

Constructing Self-motivated Pyramid Curriculums for Cross-Domain Semantic Segmentation: A Non-Adversarial Approach

Qing Lian¹, Fengmao Lv¹, Lixin Duan¹, Boqing Gong²

¹University of Electronic Science and Technology of China ²Google

{lianqing1997, lxduan}@gmail.com, fengmaolv@126.com, boqinggo@outlook.com

Abstract

We propose a new approach, called *self-motivated pyramid curriculum domain adaptation (PyCDA)*, to facilitate the adaptation of semantic segmentation neural networks from synthetic source domains to real target domains. Our approach draws on an insight connecting two existing works: curriculum domain adaptation and self-training. Inspired by the former, PyCDA constructs a pyramid curriculum which contains various properties about the target domain. Those properties are mainly about the desired label distributions over the target domain images, image regions, and pixels. By enforcing the segmentation neural network to observe those properties, we can improve the network's generalization capability to the target domain. Motivated by the self-training, we infer this pyramid of properties by resorting to the semantic segmentation network itself. Unlike prior work (e.g., logistic regression or discriminator networks) or to solve min max problems which are often difficult to optimize. We report state-of-the-art results for the adaptation from both GTAV and SYNTHIA to Cityscapes, two popular settings in unsupervised domain adaptation for semantic segmentation¹.

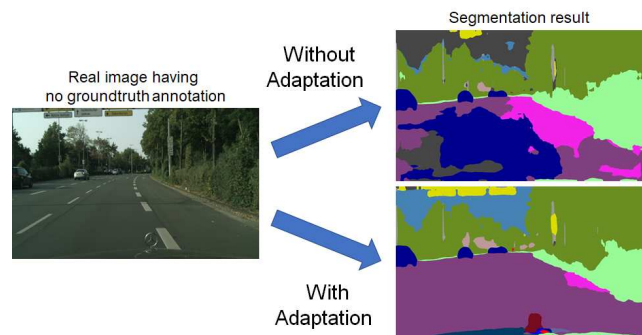
1. Introduction

The objective of semantic segmentation is to assign a semantic label to every pixel of an image. Over the past few years, a great amount of effort has been made by the community to tackle this problem [23, 1, 41, 3, 21], leading to sophisticated and high-performing deep convolutional neural networks as the main solution. However, to collect and label images for training such networks is a very daunting work [6]. To alleviate the heavy annotation burden, a promising alternative is to employ photo-realistic simulators to efficiently collect and label training data. Richter et al. [28] use the GTAV game engine to aid user annotations,

¹Code available at <https://github.com/lianqing11/pycda>



(a) Synthetic images with labeling-free pixel-wise groundtruth annotations.



(b) Segmentation results of a real image with and without adaptation.

Figure 1: Unsupervised domain adaptation for semantic segmentation. The segmentation results for real images can be significantly improved by explicit domain adaptation techniques when we adapt a segmentation model trained using synthetic imagery.

resulting a dataset of 25 thousand synthetic urban scene images labeled in only 49 hours (about 7 seconds per image).

The clear visual mismatch between the synthetic (source) domains and real (target) domains (cf. Figs. 3 and 4), however, inevitably causes significant performance degradation when one applies the model trained on the source domain to the real images of the target domain.

In order to better take advantage of the synthetic im-

agery for the semantic segmentation of real scenes, we propose a novel domain adaptation approach by drawing an intriguing connection between two previous works: curriculum domain adaptation [39, 38] and self-training [43]. This connection naturally leads to self-motivated pyramid curriculums and a new training algorithm for the cross-domain adaptation of semantic segmentation networks. Compared with the prevalent adversarial training methods in domain adaptation [4, 18, 40, 35, 27, 5, 17, 19, 33], our approach gives rise to results on par with or better than theirs and yet is lighter-weight, without the need of learning extra discriminator nets, and easier to optimize, without the need of solving any min max problems. More importantly, it outperforms both the original curriculum adaptation [38] and the original self-training method [43].

In particular, we view self-training from the perspective of curriculum domain adaptation. They share the same algorithmic format. On the one hand, the self-training alternates between two sub-tasks: 1) estimating pseudo labels for the target domain’s pixels and 2) updating the weights of the segmentation network by using both the source labels and the pseudo target labels. On the other hand, the curriculum adaptation first 1) constructs a curriculum, i.e., infers properties of the target domain in the form of frequency distributions of the class labels over an image (or image region) and then 2) updates the network’s weights using the source labels and the target domain’s properties. Due to this analogy, we may view the pseudo labels in self-training as one of the properties about the target domain. More interestingly, the second steps of the two works share exactly the same form in math — a cross-entropy loss between a frequency distribution / pseudo label and a differentiable function of the network’s predictions.

Immediately, our approach follows the above analogy. We add the pseudo labels in self-training to the curriculum as the finest layer of properties about the target domain images. On top of that, we build a pyramid of which a layer comprises image regions of a certain size. This pyramid design resembles the original curriculum domain adaptation work, in which the frequency distributions are counted over a global image and some superpixels, respectively — in other words, a simple two-layer pyramid.

In addition to enriching the original curriculum by the pseudo labels, we also improve it in two ways. One is to replace the superpixels by small squared regions to significantly save computation cost. The other is to infer the target domain properties — label distributions over the squared regions and full target domain images — by the semantic segmentation network itself. In each iteration of the training phase, we infer those properties from the network’s pixel-wise predictions over the target domain images and then use a loss defined over those properties to update the network by backpropagation.

Our main contribution is two-fold. One is that we provide a new insight connecting the self-training for adapting segmentation networks [43] and the curriculum adaptation method [39, 38]. The other is that, inspired by the connection, we propose a novel self-motivated pyramid curriculum for the domain adaptation of semantic segmentation networks. Extensive experiments show that it outperforms either [43] or [38] individually. Moreover, it is on par with or better than state-of-the-art adversarial adaptation methods without the need of maintaining an extra discriminator network or carefully tuning the optimization procedure for min max problems.

2. Related Work

Semantic segmentation. Semantic segmentation is the task that assigns labels in pixel level for an image which plays a vital role in lots of tasks including autonomous driving, disease detection, etc. In the following, we briefly review some of the works with a focus on CNN-based methods. Driven by the powerful deep neural networks [16], pixel-level prediction tasks achieve great progress mostly following the design of replacing the softmax layer in classification with the pixel-wise softmax [23]. To enlarge the receptive fields and feature resolutions, methods of [1, 2, 3, 41] employ dilated convolution. To utilize different context information and multiple features, some extend the dilated convolution to a pyramid [1, 2, 3] or resize it with multiple sizes [41].

Domain adaptation. A basic assumption in conventional machine learning is that the training and test data are drawn i.i.d. from the same underlying distribution. However, this does not always hold in real world scenarios, resulting in significant performance drops when the training and test data exist distribution mismatches. Domain adaptation aims to rectify this mismatch and make the model generalize well onto the test domain. Domain adaptation has been mostly addressed for image classification problems in computer vision [8, 9, 15, 14, 13, 24, 25]. Recent works start to study deep neural networks including learning domain-invariant models [13, 32, 12] and target specific models [30, 11].

Domain adaptation for semantic segmentation. Recent work on domain adaptation mainly proceeds in two directions. One is based on a curriculum learning strategy. Zhang et al. first learns to solve easy tasks in the target domain and then use them to regularize semantic segmentation [39, 38]. Dai et al. construct a curriculum by simulating foggy images of different fog densities [7]. In this work, we also view the self-training [43] as a curriculum-style domain adaptation method. The other line of work is to reduce the domain shift in feature space or output space and tries to seek a better way to align both domains in an intermediate layer [4, 18, 40, 35, 27, 42, 5, 17, 19, 33, 26]. In [18], Hoffman et al. combine global and local alignment methods with

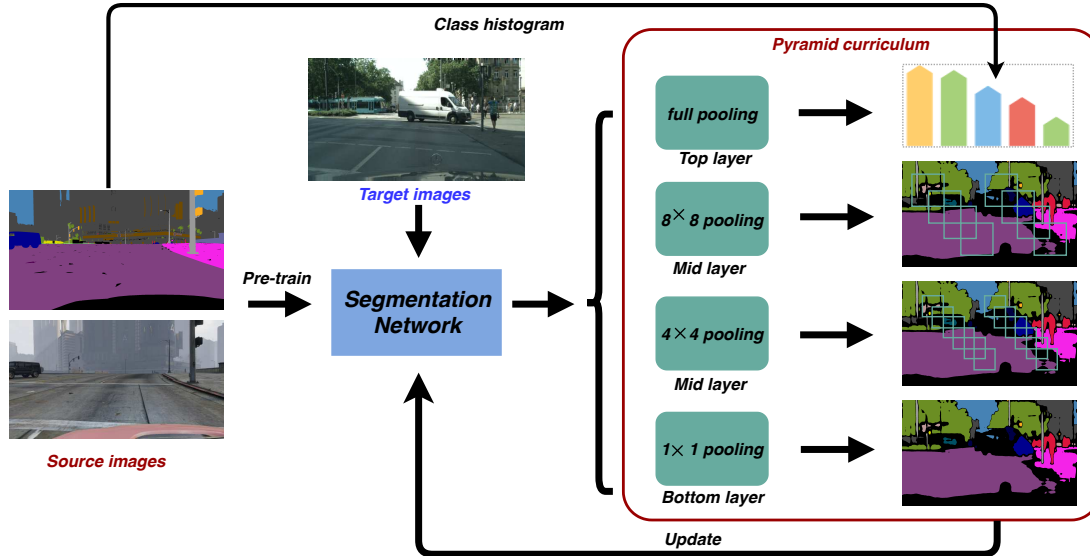


Figure 2: Overview of our self-motivated pyramid curriculum domain adaptation (PyCDA) approach to segmentation.

a domain adversarial training loss. In [5], Chen et al. further propose a global and class-specific feature alignment approach guided by the soft pseudo labels in the target domain. In [35], Tsai et al. proposed to align both domains at the structured output space level. In [4, 40], they focus on utilizing spatial information to help the discriminator distinguish the domains better. In [26, 42], they focus on solving the problem of class boundary or outliers in the domain adaptation. In [17, 27, 33], they propose to learn domain adaptive segmentation networks through directly translating the source images to the target ones at the pixel level. We refer readers to [38, Section 5] for a more thorough review.

3. Approach

In this section, we first reveal the connection between the curriculum domain adaptation (CDA) [38] and the self-training (ST) for adaptation [43]. This connection naturally leads to the training algorithm of this paper, dubbed self-motivated pyramid curriculum domain adaptation (PyCDA), for the semantic segmentation task.

3.1. CDA vs. ST

Denote by $I_t \in \mathbb{R}^{H \times W}$ a target domain image and $\hat{Y}_t \in \mathbb{R}^{H \times W \times C}$ the corresponding output of a semantic segmentation network, where H , W , and C are respectively the height, width, and number of possible classes of the input image. Most neural segmentation networks employ a pixel-wise softmax function at the output layer so that $\hat{Y}_t(i, j) \in \mathbb{R}^C$ is a probability vector satisfying $\sum_c \hat{Y}_t(i, j, c) = 1, \forall i, j$ and $\hat{Y}_t(i, j, c) \geq 0, \forall i, j, c$. Similarly, denote by I_s and \hat{Y}_s a source domain image and the corresponding prediction by a network. In unsupervised do-

main adaptation, the groundtruth labels $\{Y_s\}$ of the source domain (S) are given but the learner has no access to the labels of the target domain (T).

CDA [38] learns a semantic segmentation network by minimizing the following objective function:

$$\min \sum_{s \in S} \mathcal{L}(Y_s, \hat{Y}_s) + \lambda \sum_{t \in T} \sum_{k \in \mathcal{P}_t^1} \mathcal{C}(p_t^k, \hat{p}_t^k) \quad (1)$$

where the first term sums up pixel-wise cross-entropy losses over the source domain images ($s \in S$), each summand of the second term is a cross-entropy loss over a target domain image ($t \in T$) between two label distributions which indicate the proportion of each class in the image t or in a region of it, and the set \mathcal{P}_t^1 collects all the label distributions for the target image t . As a concrete example, a desired label distribution p_t^0 over a target domain image I_t is calculated by

$$p_t^0(c) = \frac{1}{WH} \sum_{i=1}^W \sum_{j=1}^H Y_t(i, j, c), \quad \forall c, \quad (2)$$

and the property \hat{p}_t^0 predicted by the segmentation network, which can be obtained in a similar way as above from the network prediction \hat{Y}_t , is supposed to match the desired label distribution p_t^0 . In this example, the label distribution over a full image captures a *global* property. The others calculated within an image region capture the *local* properties of a target domain image. Accurate readers may wonder how to estimate the desired properties $p_t^k, k \in \mathcal{P}_t^1$, in practice because the target labels are actually unknown in unsupervised domain adaptation; we explain it below.

The name of CDA attributes to the following easy-to-difficult curriculum. Compared with the pixel-wise predictions, it is relatively easy to obtain these label distributions,

especially when the images are about urban scenes which share common objects and spatial layouts. Zhang et al. [38] train separate logistic regression models and support vector machines to estimate these distributions $p_t^k, k \in \mathcal{P}_t^1$. In this paper, however, we estimate them in a self-training fashion by the segmentation network itself.

Self-training (ST) [43] considers the unknown labels of the target domain images as latent variables. It alternates between 1) inferring the values of the latent target labels and 2) updating the network’s weights. The second step essentially solves the following problem:

$$\min \sum_{s \in S} \mathcal{L}(Y_s, \hat{Y}_s) + \lambda \sum_{t \in T} \sum_{(i,j) \in \mathcal{P}_t^2} \mathcal{C}(Y_t(i,j), \hat{Y}_t(i,j)),$$

where \mathcal{P}_t^2 is the set of pixels of the target image I_t whose pseudo labels $\{Y_t(i,j)\}$ are inferred — the other pixels are with null labels because the network predictions at those positions are probably below a certain threshold.

Connection between the two. The objective functions of CDA and ST are remarkably alike. The only difference between them is on the sets \mathcal{P}_t^1 and \mathcal{P}_t^2 , where the former collects label distributions over the global target domain image I_t and some of its local regions, and the latter is the pixels of I_t that are pseudo-labeled by the first step in ST. *What if we union the two sets, i.e., $\mathcal{P}_t^1 \cup \mathcal{P}_t^2$?* They indeed seem like mutually complementary. Thanks to CDA, we conjecture that the finest-grained pixel-level pseudo labels P_t^2 may be enhanced by the label distributions of the fine-grained image regions and the coarsest-grained full image. Due to ST, we reasonably expect the label distributions may be derived from the segmentation network itself, without the need of resorting to any additional models (e.g., logistic regression or SVM used in [38]). Following this line of reasoning, we devise our approach as the following.

3.2. Self-motivated pyramid CDA (PyCDA)

We propose a self-motivated pyramid curriculum for the domain adaptation (PyCDA) of semantic segmentation tasks. The idea faithfully follows the insight above, i.e., we union the two sets $\mathcal{P}_t^1 \cup \mathcal{P}_t^2$ used in CDA and ST, respectively. This union results in a pyramid with at least three layers: pixels on the bottom layer, small image regions in the middle, and a full image on the top. Note that this pyramid is built for a target domain image, not source domain images. Similarly to CDA, we will infer the label distribution over the top-layer full image and label distributions, or more concretely one-hot vectors, for the middle-layer small image regions. Similarly to ST, we want to assign pseudo labels to some of the bottom-layer pixels. Please refer to Fig. 2 for an overview of the proposed PyCDA approach.

Superpixels vs. pixel squares. Before describing the main approach, we first discuss how to partition an image into

small regions. Zhang et al. employ in their original CDA work [38] non-overlapping superpixels, which incur additional computation overhead. We replace the superpixels by overlapped squares instead. While the pixel squares do not track object boundaries, their squared shape enables fast GPU computation as demonstrated below when we infer the label distributions for them. Moreover, as the squares are sufficiently small, most of them each cover pixels of the same class. In the experiments, we use the squares of 4×4 and 8×8 for the middle layers of the pyramid in PyCDA.

Self-motivated inference of target domain properties. In order to estimate the pseudo label $Y_t(i,j)$ of a target domain pixel (i,j) , we use as simple as a thresholding method. Denoting by $c^* \leftarrow \arg \max_c \hat{Y}_t(i,j,c)$, we have

$$Y_t(i,j) = \begin{cases} c^* & \text{if } \hat{Y}_t(i,j,c^*) > 0.5 \\ \text{null} & \text{otherwise} \end{cases} \quad (3)$$

where \hat{Y}_t is the output of a segmentation network at a training iteration. We say a pixel survives this step and will add it to the bottom layer of the pyramid curriculum, if its pseudo label is not null. Alternatively, one may use the self-paced policy design [43] to estimate the pseudo labels.

We employ a similar strategy to decide to which class each pixel square belongs to. Denote by (i_0, j_0) a square (e.g., the coordinates of the top-left corner of this square). We take an average pooling of the network’s predictions

$$\hat{Y}_{\text{square}}(i_0, j_0, c) \leftarrow \text{mean}_{(i,j) \in \text{square}}(\hat{Y}_t(i,j,c)) \quad (4)$$

and then threshold the pooled value $\hat{Y}_{\text{square}}(i_0, j_0, c)$ to determine the label of the square, i.e., by replacing \hat{Y}_t with \hat{Y}_{square} in eq. (3). In implementation, we leverage efficient GPU operations by adding an average pooling layer after the network’s output layer. This label for a pixel square can be converted to a one-hot vector and regarded as a label distribution over this square.

Finally, we want to estimate the label distribution over a full target domain image. Zhang et al. [38] gives a few candidate methods for this sub-task, and we use the most computation-efficient one in this work. Particularly, no matter for which target domain image, we transfer to it the mean of the label distributions of all source domain images. As shown in the the experiments of [38], this actually gives rise to results on par with learning a logistic regression model from the source domain probably because the images of urban scenes share common objects and layouts. For the target domains beyond urban scenes, alternative sophisticated algorithms are desired to reliably estimate the label distributions for the target domain images.

PyCDA. We are now ready to present the overall objective function of the proposed PyCDA approach:



Figure 3: Sample images from the GTAV [28] and SYNTHIA [29] datasets.

$$\min \frac{1}{|S|} \sum_{s \in S} \mathcal{L}(Y_s, \hat{Y}_s) + \frac{\lambda_1}{|T|} \sum_{t \in T} \mathcal{C}(p_t^0, \hat{p}_t^0) + \frac{\lambda_2}{|\mathcal{P}|} \sum_{(t,k) \in \mathcal{P}} \mathcal{C}(Y_t^k, \hat{Y}_t^k), \quad (5)$$

where λ_1 and λ_2 are pre-defined trade-off parameters, and $\mathcal{P} = \{(t, k) | \forall t \in T, k \in \mathcal{P}_t^1 \cup \mathcal{P}_t^2\}$ denotes the label distributions (resp., pseudo labels) over the squared regions (resp., pixels) of the target images. Note that in Eq. (5), the second term is a cross-entropy loss defined on the label distributions over target images, and the last term takes care of the pixels and squared regions of the target domain. In the experiments, we set $\lambda_1 = 1$, $\lambda_2 = 0.5$ and tune other free parameters (e.g., learning rate) via a validation set.

Effect of the pyramid curriculum. We have described a pyramid curriculum which consists of the global target domain images on the top, pixels at the bottom, and pixel squares in between. From the CDA point of view [38], the label distributions over the top-layer images macroscopically hint the network *how* to update its predictions while the label distributions over the middle-layer pixel squares microscopically indicate the network *where* to update. The pseudo labels of bottom-layer pixels give even more precise supervision to the network. From the ST perspective [43], the middle-layer pixel squares may be viewed as a way of consensus voting so that the pseudo labels (of the squares) can be estimated more reliably than thresholding the prediction at an isolated pixel. The label distributions over the target domain images act like a prior over the classes, playing a similar role to the class-balanced formulation in [43].

4. Experiments

In this section, we conduct extensive experiments on simulation-to-real unsupervised domain adaptation for the semantic segmentation task. We compare the proposed PyCDA with several state-of-the-art methods. A majority of

them uses adversarial training to bring closer the source and target domains on the feature level (ROAD [4]), on both features and pixels (FCAN [40], CyCADA [17]), on the output maps (OutputAdapt [35], CLAN [26]), and combining adversarial learning with entropy minimization (ADVENT [36]). In contrast, our PyCDA approach, along with CDA [38], and ST [43], adapts the neural networks by posterior regularization instead. The difference between the adversarial training methods and ours is significant for practical applications because the min max adversarial problems are often hard to optimize and have to maintain an additional discrimination network. Please note that we only compare results obtained from single models without using any ensemble strategy.

4.1. Experimental setup

We follow the experimental setup of previous works [38, 4, 40, 43] and use the standard benchmark settings (i.e., “GTAV to Cityscapes” and “SYNTHIA to Cityscapes”) in the experiments.

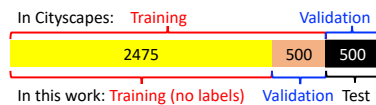
- **Cityscapes** [6] is a popular dataset to benchmark semantic segmentation models. The images are collected in the real world by vehicle-carried cameras. This dataset focuses on urban scenes, covering 50 cities in Germany and nearby countries. Its official data partition has 2,975 images in the training set, 500 in the validation set and 1,175 in the test set. In total, 19 classes of the semantic labels are compatible with GTAV and 16 with SYNTHIA.
- **GTAV** [28] is a large-scale dataset with 24,966 synthetic urban scene images collected from a near-realistically rendered computer game called Grand Theft Auto V (GTA or GTAV). We consider all the 19 semantic classes of GTAV for the adaptation to Cityscapes.
- **SYNTHIA** [29] is another synthetic image dataset and provides a particular subset, called SYNTHIA-

Table 1: Comparison results (in %) adapting from GTAV to Cityscapes. All the prior methods but CDA use the whole training set of Cityscapes in training — thus their models see more target images than ours — and do not leave out a separate validation set for model selection. Unlike the others, FCAN directly works with the original image size. OutputAdapt (ResNet-101) pre-trains the model using both ImageNet [31] and MS COCO [22].

Network	Method	road	sdwk	bldng	wall	fence	pole	light	sign	veg	trm	sky	psn	rider	car	truck	bus	train	moto	bike	mIoU	
VGG-16	Source only [32]	25.9	10.9	50.5	3.3	12.2	25.4	28.6	13.0	78.3	7.3	63.9	52.1	7.9	66.3	5.2	7.8	0.9	13.7	0.7	24.9	
	Source only (ours)	56.0	12.2	71.6	8.5	17.8	19.5	14.5	3.1	73.2	3.8	46.0	38.8	4.4	70.7	15.1	2.5	2.2	1.4	0.1	24.3	
	CDA [38]	72.9	30.0	74.9	12.1	13.2	15.3	16.8	14.1	79.3	14.5	75.5	35.7	10.0	62.1	20.6	19.0	0.0	19.3	12.0	31.4	
	ST [43]	83.8	17.4	72.1	14.3	2.9	16.5	16.0	6.8	81.4	24.2	47.2	40.7	7.6	71.7	10.2	7.6	0.5	11.1	0.9	28.1	
	CBST [43]	66.7	26.8	73.7	14.8	9.5	28.3	25.9	10.1	75.5	15.7	51.6	47.2	6.2	71.9	3.7	2.2	5.4	18.9	32.4	30.9	
	ROAD [4]	85.4	31.2	78.6	27.9	22.2	21.9	23.7	11.4	80.7	29.3	68.9	48.5	14.1	78.0	19.1	23.8	9.4	8.3	0.0	35.9	
	CyCADA [17]	85.2	37.2	76.5	21.8	15.0	23.8	22.9	21.5	80.5	31.3	60.7	50.5	9.0	76.9	17.1	28.2	4.5	9.8	0.0	35.4	
	CLAN [26]	88.0	30.6	79.2	23.4	20.5	26.1	23.0	14.8	81.6	34.5	72.0	45.8	7.9	80.5	26.6	29.9	0.0	10.7	0.0	36.6	
	ADVENT [36]	86.8	28.5	78.1	27.6	24.2	20.7	19.3	8.9	78.8	29.3	69.0	47.9	5.9	79.8	25.9	34.1	0.0	11.3	0.3	35.6	
	PyCDA (ours)	86.7	24.8	80.9	21.4	27.3	30.2	26.6	21.1	86.6	28.9	58.8	53.2	17.9	80.4	18.8	22.4	4.1	9.7	6.2	37.2	
ResNet-38	Source only [43]	70.0	23.7	67.8	15.4	18.1	40.2	41.9	25.3	78.8	11.7	31.4	62.9	29.8	60.1	21.5	26.8	7.7	28.1	12.0	35.4	
	ST [43]	90.1	56.8	77.9	28.5	23.0	41.5	45.2	39.6	84.8	26.4	49.2	59.0	27.4	82.3	39.7	45.6	20.9	34.8	46.2	41.5	
	CBST [43]	86.8	46.7	76.9	26.3	24.8	42.0	46.0	38.6	80.7	15.7	48.0	57.3	27.9	78.2	24.5	49.6	17.7	25.5	45.1	45.2	
	PyCDA (ours)	92.3	49.2	84.4	33.4	30.2	33.3	37.1	35.2	86.5	36.9	77.3	63.3	30.5	86.6	34.5	40.7	7.9	17.6	35.5	48.0	
ResNet-101	Source only [35]	75.8	16.8	77.2	12.5	21.0	25.5	30.1	20.1	81.3	24.6	70.3	53.8	26.4	49.9	17.2	25.9	6.5	25.3	36.0	36.6	
	Source only (ours)	73.8	16.0	66.3	12.8	22.3	29.0	30.3	10.2	77.7	19.0	50.8	55.2	20.4	73.6	28.3	25.6	0.1	27.5	12.1	34.2	
	ROAD [4]	76.3	36.1	69.6	28.6	22.4	28.6	29.3	14.8	82.3	35.3	72.9	54.4	17.8	78.9	27.7	30.3	4.0	24.9	12.6	39.4	
	OutputAdapt [35]	86.5	36.0	79.9	23.4	23.3	23.9	35.2	14.8	83.4	33.3	75.6	58.5	27.6	73.7	32.5	35.4	3.9	30.1	28.1	42.4	
	FCAN [40]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	46.6
	CLAN [26]	87.0	27.1	79.6	27.3	23.3	28.3	35.5	24.2	83.6	27.4	74.2	58.6	28.0	76.2	33.1	36.7	6.7	31.9	31.4	43.2	
	ADVENT [36]	87.6	21.4	82.0	34.8	26.2	28.5	35.6	23.0	84.5	35.1	76.2	58.6	30.7	84.8	34.2	43.4	0.4	28.4	35.3	44.8	
PyCDA (ours)	90.5	36.3	84.4	32.4	28.7	34.6	36.4	31.5	86.8	37.9	78.5	62.3	21.5	85.6	27.9	34.8	18.0	22.9	49.3	47.4		

RANDCITYSCAPES, to pair with Cityscapes. This subset contains 9,400 images which are automatically labeled with 12 object categories, one void class, and some unnamed classes. Following [38], we manually align four unnamed classes with their counterparts in Cityscapes, forming 16 common classes between SYNTHIA and Cityscapes.

In this work, we consider Cityscapes containing real images as the target domain, while GTAV and SYNTHIA are respectively used as the source domain (sample images from the two domains are shown in Fig. 3). Since the groundtruth labels from the official test set of Cityscapes are not publicly available, by strictly following [38], we take the official validation set as our test set for final evaluation. Also, 500 images are randomly selected from the official training set for validation, and the remaining 2,475 images are served as unlabeled training data from the target domain.



Evaluation. We directly use the evaluation code released alongside with Cityscapes, where PASCAL VOC intersection-over-union (IoU) [10] is used as the evaluation metric. Specifically, for each class, we have $\text{IoU} = \frac{\text{TP}}{\text{TP} + \text{FP} + \text{FN}}$, where TP, FP and FN are the numbers of true positive, false positive and false negative pixels, respectively, over the whole test set. In addition to the per-class

IoUs, we also report the mean of those IoUs (i.e., mIoU) over all classes. Note that in the experiments, we resize the images before feeding them to the segmentation network, so we resize the output segmentation mask back to the original size when running the evaluation code.

Implementation details. Since existing state-of-the-art methods use different base segmentation networks as their backbones, we employ the following ones for a wider range of comparison: 1) FCN8s [23] with VGG-16 [34]; 2) ResNet-38 [37]; and 3) PSP-Net [41] with ResNet-101 [16]. All the base networks are pre-trained on ImageNet [31]. Regarding the data pre-processing, we firstly resize images to the same width (1024) while preserving the original aspect ratios. During training, we randomly crop regions and feed them to the network. During testing, we feed the whole images (whose width are 1024) to the network. During evaluation, we resize the output segmentation mask back to the original image size (2048 × 1024) in order to calculate the mIoUs. Regarding the training pipeline, we firstly train the model in the source images with 30000 iterations. And then we fine-tune the model using our PyCDA framework with another 30000 iterations. The training is optimized SGD with momentum of 0.9. Using the validation data, we set our initial learning rate = 0.016 and decrease it ten times in the fine-tuning stage. In the test stage, we apply adabn [20] that change the mean and variance of batch-normalization layers, which were computed over the images of both do-

Table 2: Comparison results (in %) adapting from SYNTHIA to Cityscapes. mIoU* denotes the mean IoU over 13 classes excluding those marked with *. All the prior methods but CDA use the whole training set of Cityscapes in training — thus their models see more target images than ours — and do not leave out a separate validation set for model selection.

Network	Method	road	sdwk	blng	wall*	fence*	pole*	light	sign	veg	sky	psn	rider	car	bus	mcycl	bcycl	mIoU	mIoU*
VGG -16	Source only [4]	4.7	11.6	62.3	10.7	0.0	22.8	4.3	15.3	68.0	70.8	49.7	6.4	60.5	11.8	2.6	4.3	25.4	28.7
	Source only (ours)	50.1	20.0	49.4	0.0	0.0	16.3	0.0	0.0	69.9	54.2	43.9	4.7	43.1	6.1	0.1	0.1	22.4	26.3
	CDA [38]	57.4	23.1	74.7	0.5	0.6	14.0	5.3	4.3	77.8	73.7	45.0	11.0	44.8	21.2	1.9	20.3	29.7	35.4
	ST [43]	0.2	14.5	53.8	1.6	0.0	18.9	0.9	7.8	72.2	80.3	48.1	6.3	67.7	4.7	0.2	4.5	23.9	27.8
	CBST [43]	69.6	28.7	69.5	12.1	0.1	25.4	11.9	13.6	82.0 81.9	49.1	14.5	66.0	6.6	3.7	32.4	35.4	40.4	
	ROAD [4]	77.7	30.0	77.5	9.6	0.3	25.8	10.3	15.6	77.6	79.8	44.5	16.6	67.8	14.5	7.0	23.8	36.2	41.8
	CLAN [26]	80.4	30.7	74.7	-	-	-	1.4	8.0	77.1	79.0	46.5	8.9	73.8	18.2	2.2	9.9	-	39.3
	ADVENT [36]	67.9	29.4	71.9	6.3	0.3	19.9	0.6	2.6	74.9	74.9	35.4	9.6	67.8	21.4	4.1	15.5	31.4	36.6
	PyCDA (ours)	80.6	26.6	74.5	2.0	0.1	18.1	13.7	14.2	80.8	71.0	48.0	19.0	72.3	22.5	12.1	18.1	35.9	42.6
ResNet -101	Source only [35]	55.6	23.8	74.6	-	-	-	6.1	12.1	74.8	79.0	55.3	19.1	39.6	23.3	13.7	25.0	-	38.6
	Source only (ours)	55.6	22.7	68.6	4.3	0.1	23.0	5.6	9.1	77.2	75.9	54.7	8.7	81.5	23.9	8.4	8.8	33.0	38.5
	OutputAdapt [35]	84.3	42.7	77.5	-	-	-	4.7	7.0	77.9	82.5	54.3	21.0	72.3	32.2	18.9	32.3	-	46.7
	CLAN [26]	81.3	37.0	80.1	-	-	-	16.1	13.7	78.2	81.5	53.4	21.2	73.0	32.9	22.6	30.7	-	47.8
	ADVENT [36]	85.6	42.2	79.7	8.7	0.4	25.9	5.4	8.1	80.4	84.1	57.9	23.8	73.3	36.4	14.2	33.0	41.2	48.0
PyCDA (ours)	75.5	30.9	83.3	20.8	0.7	32.7	27.3	33.5	84.7	85.0	64.1	25.4	85.0	45.2	21.2	32.0	46.7	53.3	

mains during training, to the mean and variance over the target domain only.

4.2. Results on “GTAV to Cityscapes”

We report the results of unsupervised domain adaptation from GTAV to Cityscapes compared with existing state-of-the-arts in Table 1. Note that all the prior methods but CDA use the whole training set of Cityscapes in training — *implying their models see more target images than ours* — and do not leave out a separate validation set for model selection. Unlike the others, FCAN directly works with the original high-resolution images. OutputAdapt (ResNet-101) pre-train the model using both ImageNet [31] and MS COCO [22]).

We draw the following observations. First, all domain adaptation methods significantly outperform the respective “source only” baselines which train segmentation networks by using only synthetic source images. Such results clearly demonstrate the benefit of explicitly using domain adaptation techniques to improve the transfer from synthetic images to real ones. Moreover, comparing our full approach (PyCDA) with the existing ones in terms of mIoU, PyCDA gives rise to the best results thus far for the adaptation from GTAV to Cityscapes. Note that the second best FCAN is a two-stage method with style transfer on the image pixel level and then an adversarial training of the features. The style transfer stage runs extremely slow, consuming about one to two hours per image. PyCDA is orthogonal to both the image style transfer and adversarial training, so our results could be further improved if we apply the style transfer to the images of the two domains.

When compared with the distribution matching methods (FCN-wild [18], ROAD [4], OutputAdapt [35] and FCAN [40]), PyCDA is particularly good at the dominant classes, such as “road”, “building”, “vegetation”, and “car”.

Meanwhile, PyCDA is better than CBST [43] at classifying small objects, such as “rider”, “wall”, and “fence”, etc.

Table 3: Ablation study of PyCDA (in mIoU%).

Experiment Setting	GTAV		SYNTHIA	
	VGG-16	Res-101	VGG-16	Res-101
Source only	24.3	34.2	22.4	33.0
Top	28.0	42.0	28.7	40.7
CDA [38]	29.7	-	31.4	-
Bottom	32.6	40.6	31.3	41.0
ST [43]	28.1	-	23.9	-
top + bottom	34.9	46.2	35.1	44.8
top + pixel squares	35.4	46.3	35.4	45.6
top + superpixels	35.2	46.3	35.2	45.9
PyCDA	37.2	47.4	35.9	46.7

4.3. Results on “SYNTHIA to Cityscapes”

To further validate the effectiveness of PyCDA, we also conduct experiments by using SYNTHIA as the source domain. FCN8s with VGG-16 and PSP-Net with ResNet-101 as employed as backbones to evaluate different methods. The IoU results on this setting are summarized in Table 2. From the results, We can clearly see that our PyCDA again outperforms the existing state-of-the-arts by a large margin when using different backbones, and also similar observations can be drawn as in the “GTAV to Cityscapes” setting.

4.4. Ablation study

To analyze the effectiveness of our PyCDA, we conduct ablation study by using the above two settings, i.e., taking GTAV and SYNTHIA as the source domains, respectively. Note that PyCDA connects curriculum domain adaptation (CDA) with self-training (ST), and it can be viewed as a

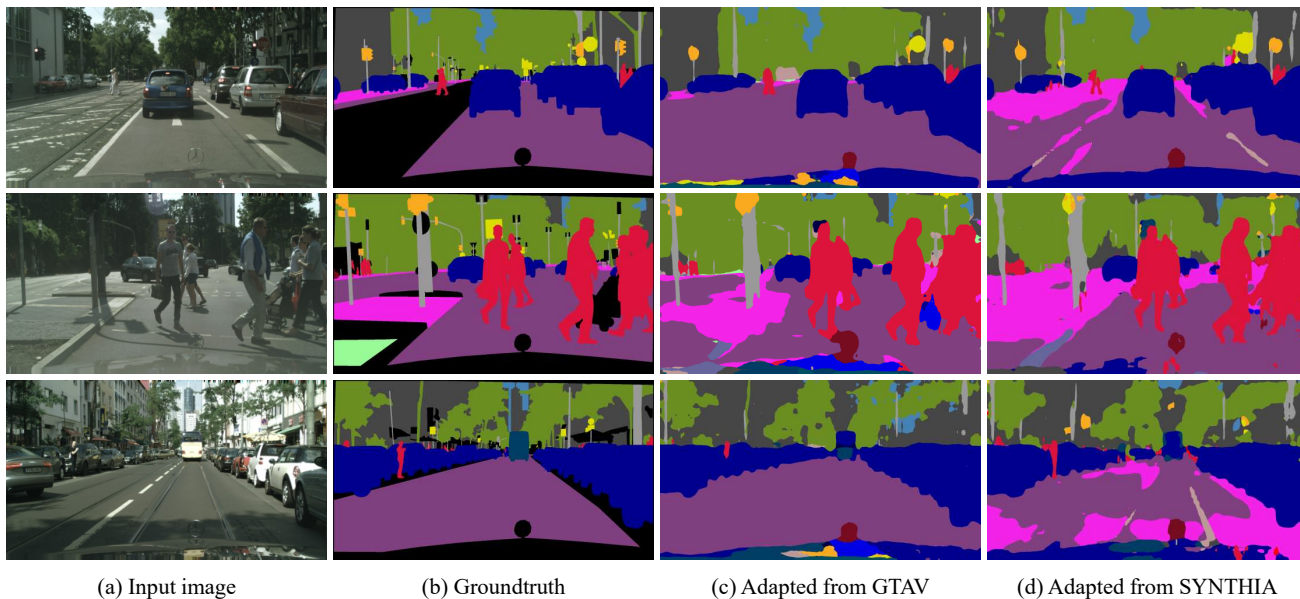


Figure 4: Some qualitative segmentation results on the target domain. (a) displays the target images, and their corresponding groundtruth segmentation masks are shown in (b). (c) and (d) display the segmentation results obtained from our PyCDA models adapted from GTAV and SYNTHIA, respectively.

pyramid constructed by multiple levels from pixels (bottom) to label distributions of full images (top). In this case, we evaluate PyCDA by comparing its counterparts which eliminate different levels. Specifically, we denote “top + bottom” and “top + pixel squares” as connecting CDA with ST at the levels of pixels and pixel squares, respectively. From Table 3, we can see that connect CDA with ST in either pixel level or pixel squares level outperform both CDA and ST in a large margin, which demonstrates the effectiveness of connecting both methods. And PyCDA, which considers both pixels and pixel squares, gets further boosted by a fairly big margin.

Superpixels vs. pixel squares. As discussed in Section 3, it is time-consuming to generate superpixels (about 3.6s per image). In order to avoid the computation overhead, we switch to pixel squares in our PyCDA. As turned out in Table 3, the mIoU performance of using pixel squares achieves comparable results to that of using superpixels.

4.5. Qualitatively comparing GTAV and SYNTHIA

One may wonder how the two source domains of synthetic imagery differ from each other and what effects the difference could cause on the target domain of real images. Fig. 3 shows some example images of the two domains. While GTAV images are vehicle-centric, there are more diverse views in SYNTHIA. In Fig. 4, we give some qualitative results obtained by our PyCDA models adapted from GTAV and SYNTHIA, respectively. In general, the segmentation results of PyCDA adapted from GTAV is better

than that adapted from SYNTHIA, especially for the dominant “road” class. This observation can also be verified by the superior IoU for “road” (90.5%, Table 1) of the PyCDA model trained based on GTAV. Given all those results, we believe GTAV is visually more similar to the real self-driving scenes than SYNTHIA in terms of both visual appearances and spatial layouts.

5. Conclusion

We propose a novel method called self-motivated pyramid curriculum domain adaptation (PyCDA) for pixel-level semantic segmentation. PyCDA provides a new perspective of insight, which connects self-training for adapting segmentation networks and curriculum domain adaptation. More specifically, PyCDA constructs a curriculum based on a pyramid of pixel squares at different sizes in each real image, including the image itself as the top layer and pixels as the bottom layer. This curriculum is self-motivated, because the label distributions over the pyramid are derived from the same network of the previous iteration. By forming such a pyramid of pixel squares, we are able to better preserve and capture local information for objects appearing at different scales. Extensive experiments on two benchmark settings (i.e., “GTAV to Cityscapes” and “SYNTHIA to Cityscapes”) clearly demonstrate the effectiveness of PyCDA when compared with other state of the arts.

Acknowledgements. This work is supported by National Natural Science Foundation of China (Grant No. 61772118).

References

- [1] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 40(4):834–848, 2018. 1, 2
- [2] Liang-Chieh Chen, George Papandreou, Florian Schroff, and Hartwig Adam. Rethinking atrous convolution for semantic image segmentation. *arXiv preprint arXiv:1706.05587*, 2017. 2
- [3] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *ECCV*, 2018. 1, 2
- [4] Yuhua Chen, Wen Li, and Luc Van Gool. Road: Reality oriented adaptation for semantic segmentation of urban scenes. In *CVPR*, 2018. 2, 3, 5, 6, 7
- [5] Yi-Hsin Chen, Wei-Yu Chen, Yu-Ting Chen, Bo-Cheng Tsai, Yu-Chiang Frank Wang, and Min Sun. No more discrimination: Cross city adaptation of road scene segmenters. In *ICCV*, 2017. 2, 3
- [6] 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. 1, 5
- [7] Dengxin Dai, Christos Sakaridis, Simon Hecker, and Luc Van Gool. Curriculum model adaptation with synthetic and real data for semantic foggy scene understanding. *International Journal of Computer Vision*, pages 1–23, 2019. 2
- [8] Lixin Duan, Ivor W. Tsang, and Dong Xu. Domain transfer multiple kernel learning. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(3):465–479, 2012. 2
- [9] Lixin Duan, Dong Xu, Ivor W. Tsang, and Jiebo Luo. Visual event recognition in videos by learning from web data. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 34(9):1667–1680, 2012. 2
- [10] Mark Everingham, S. M. Ali Eslami, Luc Van Gool, Christopher K. I. Williams, John Winn, and Andrew Zisserman. The pascal visual object classes challenge: A retrospective. *International Journal of Computer Vision*, 111(1):98–136, 2015. 6
- [11] Geoff French, Michal Mackiewicz, and Mark Fisher. Self-ensembling for visual domain adaptation. In *ICLR*, 2018. 2
- [12] Chuang Gan, Tianbao Yang, and Boqing Gong. Learning attributes equals multi-source domain generalization. In *CVPR*, 2016. 2
- [13] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(59):1–35, 2016. 2
- [14] Boqing Gong, Kristen Grauman, and Fei Sha. Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation. In *ICML*, 2013. 2
- [15] Boqing Gong, Yuan Shi, Fei Sha, and Kristen Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *CVPR*, 2012. 2
- [16] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 2, 6
- [17] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A. Efros, and Trevor Darrell. Cycada: Cycle consistent adversarial domain adaptation. In *ICML*, 2018. 2, 3, 5, 6
- [18] Judy Hoffman, Dequan Wang, Fisher Yu, and Trevor Darrell. Fcns in the wild: Pixel-level adversarial and constraint-based adaptation. *arXiv preprint arXiv:1612.02649*, 2016. 2, 7
- [19] Weixiang Hong, Zhenzhen Wang, Ming Yang, and Junsong Yuan. Conditional generative adversarial network for structured domain adaptation. In *CVPR*, 2018. 2
- [20] Yanghao Li, Naiyan Wang, Jianping Shi, Xiaodi Hou, and Jiaying Liu. Adaptive batch normalization for practical domain adaptation. *Pattern Recognition*, 80:109–117, 2018. 6
- [21] Guosheng Lin, Anton Milan, Chunhua Shen, and Ian Reid. RefineNet: Multi-path refinement networks for high-resolution semantic segmentation. In *CVPR*, 2017. 1
- [22] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollr, and C. Lawrence Zitnick. Microsoft coco: Common objects in context. In *ECCV*, 2014. 6, 7
- [23] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 1, 2, 6
- [24] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael I. Jordan. Learning transferable features with deep adaptation networks. In *ICML*, 2015. 2
- [25] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I. Jordan. Deep transfer learning with joint adaptation networks. In *ICML*, 2017. 2
- [26] Yawei Luo, Liang Zheng, Tao Guan, Junqing Yu, and Yi Yang. Taking a closer look at domain shift: Category-level adversaries for semantics consistent domain adaptation. *arXiv preprint arXiv:1809.09478*, 2018. 2, 3, 5, 6, 7
- [27] Zak Murez, Soheil Kolouri, David Kriegman, Ravi Ramamoorthi, and Kyungnam Kim. Image to image translation for domain adaptation. In *CVPR*, 2018. 2, 3
- [28] Stephan R. Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In *ECCV*, 2016. 1, 5
- [29] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio M. Lopez. The synthia dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *CVPR*, 2016. 5
- [30] Shu Rui, Bui Hung H., Narui Hirokazu, and Ermon Stefano. A dirt-t approach to unsupervised domain adaptation. In *ICLR*, 2018. 2
- [31] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, and et al. Imagenet large scale visual recognition challenge. *International Journal of Computer Vision*, 115(3):211–252, 2015. 6, 7

- [32] Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *CVPR*, 2018. 2, 6
- [33] Swami Sankaranarayanan, Yogesh Balaji, Arpit Jain, Ser Nam Lim, and Rama Chellappa. Learning from synthetic data: Addressing domain shift for semantic segmentation. In *CVPR*, 2018. 2, 3
- [34] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014. 6
- [35] Yi-Hsuan Tsai, Wei-Chih Hung, Samuel Schulter, Kihyuk Sohn, Ming-Hsuan Yang, and Manmohan Chandraker. Learning to adapt structured output space for semantic segmentation. In *CVPR*, 2018. 2, 3, 5, 6, 7
- [36] Tuan-Hung Vu, Himalaya Jain, Maxime Bucher, Matthieu Cord, and Patrick Perez. Advent: Adversarial entropy minimization for domain adaptation in semantic segmentation. In *CVPR*, 2019. 5, 6, 7
- [37] Zifeng Wu, Chunhua Shen, and Anton van den Hengel. Wider or deeper: Revisiting the resnet model for visual recognition. *arXiv preprint arXiv:1611.10080*, 2016. 6
- [38] Yang Zhang, Philip David, Hassan Foroosh, and Boqing Gong. A curriculum domain adaptation approach to the semantic segmentation of urban scenes. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, pages 1–1, 2019. 2, 3, 4, 5, 6, 7
- [39] Yang Zhang, Philip David, and Boqing Gong. Curriculum domain adaptation for semantic segmentation of urban scenes. In *ICCV*, 2017. 2
- [40] Yiheng Zhang, Zhaofan Qiu, Ting Yao, Dong Liu, and Tao Mei. Fully convolutional adaptation networks for semantic segmentation. In *CVPR*, 2018. 2, 3, 5, 6, 7
- [41] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *CVPR*, 2017. 1, 2, 6
- [42] Xinge Zhu, Hui Zhou, Ceyuan Yang, Jianping Shi, and Dahua Lin. Penalizing top performers: Conservative loss for semantic segmentation adaptation. In *ECCV*, 2018. 2, 3
- [43] Yang Zou, Zhiding Yu, B.V.K. Vijaya Kumar, and Jinsong Wang. Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In *ECCV*, 2018. 2, 3, 4, 5, 6, 7