Cross-domain learning using optimized pseudo labels: toward adaptive car detection in different weather conditions and urban cities

Ke Wang1 • Lianhua Zhang2 • Qin Xia3 • Liang Pu2 • Junlan Chen4

Received: 20 November 2020 / Accepted: 4 October 2021 / Published online: 26 October 2021
© The Author(s), under exclusive licence to Springer-Verlag London Ltd., part of Springer Nature 2021

Abstract
Convolutional neural networks based object detection usually assumes that training and test data have the same distribution, which, however, does not always hold in real-world applications. In autonomous vehicles, the driving scene (target domain) consists of unconstrained road environments which cannot all possibly be observed in training data (source domain) and this will lead to a sharp drop in the accuracy of the detector. In this paper, we propose a domain adaptation framework based on pseudo-labels to solve the domain shift. First, the pseudo-labels of the target domain images are generated by the baseline detector (BD) and optimized by our data optimization module to correct the errors. Then, the hard samples in a single image are labeled based on the optimization results of pseudo-labels. The adaptive sampling module is approached to sample target domain data according to the number of hard samples per image to select more effective data. Finally, a modified knowledge distillation loss is applied in the retraining module, and we investigate two ways of assigning soft-labels to the training examples from the target domain to retrain the detector. We evaluate the average precision of our approach in various source/target domain pairs and demonstrate that the framework improves over 10% average precision of BD on multiple domain adaptation scenarios on the Cityscapes, KITTI, and Apollo datasets.

Keywords Domain adaptation • Object detection • Autonomous vehicles • Pseudo-labels

1 Introduction

Research on autonomous vehicles (AVs) has developed rapidly in recent years. However, just as explored in our previous work [40, 41, 45], the adaptability in different cities is becoming a huge challenge for the development of AVs. Detection is a critical technology in AVs, which can obtain information about the driving scene ahead [40, 42–44]. Similarly, the domain shift cannot be neglected when the detector is applied to various weather conditions or various types of urban scenes. Although many types of research are devoted to resolving domain shift, there are still some challenges in AVs: 1) Domain adaptation methods or frameworks should take into account the real-time detection requirements of AVs; 2) It is difficult to select samples that have a large impact on domain adaptation among the massive driving scene (target domain) data (Fig. 1).

CNN-based object detection has made remarkable achievements in recent years. RCNN [9] and Fast RCNN [8] use CNN for feature extraction and classification. In
Faster RCNN (FRCNN) [31], the region proposal network (RPN) was introduced to refine the position and size of the pre-defined anchor box. In SSD [26], Anchor boxes with different aspect ratios are used to classify feature maps and predict bounding boxes on different scales. YOLO [29] divides the image into grids and predicts the bounding boxes and class labels of objects centered on each cell in the grid. CASNet [15] uses a cross-attention Siamese model to improve the performance of video salient object detection. Both FRCNN and YOLOv3 [30] are widely used nowadays. However, as discussed in [4, 21], the performance of the algorithm will descend when they are deployed in diverse scenarios.

In the initial phase of developing domain adaptation, researches are based on shallow architectures [10, 14], mainly utilize feature-based and instance-based methods to align the domain distributions. [5] proposed a subspace alignment technique to directly reduce the discrepancy between domains, [28] used maximum mean discrepancy criterion to learn a domain-invariant feature transformation, [36] attempted to discover a common low-rank subspace between domains to use the source samples reconstruct the target samples, [11] estimated the weights between domains to reduce the disparity by kernel mean matching. However, compared with the deep architecture, the improvement brought by above methods is very limited.

In recent years, the development of deep learning attracts researchers’ attention to the deep domain adaptation [16, 39]. The research includes fine-tuning supervised model in the target domain and unsupervised cross-domain representation learning. For example, [17, 32] adapt to the target domain by using noisy labels and robust learning methods, [2, 12, 47, 48] explore the feature differences and use adversarial training for adaptation. There are also some methods to reduce the feature difference by reconstructing the source domain image, such as image-to-image conversion [1, 23]. Other methods combine image reconstruction and domain difference techniques for adaptation, such as using image conversion and pseudo-label to fine-tune the model [13], adapting at the pixel level and feature level [35], using weak self-training and adversarial background score regularization to reduce the negative impact of pseudo-labels and domain feature shift [18]. These algorithms are based on FRCNN, which only considers detection accuracy but not efficiency. Actually, they are not suitable for AVs that require high detection speed.

In this paper, we address the domain shift through pseudo-labels and robust training. We divide domain adaptation into real-time inference phase and optimization & retraining phase. In the real-time inference phase, the BD, which is mounted on the AV, needs fast detection speed and high detection accuracy. YOLOv3 is considered to be a widely used generic detector with the above conditions [25], so we choose it as the BD. In theory, the recently developed YOLOv4 and YOLOv5 and other detectors can also be easily extended as BD. We observed that although BD performs poorly in the target domain, it can still be used for low-accuracy detection. The pseudo-label with low-accuracy results will be saved in this phase. In the optimization and retraining phase, we correct the pseudo-label by combining the low-accuracy pseudo-label with the results of designed detector. Then we retrain BD by mixing optimized data with the source domain data. Besides, we approach a more robust training method to accommodate pseudo-label noise. Considering the car is one of the main participants of urban traffic, this paper verifies the effectiveness of our framework through car detection in different driving scenarios. But our general framework, theoretically, can also be extended to the detection of other objects (e.g., pedestrian and traffic light).

The contributions of this paper are summarized as follows: (1) A pseudo-label optimization method was proposed to correct the low accuracy results generated by BD. (2) An adaptive sampling method was introduced to extract more effective data, which can significantly improve the adaptation of detector in the target domain. (3) A modified knowledge distillation loss and label smoothing method were applied to reduce the adverse effects of the label noise in the retraining stage.

The following content is arranged in this layout: The details of our method are explained in Sect. 2. Section 3 describes the datasets and results of various cross-domain experiments. The conclusion and future works are summarized in Sect. 4.
2 Methods

2.1 Domain adaptation framework

We represent the training data space as the source domain \( (S) \) and the test data space as the target domain \( (T) \). Figure 2 shows our domain adaptation framework. BD is initialized by source domain data \( (D_S) \) and applied on \( T \). Then the pseudo-labels are generated by BD. To make use of these noise data, we designed the following modules:

1. **Pseudo data optimization module** Correct the errors of pseudo-labels. We developed a FRCNN-based module to correct three common errors in object detection: (1) inaccurate bounding box location and size; (2) mistakes in object labels (False Positive); (3) missed detection of objects (True Negative).

2. **Data sampling module** Count the number of hard samples in a single image which from \( T \), and adaptively sample the data according to the hard sample amounts to preserve and utilize the data that has a significant impact on the detector.

3. **Retraining module** Employ a modified distillation loss to retrain the detector. We approach two different ways to assign soft-labels to reduce the effects of noise labels.

2.2 Pseudo data optimization module

2.2.1 Low accuracy data generated by BD

We assume the vehicle-mounted BD is initialized by \( D_S \), and the environment captured by the on-board camera is \( T \). When the confidence of the object in an image is greater than the threshold, the label will be saved as the pseudo-label.

2.2.2 Module structure

The pseudo data optimization module trained by \( D_S \) is proposed to reduce the errors contained in the pseudo-label. The module is based on FRCNN and takes LocNet [7] as the location method. In the pseudo-label optimization phase, the proposal region will be replaced by the pseudo-label region. Figure 3 shows the details of the module structure.

LocNet-based localization method In this part, In-Out probabilities are adopted as the conditional probability. In stage 1, taking the center point of the proposal as the origin, the area is enlarged by \( S_x \) and \( S_y \) on the X-axis and Y-axis to generate the search region. Then the search region will be divided into \( M \) parts in the horizontal regions (rows) and vertical regions (columns), respectively. The vectors \( p_x = \{p_x(i)\}_{i=1}^M \) and \( p_y = \{p_y(i)\}_{i=1}^M \), respectively, represent the conditional probabilities of each column and row of the search region to be inside the bounding box. If at least part of the area corresponding to the row or column is inside bounding box, then the row or column is considered to be within the box. The ground truth bounding box is denoted as \( B_{GT} = \{B_{GT}^1, B_{GT}^2, B_{GT}^3, B_{GT}^4\} \), while \( \{B_{GT}^1, B_{GT}^2\} \) is the left-top coordinates and \( \{B_{GT}^3, B_{GT}^4\} \) is the right-bottom coordinates. The object candidate probability vector \( T = \{T_x, T_y\} \) can be defined as follows:

\[
\forall i \in \{1, \ldots, M\}, T_x(i) = \begin{cases} 
1, & \text{if } B_{GT}^1 \leq i \leq B_{GT}^2, \\
0, & \text{otherwise}.
\end{cases} \\
\forall i \in \{1, \ldots, M\}, T_y(i) = \begin{cases} 
1, & \text{if } B_{GT}^3 \leq i \leq B_{GT}^4, \\
0, & \text{otherwise}.
\end{cases}
\]

(1)

In stage 2, the input search region is projected onto the feature map of the conv5_3 layer of vgg16 [37] for ROI pooling, and then through the convolutional layer to do max pooling on the X-axis and Y-axis, respectively. The prediction vector \( p = \{p_x, p_y\} \) is used to predict the
bounding box of search region and should be as close to $T$ as possible. The prediction bounding box $B_p = \{B_p^l, B_p^r, B_p^t, B_p^b\}$ (left-top and right-bottom coordinates) can be acquired by maximizing the likelihood of $B_p$:

$$
\arg\max_{B_p^l, B_p^r, B_p^t, B_p^b} \prod_{i \in \{B_p^l, B_p^r, B_p^t, B_p^b\}} p_x(i) \prod_{i \in \{B_p^l, B_p^r, B_p^t, B_p^b\}} (1 - p_s(i))
$$

The location loss is replaced by $L_{loc}^D(p, T)$:

$$
L_{loc}^D(p, T) = \sum_{a \in \{x, y\}} \sum_{i = 1}^{M} T_a(i) \log(p_a(i)) + \tilde{T}_a(i) \log(\tilde{p}_a(i))
$$

where $\tilde{T}_a(i) = 1 - T_a(i)$ and $\tilde{p}_a(i) = 1 - p_a(i)$.

### 2.2.3 Pseudo label refinement

Bounding box refinement As shown in Fig. 3, we replace the proposal to the pseudo-label region in stage 1 to refine the bounding box in the optimization phase. We assume $d_D$ is the confidence of the proposal in stage 2. The box from the LocNet-based localization method will be applied as the new bounding box when $d_D \geq \theta$.

#### False positive (FP) correction

An area in the image may be incorrectly identified as a true sample (e.g., a building is mislabeled as a car). For the pseudo-label region to be optimized, we set the score evaluation function $d_{\text{optim}}$ as follows:

$$
d_{\text{optim}} = \alpha d_{\text{pseudo}} + \beta d_D
$$

where $d_{\text{pseudo}}$ is the score predicted by BD, $\alpha$ and $\beta$ are the weight parameters. We estimate whether the area is a FP area through the value of $d_{\text{optim}}$. The details are shown in Sect. 2.3.

#### False negative (FN) correction

Some objects in the image could be missed detection by BD. We adopt a pre-trained model on COCO [24] dataset to mine these samples. ResNetXt-based [46] FRCNN is an ideal model, which is easily available and widely used. Although the detection speed is slow, the real-time inference is not required when optimizing pseudo-labels. We employ the FRCNN model as the FN model, which with ResNetXt-101 as the backbone, pre-trained on the COCO dataset. We assume the confidence of FN model is $d_N$ and add the region to our set of pseudo-labels when $d_N \geq \theta$. 

---

Fig. 3 Network structure of the pseudo data optimization module. In the training phase, the RPN will generate a rectangular proposal area, and then the proposal area will be expanded by $S_x$ and $S_y$ on the $X$-axis and $Y$-axis of the image to generate a search region for the next stage. In the optimization phase, to refine the bounding box, the proposal is replaced by the pseudo-label region. Then the pseudo-label region is expanded into the search region.
2.3 Retraining module

The target domain data (\(D_T\)) still have noise after the pseudo data optimization module. To accommodate these noises when retraining BD, we build a more robust retraining module.

2.3.1 Training on pseudo labels

When retraining BD with \(D_T\) and \(D_S\), the sample \(X_i\), which is the \(i\)-th training sample, will be considered as a positive sample and labeled as 1 if \(X_i\) from the ground truth of \(S\) or Sect. 2.2. The label \(y_i\) is defined as:

\[
y_i = \begin{cases} 
1, & \text{if } X_i \text{ is a pos. sample,} \\
0, & \text{if } X_i \text{ is a neg. sample.} 
\end{cases}
\]

The binary cross entropy loss is used to retrain the classification branch. For the \(i\)-th training sample, the loss is defined as:

\[
L_i(y_i, p_i) = -[y_i \log(p_i) + (1 - y_i) \log(1 - p_i)]
\]

(6)

where the label \(y_i \in \{0, 1\}\) and the model’s predicted posterior \(p_i \in [0, 1]\).

2.3.2 Pseudo label smoothing

Previous research [22] has shown that label smoothing can avoid the adverse effects of noise labels. Therefore, we further smooth the pseudo-labels. The positive labels from \(T\) are consists of two parts: (1) the pseudo-label region from BD; (2) the region from FN correction. We exploit two different ways to generate soft labels.

Soft labels of cross-domain score mapping \(s_{\text{remap}}\): Following the practice in [32], we assume that the soft score \(s_i\) in \(T\) has the same distribution as the score of \(S\). Let the score distribution of BD on \(T\) has p.d.f. \(f(x)\) and the score distribution on \(S\) has p.d.f. \(g(x)\). The cumulative distributions of them would be \(F(x) = \int_0^x f(t)dt\) and \(G(x) = \int_0^x g(r)dr\), respectively. We use a histogram to remap the distribution from \(T\) to match the score distribution from \(S\), i.e., replace the target domain score \(x\) with \(G^{-1}(F(x))\). We map the scores by linear interpolation. For the regions from FN correction, the soft score is the confidence threshold \(\theta\). The soft labels after remapping \(\tilde{s}_i\) is defined as:

\[
\tilde{s}_i = \begin{cases} 
\text{dist}\text{, if } X_i \text{ from BD,} \\
\theta, & \text{if } X_i \text{ from FN correction.} 
\end{cases}
\]

(7)

where \(\text{dist}\) is the score after remapping.

2.3.2.1 Soft labels with optimization \(\tilde{s}_{\text{optim}}\): To take advantage of the pseudo-label confidence, we assign the soft score for the pseudo data from Sect. 2.2. Considering that there are some false samples with high confidence in pseudo labels, we additionally set a high confidence threshold \(\theta_h\) (\(\theta_h > \theta\)) to filter the pseudo label. In FP correction, the soft score \(d_{\text{optim}}^{\text{FP}}\) is defined as:

\[
d_{\text{optim}}^{\text{FP}} = \begin{cases} 
x d_{\text{pseudo}} + \beta d_D, & \text{if } d_D \geq \theta_h, \\
\theta_h, & \text{if } d_D < \theta_h \text{ and } d_{\text{pseudo}} \geq \theta_h \\
\theta, & \text{if } d_D < \theta_h \text{ and } d_{\text{pseudo}} < \theta_h 
\end{cases}
\]

(8)

In FN correction, the soft score \(d_{\text{optim}}^{\text{FN}}\) is:

\[
d_{\text{optim}}^{\text{FN}} = \begin{cases} 
\theta_h, & \text{if } d_N \geq \theta_h, \\
\theta, & \text{if } d_N < \theta_h.
\end{cases}
\]

(9)

After FP correction, the pseudo-label region from BD will be verified in FN correction again. The soft label of the region \(X_i\) will be defined as \(\tilde{s}_i = 0\) if \(d_{\text{optim}}^{\text{FP}} = \theta\) and \(d_N < \theta\).

Then the soft labels with optimization is:

\[
\tilde{s}_i = \begin{cases} 
d_{\text{optim}}^{\text{FP}}, & \text{if } X_i \text{ from BD,} \\
d_{\text{optim}}^{\text{FN}}, & \text{if } X_i \text{ from FN correction,} \\
0, & \text{if } X_i \text{ from BD and } d_{\text{pseudo}} < \theta_h.
\end{cases}
\]

(10)

The distillation loss for \(X_i \in \{S, T\}\) after label smoothing is as follows now:

\[
L_i^{\text{dist}} = \begin{cases} 
L_i(y_i, p_i), & \text{if } X_i \in S, \\
L_i(\tilde{s}_i, p_i), & \text{if } X_i \in T.
\end{cases}
\]

(11)

2.4 Data sampling module

Large amounts of data are generated when a car is driving. More data may bring better improvements, but it will also consume more computing resources. Hence, to collect more effective data in \(T\), we adopt and improve the adaptive sampling method [27].

We define the soft label region that is not detected by BD as the suspected FN region, and the region that is only detected by BD is the suspected FP region. The region among them is labeled as hard sample (HS). The images are arranged in descending order according to the number of HS per image. For the ordered total sample \(K\), our sampling number is \(s\), and we have a sampling adaptation parameter \(\mu\). The sampling method is hard sampling when \(\mu = 1\). For the adaptive sampling method, random sampling is applied to select \(s\) samples from the data in the sampling area \((1, \mu s)\). The adaptive sampling method is defined as:

\[
\begin{cases}
\mu = 1, & \text{hard sampling,} \\
\min(\mu s, K) = K, & \text{random sampling,} \\
\text{otherwise,} & \text{adaptive sampling.}
\end{cases}
\]

\(\square\) Springer
3 Results and discussion

In this section, we use the YOLOv3 trained by $D_S$ as BD and test it on $T$. Therefore, BD has no ground truth information about $T$. We evaluated our method in experiments with cross weather adaptation, cross-platform adaptation, adaptation to the city with more complex scenes, and adaptation to local urban. Remap and optim represent soft labels of cross-domain score mapping and soft labels with optimization, respectively. Also, to verify the effect of pseudo-label smoothing, we recorded the results when it was not used (denoted as original). Noted that there is no FP correction for remap and original labels. The results of validation set are shown in tables. Additionally, the results trained on $T$ (oracle) are revealed to illustrate the existing gap between the domains. The experiments are tested on Nvidia Titan XP with 12GB memory.

3.1 Implement details

3.1.1 YOLOv3

In all the experiments, we train the detector 150 epochs with Darknet-53 pre-trained on ImageNet [20]. We resize the image’s largest side to 416 pixels during the training and inference stage. We report the average precision (AP) with a threshold of 0.5 for evaluation. The other setting is the same as the original YOLOv3.

3.1.2 Pseudo data optimization module

The shorter side of the image is resized to 600 pixels during the training and inference stage. Following the practice in [7], the $S_x$ and $S_y$ are setting to 1.2 which expanding the length and width of the area by 1.2 times, respectively. The $M$ is set as 28. The threshold of $p_c$ and $p_t$ is 0.5. Stochastic gradient descend (SGD) optimization algorithm is used during training, with the learning rate of 0.0001 and weight decay of 0.00005. The training process is stopped at 25 epochs.

3.1.3 Retraining module

Through our small-scale test experiments, we determined the following parameter’s value. We set the confidence threshold $\theta = 0.5$ and the high confidence threshold $\theta_h = 0.7$. In Eq. (4), we set the $x = d_{\text{pseudo}}$ and $\beta = d_D$. The $d_{\text{optim}}$ now is $d_{\text{optim}} = d_{\text{pseudo}}^2 + d_D^2$. When the $d_{\text{optim}} > 1$, the $d_{\text{optim}}$ is set to 1. So the ranges of $d_{\text{optim}}$ is [0.5, 1].

3.2 Experimental datasets

Different driving datasets are collected in different cities using different driving platforms. Urban environments and perspectives of the image are quite different, which is suitable for domain adaptation experiments. The following datasets are adopted to conduct experiments. Some datasets are used as source and target domain, and some datasets are only used as the source or target domain.

1. **Cityscapes** [3] is a real city scene dataset collected by a driving recorder in Germany and around countries containing 3475 images, which 2975 images for training and 500 images for validation. The image size is $2048 \times 1024$ pixels. We report results on the validation set because the test set does not contain annotations. We use the tightest bounding box of an instance segmentation mask as ground truth.

2. **Foggy Cityscapes** [34] is the foggy version of Cityscapes. There are three levels of foggy images, which are generated by the depth information of Cityscapes images. Foggy Cityscapes have the same data division as Cityscapes. In our experiment, we use all three levels of foggy images in the training set as $D_T$. We use the highest level (least visibility) of 500 foggy images for testing to compare with other methods.

3. **KITTI** [6] is another dataset for autonomous driving, containing 7481 images of real traffic scenes with the image size of $1242 \times 375$ pixels, including freeways, urban, and rural areas. All data are used as $D_S$.

4. **ApolloCar3D** [38] is a large-scale dataset contains a diverse set of stereo video sequences recorded in street scenes from different Chinese cities, with high quality annotations of more than 5000 frames. The training dataset contains 4083 training images and 200 validation images with the image size of $3384 \times 2710$ pixels. As with the Cityscapes, we use the tightest bounding box of an instance segmentation mask as ground truth and report the results on the validation set.
5. Local Urban Driving Video (LUDV). The video is captured by dashcam at Jiangbei District, Chongqing, China. It is 15 minutes long and contains 22514 frames in total. The image size is 1920×1080. 500 images are randomly sampled and annotated, which are used as the validation set (Fig. 5).

3.3 Domain adaptation for detection

In this section, we evaluated our method through the following experiments: (1) Cross weather adaptation; (2) Cross platform adaptation; (3) Cross cities adaptation; (4) Adaptation to local urban. In Fig. 6, we show the improvement of our method to BD and the gap with the ground truth.

3.3.1 Cross weather adaptation

The change in weather conditions will significantly affect the accuracy of detection. The most basic requirement of an AV is that the detector can maintain high accuracy in all weather conditions. Considering the cost of annotations, it is not possible to collect the data of many weather conditions. Therefore, models must be adaptable to various weather conditions. In the weather adaptation experiment, we evaluate our method through Cityscapes and Foggy Cityscapes. The Cityscapes is used as $S$ and the Foggy Cityscapes as $T$ (denoted by “Cityscapes Foggy Cityscapes”). We use the adaptive sampling to sample $s = 2500$ images from the pseudo-label set with 8925 images (three levels foggy images from the training set). We set the sampling adaptation parameter $\mu = 2$. At the same time, we use the random sampling to select the same number of images in Cityscapes.

3.3.2 Cross platform adaptation

The dataset captured by different platforms has different characteristics such as scenes and viewpoints. Also, the camera settings and frame sizes can also influence the visual appearance and image quality. These discrepancies are the domain shift reasons. In cross platform adaptation, we report the results of adaptation from different cameras. The KITTI dataset is used as $S$ and the Cityscapes is used as $T$ (denotes by “KITTI Cityscapes”). These two datasets have big differences in image size, scene, and perspective. In this experiment, we sample $s = 2000$ images from $T$ with the sampling adaptation parameter $\mu = 2$ by the adaptive sampling. The same number of images are sampled by random sampling in $S$.

Table 2 shows the results of cross platform adaptation. Compared with BD, the remap label improves the AP by 34.4%, while the optim label improves the AP by 32.7%. Up to 1.2% AP was promoted when label smoothing was applied. The results are all close to the oracle result. The results of our method also show better performance than the FRCNN-based method.

3.3.3 Cross cities adaptation

In the above two experiments, the traffic scenes of the KITTI dataset and the Cityscapes are kind of simple. In fact, the car faces much more crowded and complicated traffic conditions in many real-world scenes. To this end, we conducted the cross cities adaptation to verify the effectiveness of our method in the face of more complex scenarios. Considering the ApolloCar3D has much more complicated city environment than Cityscapes, we use the Cityscapes and ApolloCar3D datasets as $S$ and $T$ (denotes by “Cityscapes Apollo”), respectively. There are significant differences in image size, traffic complexity and perspective between the two datasets. In this experiment, we apply the same sampling method and sample the same number of images as the cross-platform adaptation.

Table 3 reports the results of our method. Our method still performs well when facing more complex and crowded scenes. In this experiment, the remap label improves 10.6% AP and the optim label improves 14.4% AP on validation.
set compared with BD. The optim label also gets a better result than the original, improves 3.2% AP.

3.3.4 Adaptation to local urban

Researchers usually use random sampling to create datasets when faced with video data. However, the sample obtained by random sampling is not necessarily the most effective data for improving the detector. This experiment is adopted to explore the effect of the data sampling module. In this

Table 1 The results comparing our method with baseline and other FRCNN based methods for weather adaptation

| Cityscapes→foggy cityscapes | Car AP (%) |
|----------------------------|------------|
| Method                     |            |
| Baseline                   | 21.9       |
| FRCNN in the wild [2]      | 40.5       |
| Diversify and match [19]   | 44.3       |
| Strong–weak align [33]     | 43.5       |
| Ours                       |            |
| Original                   | 39.9       |
| Remap                      | 43.2       |
| Optim                      | **45.2**   |
| Oracle                     | 49.7       |

We record the AP on the Foggy Cityscapes validation set. The last row shows the results of the base detector if the target domain annotations are available.

Table 2 Cross platform adaptation using KITTI and Cityscapes

| KITTI→cityscapes | Car AP (%) |
|------------------|------------|
| Method           |            |
| Baseline         | 14.6       |
| FRCNN in the wild| 38.5       |
| Ours             |            |
| Original         | 49.5       |
| Remap            | **50.7**   |
| Optim            | 47.3       |
| Oracle           | 54.2       |

We evaluate the AP of car class across the two domains.

Table 3 Adaptation from a simple scene to a complex scene

| Cityscapes→apollo | Car AP (%) |
|-------------------|------------|
| Method            |            |
| Baseline          | 15.9       |
| Ours              |            |
| Original          | 27.1       |
| Remap             | 26.5       |
| Optim             | **30.3**   |
| Oracle            | 44.6       |
section, LUDV is used as \( T \) and Cityscapes as \( S \). In LUDV, 19719 images with pseudo-label are saved by BD. Images of validation set are randomly sampled from them, while the remaining images are used as \( D_T \). All 2975 images in \( D_S \) and the same amount of images in \( D_T \) composed the mixed domain. As shown in Table 4, the samples from \( D_T \) are collected by different sampling methods. Noted that this experiment only used \( \text{optim} \) label because the other labels have no FP correction. Moreover, Fig. 7 shows the statistics of label amounts in this experiment, including the number of hard samples (HS) and the others (ES).

As shown in Fig. 7, when the sample amounts are the same, hard sampling have the largest number of labels, including 20038 HS and 45978 ES. Besides, HS has the highest proportion (30.4%). However, most of the samples collected are from similar scenes, so the result (54.6%) is lower than random sampling (54.8%). Giving consideration to the label amounts and scene diversity, adaptive sampling gets better performance than random sampling. The result (55.9%) is 1.1% higher than random sampling when \( \mu = 3 \). In addition, the result is better than others when using all \( D_T \) (59.0%). But it is not suitable when faced with much more video data considering the computing resources.

Table 4 also shows the precision (P) and recall (R) of each experiment to explore the impact of different sampling methods on both. It can be seen from the table that the number of HS is negatively correlated with the precision and positive correlation with the recall when the sample size is the same. The possible reason for this phenomenon is that the label noise in HS limits the improvement of precision, while the rich scene information improves the recall. This is also the direction for our further research and improvement.

### 3.3.5 Influence of IoU threshold

The threshold of intersection over union (IoU) controls the predicted bounding boxes, which can also impact the results of the test data. In the previous experiments, the threshold is set as 0.5. In this section, we tune the IoU threshold in our test to study its influence. We found that the value of the IoU threshold will greatly affect the test data results when the camera parameters have a big difference. So, we choose KITTI and Cityscapes to do the experiment.

Figure 8 shows the results of our experiment. With the increase in the IoU threshold, the AP of the car drops for all models. This is because the increase in the IoU threshold filters out many bounding boxes that are not so accurate, which leads to the decrease in detection precision and recall values. From Fig. 8, the AP of the baseline is close to 0 when the IoU threshold reaches 0.8. It means the baseline model can hardly locate the object accurately. Through the adaptation, our methods \((\text{remap} \text{ label and } \text{optim} \text{ label})\) still maintain a good performance when the threshold reaches 0.8. In Fig. 8, we can also see that our methods are stable under different IoU thresholds, and the gap with \( \text{oracle} \) has been kept within a certain range. This shows the effectiveness of our pseudo data optimization module.
4 Conclusions

To improve the adaptability of the vehicle-mounted object detector in various urban environments, this article employs a pseudo-label-based domain adaptation method to retrain the detector. Through our method, the detection model can adapt to the target domain without the annotations. In the real-time inference phase, BD generates low accuracy pseudo-label. In the optimization and retraining phase, the pseudo-label is corrected and used to retrain the detector. We evaluated our method through weather adaptation, cross-platform adaptation, and cross cities adaptation experiments. In weather adaptation and cross-platform adaptation, the maximum improvement of our method for BD is 23.3% AP and 34.4% AP, respectively. We compared our results, which close to the oracle, with other FRCNN-based methods. The adapted detector achieves higher accuracy while ensuring the detection speed. In the cross cities adaptation experiment, our method achieves a maximum improvement of 14.4% AP, which has a big improvement to BD. Moreover, we also explored the effect of each module by several designed experiments.

Future works focus on three perspectives: (1) Apply the framework into the system-on-chip and deploy on a real vehicle; (2) Transfer the pseudo-label data via 5G. Through cloud computing power, we can optimize the pseudo-labels and retrain the detector. (3) Employ more advanced algorithms to explore the gap between the domains. Reinforcement learning (RL) is a promising method to minimize the domain gap.

Acknowledgement This research was funded by Research supported by Chongqing Social Science Planning Project (No: 2018QNII16) National Natural Science Foundation of China (51605054), Chongqing Municipal Natural Science Foundation (cstc2020jyj-sxmX0079) Chongqing Technology Innovation and application development project (cstc2020jcsx-dxwtB0057 and cstc2020jcsx-sxmX0109) Fundamental Research Funds for the Central Universities (2020CDJ-LHZZ-042).

Authors contribution KW: Conceptualization, methodology, writing—reviewing and editing. LZ: Data curation, writing—original draft preparation. QX: Visualization, Investigation. LP: Supervision. JC: Validation.

Declarations

Conflict of interest The authors declare that they have no conflict of interest.

References

1. Arruda VF, Paixão TM, Berriel RF, De Souza AF, Badue C, Sebe N, Oliveira-Santos T (2019) Cross-domain car detection using 585 unsupervised image-to-image translation: from 586 day to night. IEEE Int Joint Conf Neural Netw. https://doi.org/10.1109/ IJCNN.2019.8852008
2. Chen Y, Li W, Sakaridis C, Dai D, Van Gool L (2018) Domain adaptive faster r-cnn for object detection in the wild. IEEE/CVF Conf Comput Vis Pattern Recognit. https://doi.org/10.1109/ CVPR.2018.00352
3. Cordts M, Omran M, Ramos S, Rehfeld T, Enzweiler M, Benenson R, Franke U, Roth S, Schiele B (2016) In: 2016 IEEE conference on computer vision and pattern recognition (CVPR), IEEE Conference on Computer Vision and Pattern Recognition, pp. 3213–3223, IEEE Comp Soc Comp Vis Fdh (2016). https:// doi.org/10.1109/CVPR.2016.350. 2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), Seattle, WA, JUN 27-30, 2016
4. Farahani A, Voghoei S, Rasheed K, Arabnia HR (2020) A brief review of domain adaptation
5. Fernando B, Habrard A, Sebban M, Tuytelaars T (2013) Unsupervised visual domain adaptation using subspace alignment. IEEE Int Conf Comput Vis
6. Geiger A, Lenz P, Stiller C, Urtasun R (2013) Vision meets robotics: the KITTI dataset. Int J Robot Res 32(11):1231–1237. https://doi.org/10.1177/0278364913491297
7. Gidaris S, Komodakis N (2016) Locnet: improving localization accuracy for object detection. IEEE Conf Comput Vis Pattern Recogn. https://doi.org/10.1109/CVPR.2016.92
8. Girshick R (2015) Fast r-cnn. In: Proceedings of the IEEE international conference on computer vision, pp. 1440-1448 (2015). https://doi.org/10.1109/ICCV.2015.169
9. Girshick R, Donahue J, Darrell T, Malik J (2014) Rich feature hierarchies for accurate object detection and semantic segmentation. In: 2014 IEEE conference on computer vision and pattern recognition (CVPR), pp. 580–587. https://doi.org/10.1109/CVPR.2014.81
10. Gopalan R, Li R, Chellappa R (2011) Domain adaptation for object recognition: an unsupervised approach. In: 2011 IEEE International conference on computer vision (ICCV), pp. 640 999-1006 (2011). https://doi.org/10.1109/ICCV.2011.6126344
11. Gretton A, Smola A, Huang J, Schmittfull M, Schwaighofer A, Lawrence N (2009) Covariate shift by kernel mean matching. In: Dataset shift in machine learning, pp. 131–160
12. He Z, Zhang L (2019) Multi-adversarial faster-rcnn for unrestricted object detection. In: 2019 IEEE/CVF international conference on computer vision (ICCV 2019), pp. 6667–6676. https://doi.org/10.1109/ICCV.2019.00677
13. Inoue N, Furuta R, Yamasaki T, Aizawa K (2018) Cross-domain weakly-supervised object detection through progressive domain adaptation. In: 2018 IEEE/CVF conference on computer vision and pattern recognition (CVPR), pp. 5001–5009. https://doi.org/10.1109/CVPR.2018.00525
14. Jhuo I, Liu D, Lee DT, Chang S (2012) Robust visual domain adaptation with low-rank reconstruction. In: 2012 IEEE conference on computer vision and pattern recognition (CVPR), pp. 2168–2175. https://doi.org/10.1109/CVPR.2012.6247924
15. Ji Y, Zhang H, Jie Z, Ma L, Wu, QMJ: Casenet, (2020) A cross-attention siamese network for video salient object detection. IEEE Trans Neural Netw Learn Syst PP(99):1–15
16. Kandaswamy C, Monteiro JC, Silva LM, Cardoso JS (2017) Multi-source deep transfer learning for cross-sensor biometrics. Neural Comput Appl 28(9:SI):2461–2475. https://doi.org/10.1007/s00521-016-2325-5
17. Khodabandeh M, Vahdat A, Ranjarb M, Macready WG (2019) A robust learning approach to domain adaptive object detection. In: 2019 IEEE/cvf international conference on computer vision
26. Liu W, Anguelov D, Erhan D, Szegedy C, Reed S, Fu CY, Berg L (2017) Learning from
24. Lin TY, Maire M, Belongie S, Hays J, Perona P, Ramanan D,
22. Li Y, Yang J, Song Y, Cao L, Luo J, Li LJ (2017) Learning from
20. Krizhevsky A, Sutskever I, Hinton GE (2017) Imagenet classi-
19. Kim T, Jeong M, Kim S, Choi S, Kim C (2019) Diversify and
18. Kim S, Choi J, Kim T, Kim C (2019) Self-training and adver-
17. Krizhevsky A, Sutskever I, Hinton GE (2017) Imagenet classi-
16. Kim S, Choi J, Kim T, Kim C (2019) Self-training and adver-
15. Krizhevsky A, Sutskever I, Hinton GE (2017) Imagenet classi-
14. Kim S, Choi J, Kim T, Kim C (2019) Self-training and adver-
13. Kim S, Choi J, Kim T, Kim C (2019) Self-training and adver-
12. Kim S, Choi J, Kim T, Kim C (2019) Self-training and adver-
11. Kim S, Choi J, Kim T, Kim C (2019) Self-training and adver-
10. Kim S, Choi J, Kim T, Kim C (2019) Self-training and adver-
9. Kim S, Choi J, Kim T, Kim C (2019) Self-training and adver-
8. Kim S, Choi J, Kim T, Kim C (2019) Self-training and adver-
7. Kim S, Choi J, Kim T, Kim C (2019) Self-training and adver-
6. Kim S, Choi J, Kim T, Kim C (2019) Self-training and adver-
5. Kim S, Choi J, Kim T, Kim C (2019) Self-training and adver-
4. Kim S, Choi J, Kim T, Kim C (2019) Self-training and adver-
3. Kim S, Choi J, Kim T, Kim C (2019) Self-training and adver-
2. Kim S, Choi J, Kim T, Kim C (2019) Self-training and adver-
1. Kim S, Choi J, Kim T, Kim C (2019) Self-training and adver-

Publisher's Note Springer Nature remains neutral with regard to jurisdictional claims in published maps and institutional affiliations.