Domain Adaptive Faster R-CNN for Object Detection in the Wild

Yuhua Chen[1] Wen Li[1] Christos Sakaridis[1] Dengxin Dai[1] Luc Van Gool[1,2]
[1]Computer Vision Lab, ETH Zurich [2]VISICS, ESAT/PSI, KU Leuven
{yuhua.chen,liwen,csakarid,dai,vangool}@vision.ee.ethz.ch

Abstract

Object detection typically assumes that training and test data are drawn from an identical distribution, which, however, does not always hold in practice. Such a distribution mismatch will lead to a significant performance drop. In this work, we aim to improve the cross-domain robustness of object detection. We tackle the domain shift on two levels: 1) the image-level shift, such as image style, illumination, etc.*, and 2) the instance-level shift, such as object appearance, size,* etc. *We build our approach based on the recent state-of-the-art Faster R-CNN model, and design two domain adaptation components, on image level and instance level, to reduce the domain discrepancy. The two domain adaptation components are based on $\mathcal{H}$-divergence theory, and are implemented by learning a domain classifier in adversarial training manner. The domain classifiers on different levels are further reinforced with a consistency regularization to learn a domain-invariant region proposal network (RPN) in the Faster R-CNN model. We evaluate our newly proposed approach using multiple datasets including Cityscapes, KITTI, SIM10K,* etc. *The results demonstrate the effectiveness of our proposed approach for robust object detection in various domain shift scenarios.*

1. Introduction

Object detection is a fundamental problem in computer vision. It aims at identifying and localizing all object instances of certain categories in an image. Driven by the surge of deep convolutional networks (CNN) [32], many CNN-based object detection approaches have been proposed, drastically improving performance [21, 51, 20, 8, 19, 39].

While excellent performance has been achieved on the benchmark datasets [12, 37], object detection in the real world still faces challenges from the large variance in viewpoints, object appearance, backgrounds, illumination, image quality, *etc*., which may cause a considerable domain shift between the training and test data. Taking autonomous driving as an example, the camera type and setup used in a particular car might differ from those used to collect training data, and the car might be in a different city where the appearance of objects is different. Moreover, the autonomous driving system is expected to work reliably under different weather conditions (*e.g.* in rain and fog), while the training data is usually collected in dry weather with better visibility. The recent trend of using synthetic data for training deep CNN models presents a similar challenge due to the visual mismatch with reality. Several datasets focusing on autonomous driving are illustrated in Figure 1, where we can observe a considerable domain shift.

Figure 1. **Illustration of different datasets for autonomous driving:** From top to bottom-right, example images are taken from: *KITTI*[17], *Cityscapes*[5], *Foggy Cityscapes*[49], *SIM10K*[30]. Though all datasets cover urban scenes, images in those dataset vary in style, resolution, illumination, object size, *etc*. The visual difference between those datasets presents a challenge for applying an object detection model learned from one domain to another domain.

Such domain shifts have been observed to cause significant performance drop [23]. Although collecting more training data could possibly alleviate the impact of domain shift, it is non-trivial because annotating bounding boxes is an expensive and time-consuming process. Therefore, it is highly desirable to develop algorithms to adapt object detection models to a new domain that is visually different from the training domain.

In this paper, we address this cross-domain object detection problem. We consider the unsupervised domain adaptation scenario: full supervision is given in the source domain while no supervision is available in the target domain.

Thus, the improved object detection in the target domain should be achieved at no additional annotation cost.

We build an end-to-end deep learning model based on the state-of-the-art Faster R-CNN model [48], referred to as Domain Adaptive Faster R-CNN. Based on the covariate shift assumption, the domain shift could occur on image level (*e.g*, image scale, image style, illumination, *etc*.) and instance level (*e.g*, object appearance, size, *etc*.), which motivates us to minimize the domain discrepancy on both levels. To address the domain shift, we incorporate two domain adaptation components on image level and instance level into the Faster R-CNN model to minimize the $\mathcal{H}$-divergence between two domains. In each component, we train a domain classifier and employ the adversarial training strategy to learn robust features that are domain-invariant. We further incorporate a consistency regularization between the domain classifiers on different levels to learn a domain-invariant region proposal network (RPN) in the Faster R-CNN model.

The contribution of this work can be summarized as follows: 1) We provide a theoretical analysis of the domain shift problem for cross-domain object detection from a probabilistic perspective. 2) We design two domain adaptation components to alleviate the domain discrepancy at the image and instance levels, resp. 3) We further propose a consistency regularization to encourage the RPN to be domain-invariant. 4) We integrate the proposed components into the Faster R-CNN model, and the resulting system can be trained in an end-to-end manner.

We conduct extensive experiments to evaluate our Domain Adaptive Faster R-CNN using multiple datasets including *Cityscapes* [5], *KITTI* [17], *SIM 10k* [30], *etc*. The experimental results clearly demonstrate the effectiveness of our proposed approach for addressing the domain shift of object detection in multiple scenarios with domain discrepancies.

2. Related Work

Object Detection: Object detection dates back a long time, resulting in a plentitude of approaches. Classical work [9, 13, 56] usually formulated object detection as a sliding window classification problem. The rise of deep convolutional networks(CNNs) [32] finds its origin in object detection, where its successes have led to a swift paradigm shift. Among the large number of approaches proposed [21, 51, 20, 19, 39, 8], region-based CNNs (R-CNN) [21, 20, 60] have received significant attention due to their effectiveness. This line of work was pioneered by R-CNN [21], which extracts region proposals from the image and a network is trained to classify each region of interest (ROI) independently. The idea has been extended by [20, 26] to share the convolution feature map among all ROIs. Faster R-CNN [21] produces object proposals with a Region Proposal Network (RPN). It achieved state-of-the-art results and laid the foundation for many follow-up works [19, 39, 8, 36, 60]. Faster R-CNN is also highly flexible and can be extended to other tasks, *e.g.* instance segmentation [7]. However, those works focused on the conventional setting without considering the domain adaptation issue for object detection in the wild. In this paper, we choose Faster R-CNN as our base detector, and improve its generalization ability for object detection in a new target domain.

Domain Adaptation: Domain adaptation has been widely studied for image classification in computer vision [10, 11, 33, 23, 22, 14, 52, 40, 15, 18, 50, 45, 43, 35]. Conventional methods include domain transfer multiple kernel learning [10, 11], asymmetric metric learning [33], subspace interpolation [23], geodesic flow kernel [22], subspace alignment [14], covariance matrix alignment [52, 57], *etc*. Recent works aim to improve the domain adaptability of deep neural networks, including [40, 15, 18, 50, 45, 43, 34, 24, 41, 42]. Different from those works, we focus on the object detection problem, which is more challenging as both object location and category need to be predicted.

A few recent works have also been proposed to perform unpaired image translation between two sets of data, which can be seen as pixel-level domain adaptation [62, 31, 59, 38]. However, it is still a challenging issue to produce realistic images in high resolution as required by real-world applications like autonomous driving.

Domain Adaptation Beyond Classification: Compared to the research in domain adaptation for classification, much less attention has been paid to domain adaptation for other computer vision tasks. Recently there are some works concerning tasks such as semantic segmentation [4, 27, 61], fine-grained recognition [16] *etc*. For the task of detection, [58] proposed to mitigate the domain shift problem of the deformable part-based model (DPM) by introducing an adaptive SVM. In a recent work [47], they use R-CNN model as feature extractor, then the features are aligned with the subspace alignment method. There also exists work to learn detectors from alternative sources, such as from images to videos [54], from 3D models [46, 53], or from synthetic models [25]. Previous works either cannot be trained in an end-to-end fashion, or focus on a specific case. In this work, we build an end-to-end trainable model for object detection, which is, to the best of our knowledge, the first of its kind.

3. Preliminaries

3.1. Faster R-CNN

We briefly review the Faster R-CNN [60] model, which is the baseline model used in this work. Faster R-CNN is a two-stage detector mainly consisting of three major com-

ponents: shared bottom convolutional layers, a region proposal network (RPN) and a region-of-interest (ROI) based classifier. The architecture is illustrated in the left part of Figure 2.

First an input image is represented as a convolutional feature map produced by the shared bottom convolutional layers. Based on that feature map, RPN generates candidate object proposals, whereafter the ROI-wise classifier predicts the category label from a feature vector obtained using ROI-pooling. The training loss is composed of the loss of the RPN and the loss of the ROI classifiers:

$$L_{det} = L_{rpn} + L_{roi} \tag{1}$$

Both training loss of the RPN and ROI classifiers have two loss terms: one for classification as how accurate the predicted probability is, and the other is a regression loss on the box coordinates for better localization. Readers are referred to [60] for more details about the architecture and the training procedure.

3.2. Distribution Alignment with $\mathcal{H}$-divergence

The $\mathcal{H}$-divergence [1] is designed to measure the divergence between two sets of samples with different distributions. Let us denote by $\mathbf{x}$ a feature vector. A source domain sample can be denoted as $\mathbf{x}_{\mathcal{S}}$ and a target domain sample as $\mathbf{x}_{\mathcal{T}}$. We also denote by $h : \mathbf{x} \to \{0, 1\}$ a domain classifier, which aims to predict the source samples $\mathbf{x}_{\mathcal{S}}$ to be 0, and target domain sample $\mathbf{x}_{\mathcal{T}}$ to be 1. Suppose $\mathcal{H}$ is the set of possible domain classifiers, the $\mathcal{H}$-divergence defines the distance between two domains as follows:

$$d_{\mathcal{H}}(\mathcal{S}, \mathcal{T}) = 2\left(1 - \min_{h \in \mathcal{H}}\left(err_{\mathcal{S}}(h(\mathbf{x})) + err_{\mathcal{T}}(h(\mathbf{x}))\right)\right).$$

where $err_{\mathcal{S}}$ and $err_{\mathcal{T}}$ are the prediction errors of $h(\mathbf{x})$ on source and target domain samples, resp. The above definition implies that the domain distance $d_{\mathcal{H}}(\mathcal{S}, \mathcal{T})$ is inversely proportional to the error rate of the domain classifier h. In other words, if the error is high for the best domain classifier, the two domains are hard to distinguish, so they are close to each other, and v.v.

In deep neural networks, the feature vector $\mathbf{x}$ usually comprises the activations after a certain layer. Let us denote by f the network that produces $\mathbf{x}$. To align the two domains, we therefore need to enforce the networks f to output feature vectors that minimize the domain distance $d_{\mathcal{H}}(\mathcal{S}, \mathcal{T})$ [15], which leads to:

$$\min_{f} d_{\mathcal{H}}(\mathcal{S}, \mathcal{T}) \Leftrightarrow \max_{f} \min_{h \in \mathcal{H}} \{err_{\mathcal{S}}(h(\mathbf{x})) + err_{\mathcal{T}}(h(\mathbf{x}))\}.$$

This can be optimized in an adversarial training manner. Ganin and Lempitsky [15] implemented a gradient reverse layer (GRL), and integrated it into a CNN for image classification in the unsupervised domain adaptation scenario.

4. Domain Adaptation for Object Detection

Following the common terminology in domain adaptation, we refer to the domain of the training data as source domain, denoted by $\mathcal{S}$, and to the domain of the test data as target domain, denoted by $\mathcal{T}$. For instance, when using the Cityscapes dataset for training and the KITTI dataset for testing, $\mathcal{S}$ is the Cityscapes dataset and $\mathcal{T}$ represents the KITTI dataset.

We also follow the classic setting of unsupervised domain adaptation, where we have access to images and full supervision in the source domain (*i.e.*, bounding box and object categories), but only unlabeled images are available for the target domain. Our task is to learn an object detection model adapted to the unlabeled target domain.

4.1. A Probabilistic Perspective

The object detection problem can be viewed as learning the posterior $P(C, B|I)$, where I is the image representation, B is the bounding-box of an object and $C \in \{1, \ldots, K\}$ the category of the object (K being the total number of categories).

Let us denote the joint distribution of training samples for object detection as $P(C, B, I)$, and use $P_{\mathcal{S}}(C, B, I)$ and $P_{\mathcal{T}}(C, B, I)$ to denote the source domain joint distribution and the target domain joint distribution, resp. Note that here we use $P_{\mathcal{T}}(C, B, I)$ to analyze the domain shift problem, although the bounding box and category annotations (*i.e.*, B and C) are unknown during training. When there is a domain shift, $P_{\mathcal{S}}(C, B, I) \neq P_{\mathcal{T}}(C, B, I)$.

Image-Level Adaptation: Using the Bayes's Formula, the joint distribution can be decomposed as:

$$P(C, B, I) = P(C, B|I)P(I). \tag{2}$$

Similar to the classification problem, we make the covariate shift assumption for objection detection, *i.e.*, the conditional probability $P(C, B|I)$ is the same for the two domains, and the domain distribution shift is caused by the difference on the marginal distribution $P(I)$. In other words, the detector is consistent between two domains: given an image, the detection results should be the same regardless of which domain the image belongs. In the Faster R-CNN model, the image representation I is actually the feature map output of the base convolutional layers. Therefore, to handle the domain shift problem, we should enforce the distribution of image representation from two domains to be the same (*i.e.*, $P_{\mathcal{S}}(I) = P_{\mathcal{T}}(I)$), which is referred to as *image-level adaptation*.

Instance-Level Adaptation: On the other hand, the joint distribution can also be decomposed as:

$$P(C, B, I) = P(C|B, I)P(B, I). \tag{3}$$

With the covariate shift assumption, *i.e.*, the conditional probability $P(C|B, I)$ is the same for the two domains, we

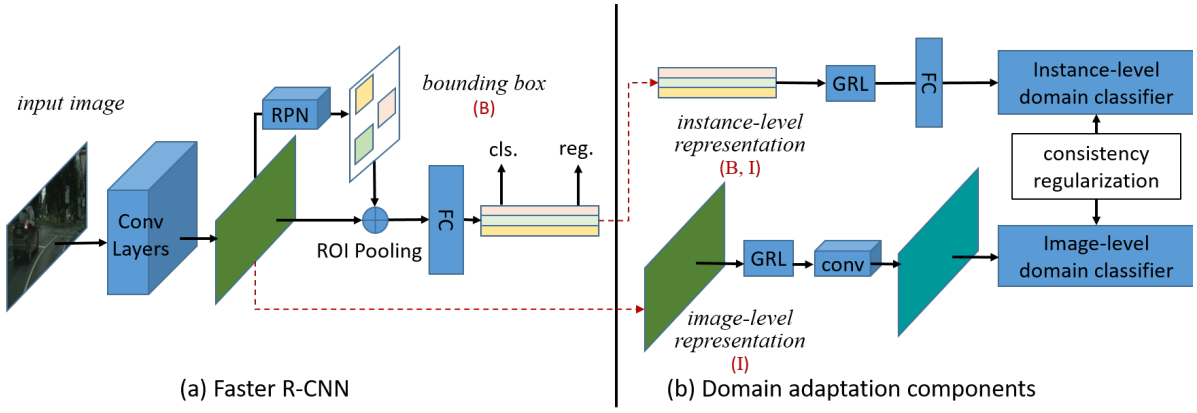


Figure 2. **An overview of our Domain Adaptive Faster R-CNN model:** we tackle the domain shift on two levels, the image level and the instance level. A domain classifier is built on each level, trained in an adversarial training manner. A consistency regularizer is incorporated within these two classifiers to learn a domain-invariant RPN for the Faster R-CNN model.

have that the domain distribution shift is from the difference in the marginal distribution $P(B, I)$. Intuitively, this implies the semantic consistency between two domains: given the same image region containing an object, its category labels should be the same regardless of which domain it comes from. Therefore, we can also enforce the distribution of instance representation from two domains to be the same (*i.e.*, $P_{\mathcal{S}}(B, I) = P_{\mathcal{T}}(B, I)$). We refer to it as instance-level alignment.

Here the instance representation (B, I) refers to the features extracted from the image region in the ground truth bounding box for each instance. Although the bounding-box annotation is unavailable for the target domain, we can obtain it via $P(B, I) = P(B|I)P(I)$, where $P(B|I)$ is a bounding box predictor (*e.g*, RPN in Faster R-CNN). This holds only when $P(B|I)$ is domain-invariant, for which we provide a solution below.

Joint Adaptation: Ideally, one can perform domain alignment on either the image or instance level. Considering that $P(B, I) = P(B|I)P(I)$ and the conditional distribution $P(B|I)$ is assumed to be the same and non-zero for two domains, thus we have:

$$P_{\mathcal{S}}(I) = P_{\mathcal{T}}(I) \Leftrightarrow P_{\mathcal{S}}(B, I) = P_{\mathcal{T}}(B, I). \tag{4}$$

In other words, if the distributions of the image-level representations are identical for two domains, the distributions of the instance-level representations are also identical, and v.v. Yet, it is generally non-trivial to perfectly estimate the conditional distribution $P(B|I)$. The reasons are two-fold: 1) in practice it may be hard to perfectly align the marginal distributions $P(I)$, which means the input for estimating $P(B|I)$ is somehow biased, and 2) the bounding box annotation is only available for source domain training data, therefore $P(B|I)$ is learned using the source domain data only, which is easily biased toward the source domain.

To this end, we propose to perform domain distribution alignment on both the image and instance levels, and to apply a consistency regularization to alleviate the bias in estimating $P(B|I)$. As introduced in Section 3.2, to align the distributions of two domains, one needs to train a domain classifier $h(\mathbf{x})$. In the context of object detection, $\mathbf{x}$ can be the image-level representation I or the instance-level representation (B, I). From a probabilistic perspective, $h(\mathbf{x})$ can be seen as estimating a sample $\mathbf{x}$'s probability belonging to the target domain.

Thus, by denoting the domain label as D, the image-level domain classifier can be viewed as estimating $P(D|I)$, and the instance-level domain classifier can be seen as estimating $P(D|B, I)$. By using the Bayes' theorem, we obtain:

$$P(D|B, I)P(B|I) = P(B|D, I)P(D|I). \tag{5}$$

In particular, $P(B|I)$ is a domain-invariant bounding box predictor, and $P(B|D, I)$ a domain-dependent bounding box predictor. Recall that in practice we can only learn a domain-dependent bounding box predictor $P(B|D, I)$, since we have no bounding box annotations for the target domain. Thus, by enforcing the consistency between two domain classifiers, *i.e.*, $P(D|B, I) = P(D|I)$, we could learn $P(B|D, I)$ to approach $P(B|I)$.

4.2. Domain Adaptation Components

This section introduces two domain adaptation components for the image and instance levels, used to align the feature representation distributions on those two levels.

Image-Level Adaptation: In the Faster R-CNN model, the image-level representation refers to the feature map outputs of the base convolutional layers (see the green parallelogram in Figure 2). To eliminate the domain distribution mismatch on the image level, we employ a patch-based domain classifier as shown in the lower right part of Figure 2.

In particular, we train a domain classifier on each activation from the feature map. Since the receptive field of each

activation corresponds to an image patch of the input image I_i, the domain classifier actually predicts the domain label for each image patch.

The benefits of this choice are twofold: 1) aligning image-level representations generally helps to reduce the shift caused by the global image difference such as image style, image scale, illumination, *etc*. A similar patch-based loss has shown to be effective in recent work on style transfer [29], which also deals with the global transformation, and 2) the batch size is usually very small for training an object detection network, due to the use of high-resolution input. This patch-based design is helpful to increase the number of training samples for training the domain classifier.

Let us denote by D_i the domain label of the i-th training image, with $D_i = 0$ for the source domain and $D_i = 1$ for the target domain. We denote as $\phi_{u,v}(I_i)$ the activation located at (u, v) of the feature map of the i-th image after the base convolutional layers. By denoting the output of the domain classifier as $p_i^{(u,v)}$ and using the cross entropy loss, the image-level adaptation loss can be written as:

$$\mathcal{L}_{img} = -\sum_{i,u,v} \Big[D_i \log p_i^{(u,v)} + (1 - D_i) \log(1 - p_i^{(u,v)}) \Big]. \tag{6}$$

As discussed in Section 3.2, to align the domain distributions, we should simultaneously optimize the parameters of the domain classifier to minimize the above domain classification loss, and also optimize the parameters of the base network to maximize this loss. For the implementation we use the gradient reverse layer (GRL) [15], whereas the ordinary gradient descent is applied for training the domain classifier. The sign of the gradient is reversed when passing through the GRL layer to optimize the base network.

Instance-Level Adaptation: The instance-level representation refers to the ROI-based feature vectors before feeding into the final category classifiers (*i.e.*, the rectangles after the "FC" layer in Figure 2). Aligning the instance-level representations helps to reduce the local instance difference such as object appearance, size, viewpoint *etc*. Similar to the image-level adaptation, we train a domain classifier for the feature vectors to align the instance-level distribution. Let us denote the output of the instance-level domain classifier for the j-th region proposal in the i-th image as $p_{i,j}$. The instance-level adaptation loss can now be written as:

$$\mathcal{L}_{ins} = -\sum_{i,j} \Big[D_i \log p_{i,j} + (1 - D_i) \log(1 - p_{i,j}) \Big]. \tag{7}$$

We also add a gradient reverse layer before the domain classifier to apply the adversarial training strategy.

Consistency Regularization: As analyzed in Section 4.1, enforcing consistency between the domain classifier on different levels helps to learn the cross-domain robustness of bounding box predictor (*i.e.*, RPN in the Faster R-CNN model). Therefore, we further impose a consistency regularizer. Since the image-level domain classifier produces an output for each activation of the image-level representation I, we take the average over all activations in the image as its image-level probability. The consistency regularizer can be written as:

$$L_{cst} = \sum_{i,j} \| \frac{1}{|I|} \sum_{u,v} p_i^{(u,v)} - p_{i,j} \|_2, \tag{8}$$

where $|I|$ denotes the total number of activations in a feature map, and $\| \cdot \|$ is the ℓ_2 distance.

4.3. Network Overview

An overview of our network is shown in Figure 2. We augment the Faster R-CNN base architecture with our domain adaptation components, which leads to our Domain Adaptive Faster R-CNN model.

The left part of Figure 2 is the original Faster R-CNN model. The bottom convolutional layers are shared between all components. Then the RPN and ROI pooling layers are built on top, followed by two fully connected layers to extract the instance-level features.

Three novel components are introduced in our Domain Adaptive Faster R-CNN. The image-level domain classifier is added after the last convolution layer and the instance-level domain classifier is added to the end of the ROI-wise features. The two classifiers are linked with a consistency loss to encourage the RPN to be domain-invariant. The final training loss of the proposed network is a summation of each individual part, which can be written as:

$$L = L_{det} + \lambda(L_{img} + L_{ins} + L_{cst}) \tag{9}$$

where λ is a trade-off parameter to balance the Faster R-CNN loss and our newly added domain adaptation components. The network can be trained in an end-to-end manner using a standard SGD algorithm. Note that the adversarial training for domain adaptation components is achieved by using the GRL layer, which automatically reverses the gradient during propagation. The overall network in Figure 2 is used in the training phase. During inference, one can remove the domain adaptation components, and simply use the original Faster R-CNN architecture with adapted weights.

5. Experiments

5.1. Experiment Setup

We adopt the unsupervised domain adaptation protocol in our experiments. The training data consists of two parts:

the *source training data* for which images and their annotations (bounding boxes and object categories) are provided, and the *target training data* for which only unlabeled images are available.

To validate the proposed approach, for all domain shift scenarios, we report the final results of our model as well as the results by combining different components (*i.e.*, image-level adaptation, instance-level adaptation, and the consistency regularization). To our best knowledge, this is the first work proposed to improve Faster R-CNN for cross-domain object detection. We include the original Faster R-CNN model as a baseline, which is trained using the source domain training data, without considering domain adaptation. For all experiments, we report mean average precisions (mAP) with a threshold of 0.5 for evaluation.

Unless otherwise stated, all training and test images are resized such that the shorter side has a length of 500 pixels to fit in GPU memory, and we set $\lambda = 0.1$ for all experiments. We follow [48] to set the hyper-parameters. Specifically, the models are initialized using weights pretrained on ImageNet. We finetune the network with a learning rate of 0.001 for 50k iterations and then reduce the learning rate to 0.0001 for another 20k iterations. Each batch is composed of 2 images, one from the source domain and one from the target domain. A momentum of 0.9 and a weight decay of 0.0005 is used in our experiments.

5.2. Experimental Results

In this section we evaluate our proposed Domain Adaptive Faster R-CNN model for object detection in three different domain shift scenarios: 1) ***learning from synthetic data***, where the training data is captured from video games, while the test data comes from the real world; 2) ***driving in adverse weather***, where the training data is taken in good weather conditions, while the test data in foggy weather; 3) ***cross camera adaptation***, where the training data and test data are captured with different camera setups.

5.2.1 Learning from Synthetic Data

As computer graphics technique advances, using synthetic data to train CNNs becomes increasingly popular. Nonetheless, synthetic data still exhibits a clear visual difference with real world images, and usually there is a performance gap with models trained on real data. Our first experiment is to investigate the effectiveness of the proposed method in this scenario. We use the *SIM 10k* [30] dataset as the source domain, and the *Cityscapes* dataset as the target domain.

Datasets: *SIM 10k* [30] consists of $10,000$ images which are rendered by the gaming engine *Grand Theft Auto*(GTAV). In *SIM 10k* , bounding boxes of $58,701$ cars are provided in the $10,000$ training images. All images are used in the training. The *Cityscapes* [5] dataset is an urban scene dataset for driving scenarios. The images are captured by a car-mounted video camera. It has $2,975$ images in the training set, and 500 images in the validation set. We use the unlabeled images from the training set as the target domain to adapt our detector, and the results are reported on the validation set. There are 8 categories with instance labels in *Cityscapes* , but only *car* is used in this experiment since only *car* is annotated in *SIM 10k* . Note that the *Cityscapes* dataset is not dedicated to detection, thus we take the tightest rectangles of its instance masks as ground-truth bounding boxes.

	img	ins	cons	car AP
Faster R-CNN				30.12
Ours	✓			33.03
		✓		35.79
	✓	✓		37.86
	✓	✓	✓	**38.97**

Table 1. The average precision (AP) of *Car* on the *Cityscapes* validation set. The models are trained using the *SIM 10k* dataset as the source domain and the *Cityscapes* training set as the target domain. *img* is short for *image-level alignment*, *ins* for *instance-level alignment* and *cons* is short for our *consistency loss*

Results: The results of the different methods are summarized in Table 1. Specifically, compared with Faster R-CNN, we achieve $+2.9\%$ performance gain using the image-level adaptation component only, and $+5.6\%$ using instance-level alignment only. This proves that our proposed image-level adaptation and instance-level adaptation components can reduce the domain shift on each level effectively. Combining those two components yields an improvement of 7.7%, which validates our conjecture on the necessity of reducing domain shifts on both levels. By further applying the consistency regularization, our Domain Adaptive Faster R-CNN model improves the Faster R-CNN model by $+8.8\%$, which achieves 38.97% in terms of AP.

5.2.2 Driving in Adverse Weather

We proceed with our evaluation by studying domain shift between weather conditions. Weather condition is an important source of domain discrepancy, as scenes are visually different as weather conditions change. Whether a detection system can perform faithfully in different weather conditions is critical for a safe autonomous driving system [44, 49]. In this section, we investigate the ability to detect objects when we adapt a model from normal to foggy weather.

Datasets: *Cityscapes* is used as our source domain, with images dominantly obtained in clear weather. In this experiment we report our results on categories with instance annotations: *person*, *rider*, *car*, *truck*, *bus*, *train*, *motorcycle* and *bicycle*.

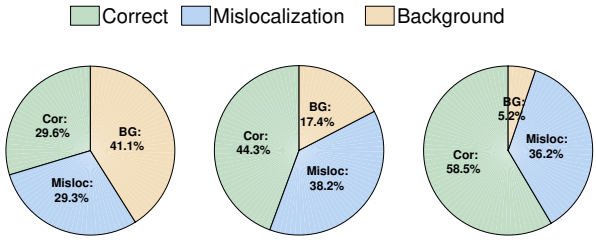
For the target domain, we use the *Foggy Cityscapes*

	img	ins	cons	person	rider	car	truck	bus	train	mcycle	bicycle	mAP
Faster R-CNN				17.8	23.6	27.1	11.9	23.8	9.1	14.4	22.8	18.8
Ours	✓			22.9	30.7	39.0	20.1	27.5	17.7	21.4	25.9	25.7
		✓		23.6	30.6	38.6	20.8	**40.5**	12.8	17.1	26.1	26.3
	✓	✓		24.2	**31.2**	39.1	19.1	36.2	19.2	17.1	27.0	26.6
	✓	✓	✓	**25.0**	31.0	**40.5**	**22.1**	35.3	**20.2**	**20.0**	**27.1**	**27.6**

Table 2. Quantitative results on the *Foggy Cityscapes* validation set, models are trained on the *Cityscapes* training set.

	img	ins	cons	K → C	C → K
Faster R-CNN				30.2	53.5
Ours	✓			36.6	60.9
		✓		34.6	57.6
	✓	✓		37.3	62.7
	✓	✓	✓	**38.5**	**64.1**

Table 3. Quantitative analysis of adaptation result between *KITTI* and *Cityscapes* . We report AP of *Car* on both directions. *e.g.* K → C and C → K.



(a) Faster RCNN (b) Ours (Ins. Only) (c) Ours (Img Only)

Figure 3. Error Analysis of Top Ranked Detections

dataset that was recently presented in [49]. *Foggy Cityscapes* is a synthetic foggy dataset in that it simulates fog on real scenes. The images are rendered using the images and depth maps from *Cityscapes* . Examples can be found at Figure 1 and also in the original paper [49]. The semantic annotations and data split of *Foggy Cityscapes* are inherited from *Cityscapes* , making it ideal to study the domain shift caused by weather condition.

Result: Table 2 presents our results and those of other baselines. Similar observations apply as in the learning from synthetic data scenario. Combining all components, our adaptive Faster R-CNN improves the baseline Faster R-CNN model by $+8.6\%$. Besides, we can see that the improvement generalizes well across different categories, which suggests that the proposed technique can also reduce domain discrepancy across different object classes.

5.2.3 Cross Camera Adaptation

Domain shift commonly exists even between real datasets taken under similar weather conditions, as different dataset are captured using different setups, with different image quality/resolution, and usually exhibit some data bias when collecting the dataset [55]. For detection, different datasets also vary drastically in scale, size and class distribution, sometimes it is difficult to determine the source of a domain shift. In this part, we focus on studying adaptation between two real datasets, as we take *KITTI* and *Cityscapes* as our datasets.

Datasets: We use *KITTI* training set which contains $7,481$ images. The dataset is used in both adaptation and evaluation. Images have original resolution of 1250×375, and are resized so that shorter length is 500 pixels long. *Cityscapes* is used as the other domain. Consistent with the first experiment, we evaluate our method using AP of *car*,

Results: We apply the proposed method in both adaptation directions, we denote *KITTI* to *Cityscapes* as $K \rightarrow C$ and vice versa. Table 3 compares our method to other baselines. A clear performance improvement is achieved by our proposed Adaptive Faster R-CNN model over other baselines. And our method is useful for both adaptation directions $K \rightarrow C$ and $C \rightarrow K$.

5.3. Error Analysis on Top Ranked Detections

In the previous sections, we have shown that both image-level and instance-level alignment help to decrease domain discrepancy. To further validate the individual effect of image-level adaptation and instance-level adaptation, we analyze the accuracies caused by most confident detections for models using adaptation components on different levels.

We use *KITTI* → *Cityscapes* as a study case. We select $20,000$ predictions with highest confidence for the vanilla Faster R-CNN model, our model with only image-level adaptation, and our model with only instance-level adaptation, respectively. Inspired by [28], we categorize the detections into 3 error types: **correct:** The detection has an overlap greater than 0.5 with ground-truth. **mis-localized:** The detection has a overlap with ground-truth of 0.3 to 0.5, and **background:** the detection has an overlap smaller than 0.3, which means it takes a background as a false positive.

The results are shown in Figure 3. From the figure we can observe that each individual component (image-level or instance-level adaptation) improves the number of correct detections (green color), and dramatically reduces the

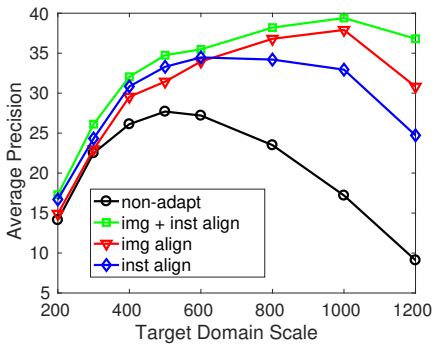


Figure 4. **AP at different scales**: Source images from *KITTI* are fixed at a scale of 500 pixels, and we resize the target images from *Cityscapes* to different scales.

number of false positives (other colors). Moreover, we also observe that the model using instance-level alignment gives higher background error than the model using image-level alignment. The reason might be the image-level alignment improves RPN more directly, which produces region proposals with better localization performance.

5.4. Image-level *v.s.* Instance-level Alignment

Image scale has shown to play a vital role in many computer vision tasks [3, 2, 6]. To further analyze the impact of image-level and instance-level adaptation, we conduct experiment on *KITTI* $\rightarrow$ *Cityscapes* by varying the image scales. Because different cameras are used in two datasets, the different camera parameters might lead to a scale drift between two domains.

In particular, we refer to the shorter length of an image as its *scale*. To study how image scale affects our two domain adaptation components, we vary the size of images in the target domain to see how this affects the behavior of the two components while the scale in the source domain is fixed to 500 pixels. For efficiency, we use a smaller VGG-M model as the backbone, and all other settings remain identical.

We plot the performance of different models in Figure 4. By varying the scale of target images, we observe that the performance of the vanilla Faster R-CNN (*i.e.*, non-adapt) drops significantly when the scales are mismatched. Comparing the two adaptation models, the image-level adaptation model is more robust to scale change than the instance-level adaptation model.

The reason behind this is that the scale change is a global transformation, which affects all instances and background. And in our design, global domain shift is mainly tackled by image-level alignment, and instance-level alignment is used to minimize instance-level discrepancy. When there is a serious global domain shift, the localization error of instance proposals goes up, thus the accuracy of instance-level alignment is damaged by deviating proposals. Nevertheless, using both always yields the best results across all scales. Contrary to the vanilla Faster R-CNN, our model can benefit from high resolution of target images, and performs increasingly better as the scale rises from 200 to $1,000$ pixels.

	Faster R-CNN	Ours(w/o)	Ours
mIoU	18.8	28.5	**30.3**

Table 4. Mean best Overlap between with groundtruth bounding boxes by top 300 proposals from RPN in different models, in which Ours(w/o) denotes our model without using consistency regularization.

5.5. Consistency Regularization

As discussed in Section 4.2, we impose a consistency regularization on domain classifiers at two different levels for learning a robust RPN. To show the benefit of using consistency regularization, we take *KITTI* $\rightarrow$ *Cityscapes* as an example to study the performance of RPN before and after using the consistency regularization in Table 4. The maximum achievable mean overlap between the top 300 proposals from RPN and the ground-truth is used for measurement. The vanilla Faster R-CNN model is also included as a baseline. As shown in the table, without using consistency regularizer, our model improves Faster R-CNN from 18.8% to 28.5% in terms of mIoU, due to the use of image-level and instance-level adaptation. By further imposing the consistency regularizer, the performance of RPN can be further improved to 30.3%, which indicates the consistency regularizer encourages the RPN to be more robust.

6. Conclusion

In this paper, we have introduced the Domain Adaptive Faster R-CNN model, an effective approach for cross-domain object detection. With our approach, one can obtain a robust object detector for a new domain without using any additional labeled data. Our approach is built on the state-of-the-art Faster R-CNN model. Based on our theoretical analysis for cross-domain object detection, we propose an image-level adaptation component and an instance-level component to alleviate the performance drop caused by domain shift. The adaptation components are based on adversarial training of $\mathcal{H}$-divergence. A consistency regularizer is further applied to learn a domain-invariant RPN. Our model can be trained end-to-end using the standard SGD optimization technique. Our approach is validated on various domain shift scenarios, and the adaptive method outperforms baseline Faster R-CNN by a clear margin, thus demonstrating its effectiveness for cross-domain object detection.

Acknowledgements This work is supported by armasuisse. Wen Li is the corresponding author. Christos Sakaridis and Dengxin Dai are supported by Toyota Motor Europe via TRACE-Zurich.

References

[1] S. Ben-David, J. Blitzer, K. Crammer, A. Kulesza, F. Pereira, and J. W. Vaughan. A theory of learning from different domains. *Machine learning*, 79(1):151–175, 2010.

[2] L.-C. Chen, Y. Yang, J. Wang, W. Xu, and A. L. Yuille. Attention to scale: Scale-aware semantic image segmentation. In *CVPR*, 2016.

[3] Y. Chen, D. Dai, J. Pont-Tuset, and L. Van Gool. Scale-aware alignment of hierarchical image segmentation. In *CVPR*, 2016.

[4] Y. Chen, W. Li, and L. Van Gool. ROAD: Reality oriented adaptation for semantic segmentation of urban scenes. In *CVPR*, 2018.

[5] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele. The Cityscapes dataset for semantic urban scene understanding. In *CVPR*, 2016.

[6] D. Dai, Y. Wang, Y. Chen, and L. Van Gool. Is image super-resolution helpful for other vision tasks? In *WACV*, 2016.

[7] J. Dai, K. He, and J. Sun. Instance-aware semantic segmentation via multi-task network cascades. In *CVPR*, 2016.

[8] J. Dai, Y. Li, K. He, and J. Sun. R-FCN: Object detection via region-based fully convolutional networks. In *NIPS*, 2016.

[9] N. Dalal and B. Triggs. Histograms of oriented gradients for human detection. In *CVPR*, 2005.

[10] L. Duan, I. W. Tsang, and D. Xu. Domain transfer multiple kernel learning. *TPAMI*, 34(3):465–479, 2012.

[11] L. Duan, D. Xu, I. W. Tsang, and J. Luo. Visual event recognition in videos by learning from web data. *TPAMI*, 34(9):1667–1680, 2012.

[12] M. Everingham, L. Van Gool, C. K. Williams, J. Winn, and A. Zisserman. The Pascal visual object classes (VOC) challenge. *IJCV*, 88(2):303–338, 2010.

[13] P. F. Felzenszwalb, R. B. Girshick, D. McAllester, and D. Ramanan. Object detection with discriminatively trained part-based models. *TPAMI*, 32(9):1627–1645, 2010.

[14] B. Fernando, A. Habrard, M. Sebban, and T. Tuytelaars. Unsupervised visual domain adaptation using subspace alignment. In *ICCV*, 2013.

[15] Y. Ganin and V. Lempitsky. Unsupervised domain adaptation by backpropagation. In *ICML*, 2015.

[16] T. Gebru, J. Hoffman, and L. Fei-Fei. Fine-grained recognition in the wild: A multi-task domain adaptation approach. *arXiv:1709.02476*, 2017.

[17] A. Geiger, P. Lenz, C. Stiller, and R. Urtasun. Vision meets robotics: The KITTI dataset. *The International Journal of Robotics Research*, 32(11):1231–1237, 2013.

[18] M. Ghifary, W. B. Kleijn, M. Zhang, D. Balduzzi, and W. Li. Deep reconstruction-classification networks for unsupervised domain adaptation. In *ECCV*, 2016.

[19] S. Gidaris and N. Komodakis. Object detection via a multi-region and semantic segmentation-aware CNN model. In *ICCV*, 2015.

[20] R. Girshick. Fast R-CNN. In *ICCV*, 2015.

[21] R. Girshick, J. Donahue, T. Darrell, and J. Malik. Rich feature hierarchies for accurate object detection and semantic segmentation. In *CVPR*, 2014.

[22] B. Gong, Y. Shi, F. Sha, and K. Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *CVPR*, 2012.

[23] R. Gopalan, R. Li, and R. Chellappa. Domain adaptation for object recognition: An unsupervised approach. In *ICCV*, 2011.

[24] P. Haeusser, T. Frerix, A. Mordvintsev, and D. Cremers. Associative domain adaptation. In *ICCV*, 2017.

[25] H. Hattori, V. Naresh Boddeti, K. M. Kitani, and T. Kanade. Learning scene-specific pedestrian detectors without real data. In *CVPR*, 2015.

[26] K. He, X. Zhang, S. Ren, and J. Sun. Spatial pyramid pooling in deep convolutional networks for visual recognition. In *ECCV*, 2014.

[27] J. Hoffman, D. Wang, F. Yu, and T. Darrell. FCNs in the wild: Pixel-level adversarial and constraint-based adaptation. *arXiv:1612.02649*, 2016.

[28] D. Hoiem, Y. Chodpathumwan, and Q. Dai. Diagnosing error in object detectors. In *ECCV*, 2012.

[29] J. Johnson, A. Alahi, and L. Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *ECCV*, 2016.

[30] M. Johnson-Roberson, C. Barto, R. Mehta, S. N. Sridhar, K. Rosaen, and R. Vasudevan. Driving in the matrix: Can virtual worlds replace human-generated annotations for real world tasks? In *ICRA*, 2017.

[31] T. Kim, M. Cha, H. Kim, J. Lee, and J. Kim. Learning to discover cross-domain relations with generative adversarial networks. In *ICCV*, 2017.

[32] A. Krizhevsky, I. Sutskever, and G. E. Hinton. Imagenet classification with deep convolutional neural networks. In *NIPS*, 2012.

[33] B. Kulis, K. Saenko, and T. Darrell. What you saw is not what you get: Domain adaptation using asymmetric kernel transforms. In *CVPR*, 2011.

[34] D. Li, Y. Yang, Y.-Z. Song, and T. M. Hospedales. Deeper, broader and artier domain generalization. In *ICCV*, 2017.

[35] W. Li, Z. Xu, D. Xu, D. Dai, and L. Van Gool. Domain generalization and adaptation using low rank exemplar SVMs. *TPAMI*, 2017.

[36] T.-Y. Lin, P. Dollár, R. Girshick, K. He, B. Hariharan, and S. Belongie. Feature pyramid networks for object detection. *arXiv:1612.03144*, 2016.

[37] T.-Y. Lin, M. Maire, S. Belongie, J. Hays, P. Perona, D. Ramanan, P. Dollár, and C. L. Zitnick. Microsoft COCO: Common objects in context. In *ECCV*, 2014.

[38] M.-Y. Liu, T. Breuel, and J. Kautz. Unsupervised image-to-image translation networks. In *NIPS*, 2017.

[39] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. Reed, C.-Y. Fu, and A. C. Berg. SSD: Single shot multibox detector. In *ECCV*, 2016.

[40] M. Long, Y. Cao, J. Wang, and M. I. Jordan. Learning transferable features with deep adaptation networks. In *ICML*, 2015.

[41] H. Lu, L. Zhang, Z. Cao, W. Wei, K. Xian, C. Shen, and A. van den Hengel. When unsupervised domain adaptation meets tensor representations. In *ICCV*, 2017.

[42] F. Maria Carlucci, L. Porzi, B. Caputo, E. Ricci, and S. Rota Bulo. AutoDIAL: Automatic domain alignment layers. In *ICCV*, 2017.

[43] S. Motiian, M. Piccirilli, D. A. Adjeroh, and G. Doretto. Unified deep supervised domain adaptation and generalization. In *ICCV*, 2017.

[44] S. G. Narasimhan and S. K. Nayar. Vision and the atmosphere. *IJCV*, 48(3):233–254, 2002.

[45] P. Panareda Busto and J. Gall. Open set domain adaptation. In *ICCV*, 2017.

[46] X. Peng, B. Sun, K. Ali, and K. Saenko. Learning deep object detectors from 3D models. In *ICCV*, 2015.

[47] A. Raj, V. P. Namboodiri, and T. Tuytelaars. Subspace alignment based domain adaptation for RCNN detector. In *BMVC*, 2015.

[48] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*, 2015.

[49] C. Sakaridis, D. Dai, and L. Van Gool. Semantic foggy scene understanding with synthetic data. *IJCV*, 2018.

[50] O. Sener, H. O. Song, A. Saxena, and S. Savarese. Learning transferrable representations for unsupervised domain adaptation. In *NIPS*, 2016.

[51] P. Sermanet, D. Eigen, X. Zhang, M. Mathieu, R. Fergus, and Y. LeCun. OverFeat: Integrated recognition, localization and detection using convolutional networks. *arXiv:1312.6229*, 2013.

[52] B. Sun, J. Feng, and K. Saenko. Return of frustratingly easy domain adaptation. In *AAAI*, 2016.

[53] B. Sun and K. Saenko. From virtual to reality: Fast adaptation of virtual object detectors to real domains. In *BMVC*, 2014.

[54] K. Tang, V. Ramanathan, L. Fei-Fei, and D. Koller. Shifting weights: Adapting object detectors from image to video. In *NIPS*, 2012.

[55] A. Torralba and A. A. Efros. Unbiased look at dataset bias. In *CVPR*, 2011.

[56] P. Viola and M. Jones. Rapid object detection using a boosted cascade of simple features. In *CVPR*, 2001.

[57] Y. Wang, W. Li, D. Dai, and L. Van Gool. Deep domain adaptation by geodesic distance minimization. *arXiv:1707.09842*, 2017.

[58] J. Xu, S. Ramos, D. Vázquez, and A. M. Lopez. Domain adaptation of deformable part-based models. *TPAMI*, 36(12):2367–2380, 2014.

[59] Z. Yi, H. Zhang, P. T. Gong, et al. DualGAN: Unsupervised dual learning for image-to-image translation. In *ICCV*, 2017.

[60] L. Zhang, L. Lin, X. Liang, and K. He. Is faster R-CNN doing well for pedestrian detection? In *ECCV*, 2016.

[61] Y. Zhang, P. David, and B. Gong. Curriculum domain adaptation for semantic segmentation of urban scenes. In *ICCV*, 2017.

[62] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017.