ADVENT: Adversarial Entropy Minimization for Domain Adaptation in Semantic Segmentation

Tuan-Hung Vu[1] Himalaya Jain[1] Maxime Bucher[1] Matthieu Cord[1,2] Patrick Pérez[1]

[1]valeo.ai, Paris, France   [2]Sorbonne University, Paris, France

Abstract

Semantic segmentation is a key problem for many computer vision tasks. While approaches based on convolutional neural networks constantly break new records on different benchmarks, generalizing well to diverse testing environments remains a major challenge. In numerous real world applications, there is indeed a large gap between data distributions in train and test domains, which results in severe performance loss at run-time. In this work, we address the task of unsupervised domain adaptation in semantic segmentation with losses based on the entropy of the pixel-wise predictions. To this end, we propose two novel, complementary methods using (i) an entropy loss and (ii) an adversarial loss respectively. We demonstrate state-of-the-art performance in semantic segmentation on two challenging "synthetic-2-real" set-ups[1] and show that the approach can also be used for detection.

1. Introduction

Semantic segmentation is the task of assigning class labels to all pixels in an image. In practice, segmentation models often serve as the backbone in complex computer vision systems like autonomous vehicles, which demand high accuracy in a large variety of urban environments. For example, under adverse weathers, the system must be able to recognize roads, lanes, sideways or pedestrians despite their appearances being largely different from ones in the training set. A more extreme and important example is so-called "synthetic-2-real" set-up [31, 30] – training samples are synthesized by game engines and test samples are real scenes. Current fully-supervised approaches [23, 47, 2] have not yet guaranteed a good generalization to arbitrary test cases. Thus a model trained on one domain, named as *source*, usually undergoes a drastic drop in performance when applied on another domain, named as *target*.

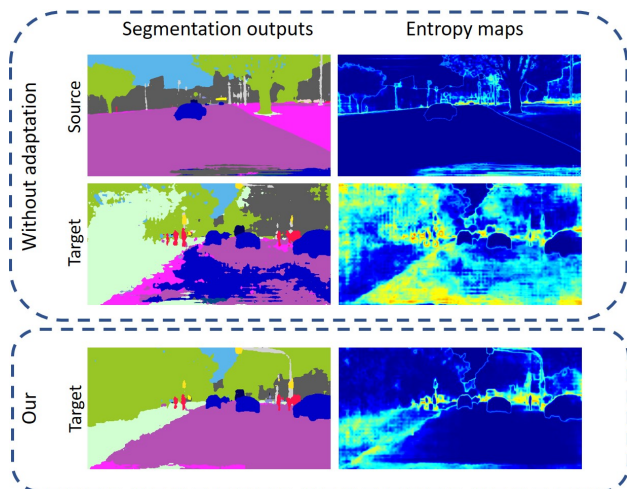[1]Code available at https://github.com/valeoai/ADVENT

Figure 1: **Proposed entropy-based unsupervised domain adaptation for semantic segmentation**. The top two rows show results on source and target domain scenes of the model trained without adaptation. The bottom row shows the result on the same target domain scene of the model trained with entropy-based adaptation. The left and right columns visualize respectively the semantic segmentation outputs and the corresponding prediction entropy maps (see text for details).

Unsupervised domain adaptation (UDA) is the field of research that aims at learning only from source supervision a well performing model on target samples. Among the recent methods for UDA, many address the problem by reducing cross-domain discrepancy, along with the supervised training on the source domain. They approach UDA by minimizing the difference between the distributions of the intermediate features or of the final outputs for source and target data respectively. It is done at single [15, 32, 44] or multiple levels [24, 25] using maximum mean discrepancies (MMD) or adversarial training [10, 42]. Other approaches include self-training [51] to provide pseudo labels or generative networks to produce target data [14, 34, 43].

Semi-supervised learning addresses a closely related

problem of learning from the data of which only a subset is annotated. Thus, it inspires several approaches for UDA, for example, self-training, generative model or class balancing [49]. *Entropy minimization* is also one of the successful approaches used for semi-supervised learning [38].

In this work, we adapt the principle of entropy minimization to the UDA task in semantic segmentation. We start from a simple observation: models trained only on source domain tend to produce *over-confident*, *i.e.*, low-entropy, predictions on source-like images and *under-confident*, *i.e.*, high-entropy, predictions on target-like ones. Such a phenomenon is illustrated in Figure 1. Prediction entropy maps of scenes from the source domain look like edge detection results with high entropy activations only along object borders. On the other hand, predictions on target images are less certain, resulting in very noisy, high entropy outputs. We argue that one possible way to bridge the domain gap between source and target is by enforcing high prediction certainty (low-entropy) on target predictions as well. To this end, we propose two approaches: direct entropy minimization using an entropy loss and indirect entropy minimization using an adversarial loss. While the first approach imposes the low-entropy constraint on independent pixel-wise predictions, the latter aims at globally matching source and target distributions in terms of *weighted self-information*.[2] We summarize our contributions as follows:

- For semantic segmentation UDA, we propose to leverage an entropy loss to directly penalize low-confident predictions on target domain. The use of this entropy loss adds no significant overhead to existing semantic segmentation frameworks.

- We introduce a novel entropy-based adversarial training approach targeting not only the entropy minimization objective but also the structure adaptation from source domain to target domain.

- To improve further the performance in specific settings, we suggest two additional practices: (i) training on specific entropy ranges and (ii) incorporating class-ratio priors. We discuss practical insights in the experiments and ablation studies.

The entropy minimization objectives push the model's decision boundaries toward low-density regions of the target domain distribution in prediction space. This results in "cleaner" semantic segmentation outputs, with more refined object edges as well as large ambiguous image regions being correctly recovered, as shown in Figure 1. The proposed models outperform state-of-the-art approaches on several UDA benchmarks for semantic segmentation, in particular the two main synthetic-2-real benchmarks, GTA5→Cityscapes and SYNTHIA→Cityscapes.

---

[2]Connection to the entropy is discussed in Section 3.

2. Related works

Unsupervised Domain Adaptation is a well researched topic for the task of classification and detection, with recent advances in semantic segmentation also. A very appealing application of domain adaptation is on using synthetic data for real world tasks. This has encouraged the development of several synthetic scene projects with associated datasets, such as Carla [8], SYNTHIA [31], and others [35, 30].

The main approaches for UDA include discrepancy minimization between source and target feature distributions [10, 24, 15, 25, 42], self-training with pseudo-labels [51] and generative approaches [14, 34, 43]. In this work, we are particularly interested in UDA for the task of semantic segmentation. Therefore, we only review the UDA approaches for semantic segmentation here (see [7] for a more general literature review).

Adversarial training for UDA is the most explored approach for semantic segmentation. It involves two networks. One network predicts the segmentation maps for the input image, which could be from source or target domain, while another network acts as a discriminator which takes the feature maps from the segmentation network and tries to predict domain of the input. The segmentation network tries to fool the discriminator, thus making the features from the two domains have a similar distribution. Hoffman *et al.* [15] are the first to apply the adversarial approach for UDA on semantic segmentation. They also have a category specific adaptation by transferring the label statistics from the source domain. A similar approach of global and class-wise alignment is used in [5] with the class-wise alignment being done using adversarial training on grid-wise soft pseudo-labels. In [4], adversarial training is used for spatial-aware adaptation along with a distillation loss to specifically address synthetic-2-real domain shift. [16] uses a residual net to make the source feature maps similar to target's ones using adversarial training, the feature maps being then used for the segmentation task. In [41], the adversarial approach is used on the output space to benefit from the structural consistency across domain. [32, 33] propose another interesting way of using adversarial training: They get two predictions on the target domain image, this is done either by two classifiers [33] or using dropout in the classifier [32]. Given the two predictions the classifier is trained to maximize the discrepancy between the distributions while the feature extractor part of the network is trained to minimize it.

Some methods build on *generative networks* to generate target images conditioned on the source. Hoffman *et al.* [14] propose Cycle-Consistent Adversarial Domain Adaptation (CyCADA), in which they adapt at both pixel-level and feature-level representation. For pixel-level adaptation they use Cycle-GAN [48] to generate target images conditioned on the source images. In [34], a generative model is

learned to reconstruct images from the feature space. Then, for domain adaptation, the feature module is trained to produce target images on source features and vice-versa using the generator module. In DCAN [43], channel-wise feature alignment is used in the generator and segmentation network. The segmentation network is learned on generated images with the content of the source and style of the target for which source segmentation map serves as the ground-truth. The authors in [50] use generative adversarial networks (GAN) [11] to align the source and target embeddings. Also, they replace the cross-entropy loss by a *conservative loss* (CL) that penalizes the easy and hard cases of source examples. The CL approach is orthogonal to most of the UDA methods, including ours: it could benefit any method that uses cross-entropy for source.

Another approach for UDA is *self-training*. The idea is to use the prediction from an ensembled model or a previous state of model as pseudo-labels for the unlabeled data to train the current model. Many semi-supervised methods [20, 39] use self-training. In [51], self-training is employed for UDA on semantic segmentation which is further extended with class balancing and spatial prior. Self-training has an interesting connection to the proposed entropy minimization approach as we discuss in Section 3.1.

Among some other approaches, [26] uses a combination of adversarial and generative techniques through multiple losses, [46] combines the generative approach for appearance adaptation and adversarial training for representation adaptation, and [45] proposes a curriculum-style learning for UDA by enforcing the consistency on local (superpixel-level) and global label distributions.

Entropy minimization has been shown to be useful for semi-supervised learning [12, 38] and clustering [17, 18]. It has also been recently applied on domain adaptation for classification task [25]. To our knowledge, we are first to successfully apply entropy based UDA training to obtain competitive performance on semantic segmentation task.

3. Approaches

In this section, we present our two proposed approaches for entropy minimization using (i) an unsupervised entropy loss and (ii) adversarial training. To build our models, we start from existing semantic segmentation frameworks and add an additional network branch used for domain adaptation. Figure 2 illustrates our architectures.

Our models are trained with a supervised loss on source domain. Formally, we consider a set $\mathcal{X}_s \subset \mathbb{R}^{H \times W \times 3}$ of sources examples along with associated ground-truth C-class segmentation maps, $\mathcal{Y}_s \subset (1, C)^{H \times W}$. Sample $\boldsymbol{x}_s$ is a $H \times W$ color image and entry $\boldsymbol{y}_s^{(h,w)} = \left[\boldsymbol{y}_s^{(h,w,c)}\right]_c$ of associated map $\boldsymbol{y}_s$ provides the label of pixel (h, w) as one-hot vector. Let F be a semantic segmen-

tation network which takes an image $\boldsymbol{x}$ and predicts a C-dimensional "soft-segmentation map" $F(\boldsymbol{x}) = \boldsymbol{P}_{\boldsymbol{x}} = \left[\boldsymbol{P}_{\boldsymbol{x}}^{(h,w,c)}\right]_{h,w,c}$. By virtue of final softmax layer, each C-dimensional pixel-wise vector $\left[\boldsymbol{P}_{\boldsymbol{x}}^{(h,w,c)}\right]_c$ behaves as a discrete distribution over classes. If one class stands out, the distribution is picky (low entropy), if scores are evenly spread, sign of uncertainty from the network standpoint, the entropy is large. The parameters θ_F of F are learned to minimize the segmentation loss $\mathcal{L}_{seg}(\boldsymbol{x}_s, \boldsymbol{y}_s) = -\sum_{h=1}^{H} \sum_{w=1}^{W} \sum_{c=1}^{C} \boldsymbol{y}_s^{(h,w,c)} \log \boldsymbol{P}_{\boldsymbol{x}_s}^{(h,w,c)}$ on source samples. In the case of training only on source domain without domain adaptation, the optimization problem simply reads:

$$\min_{\theta_F} \frac{1}{|\mathcal{X}_s|} \sum_{\boldsymbol{x}_s \in \mathcal{X}_s} \mathcal{L}_{seg}(\boldsymbol{x}_s, \boldsymbol{y}_s). \qquad (1)$$

3.1. Direct entropy minimization

For the target domain, as we do not have the annotations $\boldsymbol{y}_t$ for image samples $\boldsymbol{x}_t \in \mathcal{X}_t$, we cannot use (1) to learn F. Some methods use the model's prediction $\hat{\boldsymbol{y}}_t$ as a proxy for $\boldsymbol{y}_t$. Also, this proxy is used only for pixels where prediction is sufficiently confident. Instead of using the high-confident proxy, we propose to constrain the model such that it produces high-confident prediction. We realize this by minimizing the entropy of the prediction.

We introduce the entropy loss $\mathcal{L}_{ent}$ to directly maximize prediction certainty in the target domain. In this work, we use the Shannon Entropy [36]. Given a target input image $\boldsymbol{x}_t$, the entropy map $\boldsymbol{E}_{\boldsymbol{x}_t} \in [0, 1]^{H \times W}$ is composed of the independent pixel-wise entropies normalized to $[0, 1]$ range:

$$\boldsymbol{E}_{\boldsymbol{x}_t}^{(h,w)} = \frac{-1}{\log(C)} \sum_{c=1}^{C} \boldsymbol{P}_{\boldsymbol{x}_t}^{(h,w,c)} \log \boldsymbol{P}_{\boldsymbol{x}_t}^{(h,w,c)}, \qquad (2)$$

at pixel (h, w). An example of entropy map is shown in Figure 2. The entropy loss $\mathcal{L}_{ent}$ is defined as the sum of all pixel-wise normalized entropies:

$$\mathcal{L}_{ent}(\boldsymbol{x}_t) = \sum_{h,w} \boldsymbol{E}_{\boldsymbol{x}_t}^{(h,w)}. \qquad (3)$$

During training, we jointly optimize the supervised segmentation loss $\mathcal{L}_{seg}$ on source samples and the unsupervised entropy loss $\mathcal{L}_{ent}$ on target samples. The final optimization problem is formulated as follows:

$$\min_{\theta_F} \frac{1}{|\mathcal{X}_s|} \sum_{\boldsymbol{x}_s} \mathcal{L}_{seg}(\boldsymbol{x}_s, \boldsymbol{y}_s) + \frac{\lambda_{ent}}{|\mathcal{X}_t|} \sum_{\boldsymbol{x}_t} \mathcal{L}_{ent}(\boldsymbol{x}_t), \qquad (4)$$

with λ_{ent} as the weighting factor of the entropy term $\mathcal{L}_{ent}$.

Connection to self-training. Pseudo-labeling is a simple yet efficient approach for semi-supervised learning [21]. Recently, the approach has been applied to UDA in semantic segmentation task with an iterative self-training (ST) procedure [51]. The ST method assumes that the set $K \subset$
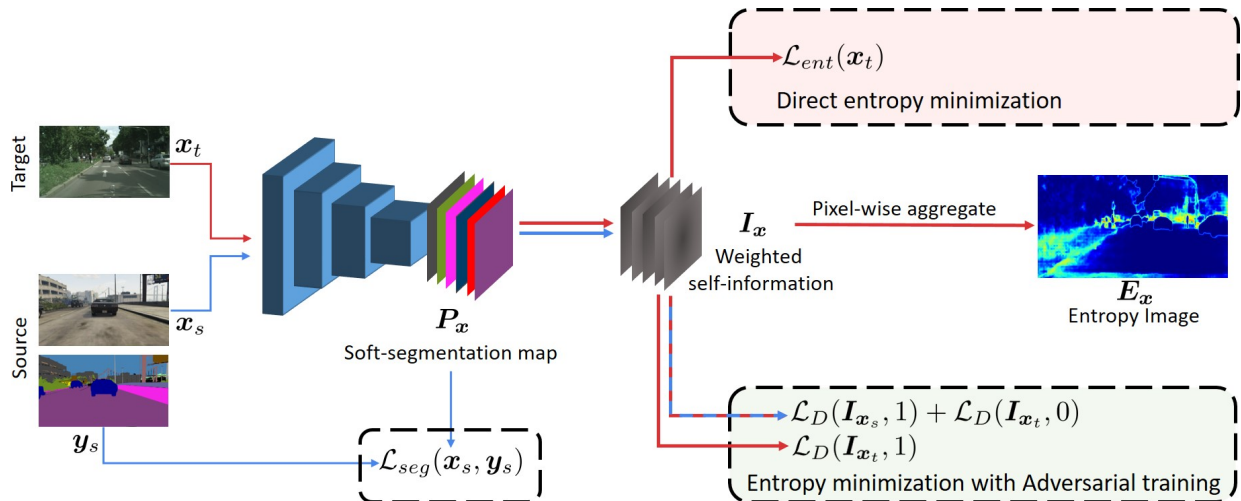
Figure 2: **Approach overview**. The figure shows our two approaches for UDA. First, *direct entropy minimization* minimizes the entropy of the target $\boldsymbol{P}_{\boldsymbol{x}_t}$, which is equivalent to minimizing the sum of weighted self-information maps $\boldsymbol{I}_{\boldsymbol{x}_t}$. In the second, complementary approach, we use adversarial training to enforce the consistency in $\boldsymbol{I}_{\boldsymbol{x}}$ across domains. Red arrows are used for target domain and blue arrows for source. An example of entropy map is shown for illustration.

$(1, H) \times (1, W)$ of high-scoring pixel-wise predictions on target samples are correct with high probability. Such an assumption allows the use of cross-entropy loss with pseudo-labels on target predictions. In practice, K is constructed by selecting high-scoring pixels with a fixed or scheduled threshold. To draw a link with entropy minimization, we write the training problem of the ST approach as:

$$\min_{\theta_F} \frac{1}{|\mathcal{X}_s|} \sum_{\boldsymbol{x}_s} \mathcal{L}_{seg}(\boldsymbol{x}_s, \boldsymbol{y}_s) + \frac{\lambda_{pl}}{|\mathcal{X}_t|} \sum_{\boldsymbol{x}_t} \mathcal{L}_{seg}(\boldsymbol{x}_t, \hat{\boldsymbol{y}}_t), \quad (5)$$

where $\hat{\boldsymbol{y}}_t$ is the one-hot class prediction for $\boldsymbol{x}_t$ and with:

$$\mathcal{L}_{seg}(\boldsymbol{x}_t, \hat{\boldsymbol{y}}_t) = - \sum_{(h,w) \in K} \sum_{c=1}^{C} \hat{\boldsymbol{y}}_t^{(h,w,c)} \log \boldsymbol{P}_{\boldsymbol{x}_t}^{(h,w,c)}. \quad (6)$$

Comparing equations (2-3) and (6), we note that our entropy loss $\mathcal{L}_{ent}(\boldsymbol{x}_t)$ can be seen as a soft-assignment version of the pseudo-label cross-entropy loss $\mathcal{L}_{seg}(\boldsymbol{x}_t, \hat{\boldsymbol{y}}_t)$. Different to ST [51], our entropy-based approach does not require a complex scheduling procedure for choosing threshold. Even, contrary to ST assumption, we show in Section 4.3 that, in some cases, training on the "hard" or "most-confused" pixels produces better performance.

3.2. Minimizing entropy with adversarial learning

The entropy loss for an input image is defined in equation (3) as the sum of independent pixel-wise prediction entropies. Therefore, a mere minimization of this loss neglects the structural dependencies between local semantics. As shown in [41], for UDA in semantic segmentation, adaptation on structured output space is beneficial. It is based on the fact that source and target domain share strong similarities in semantic layout.

In this part, we introduce a unified adversarial training framework which indirectly minimizes the entropy by having target's entropy distribution similar to the source. This allows the exploitation of the structural consistency between the domains. To this end, we formulate the UDA task as minimizing distribution distance between source and target on the *weighted self-information* space. Figure 2 illustrates our adversarial learning procedure. Our adversarial approach is motivated by the fact that the trained model naturally produces low-entropy predictions on source-like images. By aligning weighted self-information distributions of target and source domains, we indirectly minimize the entropy of target predictions. Moreover, as the adaptation is done on the weighted self-information space, our model leverages structural information from source to target.

In detail, given a pixel-wise predicted class score $\boldsymbol{P}_{\boldsymbol{x}}^{(h,w,c)}$, the self-information or "surprisal" [40] is defined as $-\log \boldsymbol{P}_{\boldsymbol{x}}^{(h,w,c)}$. Effectively, the entropy $\boldsymbol{E}_{\boldsymbol{x}}^{(h,w)}$ in (2) is the expected value of the self-information $\mathbb{E}_c[-\log \boldsymbol{P}_{\boldsymbol{x}}^{(h,w,c)}]$. We here perform adversarial adaptation on weighted self-information maps $\boldsymbol{I}_{\boldsymbol{x}}$ composed of pixel-level vectors $\boldsymbol{I}_{\boldsymbol{x}}^{(h,w)} = -\boldsymbol{P}_{\boldsymbol{x}}^{(h,w)} \cdot \log \boldsymbol{P}_{\boldsymbol{x}}^{(h,w)}$.[3] Such vectors can be seen as the disentanglement of the Shannon Entropy. We then construct a fully-convolutional discriminator network D with parameters θ_D taking $\boldsymbol{I}_{\boldsymbol{x}}$ as input and that produces domain classification outputs, *i.e.*, class label 1 (resp. 0) for the source (resp. target) domain. Similar to [11], we train the discriminator to discriminate outputs coming from source and target images, and at the same time, train the segmentation network to fool the discriminator. In detail,

---

[3]Abusing notations, '·' and 'log' stand for Hadamard product and point-wise logarithm respectively.

let $\mathcal{L}_D$ the cross-entropy domain classification loss. The training objective of the discriminator is:

$$\min_{\theta_D} \frac{1}{|\mathcal{X}_s|} \sum_{\boldsymbol{x}_s} \mathcal{L}_D(\boldsymbol{I}_{\boldsymbol{x}_s}, 1) + \frac{1}{|\mathcal{X}_t|} \sum_{\boldsymbol{x}_t} \mathcal{L}_D(\boldsymbol{I}_{\boldsymbol{x}_t}, 0), \quad (7)$$

and the adversarial objective to train the segmentation network is:

$$\min_{\theta_F} \frac{1}{|\mathcal{X}_t|} \sum_{\boldsymbol{x}_t} \mathcal{L}_D(\boldsymbol{I}_{\boldsymbol{x}_t}, 1). \quad (8)$$

Combining (1) and (8), we derive the optimization problem

$$\min_{\theta_F} \frac{1}{|\mathcal{X}_s|} \sum_{\boldsymbol{x}_s} \mathcal{L}_{seg}(\boldsymbol{x}_s, \boldsymbol{y}_s) + \frac{\lambda_{adv}}{|\mathcal{X}_t|} \sum_{\boldsymbol{x}_t} \mathcal{L}_D(\boldsymbol{I}_{\boldsymbol{x}_t}, 1), \quad (9)$$

with the weighting factor λ_{adv} for the adversarial term $\mathcal{L}_D$. During training, we alternatively optimize networks D and F using objective functions in (7) and (9).

3.3. Incorporating class-ratio priors

Entropy minimization can get biased towards some easy classes. Therefore, sometimes it is beneficial to guide the learning with some prior. To this end, we use a simple class-prior based on the distribution of the classes over the source labels. We compute the class-prior vector $\boldsymbol{p}_s$ as a ℓ_1-normalized histogram of number of pixels per class over the source labels. Now based on the predicted $\boldsymbol{P}_{\boldsymbol{x}_t}$, too large discrepancy between the expected probability for any class and class-prior $\boldsymbol{p}_s$ is penalized, using

$$\mathcal{L}_{cp}(\boldsymbol{x}_t) = \sum_{c=1}^{C} \max\left(0, \mu \boldsymbol{p}_s^{(c)} - \mathbb{E}_c(\boldsymbol{P}_{\boldsymbol{x}_t}^{(c)})\right), \quad (10)$$

where $\mu \in [0,1]$ is used to relax the class prior constraint. This addresses the fact that class distribution on a single target image is not necessarily close to $\boldsymbol{p}_s$.

4. Experiments

In this section, we present our experimental results. Section 4.1 introduces the used datasets as well as our training parameters. In Section 4.2 and Section 4.3, we report and discuss our main results. In Section 4.3, we discuss a preliminary result on entropy-based UDA for detection.

4.1. Experimental details

Datasets. To evaluate our approaches, we use the challenging *synthetic-2-real* unsupervised domain adaptation set-ups. Models are trained on fully-annotated synthetic data and are validated on real-world data. In such set-ups, the models have access to some unlabeled real images during training. To train our models, we use either GTA5 [30] or SYNTHIA [31] as source domain synthetic data, along with the training split of Cityscapes dataset [6] as target domain data. Similar set-ups have been previously used in other works [15, 14, 41, 51]. In detail:

- *GTA5→Cityscapes:* The GTA5 dataset consists of $24,966$ synthesized frames captured from a video game. Images are provided with pixel-level semantic annotations of 33 classes. Similar to [15], we use the 19 classes in common with the Cityscapes dataset.
- *SYNTHIA→Cityscapes:* We use the SYNTHIA-RAND-CITYSCAPES set[4] with $9,400$ synthesized images for training. We train our models with 16 common classes in SYNTHIA and Cityscapes. While evaluating we compare the performance on 16- and 13-class subsets following the protocol used in [51].

In both set-ups, $2,975$ unlabeled Cityscapes images are used for training. We measure segmentation performance with the standard mean-Intersection-over-Union (mIoU) metric [9]. Evaluation is done on the 500 validation images.

Network architectures. We use Deeplab-V2 [2] as the base semantic segmentation architecture F. To better capture the scene context, Atrous Spatial Pyramid Pooling (ASPP) is applied on the last layer's feature output. Sampling rates are fixed as $\{6, 12, 18, 24\}$, similar to the ASPP-L model in [2]. We experiment on the two different base deep CNN architectures: VGG-16 [37] and ResNet-101 [13]. Following [2], we modify the stride and dilation rate of the last layers to produce denser feature maps with larger field-of-views. To further improve performance on ResNet-101, we perform adaptation on multi-level outputs coming from both *conv4* and *conv5* features [41].

The adversarial network D introduced in Section 3.2 has the same architecture as the one used in DCGAN [28]. Weighted self-information maps $\boldsymbol{I}_x$ are forwarded through 4 convolutional layers, each coupled with a leaky-ReLU layer with a fixed negative slope of 0.2. At the end, a classifier layer produces classification outputs, indicating if the inputs correspond to source or target domain.

Implementation details. We employ PyTorch deep learning framework [27] in the implementations. All experiments are done on a single NVIDIA 1080TI GPU with 11 GB memory. Our model, except the adversarial discriminator mentioned in Section 3.2, is trained using Stochastic Gradient Descent optimizer [1] with learning rate 2.5×10^{-4}, momentum 0.9 and weight decay 10^{-4}. We use Adam optimizer [19] with learning rate 10^{-4} to train the discriminator. To schedule the learning rate, we follow the polynomial annealing procedure mentioned in [2].

Weighting factors of entropy and adversarial losses: To set the weight for $\mathcal{L}_{ent}$, the training set performance provides important indications. If λ_{ent} is large then the entropy drops too quickly and the model is strongly biased towards a few classes. When λ_{ent} is chosen in a suitable range however, the performance is better and not sensitive to the pre-

---

[4]A split of the SYNTHIA dataset [31] using compatible labels with the Cityscapes dataset.

(a) GTA5 → Cityscapes

Models	Appr.	road	sidewalk	building	wall	fence	pole	light	sign	veg	terrain	sky	person	rider	car	truck	bus	train	mbike	bike	mIoU
FCNs in the Wild [15]	Adv	70.4	32.4	62.1	14.9	5.4	10.9	14.2	2.7	79.2	21.3	64.6	44.1	4.2	70.4	8.0	7.3	0.0	3.5	0.0	27.1
CyCADA [14]	Adv	83.5	**38.3**	76.4	20.6	16.5	22.2	26.2	21.9	80.4	28.7	65.7	49.4	4.2	74.6	16.0	26.6	2.0	8.0	0.0	34.8
Adapt-SegMap [41]	Adv	87.3	29.8	78.6	21.1	18.2	22.5	21.5	11.0	79.7	29.6	71.3	46.8	6.5	80.1	23.0	26.9	0.0	10.6	0.3	35.0
Self-Training [51]	ST	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
Self-Training + CB [51]	ST	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
Ours (MinEnt)	Ent	85.1	18.9	76.3	32.4	19.7	19.9	21.0	8.9	76.3	26.2	63.1	42.8	5.9	80.8	20.2	9.8	0.0	14.8	0.6	32.8
Ours (AdvEnt)	Adv	86.9	28.7	78.7	28.5	25.2	17.1	20.3	10.9	80.0	26.4	70.2	47.1	8.4	81.5	26.0	17.2	**18.9**	11.7	1.6	36.1
Adapt-SegMap [41]	Adv	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
Adapt-SegMap*	Adv	85.5	18.4	80.8	29.1	24.6	27.9	33.1	20.9	83.8	31.2	75.0	57.5	28.6	77.3	32.3	30.9	1.1	28.7	35.9	42.2
Ours (MinEnt)	Ent	84.4	18.7	80.6	23.8	23.2	28.4	**36.9**	23.4	83.2	25.2	**79.4**	59.0	29.9	78.5	33.7	29.6	1.7	29.9	33.6	42.3
Ours (MinEnt + ER)	Ent	84.2	25.2	77.0	17.0	23.3	24.2	33.3	**26.4**	80.7	32.1	78.7	57.5	30.0	77.0	37.9	44.3	1.8	31.4	**36.9**	43.1
Ours (AdvEnt)	Adv	**89.9**	36.5	**81.6**	29.2	25.2	**28.5**	32.3	22.4	83.9	34.0	77.1	57.4	27.9	83.7	29.4	39.1	1.5	28.4	23.3	43.8
Ours (AdvEnt+MinEnt)	A+E	89.4	33.1	81.0	26.6	**26.8**	27.2	33.5	24.7	**83.9**	36.7	78.8	**58.7**	**30.5**	**84.8**	38.5	44.5	1.7	**31.6**	32.4	**45.5**

(b) SYNTHIA → Cityscapes

Models	Appr.	road	sidewalk	building	wall	fence	pole	light	sign	veg	sky	person	rider	car	bus	mbike	bike	mIoU	mIoU*
FCNs in the Wild [15]	Adv	11.5	19.6	30.8	4.4	0.0	20.3	0.1	11.7	42.3	68.7	51.2	3.8	54.0	3.2	0.2	0.6	20.2	22.1
Adapt-SegMap [41]	Adv	78.9	29.2	75.5	-	-	-	0.1	4.8	72.6	76.7	43.4	8.8	71.1	16.0	3.6	8.4	-	37.6
Self-Training [51]	ST	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
Self-Training + CB [51]	ST	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	36.1
Ours (MinEnt)	Ent	37.8	18.2	65.8	2.0	0.0	15.5	0.0	0.0	76	73.9	45.7	11.3	66.6	13.3	1.5	13.1	27.5	32.5
Ours (MinEnt + CP)	Ent	45.9	19.6	65.8	5.3	0.2	20.7	2.1	8.2	74.4	76.7	47.5	12.2	71.1	22.8	4.5	9.2	30.4	35.4
Ours (AdvEnt + CP)	Adv	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
Adapt-SegMap [41]	Adv	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
Adapt-SegMap* [41]	Adv	81.7	39.1	78.4	11.1	0.3	25.8	6.8	9.0	79.1	80.8	54.8	21.0	66.8	34.7	13.8	29.9	39.6	45.8
Ours (MinEnt)	Ent	73.5	29.2	77.1	7.7	0.2	**27.0**	7.1	11.4	76.7	82.1	57.2	21.3	69.4	29.2	12.9	27.9	38.1	44.2
Ours (AdvEnt)	Adv	**87.0**	**44.1**	79.7	9.6	**0.6**	24.3	4.8	7.2	80.1	83.6	56.4	23.7	72.7	32.6	12.8	**33.7**	40.8	47.6
Ours (AdvEnt+MinEnt)	A+E	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**

Table 1: **Semantic segmentation performance mIoU (%) on Cityscapes validation set of models trained on GTA5 (a) and SYNTHIA (b)**. We show results of our approaches using the direct entropy loss (MinEnt) and using adversarial training (AdvEnt). In each subtable, top and bottom parts correspond to VGG-16-based and ResNet-101-based models respectively. The "Adapt-SegMap*" denotes our retrained model of [41]. The abbreviations "Adv", "ST" and "Ent" stand for adversarial training, self-training and entropy minimization approaches.

cise value. Thus, we use the same $\lambda_{ent} = 0.001$ for all our experiments regardless of the network or the dataset. Similar arguments hold for the weight λ_{adv} in (9). We fix $\lambda_{adv} = 0.001$ in all experiments.

4.2. Results

We present experimental results of our approaches compared to different baselines. Our models achieve state-of-the-art performance in the two UDA benchmarks. In what follows, we show different behaviors of our approaches in different settings, *i.e.*, training sets and base CNNs.

GTA5→Cityscapes: We report in Table 1-a semantic segmentation performance in terms of mIoU (%) on Cityscapes validation set. Our first approach of direct entropy minimization, termed as MinEnt in Table 1-a, achieves comparable performance to state-of-the-art base-

lines on both VGG-16 and ResNet-101-based CNNs. The MinEnt outperforms Self-Training (ST) approach without and with class-balancing [51]. Compared to [41], the ResNet-101-based MinEnt shows similar results but without resorting to the training of a discriminator network. The light overhead of the entropy loss makes training time much less for the MinEnt model. Another advantage of our entropy approach is the ease of training. Indeed, training adversarial networks is generally known as a difficult task due to its instability. We observed a more stable behavior training models with the entropy loss.

Interestingly, we find that in some cases, only applying entropy loss on certain ranges works best. Such a phenomenon is observed with the ResNet-101-based models. Indeed, we get a better model by training on pixels having entropy values within the top 30% of each target sample.

The model is termed as MinEnt+ER in Table 1-a. We obtain 43.1% mIoU using this strategy on the GTA5→Cityscapes set-up. More details are given in Section 4.3.

Our second approach using adversarial training on the weighted self-information space, noted as AdvEnt, shows consistent improvement to the baselines on the two base networks. In general, AdvEnt works better than MinEnt. On the GTA5→Cityscapes UDA set-up, AdvEnt achieves state-of-the-art mIoU of 43.8. Such results confirm our intuition on the importance of structural adaptation. With the VGG-16-based network, adaptation on the weighted self-information space brings +3.3% mIoU improvement compared to the direct entropy minimization. With the ResNet-101-based network, the improvement is less, *i.e.*, +1.5% mIoU. We conjecture that, as GTA5 semantic layouts are very similar to ones in Cityscapes, the segmentation network F with high capacity base CNN like ResNet-101 is capable of learning some spatial priors from the supervision on source samples. As for lower-capacity base model like VGG-16, an additional regularization on the structured space with adversarial training is more beneficial.

By combining results of the two models MinEnt and AdvEnt, we observe a decent boost in performance, compared to results of single models. The ensemble achieves 45.5% mIoU on the Cityscape validation set. Such a result indicates that complementary information are learned by the two models. Indeed, while the entropy loss penalizes independent pixel-level predictions, the adversarial loss operates more on the image-level, *i.e.*, scene topology. Similar to [41], for a more meaningful comparison to other UDA approaches, in Table 2-a we show the performance gap between UDA models and the oracle, *i.e.*, the model trained with full supervision on the Cityscapes training set. Compared to models trained by other methods, our single and ensemble models have smaller mIoU gaps to the oracle.

In Figure 3, we illustrate some qualitative results of our models. Without domain adaptation, the model trained only on source supervision produces noisy segmentation predictions as well as high entropy activations, with a few exceptions on some classes like "building" and "car". Still, there exist many confident predictions (low entropy) which are completely wrong. Our models, on the other hand, manage to produce correct predictions at high level of confidence. We observe that overall, the AdvEnt model achieves lower prediction entropy compared to the MinEnt model.

SYNTHIA→Cityscapes: Table 1-b shows results on the 16- and 13-class subsets of the Cityscapes validation set. We notice that scene images in SYNTHIA cover more diverse viewpoints than the ones in GTA5 and Cityscape. This results in different behaviors of our approaches.

On the VGG-16-based network, the MinEnt model shows comparable results to state-of-the-art methods. Compared to Self-Training [51], our model achieves +3.6%

(a) GTA5 → Cityscapes

Method	UDA Model	Oracle	mIoU Gap (%)
FCNs in the Wild [15]	27.1	64.6	-37.5
CyCADA [14]	28.9	60.3	-31.4
Adapt-SegNet [41]	35.0	61.8	-26.8
Ours (single model)	36.1	61.8	-25.7
Adapt-SegNet [41]	42.4	65.1	-22.7
Ours (single model)	43.8	65.1	-21.3
Ours (two models)	45.5	65.1	**-19.6**

(b) SYNTHIA → Cityscapes

Method	UDA Model	Oracle	mIoU Gap (%)
FCNs in the Wild [15]	22.9	73.8	-50.9
Adapt-SegNet [41]	37.6	68.4	-30.8
Ours (single model)	36.6	68.4	-31.8
Adapt-SegNet [41]	46.7	71.7	-25.0
Ours (single model)	47.6	71.7	-24.1
Ours (two models)	48	71.7	**-23.7**

Table 2: **Performance gap between UDA models and the oracle** in GTA5→Cityscapes and SYNTHIA→Cityscapes setups. Top and bottom parts of each table correspond to VGG-16-based and ResNet-101-based models respectively.

and +4.7% on 16- and 13- class subsets respectively. However, compared to stronger baselines like the class-balanced self-training, we observe a significant drop in class "road". We argue that it is due to the large layout gaps between SYNTHIA and Cityscapes. To target this issue, we incorporate the class-ratio priors from source domain, as introduced in Section 3.3. By constraining target output distribution using class-ratio priors, noted as CP in Table 1-b, we improve MinEnt by +2.9% mIoU on both 16- and 13- class subsets. With adversarial training, we have an additional $\sim$ +1% mIoU. On the ResNet-101-based network, the AdvEnt model achieves state-of-the-art performance. Compared to the retrained model of [41], *i.e.*, Adapt-SegMap*, the AdvEnt improves the mIoUs on 16- and 13- class subsets by +1.2% and +1.8%.

Consistent with the GTA5 results above, the ensemble of the two models MinEnt and AdvEnt trained on SYNTHIA achieves the best mIoU of 41.2% and 48.0% on 16- and 13- class subsets respectively. According to Table 2-b, our models have the smallest mIoU gaps to the oracle.

4.3. Discussion

The experimental results shown in Section 4.2 have validated the advantages of our approaches. To further push the performance, we proposed two different ways to regularize the training in two particular settings. This section discusses our experimental choices and explain the intuitions behind.

GTA5→Cityscapes: Training on specific entropy ranges. In this setup, we observe that the performance of model MinEnt using ResNet-101-based network can be improved by training on target pixels having entropy values in a specific range. Interestingly, the best MinEnt model was
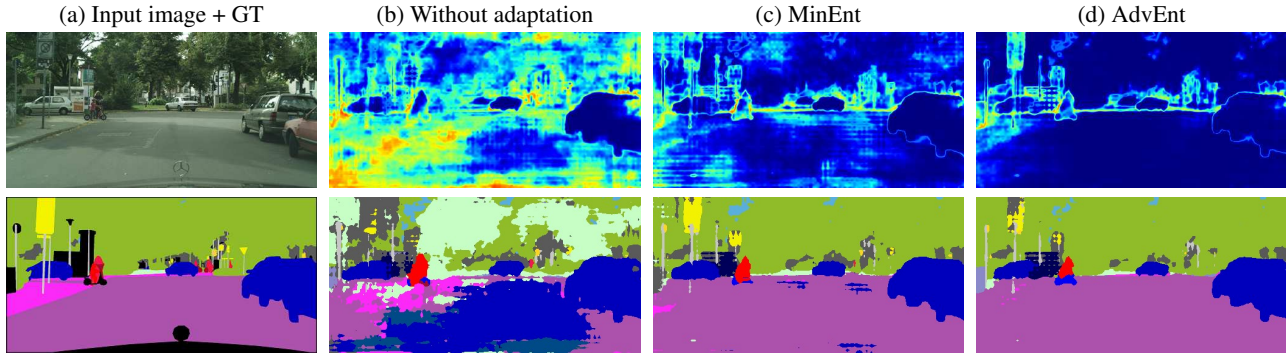
| | (a) Input image + GT | (b) Without adaptation | (c) MinEnt | (d) AdvEnt |

Figure 3: **Qualitative results in the GTA5→Cityscapes set-up**. Column (a) shows a input image and the corresponding semantic segmentation ground-truth. Column (b), (c) and (d) show segmentation results (bottom) along with prediction entropy maps produced by different approaches (top). Best viewed in colors.

| Cityscapes → Cityscapes Foggy | | | | | | | | | |
Models	person	rider	car	truck	bus	train	mcycle	bicycle	mAP
SSD-300	15.0	17.4	27.2	5.7	15.1	9.1	11.0	16.7	14.7
Ours (MinEnt)	15.8	22.0	28.3	5.0	15.2	15.0	13.0	20.6	16.9
Ours (AdvEnt)	**17.6**	**25.0**	**39.6**	**20.0**	**37.1**	**25.9**	**21.3**	**23.1**	**26.2**

Table 3: **Detection performance on *Cityscapes Foggy***.

| | (a) Without adaptation | (b) AdvEnt |

Figure 4: **Qualitative detection results on *Cityscapes Foggy***.

trained on the top 30% highest-entropy pixels on each target sample – gaining 0.8% mIoU over the vanilla model. We note that high-entropy pixels are the "most confusing" ones, *i.e.*, where the segmentation model is indecisive between multiple classes. One reason is that the ResNet-101-based model generalizes well in this particular setting. Accordingly, among the "most confusing" predictions, there is a decent amount of correct but "low confident" ones. Minimizing the entropy loss on such a set still drives the model toward the desirable direction. This assumption, however, does not hold for the VGG-16-based model.

SYNTHIA→Cityscapes: Using class-ratio prior. As discussed before, SYNTHIA has significantly different layout and viewpoints than Cityscapes. This disparity can cause very bad prediction for some classes, which is then further encouraged with minimization of entropy or their use as label proxy in self-training. Thus, it can lead to strong class biases or, in extreme cases, to missing some classes completely in the target domain. Adding class-ratio prior encourages the presence of all the classes and thus helps avoid such degenerate solutions. As mentioned in Sec. 3.3, we use μ to relax the source class-ratio prior, for example $\mu = 0$ means no prior while $\mu = 1$ implies enforcing exact class-ratio prior. Having $\mu = 1$ is not ideal as it means that each target image should follow the class-ratio from the source domain. We choose $\mu = 0.5$ to let the target class-ratio to be somewhat different from the source.

Application on UDA for object detection. The proposed entropy-based approaches are not limited to semantic segmentation and can be applied to UDA for other recognition

tasks like object detection. We conducted experiments in the UDA object detection set-up Cityscapes→Cityscapes-Foggy, similar to the one in [3]. A straight-forward application of the entropy loss and of the adversarial loss to the existing detection architecture SSD-300 [22] significantly improves detection performance over the baseline model trained only on source. In terms of mean-average-precision (mAP), compared to the baseline performance of 14.7%, the MinEnt and AdvEnt models attain mAPs of 16.9% and 26.2%. In [3], the authors report a slightly better performance of 27.6% mAP, using Faster-RCNN [29], a more complex detection architecture than ours. We note that our detection system was trained and tested on images at lower resolution, *i.e.*, 300×300. Despite these unfavorable factors, our improvement to the baseline ($+11.5\%$ mAP using AdvEnt) is larger than the one reported in [3] ($+8.8\%$). Such a preliminary result suggests the possibility of applying entropy-based approaches on UDA for detection. Table 3 reports the per-class IoU and Figure 4 shows qualitative results of the approaches on *Cityscapes Foggy*.

5. Conclusion

In this work, we address the task of unsupervised domain adaptation for semantic segmentation and propose two complementary entropy-based approaches. Our models achieve state-of-the-art on the two challenging "synthetic-2-real" benchmarks. The ensemble of the two models improves the performance further. On UDA for object detection, we show a promising result and believe that the performance can get better using more robust detection architectures.

References

[1] L. Bottou. Large-scale machine learning with stochastic gradient descent. In *COMPSTAT*. 2010. 5

[2] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. *PAMI*, 2018. 1, 5

[3] Y. Chen, W. Li, C. Sakaridis, D. Dai, and L. Van Gool. Domain adaptive faster R-CNN for object detection in the wild. In *CVPR*, 2018. 8

[4] Y. Chen, W. Li, and L. Van Gool. Road: Reality oriented adaptation for semantic segmentation of urban scenes. In *CVPR*, 2018. 2

[5] Y.-H. Chen, W.-Y. Chen, Y.-T. Chen, B.-C. Tsai, Y.-C. F. Wang, and M. Sun. No more discrimination: Cross city adaptation of road scene segmenters. In *ICCV*, 2017. 2

[6] 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. 5

[7] G. Csurka. *Domain adaptation for visual applications: A comprehensive survey*. Springer. 2

[8] A. Dosovitskiy, G. Ros, F. Codevilla, A. Lopez, and V. Koltun. CARLA: An open urban driving simulator. In *CoRL*, 2017. 2

[9] M. Everingham, S. M. A. Eslami, L. Van Gool, C. K. I. Williams, J. Winn, and A. Zisserman. The Pascal visual object classes challenge: A retrospective. *IJCV*, 2015. 5

[10] Y. Ganin and V. Lempitsky. Unsupervised domain adaptation by backpropagation. In *ICML*, 2015. 1, 2

[11] I. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. Courville, and Y. Bengio. Generative adversarial nets. In *NIPS*, 2014. 3, 4

[12] Y. Grandvalet and Y. Bengio. Semi-supervised learning by entropy minimization. In *NIPS*, 2005. 3

[13] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 5

[14] J. Hoffman, E. Tzeng, T. Park, J.-Y. Zhu, P. Isola, K. Saenko, A. Efros, and T. Darrell. CyCADA: Cycle-consistent adversarial domain adaptation. In *ICML*, 2018. 1, 2, 5, 6, 7

[15] J. Hoffman, D. Wang, F. Yu, and T. Darrell. FCNs in the wild: Pixel-level adversarial and constraint-based adaptation. *arXiv preprint arXiv:1612.02649*, 2016. 1, 2, 5, 6, 7

[16] W. Hong, Z. Wang, M. Yang, and J. Yuan. Conditional generative adversarial network for structured domain adaptation. In *CVPR*, 2018. 2

[17] H. Jain, J. Zepeda, P. Pérez, and R. Gribonval. Subic: A supervised, structured binary code for image search. In *ICCV*, 2017. 3

[18] H. Jain, J. Zepeda, P. Pérez, and R. Gribonval. Learning a complete image indexing pipeline. In *CVPR*, 2018. 3

[19] D. P. Kingma and J. Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 5

[20] S. Laine and T. Aila. Temporal ensembling for semi-supervised learning. *arXiv preprint arXiv:1610.02242*, 2016. 3

[21] D.-H. Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *ICML Workshop*, 2013. 3

[22] W. Liu, D. Anguelov, D. Erhan, C. Szegedy, S. E. Reed, C. Fu, and A. C. Berg. SSD: single shot multibox detector. In *ECCV*, 2016. 8

[23] J. Long, E. Shelhamer, and T. Darrell. Fully convolutional networks for semantic segmentation. In *CVPR*, 2015. 1

[24] M. Long, Y. Cao, J. Wang, and M. I. Jordan. Learning transferable features with deep adaptation networks. In *ICML*, 2015. 1, 2

[25] M. Long, H. Zhu, J. Wang, and M. I. Jordan. Unsupervised domain adaptation with residual transfer networks. In *NIPS*, 2016. 1, 2, 3

[26] Z. Murez, S. Kolouri, D. Kriegman, R. Ramamoorthi, and K. Kim. Image to image translation for domain adaptation. In *CVPR*, 2018. 3

[27] A. Paszke, S. Gross, S. Chintala, G. Chanan, E. Yang, Z. DeVito, Z. Lin, A. Desmaison, L. Antiga, and A. Lerer. Automatic differentiation in PyTorch. In *NIPS Workshop*, 2017. 5

[28] A. Radford, L. Metz, and S. Chintala. Unsupervised representation learning with deep convolutional generative adversarial networks. In *ICLR*, 2016. 5

[29] S. Ren, K. He, R. Girshick, and J. Sun. Faster R-CNN: Towards real-time object detection with region proposal networks. In *NIPS*, 2015. 8

[30] S. R. Richter, V. Vineet, S. Roth, and V. Koltun. Playing for data: Ground truth from computer games. In *ECCV*, 2016. 1, 2, 5

[31] G. Ros, L. Sellart, J. Materzynska, D. Vazquez, and A. M. Lopez. The SYNTHIA dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In *CVPR*, 2016. 1, 2, 5

[32] K. Saito, Y. Ushiku, T. Harada, and K. Saenko. Adversarial dropout regularization. In *ICLR*, 2018. 1, 2

[33] K. Saito, K. Watanabe, Y. Ushiku, and T. Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *CVPR*, 2018. 2

[34] S. Sankaranarayanan, Y. Balaji, A. Jain, S. Nam Lim, and R. Chellappa. Learning from synthetic data: Addressing domain shift for semantic segmentation. In *CVPR*, 2018. 1, 2

[35] A. Shafaei, J. J. Little, and M. Schmidt. Play and learn: Using video games to train computer vision models. In *BMVC*, 2016. 2

[36] C. E. Shannon. A mathematical theory of communication. *Bell system technical journal*, 1948. 3

[37] K. Simonyan and A. Zisserman. Very deep convolutional networks for large-scale image recognition. In *ICLR*, 2015. 5

[38] J. T. Springenberg. Unsupervised and semi-supervised learning with categorical generative adversarial networks. *ICLR*, 2016. 2, 3

[39] A. Tarvainen and H. Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *NIPS*, 2017. 3

[40] M. Tribus. *Thermostatics and thermodynamics*. van Nostrand, 1970. 4

[41] Y.-H. Tsai, W.-C. Hung, S. Schulter, K. Sohn, M.-H. Yang, and M. Chandraker. Learning to adapt structured output space for semantic segmentation. In *CVPR*, 2018. 2, 4, 5, 6, 7

[42] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Adversarial discriminative domain adaptation. In *CVPR*, 2017. 1, 2

[43] Z. Wu, X. Han, Y.-L. Lin, M. Gokhan Uzunbas, T. Goldstein, S. Nam Lim, and L. S. Davis. DCAN: Dual channel-wise alignment networks for unsupervised scene adaptation. In *ECCV*, 2018. 1, 2, 3

[44] H. Yan, Y. Ding, P. Li, Q. Wang, Y. Xu, and W. Zuo. Mind the class weight bias: Weighted maximum mean discrepancy for unsupervised domain adaptation. In *CVPR*, 2017. 1

[45] Y. Zhang, P. David, and B. Gong. Curriculum domain adaptation for semantic segmentation of urban scenes. In *ICCV*, 2017. 3

[46] Y. Zhang, Z. Qiu, T. Yao, D. Liu, and T. Mei. Fully convolutional adaptation networks for semantic segmentation. In *CVPR*, 2018. 3

[47] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia. Pyramid scene parsing network. In *CVPR*, 2017. 1

[48] J.-Y. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *ICCV*, 2017. 2

[49] X. Zhu. Semi-supervised learning literature survey. Technical Report 1530, Computer Sciences, University of Wisconsin-Madison, 2005. 2

[50] X. Zhu, H. Zhou, C. Yang, J. Shi, and D. Lin. Penalizing top performers: Conservative loss for semantic segmentation adaptation. In *ECCV*, September 2018. 3

[51] Y. Zou, Z. Yu, B. V. Kumar, and J. Wang. Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In *ECCV*, 2018. 1, 2, 3, 4, 5, 6, 7