ADVENT: Adversarial Entropy Minimization for Domain Adaptation in Semantic Segmentation

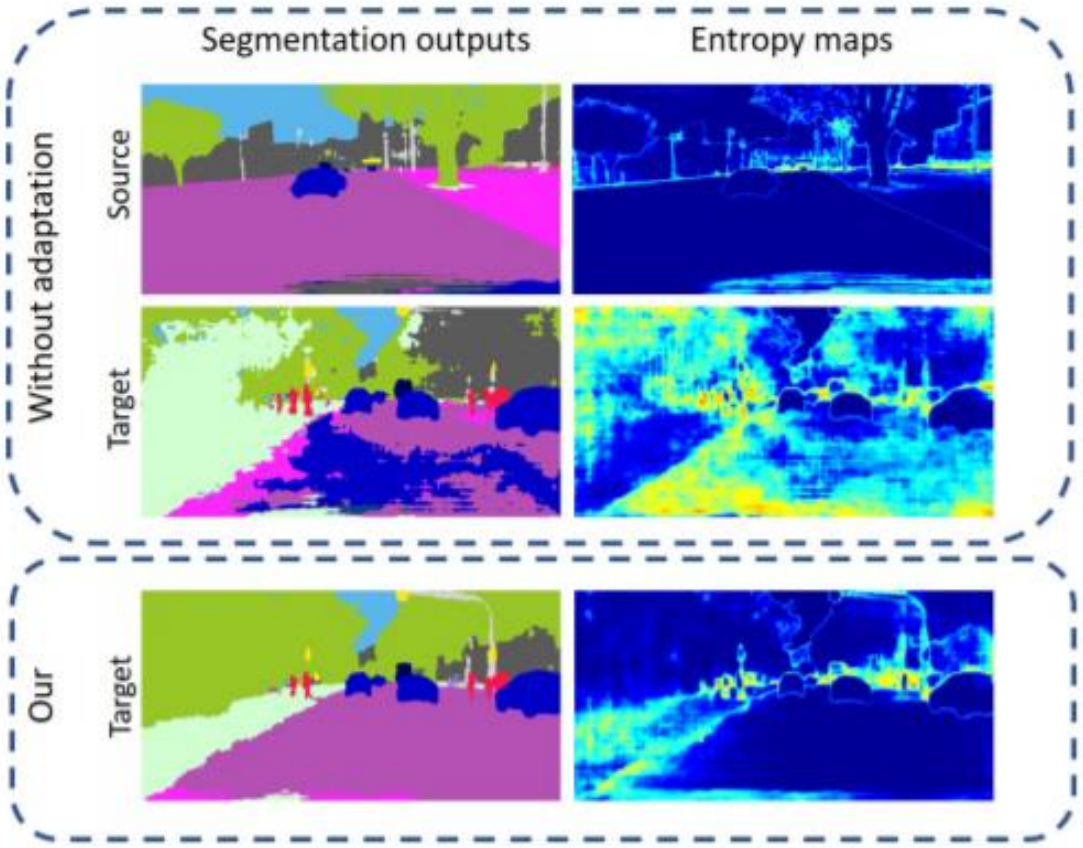
CVPR 2019

정상헌

DAVIAN

Data and Visual Analytics Lab

Motivation



- High entropy values on testing images.
- By **minimizing this entropy**, one can achieve confident classification outputs on testing images
- Furthermore, by **considering structural context** along with entropy minimization, one can achieve higher domain adaptation perf.

Proposed Method - Approach Overview

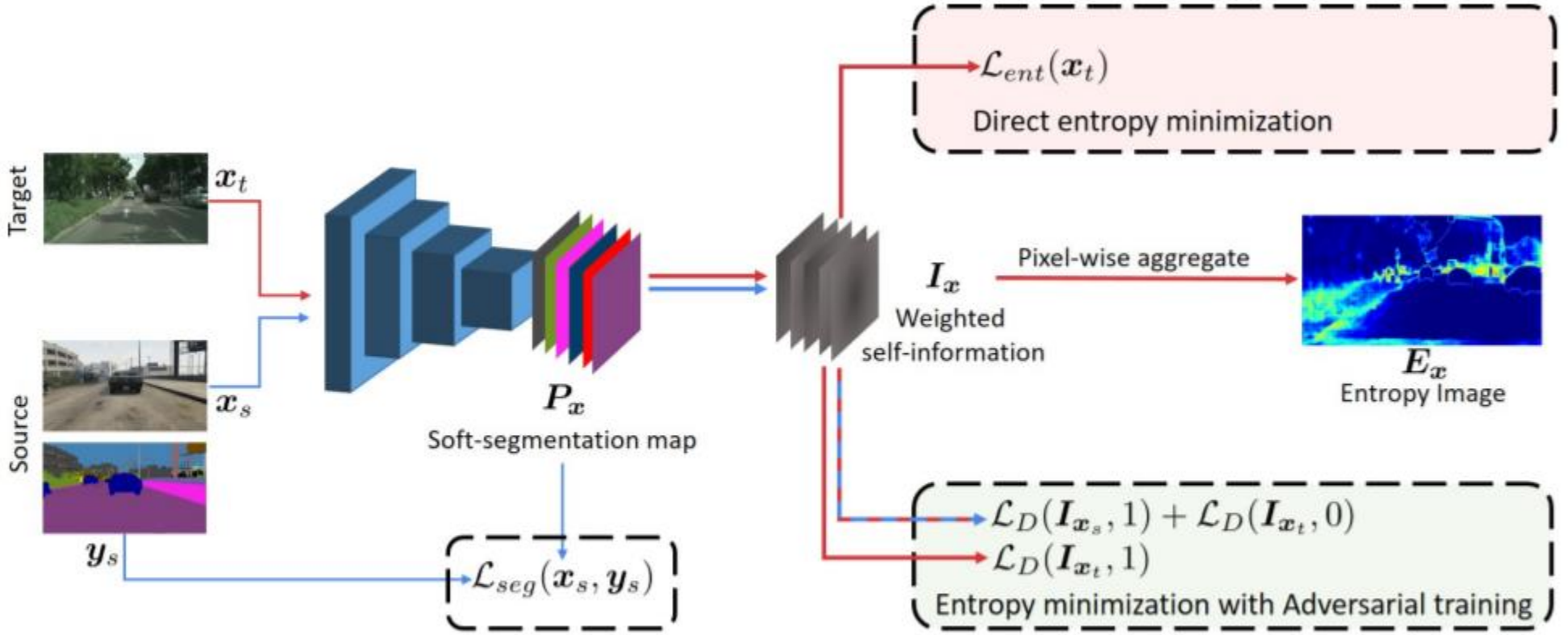


Figure 2: **Approach overview**. The figure shows our two approaches for UDA. First, *direct entropy minimization* minimizes the entropy of the target P_{x_t}, which is equivalent to minimizing the sum of weighted self-information maps I_{x_t}. In the second, complementary approach, we use adversarial training to enforce the consistency in I_x across domains. Red arrows are used for target domain and blue arrows for source. An example of entropy map is shown for illustration.

Proposed Method - Notation

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 segmentation 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:

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

Proposed Method 1 - Direct Entropy Minimization

- Most of previous approaches use the model's prediction y_t^ as a proxy for y_t.
- Instead, this paper directly minimizes the entropy in the target domain.
  - Normalized shannon entropy

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:

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

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:

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

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:

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

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

Comparison to Previous Method - Entropy Minimization <> Self-Training

- Thresholds pixels in entropy maps and uses the ones with small entropy values as pseudo-label (y_t^)
  - This requires complex threshold scheduling

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:

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

Proposed Method - Class Ratio Priors

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

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

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 $\boldsymbol{p}_s$.

Proposed Method 2 - Adversarial Training

- Direct entropy minimization neglects the structural dependencies between local semantics
  - Assumption: source and target domain share *strong similarities in semantic layout*
- Introduce fully-convolutional discriminator and adversarial training scheme.
- Feed *weighted self-information map* to the discriminator, and discriminator distinguishes it into source and 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

Proposed Method 2 - Adversarial Training

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).

Proposed Method 2 - Adversarial Training

- This approach indirectly minimizes the entropy on target domain by having target's entropy distribution similar to the source.

- This loss forces the backbone network to extract an unified representation from both source and target domains.

- How…? Proof…? - Domain Adversarial Training of Neural Network

Proof - Notation

We consider classification tasks where X is the input space and $Y = \{0, 1, \ldots, L-1\}$ is the set of L possible labels. Moreover, we have two different distributions over $X \times Y$, called the *source domain* $\mathcal{D}_{\mathrm{S}}$ and the *target domain* $\mathcal{D}_{\mathrm{T}}$. An *unsupervised domain adaptation* learning algorithm is then provided with a *labeled source sample* S drawn *i.i.d.* from $\mathcal{D}_{\mathrm{S}}$, and an *unlabeled target sample* T drawn *i.i.d.* from $\mathcal{D}_{\mathrm{T}}^{X}$, where $\mathcal{D}_{\mathrm{T}}^{X}$ is the marginal distribution of $\mathcal{D}_{\mathrm{T}}$ over X.

$$S = \{(\mathbf{x}_i, y_i)\}_{i=1}^{n} \sim (\mathcal{D}_{\mathrm{S}})^n\,; \quad T = \{\mathbf{x}_i\}_{i=n+1}^{N} \sim (\mathcal{D}_{\mathrm{T}}^{X})^{n'},$$

with $N = n + n'$ being the total number of samples. The goal of the learning algorithm is to build a classifier $\eta : X \to Y$ with a low *target risk*

$$R_{\mathcal{D}_{\mathrm{T}}}(\eta) = \Pr_{(\mathbf{x},y)\sim\mathcal{D}_{\mathrm{T}}} \Big(\eta(\mathbf{x}) \neq y\Big),$$

while having no information about the labels of $\mathcal{D}_{\mathrm{T}}$.

Proof - $\mathcal{H}$-divergence

Definition 1 (Ben-David et al., 2006, 2010; Kifer et al., 2004) *Given two domain distributions* $\mathcal{D}_{\mathrm{S}}^{X}$ *and* $\mathcal{D}_{\mathrm{T}}^{X}$ *over* X*, and a hypothesis class* $\mathcal{H}$*, the* $\mathcal{H}$-divergence *between* $\mathcal{D}_{\mathrm{S}}^{X}$ *and* $\mathcal{D}_{\mathrm{T}}^{X}$ *is*

$$d_{\mathcal{H}}(\mathcal{D}_{\mathrm{S}}^{X}, \mathcal{D}_{\mathrm{T}}^{X}) = 2 \sup_{\eta \in \mathcal{H}} \left| \Pr_{\mathbf{x} \sim \mathcal{D}_{\mathrm{S}}^{X}} [\eta(\mathbf{x}) = 1] - \Pr_{\mathbf{x} \sim \mathcal{D}_{\mathrm{T}}^{X}} [\eta(\mathbf{x}) = 1] \right|.$$

That is, the $\mathcal{H}$-divergence relies on the capacity of the hypothesis class $\mathcal{H}$ to distinguish between examples generated by $\mathcal{D}_{\mathrm{S}}^{X}$ from examples generated by $\mathcal{D}_{\mathrm{T}}^{X}$. Ben-David et al. (2006, 2010) proved that, for a symmetric hypothesis class $\mathcal{H}$, one can compute the *empirical* $\mathcal{H}$*-divergence* between two samples $S \sim (\mathcal{D}_{\mathrm{S}}^{X})^{n}$ and $T \sim (\mathcal{D}_{\mathrm{T}}^{X})^{n'}$ by computing

$$\hat{d}_{\mathcal{H}}(S, T) = 2 \left(1 - \min_{\eta \in \mathcal{H}} \left[\frac{1}{n} \sum_{i=1}^{n} I[\eta(\mathbf{x}_i) = 0] + \frac{1}{n'} \sum_{i=n+1}^{N} I[\eta(\mathbf{x}_i) = 1] \right] \right), \quad (1)$$

where $I[a]$ is the indicator function which is 1 if predicate a is true, and 0 otherwise.

Proof - Proxy $\mathcal{A}$-distance

where the examples of the source sample are labeled 0 and the examples of the target sample are labeled 1. Then, the risk of the classifier trained on the new data set U approximates the "min" part of Equation (1). Given a generalization error ϵ on the problem of discriminating between source and target examples, the $\mathcal{H}$-divergence is then approximated by

$$\hat{d}_{\mathcal{A}} = 2\,(1 - 2\epsilon)\,. \tag{3}$$

In Ben-David et al. (2006), the value $\hat{d}_{\mathcal{A}}$ is called the *Proxy $\mathcal{A}$-distance* (PAD). The *$\mathcal{A}$-distance* being defined as $d_{\mathcal{A}}(\mathcal{D}_{\mathrm{S}}^{X}, \mathcal{D}_{\mathrm{T}}^{X}) = 2 \sup_{A \in \mathcal{A}} \left| \Pr_{\mathcal{D}_{\mathrm{S}}^{X}}(A) - \Pr_{\mathcal{D}_{\mathrm{T}}^{X}}(A) \right|$, where $\mathcal{A}$ is a subset of X. Note that, by choosing $\mathcal{A} = \{A_\eta | \eta \in \mathcal{H}\}$, with A_η the set represented by the characteristic function η, the $\mathcal{A}$-distance and the $\mathcal{H}$-divergence of Definition 1 are identical.

Proof - Upper Bound of Target Risk

Theorem 2 (Ben-David et al., 2006) *Let $\mathcal{H}$ be a hypothesis class of VC dimension d. With probability $1-\delta$ over the choice of samples $S \sim (\mathcal{D}_{\mathrm{S}})^n$ and $T \sim (\mathcal{D}_{\mathrm{T}}^X)^n$, for every $\eta \in \mathcal{H}$:*

$$R_{\mathcal{D}_{\mathrm{T}}}(\eta) \leq R_S(\eta) + \sqrt{\frac{4}{n}\left(d \log \frac{2e\,n}{d} + \log \frac{4}{\delta}\right)} + \hat{d}_{\mathcal{H}}(S,T) + 4\sqrt{\frac{1}{n}\left(d \log \frac{2n}{d} + \log \frac{4}{\delta}\right)} + \beta\,,$$

with $\beta \geq \inf_{\eta^* \in \mathcal{H}} \left[R_{\mathcal{D}_{\mathrm{S}}}(\eta^*) + R_{\mathcal{D}_{\mathrm{T}}}(\eta^*)\right]$, *and*

$$R_S(\eta) = \frac{1}{n}\sum_{i=1}^{m} I\left[\eta(\mathbf{x}_i) \neq y_i\right]$$

Experiments - GTA5 → Cityscapes

(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**

Experiments - Synthia → Cityscapes

(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**

Thank you!