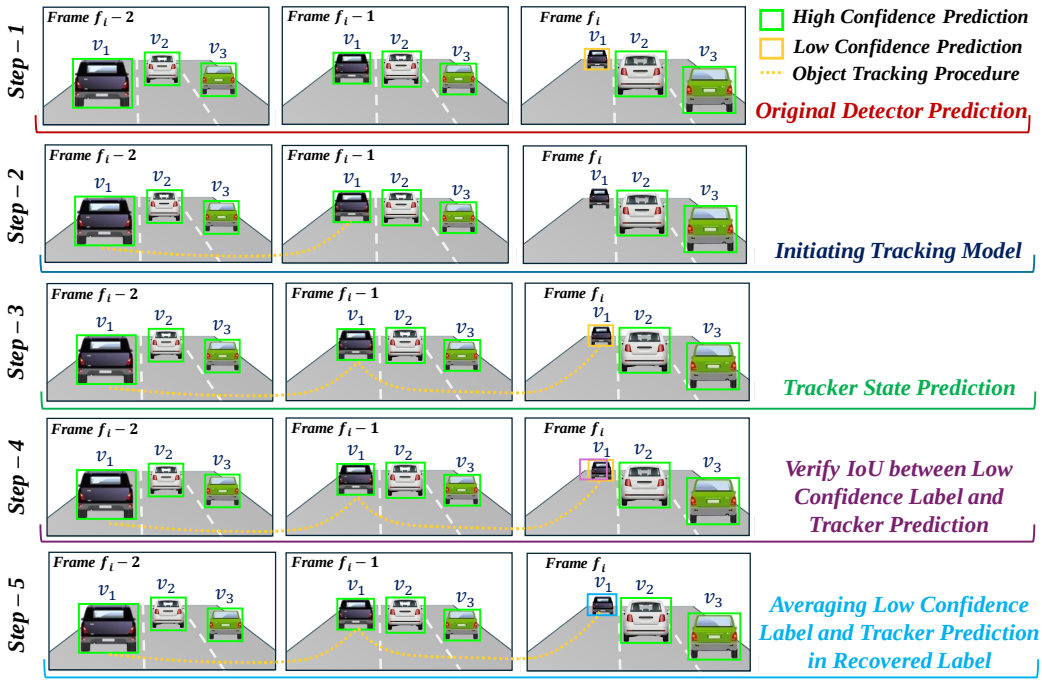


Figure 5. Procedure of recovering the poor-confidence pseudo labels.

3.2.4. Bi-Directional Label Recovery Algorithm

Algorithm 1 describes the detailed procedure of recovering the lost predictions by utilizing tracking information. From lines 1 to 4, it runs a loop for all frames in the input video sequence to execute the detector (line 2) and saves the detection results (line 3) of each frame i in high-confidence $L_{Det}^{high}[i]$ and low-confidence $L_{Det}^{Low}[i]$ label lists using a predefined confidence threshold. From lines 5 to 19, it runs a loop for each frame i to recover the predicted labels in the forward direction. In line 6, for the sake of simplicity, we define L_{curr} as a list of high-confidence labels predicted for the current frame i ($L_{curr} = L_{Det}^{high}[i]$). Similarly, in line 7, we denote l_{prev} as the predicted labels of n previous frames ($L_{prev} = L_{Det}^{high}[i-n{:}i-1]$). Then, we pass L_{curr} and L_{prev} on to the Kalman filter to obtain two track lists, such as a list of confirmed $Track_{confm}$ and list of missed tracks $Track_{miss}$. The list of confirmed tracks includes all the tracks which have a detection associated with them in the current frame i. If a tracklet has no detection label associated with the current frame i, such a tracklet is considered a missed tracklet. In lines 9 to 18, we run a loop for each track inside the $Track_{miss}$ to verify if the predicted label of tracker L_{pred} is matching with any label l in $L_{Det}^{low}[i]$ list. In lines 11 to 17, we calculate IoU score t_{IoU}^{l} between l_{pred} and low-confidence label l of $L_{Det}^{low}[i]$ for frame i. Then, we check if $t_{IoU}^{l} \geq T_{IoU}$ and the class IDs of the predicted label and the low-confidence label are the same. If both conditions are true, in line 14, we calculate the average label $Avg(l)$ using l_{pred} and l. Finally, we add $Avg(l)$ in the list of forward recovered labels defined as L_{fwd} (shown in Figure 6b). Similarly, in lines 20–34, we run a loop to recover the predictions using backward tracking. We use the information of current frame i and m future frames ($f_{(i+1)}, f_{(i+2)}, f_{(i+3)}, ..., f_{(i+m)}$) to recover the labels. The recovered labels are saved in the backward recovered labels list L_{bkw} as shown in Figure 6c.

The final step of this algorithm in line 35 is the Non-Maximal Suppression (NMS) of high-confidence detections L_{Det}^{high}, forward recovered labels L_{fwd} and the backward recovered labels L_{bkw}. The recovered labels L_{recov} are relatively more accurate pseudo labels. For an example result of Algorithm 1, Figure 6 illustrates that vehicle objects occluded by wiper are recovered by the proposed bi-directional pseudo label recovery procedure. Our recovery algorithm is also useful in recovering the missed objects in challenging scenarios.

Algorithm 1 can recover the pseudo labels that would have been missed due to rainwater on the windshield and the glare of the sun in Figure 7a and Figure 7b, respectively. These examples prove that the proposed method is an effective approach to generating accurate labels in challenging situations for the ADAS systems.

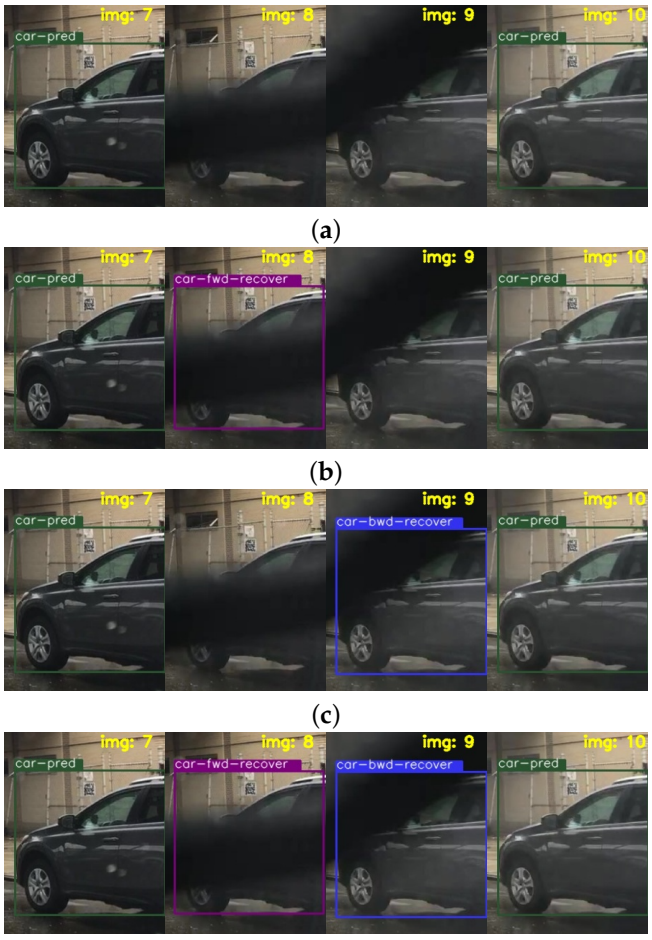
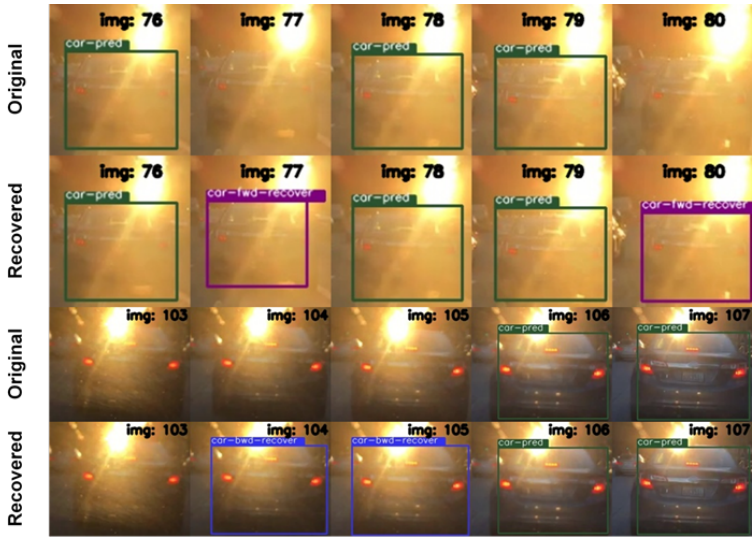


Figure 6. Example of recovering label through the forward and backward tracking: (**a**) The frames are passed through an object detector, and the pseudo labels are saved in a tracking buffer. There is a car occluded by the wiper. (**b**) The missing label for that occluded car in frame 9 is recovered by forward recovery. (**c**) The missing label in frame 8 is recovered by backward recovery. (**d**) The labels recovered by forward and backward recoveries are merged with the original predicted labels using NMS. The quality of the final predictions obtained by NMS are higher than that of the original predictions.



Figure 7. *Cont.*

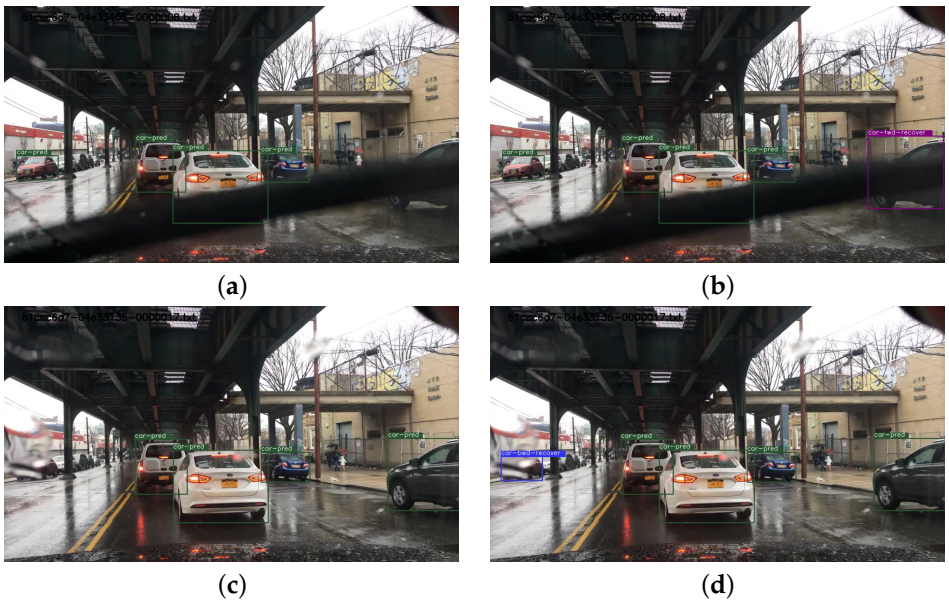


(**b**)

Figure 7. The figures show the recovered labels of occluded or distorted objects. (**a**) shows a car distorted by rainwater accumulated on the camera, (**b**) shows the scenario where the sun glare makes it difficult for the model to recognize an object. In each of these scenarios, the proposed method recovers the labels in forward (purple) and backward (blue) directions.

4. Experiment Results

The proposed method can be applied to any object detector model and any object tracker. To demonstrate the effectiveness of the proposed method, we integrate it into a popular detector called YOLOv5-m [7] and a well-known object tracker called DeepSORT [39]. We evaluate the performance of object detection before and after self-training in terms of mean average precision (mAP) at a 50% IoU threshold on the BDD100K dataset [11]. We run the simulations and experiments on a Linux-based computer equipped with 8 NVIDIA RTX 3090 GPUs and 256 GB memory size. The proposed method is applied as offline post processing to produce pseudo labels, followed by the self-training stage. Figure 8 shows the original detection results by the object detector and the boxes recovered using the proposed method.



(**a**) (**b**)

(**c**) (**d**)

Figure 8. The (**a**) image shows a vehicle occluded by the windscreen wiper. The proposed method successfully recovers the partially occluded vehicle in (**b**) that the object detector initially missed.

Similarly in (**c**), the vehicle is distorted by rainwater on the windscreen, causing the detector to fail in detecting it. The proposed method then recovers the vehicle in (**d**).

We compared the performance results of the proposed method to the performance of so-called ASTOD [30], a reference method which introduces a self-training concept with pseudo label generation.

4.1. Datasets

We used a few public datasets including BDD 100K (Berkeley Deep-Drive 100K) and Waymo datasets during the experiments as well as a set of unlabeled YouTube videos for the self-training task. The BDD dataset [11] consists of 100,000 videos. Each video is about 40 s long, 1280 × 720 (HD format), and 30 fps. There are a total of 120 million images in the dataset. The videos also come with GPS/IMU sensor data recorded by cellphones to show the rough driving trajectories. The videos are collected in multiple places, including New York, Berkley, San Francisco, and Bay Area in the US. The videos cover different weather conditions, including sunny, overcast, and rainy, as well as different times of day including morning, daytime, sunset and nighttime. Figure 9 shows the number of images in each class. Based on extensive evaluation, it is discovered that the BDD dataset provides advantages in self-training object detection models, which is attributed to its large, diverse collection of annotated images and video sequences, realistic scenarios, and rich annotations for various object classes.

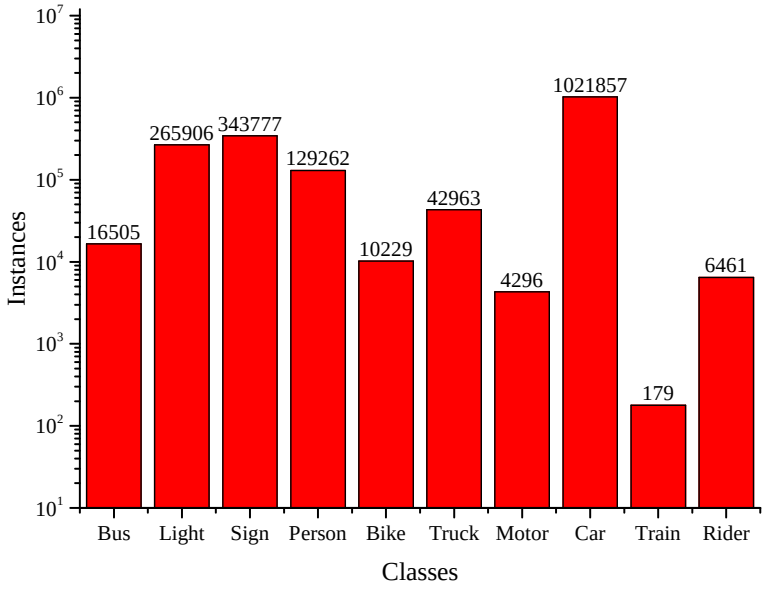


Figure 9. Statistics of different types of objects in the BDD dataset [11].

The Waymo Open Dataset is composed of 1000 segments. Each segment is 20 s long, collected at 10 Hz (100,000 frames) in diverse geographies and conditions. The objects are annotated for four classes: vehicles, pedestrians, cyclists, signs. There is a total of 11.8 million 2D bounding boxes along with the tracking IDs included in the Waymo dataset. The diverse data are collected in San Francisco, Mountain View, Los Angeles, Detroit, Seattle, and Phoenix. The data include the suburban as well as downtown areas and day and nighttime images as well.

In addition, to evaluate the methods with realistic self-training scenario, we used unlabeled videos from YouTube. We used a few driving videos provided by Seoul Walker [45]. The videos include driving roads in Seoul city during the day and nighttime.

4.2. Self-Training Scenarios

We show three experiments. In all three experiments, we used the Waymo dataset in the pre-training stage. We used a random selection of 10% (13,800 frames), 20% (27,600 frames) and 100% (138,000 frames) of the full dataset to train for multiple scenarios to observe the effect of the self-training.

On the other hand, in the self-training process, we used three different unlabeled datasets: the BDD dataset without labels, the Waymo dataset without labels, and using YouTube videos that have no labels. From each dataset, we used sequential 100,000 unlabeled frames, divided into 10 subsets. Each subset contains 10,000 unlabeled sequential frames. The pre-trained model generates pseudo labels for one split of unlabeled data (10,000 frames) at a time. The bi-directional recovery method is repeatedly conducted for 10 rounds to show the gradual improvement in self-training in a realistic scenario of ADAS or self-driving.

4.3. Object Tracker Selected for Experiment

We integrated DeepSORT [39] into the platform which includes an object detector, object tracker, and bi-directional pseudo label recovery algorithm. DeepSORT provides the robust tracking and identity management details, which allows us to maintain accurate object trajectories in dynamic environments. Compared to other object trackers, DeepSORT uses a compact CNN feature extractor (ResNet18), and thus it is an extremely lightweight tracker. The usage of both the distance and features metrics enables it to provide accurate tracking results.

4.4. Self-Training Evaluation with Waymo Dataset

We performed self-training experiments to compare the model with the existing method [30] . Table 1 shows that the proposed method outperforms the existing method [30] and gives the highest mAP@50 score. In Table 1, we start with training the object detector model [7] with a subset of the labeled Waymo dataset [12]. We perform initial training with random selection of 10% (13,800 frames) and 20% (27,600 frames) of the Waymo training dataset which consists of 138,000 total training frames. The pre-trained model is used in the first round of self-training process. A total of 10 rounds of self-training process are conducted. In each round, the following procedure is conducted. (1) A set of 10,000 unlabeled sequential frames is randomly selected from the Waymo dataset. (2) The current detection model predicts bound boxes on the selected 10,000 frames. (3) The proposed bi-directional recovery method improves the pseudo labels from the bounding boxes of all frames. (4) The detector model is self-trained with a combination of initial pre-training data with ground truth and the recovered pseudo labeled data. (5) The self-trained model is then evaluated on the Waymo validation dataset consisting of 5]k random images. (6) If the mAP score has improved, the model is updated. Otherwise, the self-trained model is discarded, and the older model is used for the next round of self-training. The above procedure is repeated for all 10 rounds. Table 1 shows that the proposed self-training method with forward recovery as well as with bi-directional recovery surpasses the results of the previous methods. For the case of pre-training with 10% of the dataset, the proposed bi-directional recovery method provides an improvement of mAP@50 by 8.7%, 9.2%, and 9.9% compared with the pre-training detector, while ASTOD [30] gives an improvement of 8.1%. For the case of pre-training with 20% of the dataset, the proposed bi-directional recovery method provides an improvement of mAP@50 by 2.1%, 2.3%, and 2.9% compared with the pre-training detector, while ASTOD [30] gives an improvement of only 1.7%. Our experiment also shows that self-training with pseudo labels generated using a constant threshold does not produce the best results. As many detected objects with a poor confidence are lost when using a constant threshold, the important information is lost, resulting in a lower mAP score. Using the recovery algorithm for only one direction, either forward or backward, does not produce the best results, as not all the information is

recovered using only one direction. Most of the information can be recovered by using the proposed bi-directional recovery.

Table 1. Comparison of Yolov5-m mAP score, initially trained with 10% and 20% of the full Waymo dataset and then self-trained with incremental 10 k Waymo data with pseudo labels generated using the proposed method. The model is evaluated using the Waymo 5 k validation dataset.

Method	Pre-Training Data	Recovery Method	Self-Training Data	mAP@50
Pretrained Model	10% Waymo (13,800) 20% Waymo (27,600)	N/A	-	44.6% 52.5%
ASTOD [30]	10% Waymo (13,800) 20% Waymo (27,600)	N/A	Waymo—100 k	52.7 (+8.1)% 54.2 (+1.7)%
High Confidence Pseudo Labels	10% Waymo (13,800) 20% Waymo (27,600)	N/A	Waymo—100 k	52.3 (+7.7)% 54.1 (+1.6)%
Forward Recovery (Proposed Method)	10% Waymo (13,800) 20% Waymo (27,600)	Forward	Waymo—100 k	53.3 (+8.7)% 54.6 (+2.1)%
Backward Recovery (Proposed Method)	10% Waymo (13,800) 20% Waymo (27,600)	Backward	Waymo—100 k	53.8 (+9.2)% 54.8 (+2.3)%
Bi-directional Recovery (Proposed Method)	10% Waymo (13,800) 20% Waymo (27,600)	Bi-directional	Waymo—100 k	54.5 (+9.9)% 55.4 (+2.9)%

4.5. Self-Training Evaluation with BDD Dataset

To further evaluate the proposed method, we initially trained the model with the full Waymo training dataset (138,000 images) and then used BDD100k for self-training. The self-training results surpass the baseline and the ASTOD [30] method results. Table 2 shows that the self-training method with forward recovery as well as with bi-directional recovery surpasses the results of previous methods. The proposed bi-directional recovery method provides an improvement of mAP@50 by 2.3%, 2.2% and 3.1% compared with the pre-training detector, while ASTOD [30] gives an improvement of only 0.3%. Figure 10 compares the performance of different recovery mechanisms over 10 rounds. The ASTOD and high-confidence methods show minor fluctuations, and forward recovery and offline BiDir recovery exhibit increasing trends, suggesting their effectiveness in improving performance over time.

Table 2. Comparison of Yolov5-m mAP score, initially trained with full Waymo training dataset and then self-trained with incremental 10 k BDD100K dataset using pseudo labels generated with the proposed bi-directional pseudo label recovery method. The models are evaluated using the Waymo 5 k validation dataset.

Method	Pre-Training Data	Recovery Method	Self-Training Data	mAP@50
Pretrained Model	Waymo (138,000)	N/A	-	57.8%
ASTOD [30]	Waymo (138,000)	N/A	BDD—100 k	58.1 (+0.3)%
High Confidence Pseudo Labels	Waymo (138,000)	N/A	BDD—100 k	57.9 (+0.1)%
Forward Recovery (Proposed Method)	Waymo (138,000)	Forward	BDD—100 k	60.1 (+2.3)%
Backward Recovery (Proposed Method)	Waymo (138,000)	Backward	BDD—100 k	60.0 (+2.2)%
Bi-directional Recovery (Proposed Method)	Waymo (138,000)	Bi-directional	BDD—100 k	60.9 (+3.1)%

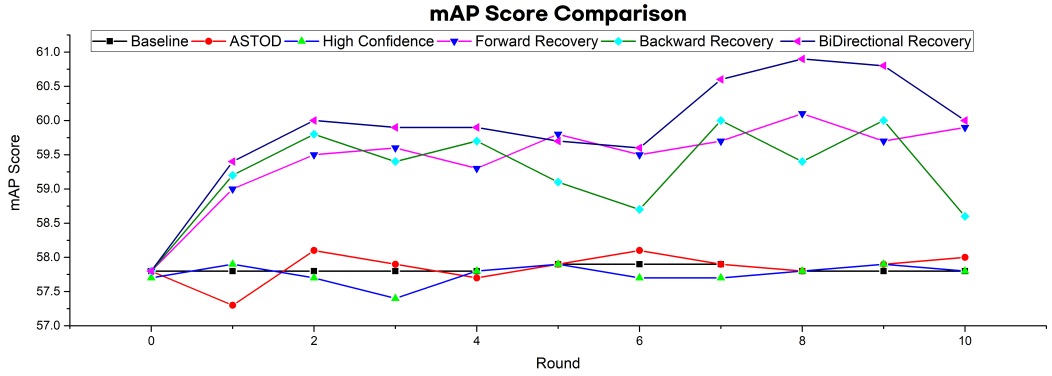


Figure 10. The mean average precision (mAP) graph illustrates the superior performance of the proposed recovery method compared to the other models. Pre-training is conducted using the Waymo dataset, followed by self-training using the BDD dataset.

4.6. Self-Training Evaluation with YouTube Videos

Table 3 shows a realistic self-training experiment using the Waymo dataset for pre-training and the labeled YouTube dataset for self-training. The first row represents the performance of a pre-trained model on Waymo data, achieving a mAP of 57.8%. ASTOD and high-confidence pseudo labels methods use the same pre-training data but yield slightly lower mAP scores of 57.7% and 57.3%, respectively. The proposed forward recovery method improves the mAP by 2% to 59.8%, while the backward recovery improves mAP by 1.7% to 59.5%. The bi-directional recovery improves mAP by 2.4% to 60.2%. These results highlight the effectiveness of the proposed self-training methods, which can gradually enhance the detection accuracy for any input video sequence. Self-training is considered crucial especially for ADAS or self-driving systems, where additional hand-labeled datasets are often hard to obtain and the cost is prohibiting by nature.

Table 3. Comparison of Yolov5-m mAP score, initially trained with full Waymo training dataset (138 k—front camera images) and then self-trained with incremental 10 k YouTube video dataset using pseudo labels generated with the proposed bi-directional pseudo label recovery method. The models are evaluated using the Waymo 5 k validation dataset.

Method	Pre-Training Data	Recovery Method	Self-Training Data	mAP@50
Pretrained Model	Waymo (138,000)	N/A	-	57.8%
ASTOD [30]	Waymo (138,000)	N/A	YouTube Videos [45]	57.7 (−0.1)%
High Confidence Pseudo Labels	Waymo (138,000)	N/A	YouTube Videos [45]	57.3 (−0.5)%
Forward Recovery (Proposed Method)	Waymo (138,000)	Forward	YouTube Videos [45]	59.8 (+2.0)%
Backward Recovery (Proposed Method)	Waymo (138,000)	Backward	YouTube Videos [45]	59.5 (+1.7)%
Bi-directional Recovery (Proposed Method)	Waymo (138,000)	Bi-directional	YouTube Videos [45]	60.2 (+2.4)%

4.7. Inference Performance Evaluation

We show that the proposed label recovery algorithm can also enhance the inference quality. In Table 4, we apply the recovery method after running the inference of YOLOv5-m using the BDD dataset. The baseline YOLOv5m [7] (without label recovery) obtains an mAP@50 of 47.32%. When the forward recovery only is applied, the detector achieves an mAP@50 of 47.64%, improving by 0.32%, while sacrificing only 3 ms in computation time. When the backward recovery only is applied, the detector improves the mAP@50 by 0.48%.

On the other hand, when both forward and backward recoveries are applied, the detector improves the mAP@50 by 0.52% at the cost of additional computation time of 7 ms. These results show that the proposed method can be integrated with any existing object detector models to improve their accuracy without using self-training with pseudo labels.

Table 4. Inference performance comparison.

Object Detector	Recovery Algorithm	Computation Time (3090 RTX)	Computation Time (Nvidia Jetson)	mAP@50
Yolov5-m (baseline)	N/A	30 ms/Image	97 ms/Image	47.32%
Yolov5-m	Forward	33 ms/Image	107 ms/Image	47.64% (+0.32)
Yolov5-m	Backward	33 ms/Image	107 ms/Image	47.80% (+0.48)
Yolov5-m	Bi-Directional	40 ms/Image	129 ms/Image	47.84% (+0.52)

5. Conclusions

In this study, a novel approach was proposed to improve the performance of the object detector model through the use of forward and backward recovery techniques. The experimental results demonstrate that the method achieves significant improvements in terms of accuracy during self-training incorporated with forward and backward recovery techniques. Unlike ASTOD, which typically discards low-confidence predictions and potentially loses valuable information, this approach refines and utilizes these predictions through bi-directional tracking. This method reduces false negatives and enhances the overall quality of pseudo labels. While ASTOD primarily relies on confidence-based filtering without leveraging future information, the bi-directional tracking algorithm uses both past and future frames to validate and recover missing labels. This ensures a more robust recovery of pseudo labels, especially in scenarios where objects are occluded or partially visible. Empirically, the approach demonstrates a notable improvement, with a 0.52% increase in mAP@50 on the BDD100K dataset, attributed to the effective recovery and inclusion of low-confidence predictions. Furthermore, the method integrates a lightweight Kalman filter-based tracker, ensuring computational efficiency and feasibility for large-scale datasets, in contrast to the more computationally intensive techniques employed by ASTOD. The findings suggest that the proposed algorithm can be a promising approach for improving the accuracy of object detection in real-world scenarios. Overall, this study contributes to the ongoing efforts to enhance the performance of object detection algorithms and has important implications for various applications in computer vision and autonomous systems. It was also observed that the method recovered some incorrect boxes. Future work will focus on improving the recovery method to exclude false positives and further enhance the performance of the object detector model.

Author Contributions: Conceptualization, S.S., H.K. and O.U.; methodology, S.S.; software, S.S.; validation, S.S., Z.A., O.U. and H.K.; formal analysis, Z.A.; investigation, S.S.; resources, H.K.; data curation, S.S. and O.U.; writing—original draft preparation, S.S.; writing—review and editing, H.K. and O.U.; visualization, S.S. and O.U.; supervision, H.K.; project administration, H.K.; funding acquisition, H.K. All authors have read and agreed to the published version of the manuscript.

Funding: This work was supported by Regional Leading Research Center (RLRC) of the National Research Foundation of Korea (NRF) grant funded by the Korean government (MSIT) (No. 2022R1A5A8026986) and supported by Institute of Information & communications Technology Planning & Evaluation (IITP) grant funded by the Korea government (MSIT) (No.2020-0-01304, Development of Self-Learnable Mobile Recursive Neural Network Processor Technology). It was also supported by the MSIT (Ministry of Science and ICT), Korea, under the Grand Information Communication Technology Research Center support program (IITP-2022-2020-0-01462) supervised by the IITP (Institute or Information & communications Technology Planning & Evaluation). This work also was supported by the Starting growth Technological R&D Program (S3318502) funded by the Ministry of SMEs and Startups (MSS, Korea).

Data Availability Statement: We used the BDD, Waymo, and YouTube video datasets. All of these datasets are publicly available. The link to each dataset is given in the reference section.

Conflicts of Interest: The authors declare no conflicts of interest. The funders had no role in the design of the study; in the collection, analyses, or interpretation of data; in the writing of the manuscript; or in the decision to publish the results.

References

1. Redmon, J.; Divvala, S.; Girshick, R.; Farhadi, A. You Only Look Once: Unified, Real-Time Object Detection. In Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Las Vegas, NV, USA, 26 June–1 July 2016.
2. Liu, W.; Anguelov, D.; Erhan, D.; Szegedy, C.; Reed, S.; Fu, C.Y.; Berg, A.C. SSD: Single Shot MultiBox Detector. In Proceedings of the European Conference on Computer Vision (ECCV), Amsterdam, The Netherlands, 11–14 October 2016.
3. Ren, S.; He, K.; Girshick, R.; Sun, J. Faster R-CNN: Towards Real-Time Object Detection with Region Proposal Networks. *IEEE Trans. Pattern Anal. Mach. Intell.* **2015**, *39*, 1137–1149. [CrossRef] [PubMed]
4. Lin, T.Y.; Goyal, P.; Girshick, R.; He, K.; Dollár, P. Focal Loss for Dense Object Detection. In Proceedings of the IEEE International Conference on Computer Vision (ICCV), Venice, Italy, 22–29 October 2017.
5. Geiger, A.; Lenz, P.; Stiller, C.; Urtasun, R. Vision meets robotics: The kitti dataset. *Int. J. Robot. Res.* **2013**, *32*, 1231–1237. [CrossRef]
6. Geiger, A.; Lenz, P.; Urtasun, R. Are we ready for autonomous driving? the kitti vision benchmark suite. In Proceedings of the KITTI Vision Benchmark Suite, Providence, RI, USA, 16–21 June 2012; IEEE: Piscataway, NJ, USA, 2012; pp. 3354–3361.
7. Jocher, G. YOLOv5 by Ultralytics. 2020. Available online: https://github.com/ultralytics/yolov5 (accessed on 3 June 2024).
8. Shah, S.; Tembhurne, J. Object detection using convolutional neural networks and transformer-based models: A review. *J. Electr. Syst. Inf. Technol.* **2023**, *10*, 54. [CrossRef]
9. Wang, Y.; Chen, X.; You, Y.; Li, L.E.; Hariharan, B.; Campbell, M.; Weinberger, K.Q.; Chao, W.L. Train in germany, test in the usa: Making 3d object detectors generalize. In Proceedings of the Making 3D Object Detectors Generalize, Seattle, WA, USA, 14–19 June 2020; pp. 11713–11723.
10. Brophy, T.; Mullins, D.; Parsi, A.; Horgan, J.; Ward, E.; Denny, P.; Eising, C.; Deegan, B.; Glavin, M.; Jones, E. A Review of the Impact of Rain on Camera-Based Perception in Automated Driving Systems. *IEEE Access* **2023**, *11*, 67040–67057. [CrossRef]
11. Yu, F.; Xian, W.; Chen, Y.; Liu, F.; Liao, M.; Madhavan, V.; Darrell, T. Bdd100k: A diverse driving video database with scalable annotation tooling. *arXiv* **2018**, arXiv:1805.04687.
12. Schwall, M.; Daniel, T.; Victor, T.; Favaro, F.; Hohnhold, H. Waymo public road safety performance data. *arXiv* **2020**, arXiv:2011.00038.
13. Blum, A.; Mitchell, T. Combining labeled and unlabeled data with co-training. In Proceedings of the COLT: Annual Workshop on Computational Learning Theory, New York, NY, USA, 24–26 July 1998; pp. 92–100.
14. Kendall, A.; Gal, Y.; Cipolla, R. Multi-Task Learning Using Uncertainty to Weigh Losses for Scene Geometry and Semantics. *arXiv* **2018**, arXiv:1705.07115
15. Tarvainen, A.; Valpola, H. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. *arXiv* **2018**, arXiv:1703.01780.
16. Englesson, E.; Azizpour, H. Consistency Regularization Can Improve Robustness to Label Noise. *arXiv* **2021**, arXiv:2110.01242.
17. Fan, Y.; Kukleva, A.; Schiele, B. Revisiting Consistency Regularization for Semi-Supervised Learning. *arXiv* **2021**, arXiv:2112.05825.
18. Zhai, X.; Oliver, A.; Kolesnikov, A.; Beyer, L. S4L: Self-Supervised Semi-Supervised Learning. *arXiv* **2019**, arXiv:1905.03670.
19. Vesdapunt, N.; Rundle, M.; Wu, H.; Wang, B. JNR: Joint-based Neural Rig Representation for Compact 3D Face Modeling. *arXiv* **2020**, arXiv:2007.06755.
20. Zhou, Z.H.; Li, M. Tri-training: Exploiting unlabeled data using three classifiers. *IEEE Trans. Knowl. Data Eng.* **2005**, *17*, 1529–1541. [CrossRef]
21. Yu, J.; Yin, H.; Gao, M.; Xia, X.; Zhang, X.; Hung, N.Q.V. Socially-Aware Self-Supervised Tri-Training for Recommendation. *arXiv* **2021**, arXiv:2106.03569.
22. Kang, K.; Li, H.; Xiao, T.; Ouyang, W.; Yan, J.; Liu, X.; Wang, X. Object Detection in Videos with Tubelet Proposal Networks. In Proceedings of the 2017 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Honolulu, HI, USA, 17–26 July 2017; pp. 889–897.
23. Crawshaw, M. Multi-Task Learning with Deep Neural Networks: A Survey. *arXiv* **2020**, arXiv:2009.09796.
24. Ruder, S. An Overview of Multi-Task Learning in Deep Neural Networks. *arXiv* **2017**, arXiv:1706.05098.
25. Gal, Y.; Islam, R.; Ghahramani, Z. Deep Bayesian Active Learning with Image Data. *arXiv* **2017**, arXiv:1703.02910.
26. Emam, Z.A.S.; Chu, H.M.; Chiang, P.Y.; Czaja, W.; Leapman, R.; Goldblum, M.; Goldstein, T. Active Learning at the ImageNet Scale. *arXiv* **2021**, arXiv:2111.12880.
27. Wang, H.; Wang, Q.; Yang, F.; Zhang, W.; Zuo, W. Data Augmentation for Object Detection via Progressive and Selective Instance-Switching. *arXiv* **2019**, arXiv:1906.00358.

28. Ghiasi, G.; Cui, Y.; Srinivas, A.; Qian, R.; Lin, T.Y.; Cubuk, E.D.; Le, Q.V.; Zoph, B. Simple Copy-Paste is a Strong Data Augmentation Method for Instance Segmentation. *arXiv* **2021**, arXiv:2012.07177.
29. Ayub, A.; Kim, H. GAN-Based Data Augmentation with Vehicle Color Changes to Train a Vehicle Detection CNN. *Electronics* **2024**, *13*, 1231. [CrossRef]
30. Vandeghen, R.; Louppe, G.; Van Droogenbroeck, M. Adaptive Self-Training for Object Detection. *arXiv* **2023**, arXiv:2212.05911.
31. Liang, X.; Liu, S.; Wei, Y.; Liu, L.; Lin, L.; Yan, S. Towards Computational Baby Learning: A Weakly-Supervised Approach for Object Detection. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 999–1007.
32. Osep, A.; Voigtlaender, P.; Luiten, J.; Breuers, S.; Leibe, B. Large-Scale Object Mining for Object Discovery from Unlabeled Video. *arXiv* **2019**, arXiv:1903.00362.
33. Misra, I.; Shrivastava, A.; Hebert, M. Watch and Learn: Semi-Supervised Learning of Object Detectors from Videos. *arXiv* **2015**, arXiv:1505.05769.
34. Singh, K.K.; Xiao, F.; Lee, Y.J. Track and Transfer: Watching Videos to Simulate Strong Human Supervision for Weakly-Supervised Object Detection. *arXiv* **2016**, arXiv:1604.05766.
35. Tang, K.; Ramanathan, V.; Fei-fei, L.; Koller, D. Shifting Weights: Adapting Object Detectors from Image to Video. In *Proceedings of the Advances in Neural Information Processing Systems*; Curran Associates, Inc.: Glasgow, UK, 2012; Volume 25.
36. Liu, G.; Zhang, F.; Pan, T.; Wang, B. Low-Confidence Samples Mining for Semi-supervised Object Detection. *arXiv* **2023**, arXiv:2306.16201.
37. Qi, C.R.; Zhou, Y.; Najibi, M.; Sun, P.; Vo, K.; Deng, B.; Anguelov, D. Offboard 3D Object Detection from Point Cloud Sequences. *arXiv* **2021**, arXiv:2103.05073.
38. Yang, B.; Bai, M.; Liang, M.; Zeng, W.; Urtasun, R. Auto4D: Learning to Label 4D Objects from Sequential Point Clouds. *arXiv* **2021**, arXiv:2101.06586.
39. Wojke, N.; Bewley, A.; Paulus, D. Simple Online and Realtime Tracking with a Deep Association Metric. *arXiv* **2017**, arXiv:1703.07402.
40. Breitenstein, M.D.; Reichlin, F.; Leibe, B.; Koller-Meier, E.; Van Gool, L. Robust tracking-by-detection using a detector confidence particle filter. In Proceedings of the 2009 IEEE 12th International Conference on Computer Vision, Kyoto, Japan, 29 September–2 October 2009; pp. 1515–1522.
41. Hua, Y.; Alahari, K.; Schmid, C. Online Object Tracking with Proposal Selection. In Proceedings of the 2015 IEEE International Conference on Computer Vision (ICCV), Santiago, Chile, 7–13 December 2015; pp. 3092–3100.
42. Breitenstein, M.D.; Reichlin, F.; Leibe, B.; Koller-Meier, E.; Van Gool, L. Online Multiperson Tracking-by-Detection from a Single, Uncalibrated Camera. *IEEE Trans. Pattern Anal. Mach. Intell.* **2011**, *33*, 1820–1833. [CrossRef]
43. Zhou, X.; Koltun, V.; Krähenbühl, P. Tracking Objects as Points. *arXiv* **2020**, arXiv:2004.01177.
44. Mclachlan, G. Mahalanobis Distance. *Resonance* **1999**, *4*, 20–26. [CrossRef]
45. SeoulWalker. SeoulWalker YouTube Channel. 8 May 2024. Available online: https://www.youtube.com/watch?v=ujIy2cFcapY (accessed on 3 June 2024).

Disclaimer/Publisher's Note: The statements, opinions and data contained in all publications are solely those of the individual author(s) and contributor(s) and not of MDPI and/or the editor(s). MDPI and/or the editor(s) disclaim responsibility for any injury to people or property resulting from any ideas, methods, instructions or products referred to in the content.