Object Detection with Self-Supervised Scene Adaptation

Zekun Zhang[1] Minh Hoai[1,2]
[1]Stony Brook University, Stony Brook, NY 11794, USA
[2]VinAI Artificial Intelligence Application and Research JSC, Hanoi, Vietnam
{zekzhang,minhhoai}@cs.stonyrbook.edu

Abstract

This paper proposes a novel method to improve the performance of a trained object detector on scenes with fixed camera perspectives based on self-supervised adaptation. Given a specific scene, the trained detector is adapted using pseudo-ground truth labels generated by the detector itself and an object tracker in a cross-teaching manner. When the camera perspective is fixed, our method can utilize the background equivariance by proposing artifact-free object mixup as a means of data augmentation, and utilize accurate background extraction as an additional input modality. We also introduce a large-scale and diverse dataset for the development and evaluation of scene-adaptive object detection. Experiments on this dataset show that our method can improve the average precision of the original detector, outperforming the previous state-of-the-art self-supervised domain adaptive object detection methods by a large margin. Our dataset and code are published at https://github.com/cvlab-stonybrook/scenes100.

1. Introduction

The need to detect objects in video streams from stationary cameras arises in many computer vision applications, including video surveillance and autonomous retail. In general, different applications require the detection of different object categories, and each computer-vision-based product will have its own detector. However, for a specific product, there is typically a *single* detector that will be used for *many* cameras/scenes. For example, a typical video surveillance product would use the same detector to detect pedestrians and vehicles for network cameras installed at different locations. Unfortunately, a single detector might not work well for all scenes, leading to trivial and unforgiving mistakes.

This fundamental problem of many computer vision products stems from the limited generalization power of a single model, due to limited training data, limited model capacity, or both. One can attempt to address this problem by using more training data, but it will incur additional cost for data collection and annotation. Furthermore, in many cases, due to the low latency requirement or the limited computing resources for inference, a product is forced to use a very lightweight network, and this network will have limited representation capacity to generalize across many scenes.

In this paper, instead of having a single scene-generic detector, we propose using scene-specific detectors. This yields higher detection performance as each detector is customized for a specific scene, and allows us to use a lightweight model without sacrificing accuracy as each detector is only responsible for one scene.

Obtaining scene-specific detectors, however, is very challenging. A trivial approach is to train a detector for each scene separately, but this requires an enormous amount of annotated training data. Instead, we propose a self-supervised method to adapt a pre-trained detector to each scene. Our method records the unlabeled video frames in the past, uses the trained detector to detect objects in those frames, and generates augmented training data based on those detections. Although the detections made by the pre-trained model can be noisy, they can still be useful for generating pseudo annotated data. We further extend those pseudo bounding boxes by applying object tracking [2, 57] along the video timeline, aiming to propagate the detections to adjacent frames to recover some of the false negatives not returned by the detector. We also use multiple detectors to obtain the pseudo labels and train the detector in a cross-teaching manner, taking the advantage of the ensemble of models [13, 24].

Exploiting the stationary nature of the camera, we propose two additional techniques to boost the detection performance: location-aware mixup and background-augmented input. The former is to generate more samples during training through object mixup [76] that contains less artifacts, based on the aforementioned pseudo boxes generated from detection and tracking. The latter involves estimating the background image and fusing it with the detector's input.

In short, the main contribution of our paper is a novel framework that utilizes self-supervision, location-aware ob-

ject mixup, and background modeling to improve the detection performance of a pre-trained object detector on scenes with stationary cameras. We also contribute a large scale and diverse dataset for the development of scene adaptive object detection, which contains sufficient quantity and quality annotations for evaluation.

2. Related Work

Despite much recent improvement in object detection research [3,5,6,8,12,18,19,21,25,28,31,36,40,43,45,47–50, 53–56,64,65,67–69,77,79], trained detectors still encounter problems with domain shift or domain gap. Scene adaptive object detection can be viewed as a special case of domain adaption, in which an object detector trained on a fully-supervised source domain is adapted to a target domain. Most research in this direction focuses on semi-supervised, weakly-supervised, or self-supervised adaptation. In addition, source-free adaptation [26, 33, 35, 71] has been proposed to address the situation where the source domain data is unavailable during adaptation.

Self-labeling methods [11,27,29,33,37,44,46,57,60,73] uses the teacher-student setup from semi-supervised image classification [30, 62]. A teacher detector trained on the source domain generates pseudo bounding boxes on the target domain images, and a student model is trained with those boxes to improve its performance on the target domain. Techniques such as weak/strong augmentation [37,44,60], knowledge distillation [11], and weight averaging [33, 37] have been used to deal with noisy pseudo labels from the teacher detector due to the domain gap. Our proposed pseudo-labeling method uses two teacher models trained on the source domain to train a student model, and their pseudo boxes are aggregated and refined. We further apply tracking as in [57], but with both forward and backward directions, to extend the pseudo labels. The pseudo boxes also form the basis for later location-aware mixup and dynamic background extraction steps.

Domain alignment methods [7, 11, 22, 33, 34, 58, 72, 78] aim to reduce the domain gap by enforcing the models to output similarly on the source and target domain at image, proposal, or instance levels. They use domain adversarial learning through gradient reversal [15, 16] or graph matching [34] for domain alignment. This approach involves adding domain alignment losses to existing object detection models and is complementary to our proposed method.

Another seemingly related research area is continual or incremental learning. Scene adaptive and domain adaptive object detection are in between task-continual learning [38] and data-continual learning [52], for that the object categories of the source and target domains are the same, but the data distribution differs. It is possible to formulate scene adaptive object detection as an online learning problem, but this approach would require additional annotations.

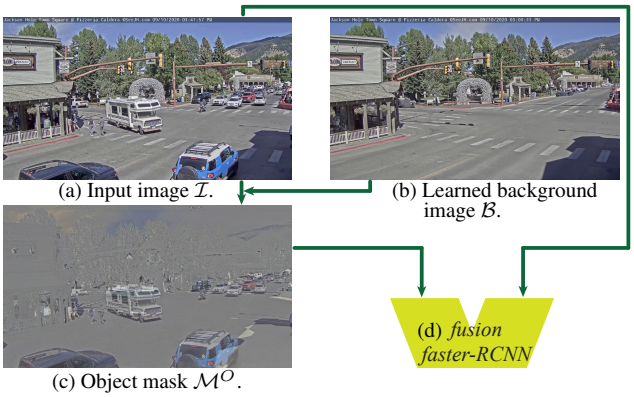


Figure 1. Inference flow of the proposed detector that can adapt to a specific scene. The background image and fusion faster-RCNN model are learned during training, discussed in Sec. 3.

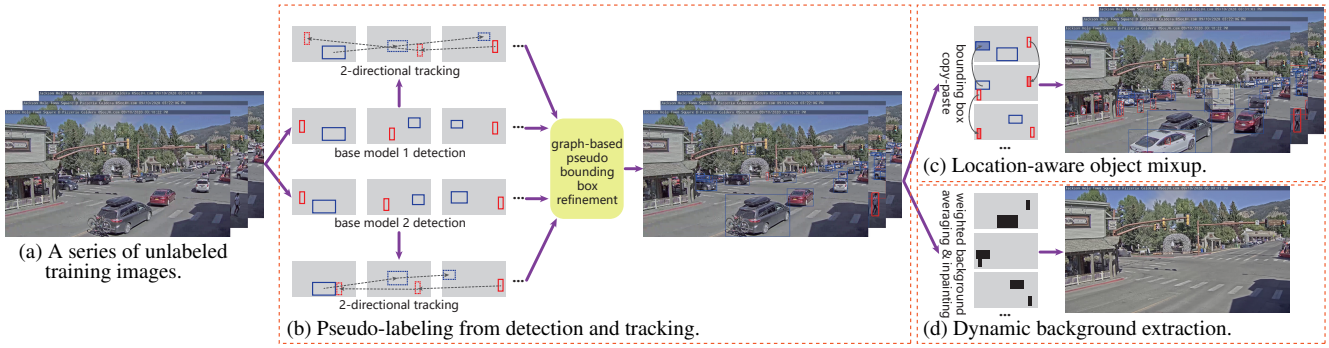
3. Methods

For an object detector pre-trained on a source dataset referred as the base model, we aim to improve its performance on video stream from a stationary camera in a given scene, with unlabeled videos captured by the same camera in the past as adaptation training samples. The categories of object of interest in the scene are also present in the source dataset, but the data distribution is shifted. The base model is adapted to each scene independently.

Assuming the distribution shift is moderate, the base detector can generate partially decent bounding boxes on the unlabeled frames. We use them as pseudo-labels to improve the base model itself. Many objects are ignored by the base detector, as their appearance can be very different from the source dataset. However, videos contain temporal correlation information, making it possible to apply object tracking in both time directions to recover some of those false negatives. The pseudo-labels are then refined to remove low-confidence boxes and duplicates. We refer to this process of obtaining pseudo bounding boxes as pseudo-labeling.

As data augmentation is beneficial for training on less reliable labels [51], we take advantages of object mixup to generate new sample images. Since the background is stationary, those images contain fewer artifacts, which is beneficial for detectors. We further adopt the idea of ensemble models [13, 24] by using two base detectors to obtain the pseudo labels, for they can be complementary to each other. The pseudo boxes are used to train one base model, which we refer as cross-teaching.

Since the job of a detector is to separate foreground objects from the background, having information about the background is advantageous. In a stationary camera video with moving objects, different parts of the background are covered by objects in different frames, making it possible to model the complete background. We use the aforementioned pseudo bounding boxes to mark the uncovered parts of the background in each frame. Combining the partial



(a) A series of unlabeled training images.

(b) Pseudo-labeling from detection and tracking.

(c) Location-aware object mixup.

(d) Dynamic background extraction.

Figure 2. Training flow of the proposed self-supervised scene adaptation object detection framework with sample images. The mixup training images with pseudo bounding boxes and background image are used in the fusion training. The details are discussed in Sec. 3.

background from a sequence of frames can give an accurate and complete background model. We then modify the Faster-RCNN [56] architecture to fuse the background with the input image to improve detection performance.

The overall structure and data flows of the proposed methods are shown in Fig. 1 and Fig. 2. Our proposed methods are highly flexible and can be configured to trade-off between performance and speed. Object tracking can be bypassed to reduce the time to get the pseudo-bounding boxes. Mixup and object mask fusion can also be disabled for faster training and inference. In Sec. 5.5 we will show ablation study results on how those components affect the performance of the adapted models. We now describe each components in more details. For consistency, the images are all from the first video in our collected dataset, which will be described in Sec. 4.

3.1. Base Detection Models on Source Domain

We use Faster-RCNN [56] in our experiments for its high precision and flexible modular architecture. Among the techniques proposed for self-supervised scene adaptation, both pseudo-labeling and mixup can be directly applied to any object detection model. The object mask fusion technique requires modification to the network architecture, but it can be easily extended to most types of detectors by adding a parallel branch to the existing network.

The object labels in MSCOCO training set are remapped to *person* and *vehicle* as described in Sec. 4.2. All objects other than *person*, *car*, *bus*, and *truck* are discarded. We take Faster-RCNNs pre-trained on the original MSCOCO training set, and finetune them on the remapped training set. We choose two models with ResNet-50 and ResNet-101 backbones as base models, referred as **M1** and **M2**.

3.2. Pseudo Labels from Detection and Tracking

We take the frames from the first training portion of each video, and adopt the idea of detection and tracking similar to [57], then apply our proposed bounding box refinement and cross-teaching schemes. Base detectors described in Sec. 3.1 are applied on the frames, and any bounding box with confidence score higher than λ_{det} is kept as a valid pseudo box.

Single-object trackers are initialized with detected objects having confidence scores higher than λ_{sot}. We use DiMP-50 [2] to track the objects in both forward and backward directions in the video timeline, and the tracked object bounding boxes are used as pseudo boxes. The reason for tracking in both directions is that if an object is moving towards the camera and appears bigger by time, backward tracking is more accurate because the initial bounding box contains more detailed texture. To prevent drifting, the maximum length of each track is capped at two seconds.

Bounding box refinement is applied to eliminate duplicated boxes of the same object from combining pseudo bounding boxes from detector and tracker. Each candidate box in the same frame is regarded as a graph node. If two boxes have the same object label, and their IoU is above a threshold λ_{iou}, then an edge is added to connect them. We assume that each connected component of the graph represents one single object, and the nodes in the component are duplicated. We only keep the node with the highest degree (*i.e.*, the node with the highest number of connected edges). An example of refinement results is shown in Fig. 2b. Refinement can effectively remove duplicated bounding boxes. However, it can neither remove false positives nor recover false negatives.

Both M1 and M2 are used in detection and tracking. The pseudo bounding boxes from both base models are then refined jointly, and the refined boxes are used to train M2 in adaptation. We refer this setting as cross-teaching.

3.3. Location-Aware Object Mixup

Mixup [76] is originally proposed as a simple but powerful data augmentation method in for image classification. It is quickly improved by many modifications [1, 23, 32, 66, 75]. Mixup is also used in object detection tasks [4, 75], where it can be applied at image or instance levels. During adaptation training, a frame with its pseudo bounding

(a) Random mixup. (b) Location-aware mixup.

Figure 3. An example of location-aware object mixup. Compared to random mixup, less artifacts are introduced.

boxes has the probability p_{mixup} of being paired with another randomly select source frame for mixup. A portion r_{mixup} of the pseudo bounding boxes in the source frame are cropped out and pasted onto the original frame, along with their locations and labels. If a pasted box covers more than α_{cover} of a pseudo box in the original frame, the pseudo box will be excluded from the pseudo labels. An example of this location-aware object mixup is shown in Fig. 2c and Fig. 3. Since the background is stationary, we propose to use location-aware mixup instead of the original random mixup to reduce the amount of artifacts in the generated mixup images and to keep the location distribution of the objects in the scene. This will be shown to be beneficial in the experiments.

3.4. Dynamic Background Extraction

For a video frame $\mathcal{I}$ of dimension $W \times H$ associated with a set of K pseudo-annotated object bounding boxes $\{(x_k^1, y_k^1, x_k^2, y_k^2) | k = 1 \dots K\}$, a background mask $\mathcal{M}$ of the same dimension as $\mathcal{I}$ can be constructed as follows. For each location (x, y) in $\mathcal{M}$, we set $M[x, y] = 0$ if (x, y) is inside of any pseudo-annotated bounding box and 1 otherwise. Then for a sequence of frame-mask pairs $\{(\mathcal{I}_l, \mathcal{M}_l) | l = 1 \dots L\}$, the background image is determined as

$$\mathcal{B} = \frac{\sum_{l=1}^{L} \mathcal{I}_l \otimes \mathcal{M}_l}{\sum_{l=1}^{L} \mathcal{M}_l}, \tag{1}$$

where $\otimes$ is the pixel-wise multiplication operator.

There might exist a location (x', y') that lies inside an object bounding box in every image, *i.e.*, $\mathcal{M}_l[x', y'] = 0$. In this case, the background at this location is never observed and its pixel value cannot be determined. In this case, we "guess" the background value by inpainting inside such areas using the inpainting algorithm of [63].

Fig. 2d shows an example background, which is reasonably accurate. Although in each video, the camera perspective is fixed, the background can change overtime due to some factors such as illumination change. To incorporate this, the background extraction is operated every T_{bg} seconds, giving a dynamic background modeling.

We construct an object mask as an additional input modality for the detector, to utilize the extracted background to improve detection performance during adaptation. For a frame $\mathcal{I}$ and its corresponding background model image $\mathcal{B}$, we define its object mask image as

$$\mathcal{M}^O = (\mathcal{I} - \mathcal{B} + 1) \times 0.5, \tag{2}$$

where we assume the pixel values in both $\mathcal{I}$ and $\mathcal{B}$ have been normalized to the range $[0, 1]$. An object mask example is shown in Fig. 1c, in which objects are clearly separated from the background. Note that since $\mathcal{I}$, $\mathcal{B}$, and $\mathcal{M}^O$ are linearly dependent, no information is lost.

Before the adaptation training process, the base detector needs to be modified to use object mask input modality and trained on the MSCOCO training set with background images. However, dynamic background extraction on MSCOCO is impossible since it only consists of static images. Therefore, we rely on the object mask annotations in the dataset and apply the same inpainting algorithm [63] to generate the background image for each training image.

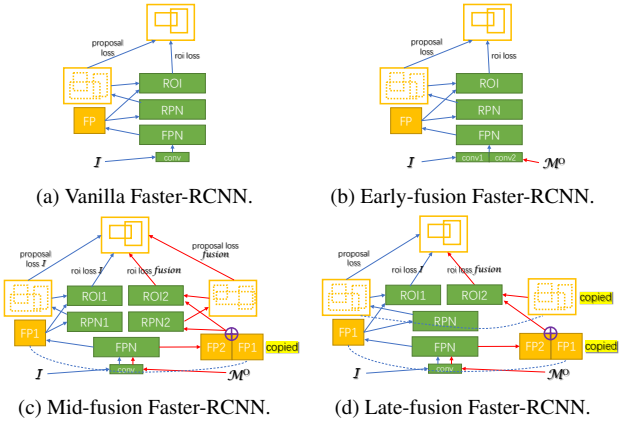
3.5. Object Mask Fusion

The architecture of Faster-RCNN [56] is depicted in Fig. 4a. First, a CNN backbone with a feature pyramid network (FPN) [39] is utilized to extract the Feature Pyramid (FP). The first layer in the CNN that takes the input image is also shown here. Then, an RPN [39] is employed to produce object bounding box proposals. Lastly, an ROI head [19,20] is used to assign object labels and refine the bounding boxes on the pooled feature maps of the proposals. The entire network is trained utilizing two sets of losses: localization loss and objectness loss from RPN, and localization loss and classification loss from the ROI head. The fusion models explained below start with the base model that is trained on the remapped MSCOCO training set.

Early-fusion. The input to the Faster-RCNN network is the stacked image $[\mathcal{I}; \mathcal{M}^O]$, with the two modalities stacked along the color channels, as shown in Fig. 4b. To enable the network to take a 6-channel image as input instead of the vanilla 3-channel image, the first layer of the backbone CNN is duplicated and stacked. The convolution kernel weights are first copied, then halved. This ensures that the feature map after the first layer conv1|conv2 is the same as in the vanilla model when $[\mathcal{I}; \mathcal{I}]$ is fed into it, providing a smooth start for adaptation training. This modification introduces only a small number of additional model parameters, and the increase in computational cost is negligible.

Mid-fusion. Both $\mathcal{I}$ and $\mathcal{M}^O$ are fed into the FPN in parallel, which yields two feature pyramid FP1 and FP2, as shown in Fig. 4c. Two sets of RPNs and ROI heads are used in parallel, and their initial weights are copied from the base model. FP1 is used in the left branch with RPN1 and ROI1, which is the same as the vanilla model. In the right branch with RPN2 and ROI2, the input feature pyramid is the fusion of FP1 and FP2:

$$\text{FP}_{fusion} = \text{FP1} \oplus \text{FP2}. \tag{3}$$



(a) Vanilla Faster-RCNN. (b) Early-fusion Faster-RCNN.

(c) Mid-fusion Faster-RCNN. (d) Late-fusion Faster-RCNN.

Figure 4. Illustration of different fusion models for Faster-RCNN. Network modules are indicated by green, and feature maps and outputs are indicated by yellow.

The fused feature pyramid has the same dimension as both FP1 and FP2. The fusion operator can be either non-parametric such as average pooling, or parametric such as convolution layers. The whole network is trained with losses from both branches

$$\mathcal{L}^{mid} = (1-\alpha_{mid})(\mathcal{L}^{\mathcal{I}}_{rpn}+\mathcal{L}^{\mathcal{I}}_{roi})+\alpha_{mid}(\mathcal{L}^{fusion}_{rpn}+\mathcal{L}^{fusion}_{roi}), \quad (4)$$

where α_{mid} is the weights of different losses.

Late-fusion is similar to mid-fusion as shown in Fig. 4d. The key difference is that only the ROI head is branched. The region proposals are only generated from FP1, and the fused feature pyramid is only used in ROI head in the right branch. As a result, the total loss is

$$\mathcal{L}^{late} = \mathcal{L}^{\mathcal{I}}_{rpn} + (1-\alpha_{late})\mathcal{L}^{\mathcal{I}}_{roi} + \alpha_{late}\mathcal{L}^{fusion}_{roi}. \quad (5)$$

Both mid-fusion and late-fusion introduce significant additional parameters and computation. $\mathcal{I}$ and $\mathcal{M}^O$ need to pass the backbone, requiring more computation even at inference time. The quantitative comparison of inference speed of different fusion models can be found in the supplementary material.

To train the fusion models in adaptation, we first fine-tune the base models on MSCOCO training set with the inpainted background models described in Sec. 3.4 until convergence. During evaluation, we use the last background image to obtain the object masks, as it is the closest to the evaluation images on the timeline.

4. Scenes100 Dataset

For our proposed scene-adaptive object detection problem, there is no existing dataset with long enough videos of stationary backgrounds for the development and evaluation of self-supervised adaptation techniques. We therefore collected a new dataset called **Scenes100**, which will be described in this section.

Figure 5. Some annotated frames from the evaluation portions. The green mask shows the part of the frame that is excluded for annotation and evaluation. Objects are labeled by bounding boxes. Please see the supplementary material for detailed descriptions.

4.1. Data Collection

We used keywords such as "live view", "street camera", and "live webcam" to search for live streams on YouTube. We look for streams that are of decent quality and have fixed camera perspective. Around 200 candidate videos were recorded between September 2020 and February 2022. We picked 100 videos to compile our Scenes100 dataset, ensuring that each video is longer than two hours and has minimum width and height of 720 pixels. We also aimed for a dataset with high diversities. The 100 videos of Scenes100 were recorded in a variety of 16 countries and territories, and during different time periods of day and year. The dataset contain videos from both indoor and outdoor scenes, with different camera perspectives. The density and scale of visual objects also vary significantly from one video to another. Some sample frames is presented in Fig. 5 to show the diversity of our dataset.

4.2. Manual Annotation

We are interested in detecting objects from two categories: *person* and *vehicle*. The *person* category includes any visible and recognizable people in the frames; this corresponds to the *person* category in MSCOCO [41] dataset. The *vehicle* category includes all vehicles with four or more wheels, so motorbikes, bicycles, tricycles are not included. This category corresponds to the *car*, *bus*, and *truck* categories in MSCOCO.

For each video, we used the first 1.5 hours for self-supervised and weakly-supervised training, and the remaining part for evaluation. Since many videos were taken with wide-angle lenses, some frames included faraway objects that were too small to be considered in evaluation. To address this, we manually drew polygon masks to exclude

dataset	contain videos	average length	frames	countries	bounding boxes	boxes per video
MSCOCO [41]	No	-	-	-	897K	-
KITTI [17]	No	-	-	1	80K	-
BDD100K [74]	Yes	40s	120M	1	1.8M	18
CityScapes [9]	Yes	1.8s	150K	2	65K	13
Scenes100	Yes	2h	21.6M	16	84K	840

Table 1. Comparison of Scenes100 with other object detection datasets (K= 10^3, M= 10^6).

these faraway parts from annotation and evaluation. We then estimated the number of visual objects in the unmasked parts by first using a pre-trained detector, and then refining the estimate through human annotation. Frames for annotation were uniformly sampled from the second portion of each video, with the number of frames being inversely proportional to the estimated number of objects in the frames. This ensured that each video had roughly the same number of annotated objects. We contracted a data annotation company to perform bounding box annotation on those sampled frames. Some annotated frame samples for evaluation is shown in Fig. 5.

4.3. Dataset Statistics

In Tab. 1, we compare Scenes100 with some popular object detection datasets. MSCOCO [41] is a fully-supervised general-purpose object detection dataset, which acts as the supervised source domain. KITTI [17] only contains object detection annotation associated to individual images. In BDD100K [74], object detection annotation is given for one key frame in each video. For CityScapes [9], we consider the finely labeled instance-level segmentation masks, which are given for the 20$^{\text{th}}$ image from a 30 frame video snippets lasting 1.8s.

The scale of Scenes100 is comparable to that of KITTI and CityScapes. However, Scenes100 has three unique features. First, the videos in Scenes100 are much longer than those in any other datasets with scenes, providing sufficient data for a model to adapt to each of them. Second, Scenes100 has greater diversity. Unlike some other datasets that are recorded in only a few cities with similar camera angles, background environments, and object types, Scenes100 covers several different countries, with varying weather conditions, road conditions, view of field, camera perspectives, and vehicle types. Third, each video in Scenes100 has a fixed camera perspective and stationary background scene, making background modeling possible.

Note that many currently available datasets are not primarily intended for self-supervised adaptive object detection tasks. They may possess other features, such as segmentation masks, multi-modal sensor data, depth maps, or annotations for object tracking. However, despite the absence of these attributes, Scenes100 holds value as the first extensive and varied dataset that features fixed camera perspectives and lengthy videos. As such, it can complement existing dataset collections and serve as a valuable resource for researchers exploring scene adaptive object detection.

5. Experiments

5.1. Evaluation Protocols

We evaluate the performance of object detectors following the COCO evaluation protocol [41] to calculate the average precision at $IoU = 50\%$ (AP^{50}) and mean average precision of different IoU thresholds from 0.5 to 0.95 (AP^m), with some modifications as follows.

Non-evaluation region: before feeding the ground-truth and detected bounding boxes to the COCO evaluator, the bounding boxes that have at least one corner inside the non-evaluation mask will be removed. So the faraway parts in the frames where objects are deemed too small and blurry will not affect the evaluation results. Please see Sec. 4.2 and Fig. 5 for more details.

Category weighting: in the standard COCO evaluation protocol, the overall multi-class AP is the simple average of the APs of the different categories. If a class is sparse in the training set and validation set, the model tends to perform poorly due to the scarcity of the training samples. In this case the average AP does not truly reflect the performance of a detector. Hence we propose an additional multi-class AP metric with weighted average where the weight is determined by the prevalence of ground truth object instances in the evaluation set. This weighted AP can better portray the performance of a detector in the case of long-tailed distribution. We refer the standard classes-mean COCO AP as AP_{co}, and the proposed weighted AP as AP_w. We refer the AP averaged over IoU thresholds and $IoU = 50\%$ as AP^m and AP^{50}.

Per-video evaluation: we aim to evaluate the performance of a scene adaption method, instead of a specific detection model. For each of the video, a model is adapted to it from the same base model trained on the source dataset. The performance of the base model and adapted model on each video is calculated individually as $\{(AP_{v,base}, AP_{v,adapt})|v = 1 \dots 100\}$. Then we evaluate the overall effectiveness of the adaptation method using the averaged AP gain as:

$$APG = \frac{1}{100}\sum_{v=1}^{100}(AP_{v,adapt} - AP_{v,base}). \quad (6)$$

5.2. Implementation Details

Our implementation is based on Detectron2 [70]. The DiMP tracker is directly taken from the official PyTracking [10] implementation without any change. More details and hyper-parameters can be found in the supplementary material. During training, we include the same number of

images from MSCOCO training set as the number of unlabeled frames from the videos. The hyper-parameters are the same for all videos. We check the training loss for all experiments, including the baseline methods, to ensure convergence. We also train "compound" models where all videos are used together for adaptation. The results and further details can be found in the supplementary material.

5.3. Domain Adaptation Baselines

We compare our methods with several domain adaptive object detection baselines. In their original papers, the methods are evaluated on other domain adaptive object detection datasets. However, those datasets do not contains videos with fixed camera perspectives, so the proposed location-aware mixup and dynamic background extraction cannot be applied. We instead apply the methods in other papers on Scenes100. For a fair comparison, all methods start with the same base model and weights, and then are adapted to and evaluated on each scene individually. We keep the values of the hyper-parameters that are specific to a baseline method used in the original papers. We use the same learning rate, batch size, and number of iterations as our experiments in those baseline methods. More details can be found in the supplementary material.

Self-Train (ST) [57] is a method that uses detection and tracking to obtain pseudo bounding boxes, similar to ours. But ST applies tracking only in one direction. It assigns different weights to the pseudo bounding boxes during training, but no refinement nor cross-teaching.

STAC [60] uses the base model to get pseudo bounding boxes, and trains the model on a strongly augmented version of the target domain image with those boxes.

Adaptive Teacher (AT) [37] also uses self-training, but the pseudo bounding boxes are detected on the fly. It utilizes exponential moving average to update the model gradually. It also uses weak/strong data augmentation, and a domain classifier to align the features of the two domains.

H^{2}FA R-CNN [72] applies feature alignment of source and target domains at various levels of Faster-RCNN. It assumes that image-level weak annotation is available for the target domain. So we use the labels from our pseudo-labeling process as image-level class labels.

TIA [78] is a method that instantiates feature alignment by using both domain classifier and auxiliary classification and localization heads. The model is trained in a mini-max manner using gradient reverse layers [15].

LODS [33] uses style enhancement as data augmentation. The pseudo bounding boxes are generated from the original images, and the features from both version of the same image are aligned. It also utilizes exponential moving average for the weights. It does not use any supervised training data from the source domain.

Model	# params	MSCOCO		Scenes100			
		AP^m_{co}	AP^{50}_{co}	AP^m_{co}	AP^{50}_{co}	AP^m_w	AP^{50}_w
M1 (R-50)	41.4M	50.05	76.48	40.52	62.12	41.28	64.65
M2 (R-101)	60.5M	51.29	77.46	41.11	63.10	41.96	65.74

Table 2. Performance of base models on MSCOCO validation set with remapped categories and Scenes100 before any adaptation. For Scenes100, APs are evaluated on each video individually and then averaged. R- stands for ResNet. # indicates the number of parameters (M= 10^6). See Sec. 5.1 for AP notation details.

5.4. Results

The performance of the base models is shown in Tab. 2, along with their number of trainable parameters. M2 is a bigger model with more parameters and representation capacity, so it outperforms M1 by a noticeable margin in all the metrics before adaptation. All the AP gains shown in other tables are based on the performance of base model M2. Note that the AP numbers on Scenes100 is already relatively high at only about 10 points below the performance on MSCOCO, so we do not observe very high (> 5 points) AP gains in the results shown later.

We compare the performance of different adaptation methods in terms of average AP gain in Tab. 3. Here we show our best combination of methods, which incorporates pseudo-labeling, location-aware mixup, and object mask mid-fusion. Among the baseline methods for domain adaptive object detection, only ST and LODS improves the performance significantly. AT only yields marginal improvement in classes-weighted AP. STAC, H^{2}FA, and TIA actually degrade the performance on Scenes100. LODS performs surprisingly well considering it does not use source domain images. However, we found that any training schedule longer than 1000 iterations would lead to severe performance degradation. The proposed method yields much higher, consistent, and stable improvement.

In general, self-training-based methods (ST, AT, STAC, and the proposed method) perform better than domain alignment-based methods (H^{2}FA and TIA). Domain alignment is usually achieved through adversarial learning, which is known to have unstable objectives and is more difficult to optimize [14, 42, 59, 61]. Therefore, these methods are more sensitive to hyperparameters and training schedules, making it challenging to determine the optimal settings for all datasets. Applying domain alignment methods directly to our scene adaptive dataset without careful tuning can cause problems as our dataset has less in-domain variance than domain adaptation datasets. Furthermore, since the scenes in our dataset are very different, they might require different settings. This implies that Scenes100 has its uniqueness compared to more generally purposed domain adaptive object detection datasets. Our proposed method is more robust to this setting and does not rely heavily on

Method	APG^m_{co}	APG^{50}_{co}	APG^m_w	APG^{50}_w
ST [57]	+0.80	+0.24	+1.39	+1.03
STAC [60]	-1.26	-5.12	-1.97	-6.64
AT [37]	-0.75	-1.11	+0.06	+0.04
H^2FA [72]	-3.10	-4.97	-3.77	-6.01
TIA [78]	-0.32	-0.37	-0.32	-0.33
LODS [33]	+0.45	+1.28	+1.02	+2.28
Proposed	**+3.76**	**+4.45**	**+3.78**	**+4.65**

Table 3. Averaged AP gain of different adaptation methods. See Sec. 5.1 for AP notation details.

Mixup	Fusion	APG^m_{co}	APG^{50}_{co}	APG^m_w	APG^{50}_w
✗	✗	+0.95	+0.54	+1.67	+1.55
Location-aware	✗	**+1.72**	**+1.67**	**+2.25**	**+2.53**
Random	✗	+1.22	+1.13	+1.76	+2.03
✗	early	+1.85	+2.12	+2.22	+2.73
✗	mid	**+3.40**	**+3.81**	**+3.67**	**+3.98**
✗	late	+3.34	+3.60	+3.38	+3.72
Location-aware	early	+2.25	+2.82	+2.59	+3.46
Location-aware	mid	**+3.76**	**+4.45**	**+3.78**	**+4.65**
Location-aware	late	+3.66	+4.10	+3.73	+4.31

Table 4. Ablation study for the proposed components of our scene-adaptive object detection method. ✗ means not being applied. Pseudo-labeling is always applied. This table shows the averaged AP gain; see Sec. 5.1 for AP notation details.

Mixup	Fusion	APG^m_{co}	APG^{50}_{co}	APG^m_w	APG^{50}_w
✗	AVG	**+3.40**	**+3.81**	**+3.67**	**+3.98**
✗	CNN	+3.25	+3.52	+3.26	+3.79
✗	ATTN	+3.22	+3.73	+3.23	+3.95
Location-aware	AVG	**+3.76**	**+4.45**	**+3.78**	**+4.65**
Location-aware	CNN	+3.60	+4.20	+3.71	+4.50
Location-aware	ATTN	+3.30	+3.89	+3.45	+4.26

Table 5. Ablation study on different feature pyramid fusion methods in term of averaged AP gain. AVG, CNN, and ATTN indicate mid-fusion based on average pooling, CNN, or attention module, respectively. ✗ means not being applied. Pseudo-labeling is always applied. See Sec. 5.1 for AP notation details.

hyperparameter tuning, as we use the same set of hyper-parameters across all 100 videos. Additional analysis and success/failure cases can be found in the supplementary material.

5.5. Ablation Study

We explore the effect of the different components in our methods by experimenting various combinations of them. For all experiments, we keep the pseudo-labeling unchanged, as it is the basis of other components. We compare different types of object mask fusion (Fusion): early, mid, and late. We also compare the proposed Location-aware object mixup with Random mixup. Tab. 4 shows the results. The experiments on hyper-parameters can be found in the supplementary material.

The proposed pseudo-labeling, mixup, and object mask fusion all benefit the adaptation performance. Applying only the pseudo-labeling already leads to AP gain higher than ST, which is more complicated involving pseudo box weighting. Location-aware mixup is also more advantageous than Random mixup, validating the assumption in Sec. 3.3 that having fewer artifacts is better for adaptation. Among three fusion options, mid-fusion yields the best result. That is perhaps because in a mid-fusion model, the RPN utilizes the fused feature pyramid, which can contain critical information of object boundaries. The combination of mixup and object mask fusion can further improve the performance.

We also compare different feature pyramid fusion methods described in Eq. (3). In our main experiments, we simply average the feature pyramids from the original image and the object mask. Now at each level of the pyramid, we use a separate CNN or an attention module to fuse the feature maps. The comparison is shown in Tab. 5. The parametric method performs slightly worse than average-pooling, probably because the newly introduced modules have randomly initialized weights and thus require additional supervised training data.

6. Summary

We have presented a self-supervised framework for scene-adaptive object detection. Base detectors and generic trackers are used to generate pseudo object labels as the adaptive training targets in a cross-teaching manner. To fully utilize the background equivariance when camera perspective is fixed, we propose artifacts-free location-aware object mixup to augment the input images, and dynamic background extraction for additional input modality to the detector. We also introduce the first large-scale scene adaptive object detection dataset, **Scenes100**, with several unique features compared to other domain adaptive object detection datasets. Our method outperforms domain adaptive object detection baselines by a large margin on Scenes100. We also conduct extensive experiments to illustrate the effectiveness of the components in our framework. We hope this work has demonstrated the importance of scene adaptation, and it will spur further research interest in this impactful but understudied area.

Acknowledgements. This research was partially supported by NSF grants IIS-1763981 & NSDF DUE-2055406.

References

[1] David Berthelot, Nicholas Carlini, Ian Goodfellow, Avital Oliver, Nicolas Papernot, and Colin Raffel. Mixmatch: A holistic approach to semi-supervised learning. In *NeurIPS*, 2019. 3

[2] Goutam Bhat, Martin Danelljan, Luc Van Gool, and Radu Timofte. Learning discriminative model prediction for tracking. *ICCV*, pages 6181–6190, 2019. 1, 3

[3] Alexey Bochkovskiy, Chien-Yao Wang, and Hong-Yuan Mark Liao. Yolov4: Optimal speed and accuracy of object detection. In *ArXiv*, 2020. 2

[4] Shahine Bouabid and Vincent Delaitre. Mixup regularization for region proposal based object detectors. In *ArXiv*, volume abs/2003.02065, 2020. 3

[5] Zhaowei Cai and Nuno Vasconcelos. Cascade r-cnn: Delving into high quality object detection. In *CVPR*, 2018. 2

[6] Nicolas Carion, Francisco Massa, Gabriel Synnaeve, Nicolas Usunier, Alexander Kirillov, and Sergey Zagoruyko. End-to-end object detection with transformers. In *ECCV*, pages 213–229, 2020. 2

[7] Yuhua Chen, Wen Li, Christos Sakaridis, Dengxin Dai, and Luc Van Gool. Domain adaptive faster r-cnn for object detection in the wild. In *CVPR*, 2018. 2

[8] Xiangxiang Chu, Zhi Tian, Yuqing Wang, Bo Zhang, Haibing Ren, Xiaolin Wei, Huaxia Xia, and Chunhua Shen. Twins: Revisiting the design of spatial attention in vision transformers. In *NeurIPS*, 2021. 2

[9] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016. 6

[10] Martin Danelljan, Goutam Bhat, Christoph Mayer, and Matthieu Paul. Pytracking. https://github.com/visionml/pytracking, 2019. 6

[11] Jinhong Deng, Wen Li, Yuhua Chen, and Lixin Duan. Unbiased mean teacher for cross-domain object detection. In *CVPR*, pages 4091–4101, 2021. 2

[12] Alexey Dosovitskiy, Lucas Beyer, Alexander Kolesnikov, Dirk Weissenborn, Xiaohua Zhai, Thomas Unterthiner, Mostafa Dehghani, Matthias Minderer, Georg Heigold, Sylvain Gelly, Jakob Uszkoreit, and Neil Houlsby. An image is worth 16x16 words: Transformers for image recognition at scale. In *ICLR*, 2021. 2

[13] Yoav Freund and Robert E. Schapire. A desicion-theoretic generalization of on-line learning and an application to boosting. In Paul Vitányi, editor, *Computational Learning Theory*, pages 23–37, Berlin, Heidelberg, 1995. Springer Berlin Heidelberg. 1, 2

[14] Angus Galloway, Anna Golubeva, Thomas Tanay, Medhat Moussa, and Graham W. Taylor. Batch normalization is a cause of adversarial vulnerability. In *ArXiv*, 2020. 7

[15] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *ICML*, ICML'15, page 1180–1189, 2015. 2, 7

[16] Yaroslav Ganin, E. Ustinova, Hana Ajakan, Pascal Germain, H. Larochelle, Francçois Laviolette, Mario Marchand, and Victor S. Lempitsky. Domain-adversarial training of neural networks. In *J. Mach. Learn. Res.*, 2016. 2

[17] Andreas Geiger, Philip Lenz, and Raquel Urtasun. Are we ready for autonomous driving? the kitti vision benchmark suite. In *CVPR*, 2012. 6

[18] Ross Girshick. Fast r-cnn. In *ICCV*, pages 1440–1448, 2015. 2

[19] Ross Girshick, Jeff Donahue, Trevor Darrell, and Jitendra Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, pages 580–587, 2014. 2, 4

[20] Kaiming He, Georgia Gkioxari, Piotr Dollár, and Ross Girshick. Mask r-cnn. In *ICCV*, pages 2980–2988, 2017. 4

[21] Kaiming He, X. Zhang, Shaoqing Ren, and Jian Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. *IEEE TPAMI*, 37:1904–1916, 2015. 2

[22] Zhenwei He and Lei Zhang. Multi-adversarial faster-rcnn for unrestricted object detection. In *ICCV*, pages 6667–6676, 2019. 2

[23] Dan Hendrycks, Norman Mu, Ekin D. Cubuk, Barret Zoph, Justin Gilmer, and Balaji Lakshminarayanan. AugMix: A simple data processing method to improve robustness and uncertainty. In *ICLR*, 2020. 3

[24] Tin Kam Ho. Random decision forests. In *ICDAR*, volume 1, pages 278–282 vol.1, 1995. 1, 2

[25] Minh Hoai and Andrew Zisserman. Talking heads: Detecting humans and recognizing their interactions. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2014. 2

[26] Jiaxing Huang, Dayan Guan, Aoran Xiao, and Shijian Lu. Model adaptation: Historical contrastive learning for unsupervised domain adaptation without source data. In *Thirty-Fifth Conference on Neural Information Processing Systems*, 2021. 2

[27] Naoto Inoue, Ryosuke Furuta, Toshihiko Yamasaki, and Kiyoharu Aizawa. Cross-domain weakly-supervised object detection through progressive domain adaptation. In *CVPR*, pages 5001–5009, 2018. 2

[28] Glenn Jocher. Yolov5. https://github.com/ultralytics/yolov5, 2020. 2

[29] Seunghyeon Kim, Jaehoon Choi, Taekyung Kim, and Changick Kim. Self-training and adversarial background regularization for unsupervised domain adaptive one-stage object detection. In *ICCV*, October 2019. 2

[30] Samuli Laine and Timo Aila. Temporal ensembling for semi-supervised learning. In *ICLR*, 2017. 2

[31] Chuyi Li, Lulu Li, Hongliang Jiang, Kaiheng Weng, Yifei Geng, Liang Li, Zaidan Ke, Qingyuan Li, Meng Cheng, Weiqiang Nie, Yiduo Li, Bo Zhang, Yufei Liang, Linyuan Zhou, Xiaoming Xu, Xiangxiang Chu, Xiaoming Wei, and Xiaolin Wei. Yolov6: A single-stage object detection framework for industrial applications. In *ArXiv*, 2022. 2

[32] Junnan Li, Richard Socher, and Steven C.H. Hoi. Dividemix: Learning with noisy labels as semi-supervised learning. In *ICLR*, 2020. 3

[33] Shuaifeng Li, Mao Ye, Xiatian Zhu, Lihua Zhou, and Lin Xiong. Source-free object detection by learning to overlook

domain style. In *CVPR*, pages 8014–8023, June 2022. 2, 7, 8, 13, 15, 16

[34] Wuyang Li, Xinyu Liu, and Yixuan Yuan. Sigma: Semantic-complete graph matching for domain adaptive object. In *CVPR*, 2022. 2

[35] Xianfeng Li, Weijie Chen, Di Xie, Shicai Yang, Peng Yuan, Shiliang Pu, and Yueting Zhuang. A free lunch for unsupervised domain adaptive object detection without source data. *AAAI*, 35(10):8474–8481, May 2021. 2

[36] Yanghao Li, Chao-Yuan Wu, Haoqi Fan, Karttikeya Mangalam, Bo Xiong, Jitendra Malik, and Christoph Feichtenhofer. Mvitv2: Improved multiscale vision transformers for classification and detection. In *CVPR*, 2022. 2

[37] Yu-Jhe Li, Xiaoliang Dai, Chih-Yao Ma, Yen-Cheng Liu, Kan Chen, Bichen Wu, Zijian He, Kris Kitani, and Peter Vajda. Cross-domain adaptive teacher for object detection. In *CVPR*, 2022. 2, 7, 8, 13, 15, 16

[38] Zhizhong Li and Derek Hoiem. Learning without forgetting. *IEEE TPAMI*, 40(12):2935–2947, dec 2018. 2

[39] Tsung-Yi Lin, Piotr Dollar, Ross Girshick, Kaiming He, Bharath Hariharan, and Serge Belongie. Feature pyramid networks for object detection. In *CVPR*, July 2017. 4

[40] Tsung-Yi Lin, Priya Goyal, Ross Girshick, Kaiming He, and Piotr Dollár. Focal loss for dense object detection. In *ICCV*, pages 2999–3007, 2017. 2

[41] Tsung-Yi Lin, Michael Maire, Serge Belongie, Lubomir Bourdev, Ross Girshick, James Hays, Pietro Perona, Deva Ramanan, C. Lawrence Zitnick, and Piotr Dollár. Microsoft coco: Common objects in context. In *ArXiv*, 2014. 5, 6

[42] Chen Liu, Mathieu Salzmann, Tao Lin, Ryota Tomioka, and Sabine Süsstrunk. On the loss landscape of adversarial training: Identifying challenges and how to overcome them. In *NeurIPS*, NIPS'20, Red Hook, NY, USA, 2020. Curran Associates Inc. 7

[43] Wei Liu, Dragomir Anguelov, Dumitru Erhan, Christian Szegedy, Scott Reed, Cheng-Yang Fu, and Alexander C. Berg. Ssd: Single shot multibox detector. In *ECCV*, 2016. 2

[44] 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. In *ICLR*, 2021. 2

[45] Ze Liu, Yutong Lin, Yue Cao, Han Hu, Yixuan Wei, Zheng Zhang, Stephen Lin, and Baining Guo. Swin transformer: Hierarchical vision transformer using shifted windows. In *ICCV*, 2021. 2

[46] Muhammad Akhtar Munir, Muhammad Haris Khan, M. Saquib Sarfraz, and Mohsen Ali. Ssal: Synergizing between self-training and adversarial learning for domain adaptive object detection. In *NeurIPS*, 2021. 2

[47] Supreeth Narasimhaswamy, Trung Nguyen, and Minh Hoai. Detecting hands and recognizing physical contact in the wild. In *Advances in Neural Information Processing Systems*, 2020. 2

[48] Supreeth Narasimhaswamy, Thanh Nguyen, Mingzhen Huang, and Minh Hoai. Whose hands are these? hand detection and hand-body association in the wild. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, 2022. 2

[49] Supreeth Narasimhaswamy, Zhengwei Wei, Yang Wang, Justin Zhang, and Minh Hoai. Contextual attention for hand detection in the wild. In *Proceedings of the International Conference on Computer Vision*, 2019. 2

[50] Thanh Nguyen, Chau Pham, Khoi Nguyen, and Minh Hoai. Few-shot object counting and detection. In *Proceedings of the European Conference on Computer Vision*, 2022. 2

[51] Kento Nishi, Yi Ding, Alex Rich, and Tobias Höllerer. Augmentation strategies for learning with noisy labels. In *CVPR*, pages 8022–8031, June 2021. 2

[52] German I. Parisi, Ronald Kemker, Jose L. Part, Christopher Kanan, and Stefan Wermter. Continual lifelong learning with neural networks: A review. *Neural Networks*, 113:54–71, 2019. 2

[53] Joseph Redmon, Santosh Divvala, Ross Girshick, and Ali Farhadi. You only look once: Unified, real-time object detection. In *ArXiv*, 2015. 2

[54] Joseph Redmon and Ali Farhadi. YOLO9000: better, faster, stronger. In *ArXiv*, 2016. 2

[55] Joseph Redmon and Ali Farhadi. Yolov3: An incremental improvement. In *ArXiv*, 2018. 2

[56] Shaoqing Ren, Kaiming He, Ross Girshick, and Jian Sun. Faster r-cnn: Towards real-time object detection with region proposal networks. In C. Cortes, N. Lawrence, D. Lee, M. Sugiyama, and R. Garnett, editors, *NeurIPS*, volume 28. Curran Associates, Inc., 2015. 2, 3, 4

[57] Aruni RoyChowdhury, Prithvijit Chakrabarty, Ashish Singh, SouYoung Jin, Huaizu Jiang, Liangliang Cao, and Erik Learned-Miller. Automatic adaptation of object detectors to new domains using self-training. In *CVPR*, 2019. 1, 2, 3, 7, 8, 13, 15, 16

[58] Kuniaki Saito, Yoshitaka Ushiku, Tatsuya Harada, and Kate Saenko. Strong-weak distribution alignment for adaptive object detection. In *CVPR*, 2019. 2

[59] Chawin Sitawarin, Supriyo Chakraborty, and David A. Wagner. Improving adversarial robustness through progressive hardening. In *ArXiv*, volume abs/2003.09347, 2020. 7

[60] Kihyuk Sohn, Zizhao Zhang, Chun-Liang Li, Han Zhang, Chen-Yu Lee, and Tomas Pfister. A simple semi-supervised learning framework for object detection. In *arXiv:2005.04757*, 2020. 2, 7, 8, 13, 15, 16

[61] Arvind P. Sridhar, Chawin Sitawarin, and David Wagner. Mitigating adversarial training instability with batch normalization. In *ICLRW*, 2021. 7

[62] Antti Tarvainen and Harri Valpola. Weight-averaged consistency targets improve semi-supervised deep learning results. In *NeurIPS*, 03 2017. 2

[63] Alexandru Cristian Telea. An image inpainting technique based on the fast marching method. *Journal of Graphics Tools*, 9:23 – 34, 2004. 4

[64] Zhi Tian, Chunhua Shen, Hao Chen, and Tong He. FCOS: Fully convolutional one-stage object detection. In *ICCV*, 2019. 2

[65] Ashish Vaswani, Noam Shazeer, Niki Parmar, Jakob Uszkoreit, Llion Jones, Aidan N Gomez, **L** ukasz Kaiser, and Illia Polosukhin. Attention is all you need. In *NeurIPS*, volume 30, 2017. 2

[66] Vikas Verma, Alex Lamb, Christopher Beckham, Amir Najafi, Ioannis Mitliagkas, David Lopez-Paz, and Yoshua Bengio. Manifold mixup: Better representations by interpolating hidden states. In *ICML*, volume 97 of *Proceedings of Machine Learning Research*, pages 6438–6447. PMLR, 09–15 Jun 2019. 3

[67] 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. In *ArXiv*, 2022. 2

[68] Chien-Yao Wang, I-Hau Yeh, and Hongpeng Liao. You only learn one representation: Unified network for multiple tasks. In *ArXiv*, 2021. 2

[69] Wenhai Wang, Enze Xie, Xiang Li, Deng-Ping Fan, Kaitao Song, Ding Liang, Tong Lu, Ping Luo, and Ling Shao. Pyramid vision transformer: A versatile backbone for dense prediction without convolutions. In *ICCV*, pages 568–578, 2021. 2

[70] Yuxin Wu, Alexander Kirillov, Francisco Massa, Wan-Yen Lo, and Ross Girshick. Detectron2. https://github.com/facebookresearch/detectron2, 2019. 6, 11

[71] Lin Xiong, Mao Ye, Dan Zhang, Yan Gan, Xue Li, and Yingying Zhu. Source data-free domain adaptation of object detector through domain-specific perturbation. *International Journal of Intelligent Systems*, 36:3746 – 3766, 2021. 2

[72] Yunqiu Xu, Yifan Sun, Zongxin Yang, Jiaxu Miao, and Yi Yang. H^2FA R-CNN: Holistic and hierarchical feature alignment for cross-domain weakly supervised object detection. In *CVPR*, pages 14329–14339, 2022. 2, 7, 8, 13, 15, 16

[73] Jihan Yang, Shaoshuai Shi, Zhe Wang, Hongsheng Li, and Xiaojuan Qi. St3d: Self-training for unsupervised domain adaptation on 3d object detection. In *CVPR*, 2021. 2

[74] 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. In *CVPR*, June 2020. 6

[75] Sangdoo Yun, Dongyoon Han, Seong Joon Oh, Sanghyuk Chun, Junsuk Choe, and Youngjoon Yoo. Cutmix: Regularization strategy to train strong classifiers with localizable features. In *ICCV*, 2019. 3

[76] Hongyi Zhang, Moustapha Cissé, Yann N. Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. In *ArXiv*, 2017. 1, 3

[77] Pengchuan Zhang, Xiyang Dai, Jianwei Yang, Bin Xiao, Lu Yuan, Lei Zhang, and Jianfeng Gao. Multi-scale vision longformer: A new vision transformer for high-resolution image encoding. *ICCV*, 2021. 2

[78] Liang Zhao and Limin Wang. Task-specific inconsistency alignment for domain adaptive object detection. In *CVPR*, 2022. 2, 7, 8, 13, 15, 16

[79] Xizhou Zhu, Weijie Su, Lewei Lu, Bin Li, Xiaogang Wang, and Jifeng Dai. Deformable DETR: deformable transformers for end-to-end object detection. In *ArXiv*, 2020. 2

Appendices

In the appendices, we show more sample frames and distribution of camera locations of Scenes100 to illustrate its diversity. We provide more implementation details and the running speed for the experiments in the main paper. We also look into the quality of the pseudo-labeling and model performance from the perspective of individual videos. Lastly we show the results of compound adaptation experiments. It is recommended to read this document on a color screen, and zoom in for fine details in the figures.

A. Sample Frames and Distribution of Camera Locations of Scenes100

In Fig. A we present sample frames along with their non-annotation masks and annotation bounding boxes from 12 videos in Scenes100 to show the diversity of the dataset. Please refer to the sub-captions for our comments for each of the scenes. The diversity of scenes shows the usefulness of the dataset. And since all videos share the same set of hyper-parameters in the adaptation experiments, it reiterates the effectiveness and robustness of the proposed self-supervised scene adaptive object detection method.

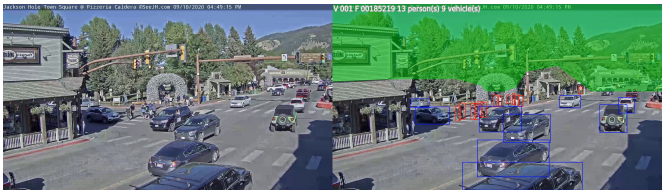
The locations of the cameras of the videos in Scenes100 is shown on a world map in Fig. B. Unlike most other object detection or scene understanding datasets, which are captured in a smaller range of locations, the videos in Scenes100 were recorded in places across the globe, giving great diversity.

B. More Implementation Details

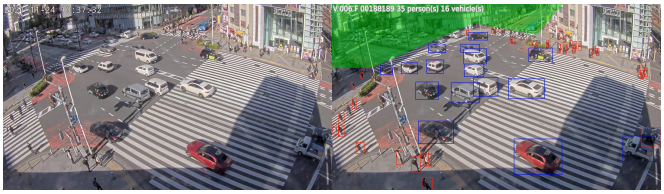
B.1. Software Libraries, Hyper-parameters, and Training Details

We start the finetuning from the models provided by the Detectron2 [70] model zoo. M1 and M2 are based on the configurations "COCO-Detection/ faster_rcnn_R_50_FPN_3x.yaml" and "COCO-Detection/ faster_rcnn_R_101_FPN_3x.yaml", respectively. We keep the weights of the backbone and RPN, but re-initialize the weights of the new ROI heads, as the number of classes changes during the object categories remapping described in the main paper. Then the whole network is trained end-to-end on remapped MSCOCO training set. We use learning rate scheduling with base of 5×10^{-4}, image batch size of 4, and ROI batch size of 128. The models are trained for 15,000 iterations. We examine the models' performance on validation set and losses periodically during training to ensure convergence.

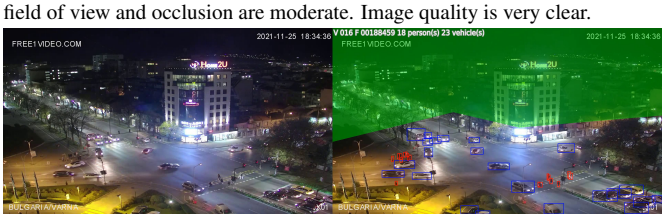
For self-supervised adaptation training, the training video portions are down-sampled uniformly to 5 frames per second, which gives 27,000 training frames per video. All spatial resolutions of the frames are kept. For pseudo-labeling, we set $\lambda_{det} = 0.5$, $\lambda_{sot} = 0.9$, and $\lambda_{iou} = 0.85$.



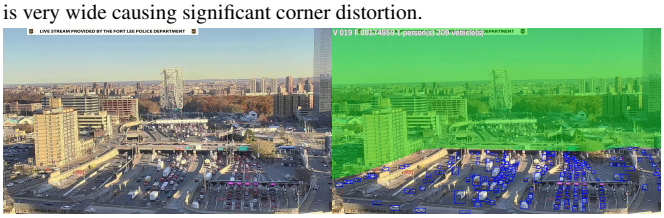
(a) Video `001` recorded in Jackson, Wyoming, USA at September 2020. The field of view and occlusion are moderate. Image quality is very clear.



(b) Video `006` recorded in Tokyo, Japan at November 2021. The field of view is very wide causing significant corner distortion.



(c) Video `016` recorded in Varna, Bulgaria at November 2021. The field of view is extremely wide. The lighting is low.



(d) Video `019` recorded in New York City, USA at November 2021. The field of view is extremely wide. Object are densely occluded.

(e) Video `040` recorded in Osaka, Japan at November 2021. It is a shadowed walkway in business district with heavy object occlusion.

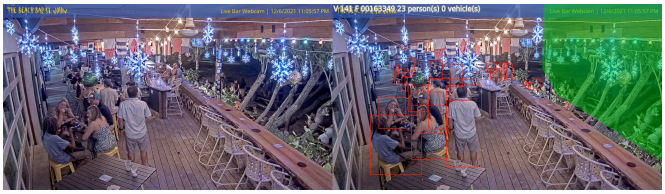
(f) Video `049` recorded in Tokyo, Japan at November 2021. The field of view is extremely wide and the objects appear very small.

(g) Video `074` recorded in San Francisco, California, USA at November 2021. There is very strong contrast between light and shadow.

(h) Video `090` recorded in Ust-Kut, Russia at December 2021. The weather is snowy, leaving mostly white background.

(i) Video `114` recorded in Kennebunkport, Maine, USA at December 2021. Most of the objects are in a strong back-light condition.

(j) Video `128` recorded in Katashina, Japan at December 2021. This is an indoor scene with people occluded by desks and chairs.



(k) Video `141` recorded in St John, U.S. Virgin Islands at December 2021. It is an indoor scene with heavy object occlusion.

(l) Video `172` recorded in Ammanford, Wales at December 2021. The lighting is very low, and motion blur of the objects is strong.

Figure A. Sample frames from Scenes100 with their non-annotation masks and annotation bounding boxes. As described in the sub-captions, the videos cover a variety of locations, weather, lighting conditions, image qualities, camera perspectives, and indoor/outdoor environments. The diversity of our dataset makes it representative for various scenes and thus useful for the understudied scene adaptive object detection task.

For location-aware mixup, we set $p_{mixup} = 0.3$, $r_{mixup} = 0.5$, and $\alpha_{cover} = 0.65$. For dynamic background extrac-

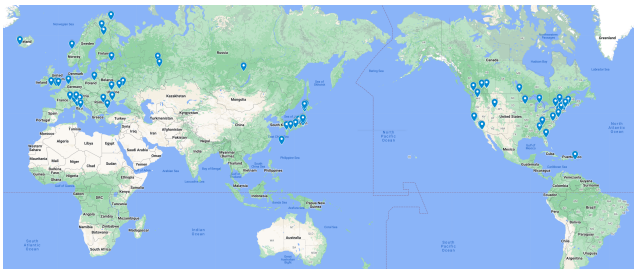


Figure B. The locations of the cameras in Scenes100 on the world map. Antarctica and Arctic regions are not included. Each location is represented by a blue pin. The number of pins is smaller than 100, as some of the videos are captured in the same city. Map is created using tools provided by Google Maps. Please see https://support.google.com/maps/answer/3145721 for its conventions on region names and borders.

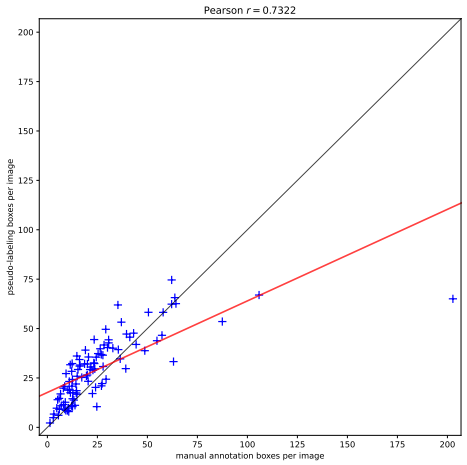


Figure C. Compare the number of object bounding boxes per image of manual annotation and pseudo-labeling. Each scatter point + represents one video. The red line — indicates the least-square linear model fit to the scatter. The Pearson correlation coefficient is displayed.

tion we set $T_{bg} = 90s$. For fusion models training, we use average pooling for feature pyramid fusion, and set $\alpha_{mid} = \alpha_{late} = 0.5$. In adaptation training, we use learning rate scheduling with base of 10^{-4}, image batch size of 4, and ROI batch size of 128. The models are trained for 20,000 iterations. We examine the models' performance on validation set and losses periodically during training to ensure convergence.

B.2. Implementation Details of Compared Baselines

The official implementation[1] of **Self-Train (ST)** [57] does not include the code for detection, tracking, and hard negative mining, so we re-implemented it in our framework.

We take the code of the core algorithm from the official implementation[2] of **STAC** [60], which is the image augmentation function set, and integrate it into our framework.

We directly use the official implementation[3] of **Adaptive Teacher (AT)** [37] and simply replace the initial base model and data loader with ours.

We directly use the official implementation[4] of **H^{2}FA R-CNN** [72], and simply replace the initial base model and data loader with ours.

The official implementation[5] of **TIA** [78] is based on another framework not compatible with our dataset and model architecture. So we implement it following the core logic in the official code base. We have to reduce the coefficient of auxiliary consistency losses by a factor of 10 to avoid training divergence. Number of training iterations is also reduced to 250, for longer training schedule leads to strong performance degradation.

The official implementation[6] of **LODS** [33] is based on another framework not compatible with our dataset and model architecture. So we implement it following the core logic in the official code base. Number of training iterations is also reduced to 250 due to performance degradation.

B.3. Inference Speed of Different Fusion Models

We show the inference throughput of vanilla faster-RCNN, early-fusion faster-RCNN, mid-fusion faster-RCNN, and late-fusion faster-RCNN in Tab. A. All models use the same R-101 backbone, and are ran on the same NVIDIA RTX 4090 GPU using the same set of images. The dataloader is carefully optimized to eliminate any possible CPU bottleneck. Please note that although mid-fusion and late-fusion introduces significantly more parameters by duplicating the RPN and ROI networks, only the branch for the fused feature pyramid will be deployed for inference. Thus they do not enlarge the model compared to vanilla faster-RCNN at inference time.

fusion model	inference throughput (images/s)
vanilla model	17.78
early-fusion	17.33
mid-fusion	11.93
late-fusion	11.96

Table A. Inference throughput of different fusion models in term of images per second. As stated in the main paper, early-fusion adds very limited computational cost, while mid-fusion and late-fusion impact the speed more significantly, but they give higher precision.

[1] https://github.com/AruniRC/detectron-self-train
[2] https://github.com/google-research/ssl_detection
[3] https://github.com/facebookresearch/adaptive_teacher
[4] https://github.com/XuYunqiu/H2FA_R-CNN
[5] https://github.com/MCG-NJU/TIA
[6] https://github.com/Flashkong/Source-Free-Object-Detection-by-Learning-to-Overlook-Domain-Style

B.4. Effect of Pseudo-Labeling Hyper-parameters

We try to exclude tracking pseudo bounding boxes from training. We also test only using M2 to generate the detection results. The results are shown in Tab. B. Including tracking bounding boxes in pseudo-labeling provides consistent performance gain. The results of only using M2 in detection also show that ensemble of models is beneficial.

We change the hyper-parameters λ_{det} and λ_{iou} for pseudo-labeling, and see how they effect the performance of the adapted models. To avoid interfering of other factors, location-aware mixup and object mask fusion are not used. The results are shown in Tab. C. The results show that increasing λ_{det} to certain level can lead to increase of APG^m, but decrease of APG^{50}. The performance will be degraded if λ_{det} is too high. Lower λ_{iou} leads to slightly higher performance.

B.5. Effect of Mixup Hyper-parameters

We change the hyper-parameters p_{mixup}, r_{mixup}, and α_{cover} for our proposed location-aware object mixup, and see how they effect the performance of the adapted models. To avoid interfering of other factors, object mask fusion is not used. The results are shown in Tab. D. It can be seen that increasing p_{mixup} or r_{mixup}, meaning stronger mixup, can improve the APs slightly.

B.6. Discussion

Please note that the results shown in Tab. C and Tab. D should not be viewed as a full-scale hyper-parameter tuning. The hyper-parameters in pseudo-labeling and location-aware mixup can interfere with each other. The situation is more complicated when object mask fusion is used. A proper tuning will require a search in the full hyper-parameter space, which is far beyond the computational capacity we have. It can also be reasonably expected that each video in Scenes100 has it own set of optimal hyper-parameters. Nevertheless, we argue that our proposed methods are mostly insensitive to the hyper-parameters, and can still perform well even no video-specific tuning is applied.

C. Individual Video Based Analysis

In the main paper, all the results are given in the form of the average over all the videos in Scenes100. However, due to their diversity, it is natural that our proposed methods perform differently on different videos. Here we take a deeper look into the individual performance of our methods.

C.1. Quantitative Assessment of Pseudo-Labeling

In Fig. C, we compare the number of manually labeled object bounding boxes per image with the number of pseudo bounding boxes after the proposed pseudo-labeling procedure. The same set of hyper-parameters are used as in the experiments. All the pseudo boxes with at least 1 corner inside the non-evaluation mask are removed for consistency. Please note that the mask is not used in the adaptation training experiments. It is clear the number of bounding boxes from pseudo-labeling is correlated with actual number of bounding boxes from human annotation. Please note that the comparison is not precise, since the frames used for pseudo-labeling and human annotation come from different parts of the videos, which means the density of objects can change. When the object density is not very high (less than 50 object per image), the correlation is strong, showing the proposed pseudo-labeling can identify most of the objects. However, when the object density is very high, meaning that there is heavy occlusion or the field of view is extremely wide, the scene becomes more difficult for object detectors and the number of pseudo bounding boxes is smaller compared to the actual number.

C.2. Correlation of AP Gains of Methods

We take a framework either from the baseline methods or from our ablation study, pair it with our best proposed method (pseudo-labeling + location-aware mixup + object mask mid-fusion, indicated by PL+MX+MF). we plot the individual AP gain after adaptation on each video of the 2 models in the pair as a scatter and calculate the Pearson correlation coefficient. The results are shown in Fig. D.

It is clear that only ST and LODS, which give moderate AP gain during adaptation, is weakly correlated with the proposed method. STAC, AT, H^{2}FA, and TIA all are not correlated with the proposed method. However, the combinations PL (pseudo-labeling), PL+MX (pseudo-labeling + location-aware mixup), PL+EF (pseudo-labeling + object mask early-fusion), PL+MF (pseudo-labeling + object mask mid-fusion), and PL+LF (pseudo-labeling + object mask late-fusion) are all strongly correlated with the best method. This indicates that our proposed methods are consistent in scene adaptation performance. The scenes that all methods perform poorly can be regarded as hard samples.

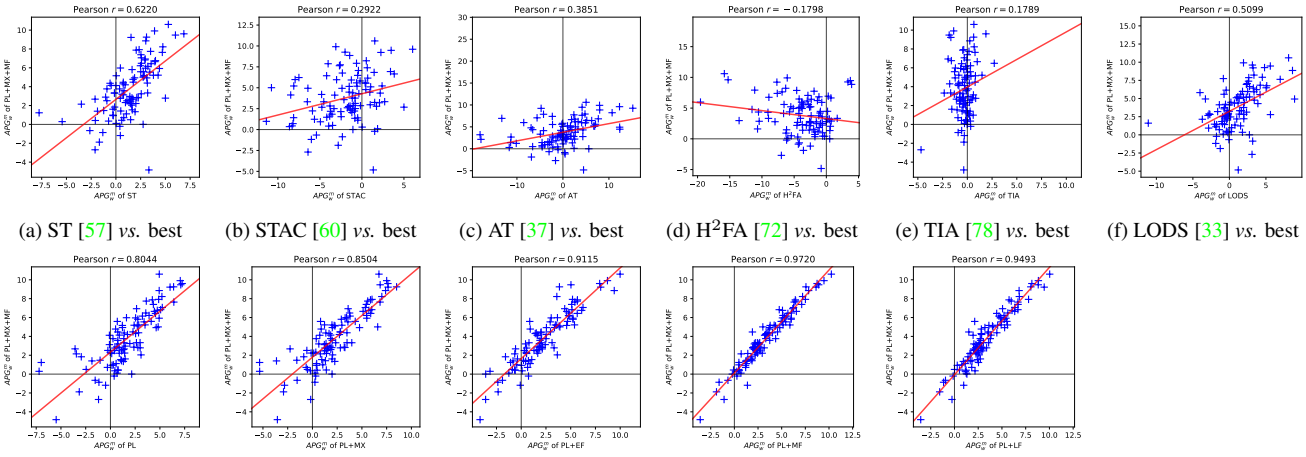
C.3. Success and Failure Cases Study

In Fig. E we present the videos that the best proposed method performs extraordinary well or bad in term of APG^m_w, and discuss the possible causes. Since it is the AP gains being examined, good performance means not only that the adapted model achieves high precision, but also that the base model cannot perform very well so there is room for improvement. Bad performance means the adaptation process actually degrade the detection ability of the base model.

Please note that the case study is qualitative and empirical. In our future work we will carry more systematic analysis on the performance, and improve our methods based on the observations.

detectors	tracking	APG^{m}_{co}	APG^{50}_{co}	APG^{m}_{w}	APG^{50}_{w}
M1+M2	✓	**+0.95**	**+0.54**	**+1.67**	**+1.55**
M1+M2	✗	+0.73	+0.22	+1.49	+1.31
M2	✓	+0.58	-0.30	+1.21	+0.67

Table B. Effects of tracking and ensemble of models on adaptation performance, in term of averaged AP gain. ^{m} and ^{50} stand for mean over IoU thresholds and $IoU = 50\%$, respectively. $_{co}$ and $_{w}$ stand for standard classes mean and proposed classes-weighted mean. ✓ and ✗ mean being and not being applied, respectively.



(a) ST [57] *vs.* best (b) STAC [60] *vs.* best (c) AT [37] *vs.* best (d) H^{2}FA [72] *vs.* best (e) TIA [78] *vs.* best (f) LODS [33] *vs.* best

(g) PL *vs.* best (h) PL+MX *vs.* best (i) PL+EF *vs.* best (j) PL+MF *vs.* best (k) PL+LF *vs.* best

Figure D. Compare the AP gains of different frameworks against the proposed best model. Each scatter point + represents one video. The red lines — indicate the least-square linear models fit to the scatters. The Pearson correlation coefficients are displayed.

λ_{det}	λ_{iou}	APG^{m}_{co}	APG^{50}_{co}	APG^{m}_{w}	APG^{50}_{w}
0.5	0.85	+0.95	+0.54	+1.67	+1.55
0.7	0.85	**+1.47**	+0.43	+1.79	+0.78
0.9	0.85	+0.33	-3.02	-0.12	-4.09
0.5	0.75	+1.28	**+0.60**	**+2.03**	**+1.75**
0.5	0.95	+0.36	+0.50	+1.00	+1.51

Table C. Effects of pseudo-labeling hyper-parameters on adaptation performance, in term of averaged AP gain. ^{m} and ^{50} stand for mean over IoU thresholds and $IoU = 50\%$, respectively. $_{co}$ and $_{w}$ stand for standard classes mean and proposed classes-weighted mean. λ_{det} is the minimum score for a detected object to be included in the pseudo bounding boxes. λ_{iou} is the IoU for 2 bounding boxes to be merged during refinement.

D. Performance of Compound Models

Here we treat Scenes100 in a manner closer to the standard domain adaptive object detection problem. All 100 videos are regarded as a whole target domain, and the models are trained on all the training frames from them, resulting a generic (compound) model for all videos. For consistency reasons, we keep all the hyper-parameters and settings during training unchanged from the individual adaptation settings used in the main paper, only to increase the number of training iterations by 20× to incorporate larger training set. After training, we still use the same independent evaluation protocol as in the main paper, and report the average AP gains in Tab. E.

Interestingly, different methods performs very differently under this compound adaptation setting compared to individual adaptation setting. ST is the only method the performs noticeably better, implying that it benefits from higher variance in the training data. AT and TIA perform significantly worse. Our proposed method sees about 1-point drop in the APs, but is still the best one by a considerable margin. This shows that the proposed method is more suitable for a fine-grained scene adaptive learning setting compared to more generic domain adaptive one. Trying to explain the vast difference between methods under different settings can be an interesting direction of research in our future work.

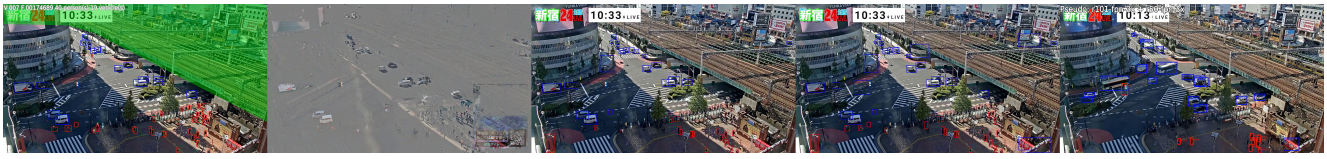
p_{mixup}	r_{mixup}	α_{cover}	APG^{m}_{co}	APG^{50}_{co}	APG^{m}_{w}	APG^{50}_{w}
0.3	0.5	0.65	+1.72	+1.67	+2.25	+2.53
0.5	0.5	0.65	+1.77	+1.72	+2.41	+2.75
0.7	0.5	0.65	**+2.06**	**+2.16**	**+2.47**	**+2.90**
0.3	0.3	0.65	+1.67	+1.67	+2.15	+2.47
0.3	0.7	0.65	+1.60	+1.59	+2.20	+2.57
0.3	0.5	0.45	+1.81	+1.70	+2.29	+2.52
0.3	0.5	0.85	+1.73	+1.74	+2.23	+2.60

Table D. Effects of locate-aware mixup hyper-parameters on adaptation performance, in term of averaged AP gain. ^{m} and ^{50} stand for mean over IoU thresholds and $IoU = 50\%$, respectively. $_{co}$ and $_{w}$ stand for standard classes mean and proposed classes-weighted mean. p_{mixup} is the probability that a frame is selected for mixup. r_{mixup} is the probability a pseudo bounding box in the mixup source frame is copied and pasted. α_{cover} is the threshold that a covered pseudo bounding box to be removed if exceeded.

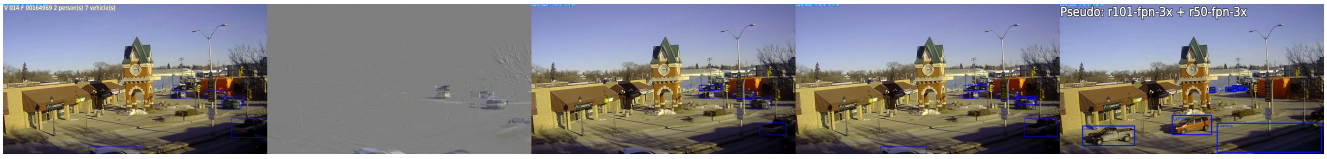
Method	Individually-adapted (main paper)				Compound adaptation			
	APG^{m}_{co}	APG^{50}_{co}	APG^{m}_{w}	APG^{50}_{w}	APG^{m}_{co}	APG^{50}_{co}	APG^{m}_{w}	APG^{50}_{w}
ST [57]	+0.80	+0.24	+1.39	+1.03	+1.69	+1.47	+1.69	+1.46
STAC [60]	-1.26	-5.12	-1.97	-6.64	-1.03	-4.81	-1.56	-6.03
AT [37]	-0.75	-1.11	+0.06	+0.04	-4.64	-6.92	-3.57	-5.50
H^{2}FA [72]	-3.10	-4.97	-3.77	-6.01	-3.78	-5.98	-3.99	-6.66
TIA [78]	-0.32	-0.37	-0.32	-0.33	-1.82	-2.76	-1.58	-2.34
LODS [33]	+0.45	+1.28	+1.02	+2.28	+0.59	+1.01	+0.69	+1.33
Proposed	**+3.76**	**+4.45**	**+3.78**	**+4.65**	**+2.77**	**+3.68**	**+2.56**	**+3.48**

Table E. Averaged AP gain of different compound adaptation models. The numbers of AP gain under individual adaptation setting are copied from the main paper for easy comparison. ^{m} and ^{50} stand for mean over IoU thresholds and $IoU = 50\%$, respectively. $_{co}$ and $_{w}$ stand for standard classes mean and proposed classes-weighted mean.

(a) Success case of video `154`, adaptation increases APG_w^m from 17.59 to 27.19. The base model misses some of the cars, probably due to the complexity of the scene and varied light and shadow conditions. The pseudo-labeling can identify most of the target objects, and the adapted model is able to pick up those cars missed by the base model.



(b) Success case of video `007`, adaptation increases APG_w^m from 25.13 to 33.36. Similar to video `154`, the base model misses many of the smaller objects. Although the pseudo-labeling cannot find all of objects, the adapted model still performs significantly better.



(c) Success case of video `014`, adaptation increases APG_w^m from 42.53 to 51.75. The base model cannot properly detect the vehicles in the shadow. However, in the object mask they are more clearly outlined and can be identified by the adapted model.

(d) Failure case of video `051`, adaptation decreases APG_w^m from 32.84 to 30.96. Adaptation seems to make the model produce less bounding boxes at the heavily-occluded regions (*e.g.* the parked cars at middle-right of the frame). This is probably due to the fact that pseudo-labeling cannot identify each object accurately at those regions.

(e) Failure case of video `093`, adaptation decreases APG_w^m from 54.91 to 50.08. The base model already performs very well, likely because the background is covered by snow and the objects are well-separated. Pseudo-labeling introduces some false positive bounding boxes, which is learned by the adapted model. Tuning the pseudo-labeling hyper-parameters can probably fix this issue.

Figure E. Some success and failure cases of the best proposed method. For each case, from left to right, 5 images are presented: i) 1 sample frame from the evaluation split with its non-annotation masks and annotation bounding boxes, ii) its corresponding object mask image, iii) the detection output of the base model on the evaluation frame, iv) the detection output of the adapted model on the evaluation frame, and v) 1 sample frame from the training split with its pseudo-bounding boxes from pseudo-labeling. For the detection output, we only show object bounding boxes with confidence score higher than 0.5.