Uncertainty Reduction for Model Adaptation in Semantic Segmentation

Prabhu Teja S [a, b]   François Fleuret [a, c]
[a]Idiap Research Institute,   [b]EPFL,   [c]University of Geneva
prabhu.teja@idiap.ch, francois.fleuret@unige.ch

Abstract

Traditional methods for Unsupervised Domain Adaptation (UDA) targeting semantic segmentation exploit information common to the source and target domains, using both labeled source data and unlabeled target data. In this paper, we investigate a setting where the source data is unavailable, but the classifier trained on the source data is; hence named "model adaptation". Such a scenario arises when data sharing is prohibited, for instance, because of privacy, or Intellectual Property (IP) issues.

To tackle this problem, we propose a method that reduces the uncertainty of predictions on the target domain data. We accomplish this in two ways: minimizing the entropy of the predicted posterior, and maximizing the noise robustness of the feature representation. We show the efficacy of our method on the transfer of segmentation from computer generated images to real-world driving images, and transfer between data collected in different cities, and surprisingly reach performance comparable with that of the methods that have access to source data.

1. Introduction

The successes of deep learning, especially in semantic segmentation, have been driven in large part by large architectures [10, 9]. While architectural improvements have made tremendous strides in improving performance, these large computation machines require vast amounts of data to be trained adequately. A few large densely labeled datasets such as Cityscapes [17], or Berkeley Deep Drive [73] exist, but producing them, for instance for a new use case due to a different environment, or sensors, is very laborious and expensive; Cordts et al. [17] reports that each image required about 90 minutes for labeling and verification. Added to the difficulty of collecting data for segmentation, minor changes in conditions like change of cities between train and test results in a drop of performance [15], as so does change in lighting conditions [19], as it violates the pivotal assumption of test and train data being sampled from the same distribution [57]. Thus, owing to the large costs, both time and economic, creating annotated datasets for each scenario is impractical, motivating the re-use or transfer of knowledge from available images to requisite application.

The problem of transfer for segmentation has been predominantly investigated in literature (see Section 4) in two settings: adapting a model trained on synthetically generated images to real images, and adapting a model to cities different than the ones it has been trained for. We too adopt these settings for study in this paper. The synthetic-to-real images adaptation has attracted a lot of attention as the cost of generating segmentation ground truth for graphically rendered frames like GTA [58], or Synthia [60] is substantially lower. Richter et al. [58] labeled a total of 24,966 frames at an average of 7 seconds per frame, a large drop from 90 minutes taken for Cityscapes. However, due the nature of their generation, these synthetic images have a significant domain gap to the real images, which results in a large drop in the performance of networks trained on synthetic data when they are used on real images. Similarly, given the existing real world datasets like Cityscapes, it is paramount that the knowledge learned on these datasets is effectively transferred to different scenarios, without having to provide annotated data for each scenario.

A large portion of the methods that has been devoted to tackling this problem, like some of the ones reviewed in Section 4, require the labeled source data to be available along with the unlabeled target data for the adaptation process. We, in this paper, focus on the problem where the source data itself is not available but the source trained classifier is [16]. This is similar to life-long learning [63], where the goal is to adapt to several tasks over several domains, and the only information payload that is carried over is the model itself. Differing from that, we are not concerned with preserving the performance on the source task. The current problem of source data-less transfer is pertinent when there exist data sharing restrictions on the source data; a common way to circumvent this is to share the trained classifier from which the input data itself cannot be reconstructed. A classical application is in medical image processing, where patient data cannot be freely shared due to privacy concerns, but a trained model can be. Another relevant application where we envisage such a setting is in search-and-rescue operations, where data is collected on a mobile drone, and the segmentor network is adapted based on only the data collected, without having the need to access the original labeled dataset. Thus the problem of domain adaptation in the absence of source data, termed *model adaptation* [16], is of practical significance.

In this work, we study the problem of *model adaptation* for semantic segmentation. To the best of our knowledge, ours is the first work to do so. In the absence of source labeled data that has been effectively exploited by previous works, we enforce auxiliary properties that are desirable in a system, namely confident predictions for the target data, and noise resilience, and thereby increased stability of classification to parameter choices. To this end, we propose a method that uses feature corruption [16, 47, 54], and entropy regularization [29, 69]. We find that having access only to the source classifier, along with unlabeled target data, can result in performance comparable to the case where source data is also available.

2. Handling the absence of labeled source data

2.1. A toy problem

To motivate our method, we consider the ideal case depicted on Figure 1. Let us consider a simple case of binary classification of $Y = \{0, 1\}$ for a scalar input feature $x \in \mathbb{R}$. The probability of classification is defined using a sigmoid function of the input x and a threshold t

$$p(Y = 1|X = x; t) = \frac{1}{1 + e^{-(x-t)}} \tag{1}$$

For illustration in Figure 1, we show the class conditional distributions, though we do not use class information. With only the information of the feature distribution $\mu_X(x)$, our goal is to reason about scenarios that are likely to generalize better. If the labels are available, traditional wisdom tells us that Figure 1c is likely the ideal scenario to attain in terms of generalization [62]. We make it a little more concrete here, in the case where the labels are not available.

In Figure 1a, the feature x is given by a source trained feature extractor, and t_S is the threshold learned. It is apparent that such a threshold is quite likely ill-suited for the target domain, as it places the decision boundary in a high density region of the feature space, contradicting the traditional cluster & continuity assumptions [7, 41, 6]. However, if we change the threshold to t_*, we expect better generalization performance. Note that this is also the classifier for which the entropy of output probability predictions over the distribution $\mu_X(x)$ is the lowest. We mathematically define this idea and show numerical simulations for these cases in Appendix B.

We would like to modify the feature extractor itself such that the class conditional distributions overlap lesser than in Figure 1a. This can be achieved by imposing an entropy penalty on output posterior predictions, and using that penalty to train the feature extractor too. We show this in Figure 1b. As the overlap of the class-conditional data distributions decreases, we expect the generalization performance to improve.

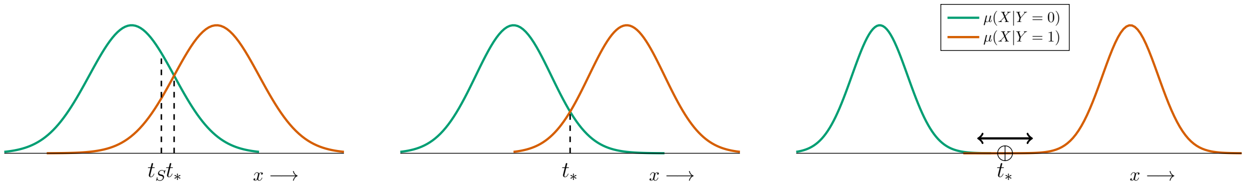
The ideal scenario is shown in Figure 1c, where a large number of thresholds can separate the two classes, and we choose one that results in the least uncertain predictions of target data. One can draw parallels to max-margin methods like the SVM, where one is interested in finding a separator that is optimally distant from all classes. In a nutshell, we see that a classifier that has stable predictions for a range of parameter choices is likely to generalize better. This is, of course, in our context where we do not have access to labeled training data. In order to accomplish this, we need to go beyond entropy quantification of the predicted labels; we enforce stability of predictions over noisy features $x \pm \epsilon$.

2.2. Proposed method

We, first, formally define the problem. Let $\mathcal{X} \in \mathbb{R}^D$ be the input, and $\mathcal{Y} \in \{1 \ldots K\}$ the labels. Let $\mathcal{S}$ with density $\mu_S(\boldsymbol{x})$ be the source domain and $\mathcal{T}$ with density $\mu_T(\boldsymbol{x})$ be the target one, where we do not have access to the labels of $\mathcal{T}$ while training. Let $X_S = \{(\boldsymbol{x}_i, y_i)\ i = 1 \ldots N_s, \boldsymbol{x}_i \sim \mu_S(\boldsymbol{x}), y_i \sim p_S(y)\}$ be the source data, and $X_T = \{(\mathbf{x}_i,)\ i = 1 \ldots N_t, \boldsymbol{x}_i \sim \mu_T(\boldsymbol{x})\}$ be the target data. Let $p_T(y)$ be the target domain label prior that is unknown to us. Here $\mu_S \neq \mu_T$. We do not have direct access to X_S, but have access to a network trained on X_S. Let us denote that network by $f(\boldsymbol{x}, \boldsymbol{\theta_S}) \equiv f_S(\boldsymbol{x})$ where $\boldsymbol{\theta_S}$ denotes the parameters of the network trained on the source data. Let g and h denote the feature extractor and classifier respectively, whose composition is f *i.e.,* $f = h \circ g$. Let also $\boldsymbol{\theta}_g$ and $\boldsymbol{\theta}_h$ be their corresponding parameters. In our case, the network g refers to the ResNet-101 backbone, and h refers to the ASPP [10] decoder, which we describe in Section 3.3. For convenience, h also subsumes the softmax layer, and thus $f(\boldsymbol{x}; \boldsymbol{\theta}) = p(\boldsymbol{y}|\boldsymbol{x}; \boldsymbol{\theta})$, where $\boldsymbol{y}$ is a vector of probabilities from which the predicted label is sampled. Summarizing, we have f_S and X_T, and our goal is to modify f_S such that its performance is improved on $\mathcal{T}$ data.

2.2.1 Optimizing the feature extractor to generate robust features

The likelihood of predictions $\boldsymbol{y}$ for an input of $\boldsymbol{x} \in \mathcal{T}$ is computed by $f(\boldsymbol{x}; \boldsymbol{\theta})$, on which an entropy penalty can be imposed [29]. However with a trivial application of this, for the illustration in Figure 1b, the network can learn to separate the distribution by placing the threshold t_* at any arbitrary point and shearing the feature distributions around



(a)

(b)

(c)

Figure 1: Given only the feature distributions in 1a to 1c in a two-class scenario, which one generalizes the best? In 1a, we show the distribution of features extracted by source trained network and the corresponding source threshold on the target data. Tuning the threshold to t_* from t_S is expected to result in better generalization. If we can modify the feature extractor itself, reducing the uncertainty of classification over the domain gives us 1b. This can be achieved by penalizing the entropy of predictions of each of the data points. We argue that while entropy is seemingly sufficient, we need to reduce the uncertainty in the predictions of the network over a wide range of parameter choices obtain better separation of the data like in 1c, and thereby better generalization. Details in Section 2.1.

it, thereby being stable to the choice of the threshold instead of attaining separation of features as in Figure 1c. For simple problems like the one in Figure 1, it can be achieved by enforcing stability to input perturbations. However in deep networks, the network can learn to denoise the inputs in the initial few layers of processing. Using stronger augmentations like the ones in Chen et al. [13] are ill-suited for segmentation, as classification networks are expected to be invariant to such noise, whereas segmentation networks are expected to be equivariant. This can be remedied by adding structured noise to inputs such that the layout of the input objects is preserved (for example, modifying the colors of objects). We achieve this by adding noise to the feature representation using dropout, similar to Ouali et al. [54], that acts as a structured noise in the input space.

Let $\hat{\boldsymbol{y}}^i = f(\boldsymbol{x}; \hat{\boldsymbol{\theta}}_i)$ be the output of the network using the i^{th} instantiation of dropout, and $\boldsymbol{y} = f(\boldsymbol{x}; \boldsymbol{\theta})$ the output for the network without dropout. In such a case, we propose to compute the *uncertainty loss* as

$$\mathcal{L}_{un} = \frac{1}{N} \sum_{i=1}^{N} \left(\hat{\boldsymbol{y}}^i - \boldsymbol{y} \right)^2 \tag{2}$$

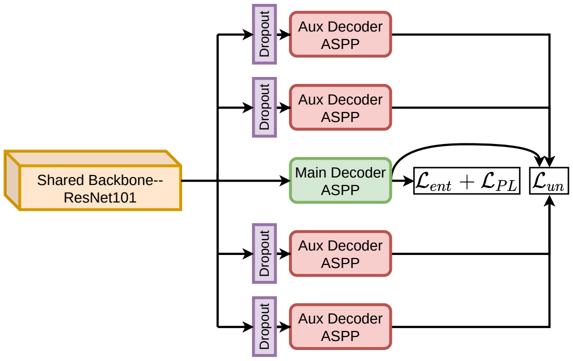


Figure 2: An illustration of the proposed method. The main decoder is trained with Equations (3) and (5), whereas the backbone feature extractor is trained with a combination of Equations (2), (3) and (5). At test time, we discard the auxiliary branches, and thus there is no additional computation at inference.

To implement Equation (2), we introduce dropout only between g and h. In the context of Bayesian neural networks, such a method has been termed *LastLayer-Dropout* elsewhere [55]. Instead of using a single branch that predicts the output with dropout-weights, we use multiple decoders $\hat{h}$ that takes in dropped out features predicted by g. Thus $\boldsymbol{y} = f(\boldsymbol{x}; \boldsymbol{\theta}) = h(g(\boldsymbol{x}; \boldsymbol{\theta}_g); \boldsymbol{\theta}_h)$ and $\hat{\boldsymbol{y}} = f(\boldsymbol{x}; \hat{\boldsymbol{\theta}}) = \hat{h}(g(\boldsymbol{x}; \boldsymbol{\theta}_g); \hat{\boldsymbol{\theta}}_h)$. Thus, each auxiliary decoder $\hat{h}$ sees only partial feature tensor, and is required to come as close to reconstructing the original label tensor. We, experimentally, find that freezing the main decoders' weights $\boldsymbol{\theta}_h$ while training the auxiliary decoders' parameters $\hat{\boldsymbol{\theta}}_h$ results in better performance. Our full method is shown in Figure 2.

The use of multiple auxiliary decoders has been proposed before in Meyerson and Miikkulainen [49] for defining pseudo-tasks for deep multi-task learning, and in Ouali et al. [54] for semi-supervised learning. Meyerson and Miikkulainen [49], however, use ground-truth labels to train each of the auxiliary classifiers. We note the similarities to several previous works that focus on generating robust representations using various kinds of feature corruptions [11, 28] that are class agnostic, and ones that are class specific like guided cutout [20]. However, we experimentally find that advanced forms of feature noising (like class dependent noising or targeted cutout) do not work as well, and we hypothesize that it is due to the unreliable predictions of the source trained network on target images.

The proposed uncertainty loss in Equation (2) improves the noise resilience of the network. In addition to that we use an entropy regularizer that minimizes the entropy of the network's predictions, that results in better empirical performance.

$$\mathcal{L}_{ent} = H\{f(\boldsymbol{x}; \boldsymbol{\theta})\} \tag{3}$$

where, $H\{\}$ is the entropy of the probability distribution over all K classes for an input $\boldsymbol{x}$. We hypothesize that this is because the dropout noise modifies the decision boundary given by the optimizer itself, and thus is beneficial to explicitly regularize the predictions given that estimate of weights. We find that we do not need a diversity enforcing loss, as specified in [37, 44], to prevent degeneracies.

2.2.2 Regularizing using the source trained model

We note that the losses in Equations (2) and (3) do not use the information in the source trained classifier, but enforce certain properties to be satisfied by the network on the target domain. In Figure 1c, an interchanged labeling *i.e.,* where class-0 is predicted class-1 and vice-versa, results in the same loss value for Equations (2) and (3). To avoid such issues, and to infuse plausible class structure to the data, we use pseudo-labeling.

Pseudo-labeling or self-training has been a mainstay method in the semi-supervised problems, prior to the deep learning era. However, owing to the availability of large-scale datasets, it has been used to great success [1, 41, 71] for several classification problems. Traditional methods use the class with the highest predicted probability as the ground-truth for each unlabeled sample. However, in the case of a domain change, the accuracy of such predicted pseudo-labels is low. So we use the following modification to the standard definition

$$\boldsymbol{y}_{PL} = \begin{cases} \arg\max f(\boldsymbol{x}, \boldsymbol{\theta}) & \text{if } \max(f(\boldsymbol{x}, \boldsymbol{\theta})) \geq \tau \\ \text{IGNORE} & \text{otherwise} \end{cases} \tag{4}$$

i.e., we only consider as pseudo-labels the samples that are at least τ confident. The samples with the *IGNORE* label do not contribute to the loss. However choosing τ is a non-trivial task, as too low a threshold will result in wrong labels, and too high a threshold will result in no target data being bootstrapped for training. In this work we adopt the strategy of class balanced thresholding [81], where τ is varied per class, such that a certain proportion of points per class are always selected. We define the pseudo labeling loss to be the cross-entropy loss with the pseudo labels as defined in Equation (4)

$$\mathcal{L}_{PL} = -\mathbb{1}_{\boldsymbol{y}_{PL}}^{T} \log(\boldsymbol{y}) \tag{5}$$

where $\mathbb{1}_{\boldsymbol{y}_{PL}}$ is one-hot encoded vector of $\boldsymbol{y}_{PL}$, and log is applied element-wise.

Thus, the overall loss function being optimized is the combination of Equations (2), (3) and (5):

$$\mathcal{L} = \mathcal{L}_{PL} + \lambda_{ent}\mathcal{L}_{ent} + \lambda_{un}\mathcal{L}_{un} \tag{6}$$

where $\lambda_{ent}, \lambda_{un}$ are the weights of the individual loss terms.

Our work is connected to recent work in interesting ways: if Equation (2) is construed to be a form of self-supervision, our method can be interpreted as a form of *test-time training* [65]. *Test-time training* proposes to use an auxiliary task at test-time that helps combat domain shift from the training set. We differ from them in that we do not update the network at test time, but do so when given target domain data. Similarly, our work, conceptually, uses self-supervision for domain adaptation, similar to [64], but doesn't need access to source labeled data. Additionally, our method can be interpreted as a form of making the network a *bit Bayesian* [38], where instead of placing Gaussian posterior on the weights of the penultimate layer's weights, we use dropout distribution. As previously mentioned, our method has similarities to pseudo-tasks, in multi-task learning [49], which uses labeled data for training.

3. Experiments

3.1. A toy problem

To elucidate the utility of each of the terms in Equation (6), we use a toy problem as shown in Figure 3. In Figure 3 a & b we show the source and target datasets; we use a rotated version of the source data as the target data. As it is in our case, we do not use the labels of the target data. We train a small two layer neural network with batchnorm and ReLU activations. We provide the exact architecture in Table 5 in the appendix. We take the outputs before the last classifier layer as the features of the network, and use the techniques that we described in Section 2.2 to train the network, except we do not use pseudo labeling for this problem. We use a feature dimensionality of 2 and plot the target features in Figure 3c with the source classifier. It is very apparent that the source network extracts features that do not transfer well. In Figure 3d, with a simple entropy regularization on the target data, we see that the performance improves tremendously. However, some of the blue points are very close to the separating line. To remedy this, our proposed feature noise decoder (detailed in Section 2.2) pushes the points away from the separating line. This can be interpreted as a form of increasing the stability of the classification, and thereby reducing uncertainty, which we hypothesize to be the key to better generalization.

3.2. Datasets and Evaluation

We demonstrate the efficacy of our method on the standard domain adaptation tasks of GTA [58]→ Cityscapes [17] (*GTA-CS*) and Synthia [60]→ Cityscapes (*SYN-CS*) and Cityscapes→NTHU Crosscity[15] (*CS-CC*), a standard test setting used in several previous works (Section 4). Cityscapes consists 2975 annotated images, each of size 2048 × 1024, that act as our training set. It has

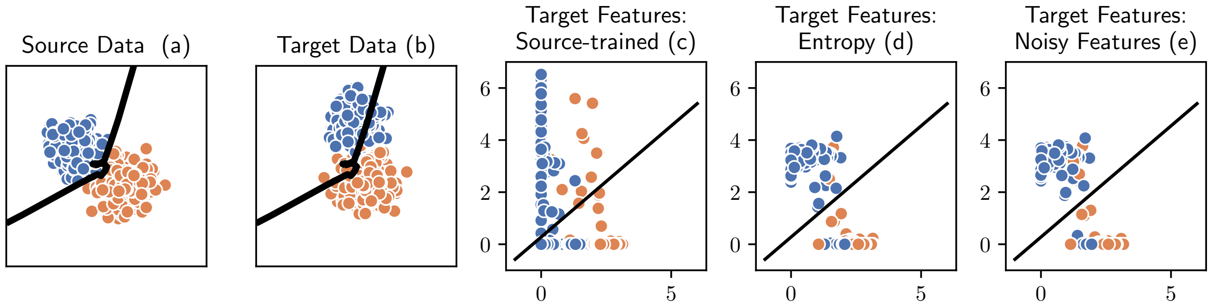


Figure 3: A toy example in $\mathbb{R}^2$ to illustrate our algorithm. The target data is unlabeled, however we shade it for illustration. The thick line in all the figures is the separator. In (a) we show the source data and the classifier trained on that, and in (b) the classifier unmodified on the target data. In (c) we visualize the features extracted by the source trained classifier on target data. With a simple entropy penalty on the target data, we get substantially better performance (d). With the additional uncertainty loss, we see that the features extracted are pulled further away from the separating line in (e). Details in Section 3.1.

500 images as the validation set, which we use to benchmark our method. It consists of 19 semantic classes for segmentation. The GTA dataset consists of 24966 frames, of size 1914 × 1052 grabbed from the famous game Grand Theft Auto. The ground truth is generated by the game renderer itself. It shares the same 19 semantic classes as Cityscapes. Synthia has 9400 images of size 1280 × 760 synthetic images, and shares 16 classes with Cityscapes. For our method, we use a network trained on Synthia or GTA, and adapt it to Cityscapes using the 2975 training images without their ground-truth labels. Crosscity dataset has been recorded in four cities: Rome, Rio, Taipei, and Tokyo with each image of resolution 2048×1024. Following [12], we use their experimental setup of 3200 unlabeled images as target training data, and 100 labeled images as target test data. This adaptation task has 13 shared classes. We use the pretrained models provided by Chen et al. [12] for the source trained model.

To evaluate our method, we use Intersection-over-Union (IoU) for each class, and its average mean-Intersection-over-Union (mIoU) over all classes. We report the metrics for all the 19 classes of *GTA-CS* adaptation, for the 16 common classes for the *SYN-CS* experiments, and for the 13 common classes in the *CS-CC* experiments. In accordance with some recent papers, we also report a mIoU* comparing only 13 classes for the Synthia to Cityscapes adaptation task.

3.3. Implementation details

To facilitate a fair comparison with relevant works, we use a DeepLab V2 network [10] with a ResNet-101 backbone [31]. Using the notation defined in Section 2, g is the ResNet-101 based feature extractor, and h is the Atrous Spatial Pyramidal Pooling (ASPP) decoder. The ResNet 101 backbone is pre-trained on ImageNet. ASPP [10] is a multi-scale decoder, that is used to aggregate multi-scale information for segmentation. It has 4 parallel atrous convolutions of various rates, which capture long-range contextual information from extracted features. We train the model on the source domain with stochastic gradient descent with a learning rate of 2.5×10^{-4} with a weight decay of 0.0005, momentum of 0.9. We use a poly learning rate decay scheduler with power of 0.9. For the adaptation experiments, we use a lower learning rate of 5×10^{-5}, with other parameters staying the same, with no learning rate decay. We also use a $10\times$ the learning rate for ASPP decoders [10] in our experiments. For all our experiments, we use $\lambda_{\text{ent}} = 1.0$, $\lambda_{un} = 0.1$. We use the defaults prescribed by Zou et al. [80] to extract class-balanced pseudo labels. For the *CS-CC* experiments, we run the adaptation for only 2 epochs, and for *GTA-CS*, *SYN-CS* for 6 epochs. Code will be available at https://git.io/JthPp.

3.4. Results

We show the performance of our proposed method on the tasks of *GTA-CS* adaptation in Table 1a, *SYN-CS* adaptation in Table 1b and *CS-CC* adaptation in Table 1c. We obtain results that are at-par or better than some of the classic works on unsupervised domain adaptation (with source data). This work is, however, not a claim that source data is not necessary for domain adaptation. The utility of source data has been exploited effectively by recent methods through style transfer techniques [72, 35], thus they obtain higher performance than us. Our proposed method improves substantially on the larger background classes, and we hypothesize this is because the source classifier can make reliable predictions for these classes. The perceptual domain gap seems to influence the transfer performance too, as we get very comparable performance for *GTA-CS* and *CS-CC* experiments, compared to the results we obtain for *SYN-CS* experiments.

Uses source data	Method	Road	Sidewalk	Building	Wall	Fence	Pole	Tra. light	Tra. Sign	Veg.	Terrain	Sky	Person	Rider	Car	Truck	Bus	Train	Motorbike	Bike	mIoU
Yes	AdaptSegNet [66]	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
	AdvEnt [69]	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
	FCAN [76]	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	46.6
	CBST [81]	91.8	53.5	80.5	32.7	21.0	34.0	28.9	20.4	83.9	34.2	80.9	53.1	24.0	82.7	30.3	35.9	16.0	25.9	**42.8**	45.9
	MRKLD [80]	91.0	**55.4**	80.0	**33.7**	21.4	**37.3**	32.9	24.5	**85.0**	34.1	80.8	57.7	24.6	84.1	27.8	30.1	**26.9**	26.0	42.3	**47.1**
	LRENT [80]	91.8	53.5	80.5	32.7	21.0	34.0	29.0	20.3	83.9	34.2	80.9	53.1	23.9	82.7	30.2	35.6	16.3	25.9	**42.8**	45.9
	Source	71.3	19.2	69.1	18.4	10.0	35.7	27.3	6.8	79.6	24.8	72.1	57.6	19.5	55.5	15.5	15.1	11.7	21.1	12.0	33.8
No	Our method	**92.3**	55.2	**81.6**	30.8	18.8	**37.1**	17.7	12.1	84.2	35.9	**83.8**	57.7	24.1	81.7	27.5	44.3	6.9	24.1	40.4	45.1

(a) Results of GTA5 → Cityscapes (*GTA-CS*) domain adaptation.

Uses source data	Method	Road	Sidewalk	Building	Wall*	Fence*	Pole*	Tra. Light	Tra. sign	Veg.	Sky	Person	Rider	Car	Bus	Motorbike	Bike	mIoU	mIoU*
Yes	AdaptSegNet [66]	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
	AdvEnt [69]	**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
	CBST [81]	68.0	29.9	76.3	10.8	1.4	33.9	22.8	29.5	77.6	78.3	60.6	28.3	81.6	23.5	18.8	39.8	42.6	48.9
	MRKLD [80]	67.7	32.2	73.9	10.7	1.6	**37.4**	22.2	31.2	80.8	80.5	**60.8**	**29.1**	**82.8**	25.0	19.4	**45.3**	**43.8**	**50.1**
	LRENT [80]	65.6	30.3	74.6	13.8	1.5	35.8	**23.1**	29.1	77.0	77.5	60.1	28.5	82.2	22.6	**20.1**	41.9	42.7	48.7
	Source	64.3	21.3	73.1	2.4	1.1	31.4	7.0	27.7	63.1	67.6	42.2	19.9	73.1	15.3	10.5	38.9	34.9	40.3
No	Our method	59.3	24.6	77.0	**14.0**	**1.8**	31.5	18.3	**32.0**	**83.1**	80.4	46.3	17.8	76.7	17.0	18.5	34.6	39.6	45.0

(b) Results of Synthia → Cityscapes (*SYN-CS*) domain adaptation.

City	Uses source data	Method	Road	Sidewalk	Building	Tra. Light	Tra. Sign	Veg.	Sky	Person	Rider	Car	Bus	Motorbike	Bike	mIoU
Rome	Yes	Cross city [15]	79.5	29.3	84.5	0.0	22.2	80.6	82.8	29.5	13.0	71.7	37.5	25.9	1.0	42.9
		MaxSquare [12]	82.9	32.6	86.7	**20.7**	**41.6**	85.0	**93.0**	47.2	**22.5**	**82.2**	53.8	**50.5**	9.9	**54.5**
	No	Source	85.0	34.7	86.4	17.5	39.0	84.9	85.4	43.8	15.5	81.8	46.3	38.4	4.8	51.0
		Our Method	**86.2**	**39.1**	**87.6**	14.3	37.8	**85.5**	88.5	**49.9**	21.9	81.6	**56.3**	40.4	**10.4**	53.8
Rio	Yes	Cross city [15]	74.2	43.9	79.0	2.4	7.5	77.8	69.5	39.3	10.3	67.9	41.2	27.9	10.9	42.5
		MaxSquare [12]	76.9	48.8	**85.2**	13.8	18.9	**81.7**	**88.1**	**54.9**	**34.0**	76.8	39.8	**44.1**	**29.7**	53.3
	No	Source	74.2	42.2	84.0	12.1	20.4	78.3	87.9	50.1	25.6	76.6	40.0	27.6	17.0	48.9
		Our Method	**82.8**	**57.0**	84.8	**17.4**	**24.0**	80.5	86.0	54.2	27.7	**78.2**	**43.8**	38.3	21.5	**53.5**
Tokyo	Yes	Cross city [15]	83.4	35.4	72.8	12.3	12.7	77.4	64.3	42.7	21.5	64.1	**20.8**	8.9	40.3	42.8
		MaxSquare [12]	81.2	30.1	77.0	12.3	**27.3**	**82.8**	**89.5**	**58.2**	**32.7**	**71.5**	5.5	**37.4**	**48.9**	**50.5**
	No	Source	81.4	28.4	**78.1**	**14.5**	19.6	81.4	86.5	51.9	22.0	70.4	18.2	22.3	46.4	47.8
		Our Method	**87.1**	**38.3**	77.2	13.7	24.4	82.6	86.9	54.1	28.0	69.6	18.5	19.2	48.0	49.8
Taipei	Yes	Cross city [15]	78.6	28.6	80.0	13.1	7.6	68.2	82.1	16.8	9.4	60.4	34.0	26.5	9.9	39.6
		MaxSquare [12]	80.7	32.5	85.5	**32.7**	15.1	78.1	**91.3**	**32.9**	7.6	69.5	44.8	52.4	**34.9**	**50.6**
	No	Source	82.6	33.0	**86.3**	16.0	**16.5**	**78.3**	83.3	26.5	8.4	70.7	36.1	47.9	15.7	46.3
		Our Method	**86.4**	**34.6**	84.6	22.4	9.9	76.2	88.3	32.8	**15.1**	**74.8**	**45.8**	**53.3**	26.7	50.1

(c) Results of Cityscapes → Cross-City (*CS-CC*) experiments.

Table 1: For all the experiments in Tables 1a to 1c we compare our proposed method with methods that use source data for adaptation. We find that our method is comparable, and in some cases better than the methods that use the source data for adaptation. In underline, we compare our results to the source trained classifier, and with an **bold** the best performance over all methods. We omit the underline if our proposed method, or the source classifier out performs the methods that use the source data for adaptation.

3.4.1 Importance of loss terms

In order to correctly attribute performance to each of the terms in Equation (6), we ablate over the loss terms in Table 2. Broadly summarizing, we find that the first choice of using pseudo-labeling results in a substantial improvement of performance over the source classifier. With each term we see that the performance increases, however as described in Section 3.4.2, our method also suffers higher variance.

Loss function	$\mathcal{L}_{PL}$	$\mathcal{L}_{ent}$	$\mathcal{L}_{ent} + \mathcal{L}_{PL}$	$\mathcal{L}_{un} + \mathcal{L}_{PL}$	$\mathcal{L}_{DT}$	$\mathcal{L}$
	Eq 5	Eq 3	Eq 3 + Eq 5	Eq 2 + Eq 5	Eq 6	Eq 6
% mIoU	42.24	19.85	42.39	42.72	44.52	45.07

Table 2: Importance of each of the loss terms proposed. $\mathcal{L}_{DT}$ refers to training with loss in Equation (6) without freezing the decoders, and the last column $\mathcal{L}$ shows the performance on freezing the main decoder. We see that each of the loss terms gives a consistent improvement over the previous loss values.

Thus, for a new use case one might expect a small but consistent improvement with pseudo-labeling, and other loss terms are useful if the method can be tuned carefully. We show various additional ablations in Appendix D, and some qualitative results in Appendix E.

3.4.2 Variance analysis

In order to obtain a better estimate of performance, we run some of the baseline methods that we use in Table 1a with five random seeds and show the performance. We use the publicly available codes from the authors. The codes were executed for a maximum of 72 hours, a limitation imposed by our computation resources. For each method, we change only the random seed for each run, and leave the rest of the hyperparameters to the default values set by the authors in their codes. In Table 3, we show various statistics computed over the obtained runs. We see that the common trend in publications on UDA is to report the best obtained performance. While it is a pragmatic choice to use the best obtained model as benchmarked on a validation set, for deployments, it induces a systemic bias in the assessment of the true performance of the system. Thus, we believe that better characterization of the system's performance is through computing the average performance and the standard deviation, in addition to the best performance obtained.

In Table 3, we find that merely changing the random seed can have a noticeable effect on the performance of some of the standard systems, an observation made before [48, 3]. The discrepancy between the maximum results in Table 3 and the reported numbers can be attributed to hardware and software discrepancies, or budget used. Vu et al. [69] also remark that one needs to run the experiments a few times to reach comparable performance [1]. Keeping in line with the standards of the domain, we report the best obtained performance in Tables 1a to 1c, and show results of variance analysis in Tables 3 and 4. Examining the mean and standard deviation obtained for our *GTA-CS* and *CS-CC* experiments, we find that while our method achieves higher maximum performance, it has a higher variance compared to the methods that use source data for adaptation process. We hypothesize this is due to the unavailability of source data for adaptation; the proxy tasks used cannot act as suitable replacements for labeled data, in controlling and guiding the optimization process. We leave this analysis to future work.

Method	**Performance estimate**	**Min**	**Reproduced Results**	**Previously Reported**
AdaptSegnet [66]	39.68 ± 1.49	37.70	42.20	42.40
ADVENT [69] (90K)	41.57 ± 0.73	40.73	42.73	43.80
ADVENT [69] (Best)	42.56 ± 0.64	41.60	42.39	
CBST [81]	44.04 ± 0.88	42.80	45.03	45.90
Proposed Method	42.44 ± 2.18	39.71	45.06	–

Table 3: Variance of the methods examined for the GTA→ Cityscapes. We show the mean, standard deviation, minimum and reproduced (maximum) performance obtained over five runs with different random seeds, and the official reported metrics from the paper. For ADVENT we show two rows to indicate the two testing strategies in their code: The first one is after 90K iterations, and the second is the best attained performance. We see that the common strategy is to report the best obtained result.

	Rome	Rio	Tokyo	Taipei
Performance estimate	53.2± 0.8	52.37± 1.08	49.2± 0.71	49.48±0.76

Table 4: Variance of proposed method for the Cityscapes→NTHU Crosscity adaptation.

4. Related Work

Semantic Segmentation Deep learning had its success in semantic segmentation with fully convolutional networks [45], which converted the full-connected layers to convolutional layers. Following this, various networks that made several architectural changes to improve accuracy metrics [74, 79, 10], and computational requirements [78, 56, 59, 53] have been proposed. To remedy the data hungriness of these networks, domain adaptation, specifically unsupervised domain adaptation, has been an oft studied problem recently

Unsupervised Domain Adaptation Based on the seminal work of adversarial domain adaptation [26] that uses a discriminator network to align features from both domains, several methods have been proposed for segmentation on similar lines. Methods have been proposed that align intermediate feature spaces [33, 61, 22, 70], output space [66, 67].

Another family of methods use pseudo-labeling to enrich the target domain training using either hard labels [81, 18,

[1] https://github.com/valeoai/ADVENT#training

75], or soft labels [80]. Chang et al. [5] use a *per-domain* autoencoder to separate domain idiosyncrasies from features relevant for cross-domain segmentation. Vu et al. [69], Chen et al. [12] propose an entropy based objective, and an adversarial alignment of the entropy maps of the two domains.

A host of techniques include style translation as a part of their network that is trained along with the segmentation network [32, 43, 14, 50], or separately [72, 35].

Model Adaptation Most of the previously mentioned methods need the explicit availability of source domain data during adaptation too, and have made tremendous strides in improving the segmentation performance in that case. A few recent papers tackle model adaptation for classification problems [44, 42, 16]. [39] proposes source-free domain adaptation in the case where label knowledge of the target domain is not available, and show their efficiency on a set of classification problems with varying levels of label overlap. As we argued in Section 1, to the best of our knowledge, this problem has not been tackled in the context of semantic segmentation. Though the task of segmentation can be construed to be one that of pixel-wise prediction, we emphasize that the techniques cannot be interchangeably used. As the resource requirements of segmentation networks are substantially higher than that of classification networks, methods proposed classification that use label conditioned generation [42] of target domain data, or use memory intensive methods for reliable estimation of pseudo-labels [36, 44] are impractical.

Self-supervision for segmentation Self-supervised learning exploits the structure in data by defining a *pretext task*, so that the network learns a good semantic representation of the input image. Examples include rotation prediction [27], context prediction [21], jig-saw puzzle solving [52],contrastive learning[13, 30]. Self-supervised learning for unsupervised domain adaptation has been proposed to train the feature extractor [64]. However, the applications of self-supervised learning to segmentation have been limited, as the invariances enforced differ widely between classification and segmentation problems. Some attempts to learn segmentation networks include using very strong perturbation techniques like CutMix [24], using consistency regularization through feature noising [54], by using clustering for pseudo-labels [40]. As described in Section 2.2, we note similarities of our method to tasks proposed in [54].

Multi-task learning and robust classification The use of multiple tasks to enrich the representations learned by a network has been used quite often in computer vision [77, 4]. These ideas have been adapted to single task learning by devising *pseudo-tasks* [49]. These pseudo-tasks train the shared network structure to learn the task in multiple ways, and can be interpreted as using self-supervision. Similarly, increasing the robustness of classification through feature noising has been studied extensively [28, 8, 68, 16, 47]. Work on learning robust networks through pseudo-ensembling by reducing the variance when dropout is used, has been proposed [2]. Ideas of large margin learning have been extended to deep learning [23].

Uncertainty modeling for neural networks Our proposed method in Section 2.2 is very similar to the uncertainty modeling for neural networks using Monte-Carlo (MC) dropout. MC-dropout [34, 25] uses dropout at test-time to sample various outputs and average them for predicted posterior. Bayesian treatments of neural networks have been important owing to their ability to predict uncertainty better, and dealing with mis-calibration problems [38]. Such modeling has been used for medical imaging to estimate the confidence of lesion segmentation [51], for autonomous driving to estimate the uncertainty of steering wheel angle prediction [46].

5. Conclusion

In this paper, we focus on the problem of domain adaptation for semantic segmentation in the absence of source data. In the absence of any labels to guide the optimization, we propose a method that reduces the uncertainty of predictions on the target domain data, that can also be interpreted as increasing the stability of the feature extractor. On the standard benchmark of tasks for semantic segmentation transfer, we obtain performance comparable with that of methods that use source data. We hope that our work gives the required fillip to the community to focus on this newer, practically significant, and challenging form of domain adaptation.

Acknowledgments

The research leading to these results was supported by the “Swiss Center for Drones and Robotics - SCDR” of the Department of Defence, Civil Protection and Sport via armasuisse S+T under project n^{o}050-38.

References

[1] Eric Arazo, Diego Ortego, Paul Albert, Noel E O'Connor, and Kevin McGuinness. Pseudo-labeling and confirmation bias in deep semi-supervised learning. In *2020 International Joint Conference on Neural Networks (IJCNN)*. IEEE, 2020. 4

[2] Philip Bachman, Ouais Alsharif, and Doina Precup. Learning with pseudo-ensembles. In *Advances in neural information processing systems*, pages 3365–3373, 2014. 8

[3] Yoshua Bengio. Practical recommendations for gradient-based training of deep architectures. In *Neural networks: Tricks of the trade*, pages 437–478. Springer, 2012. 7

[4] Hakan Bilen and Andrea Vedaldi. Integrated perception with recurrent multi-task neural networks. In *Advances in neural information processing systems*, pages 235–243, 2016. 8

[5] Wei-Lun Chang, Hui-Po Wang, Wen-Hsiao Peng, and Wei-Chen Chiu. All about structure: Adapting structural information across domains for boosting semantic segmentation. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1900–1909, 2019. 8

[6] Olivier Chapelle, Bernhard Schlkopf, and Alexander Zien. *Semi-Supervised Learning*. The MIT Press, 1st edition, 2010. 2

[7] Olivier Chapelle and Alexander Zien. Semi-supervised classification by low density separation. In *AISTATS*, volume 2005, pages 57–64. Citeseer, 2005. 2

[8] Gal Chechik, Geremy Heitz, Gal Elidan, Pieter Abbeel, and Daphne Koller. Max-margin classification of data with absent features. *Journal of Machine Learning Research*, 9(Jan):1–21, 2008. 8

[9] Liang-Chieh Chen, Yukun Zhu, George Papandreou, Florian Schroff, and Hartwig Adam. Encoder-decoder with atrous separable convolution for semantic image segmentation. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 833–851, 2018. 1

[10] 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 Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, 40:834–848, 2018. 1, 2, 5, 7

[11] Minmin Chen, Kilian Weinberger, Fei Sha, and Yoshua Bengio. Marginalized denoising auto-encoders for nonlinear representations. In *International Conference on Machine Learning*, pages 1476–1484, 2014. 3

[12] Minghao Chen, Hongyang Xue, and Deng Cai. Domain adaptation for semantic segmentation with maximum squares loss. In *Proc. of International Conference on Computer Vision (ICCV)*, 2019. 5, 6, 8, 3

[13] Ting Chen, Simon Kornblith, Mohammad Norouzi, and Geoffrey Hinton. A simple framework for contrastive learning of visual representations. *arXiv preprint arXiv:2002.05709*, 2020. 3, 8

[14] Yun-Chun Chen, Yen-Yu Lin, Ming-Hsuan Yang, and Jia-Bin Huang. Crdoco: Pixel-level domain transfer with cross-domain consistency. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 8

[15] 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 *Proceedings of the IEEE International Conference on Computer Vision*, pages 1992–2001, 2017. 1, 4, 6

[16] Boris Chidlovskii, Stephane Clinchant, and Gabriela Csurka. Domain adaptation in the absence of source domain data. In *Proceedings of the 22nd ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 451–460, 2016. 1, 2, 8

[17] 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 *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3213–3223, 2016. 1, 4

[18] 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 (IJCV)*, pages 1–23, 2019. 7

[19] Dengxin Dai and Luc Van Gool. Dark model adaptation: Semantic image segmentation from daytime to nighttime. In *2018 21st International Conference on Intelligent Transportation Systems (ITSC)*, pages 3819–3824. IEEE, 2018. 1

[20] Terrance DeVries and Graham W Taylor. Improved regularization of convolutional neural networks with cutout. *arXiv preprint arXiv:1708.04552*, 2017. 3

[21] Carl Doersch, Abhinav Gupta, and Alexei A Efros. Unsupervised visual representation learning by context prediction. In *Proceedings of the IEEE international conference on computer vision*, pages 1422–1430, 2015. 8

[22] Liang Du, Jingang Tan, Hongye Yang, Jianfeng Feng, Xiangyang Xue, Qibao Zheng, Xiaoqing Ye, and Xiaolin Zhang. SSF-DAN: separated semantic feature based domain adaptation network for semantic segmentation. In *Proc. of International Conference on Computer Vision (ICCV)*, 2019. 7

[23] Gamaleldin Elsayed, Dilip Krishnan, Hossein Mobahi, Kevin Regan, and Samy Bengio. Large margin deep networks for classification. In *Advances in neural information processing systems*, pages 842–852, 2018. 8

[24] Geoff French, Timo Aila, Samuli Laine, Michal Mackiewicz, and Graham Finlayson. Semi-supervised semantic segmentation needs strong, high-dimensional perturbations. *arXiv preprint arXiv:1906.01916*, 2019. 8

[25] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: Representing model uncertainty in deep learning. In *international conference on machine learning*, pages 1050–1059, 2016. 8, 3

[26] 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 (JMLR)*, 17(1):2096–2030, 2016. 7

[27] Spyros Gidaris, Praveer Singh, and Nikos Komodakis. Unsupervised representation learning by predicting image rotations. In *International Conference on Learning Representations*, 2018. 8

[28] Amir Globerson and Sam Roweis. Nightmare at test time: robust learning by feature deletion. In *Proceedings of the 23rd international conference on Machine learning*, pages 353–360, 2006. 3, 8

[29] Yves Grandvalet and Yoshua Bengio. Semi-supervised learning by entropy minimization. In *Actes de CAP 05, Conférence francophone sur l'apprentissage automatique*, pages 281–296, 2005. 2, 3

[30] Kaiming He, Haoqi Fan, Yuxin Wu, Saining Xie, and Ross Girshick. Momentum contrast for unsupervised visual representation learning. In *Proceedings of the IEEE/CVF Con-*

ference on Computer Vision and Pattern Recognition, pages 9729–9738, 2020. 8

[31] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 770–778, 2016. 5

[32] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *Proc. of the International Conference on Machine Learning (ICML)*, 2018. 8

[33] 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. 7

[34] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision? In *Advances in neural information processing systems*, pages 5574–5584, 2017. 8

[35] Myeongjin Kim and Hyeran Byun. Learning texture invariant representation for domain adaptation of semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12975–12984, 2020. 5, 8

[36] Youngeun Kim, Sungeun Hong, Donghyeon Cho, Hyoungseob Park, and Priyadarshini Panda. Domain adaptation without source data. *arXiv preprint arXiv:2007.01524*, 2020. 8

[37] Andreas Krause, Pietro Perona, and Ryan G Gomes. Discriminative clustering by regularized information maximization. In *Advances in neural information processing systems*, pages 775–783, 2010. 4

[38] Agustinus Kristiadi, Matthias Hein, and Philipp Hennig. Being bayesian, even just a bit, fixes overconfidence in relu networks. *arXiv preprint arXiv:2002.10118*, 2020. 4, 8, 1

[39] Jogendra Nath Kundu, Naveen Venkat, R Venkatesh Babu, et al. Universal source-free domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4544–4553, 2020. 8

[40] Mans Larsson, Erik Stenborg, Carl Toft, Lars Hammarstrand, Torsten Sattler, and Fredrik Kahl. Fine-grained segmentation networks: Self-supervised segmentation for improved long-term visual localization. In *Proceedings of the IEEE International Conference on Computer Vision*, pages 31–41, 2019. 8

[41] Dong-Hyun Lee. Pseudo-label: The simple and efficient semi-supervised learning method for deep neural networks. In *Workshop on challenges in representation learning, ICML*, volume 3, 2013. 2, 4

[42] Rui Li, Qianfen Jiao, Wenming Cao, Hau-San Wong, and Si Wu. Model adaptation: Unsupervised domain adaptation without source data. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9641–9650, 2020. 8

[43] Yunsheng Li, Lu Yuan, and Nuno Vasconcelos. Bidirectional learning for domain adaptation of semantic segmentation. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2019. 8

[44] Jian Liang, Dapeng Hu, and Jiashi Feng. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In *International Conference on Machine Learning (ICML)*, pages xx–xx, July 2020. 4, 8

[45] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3431–3440, 2015. 7

[46] Antonio Loquercio, Mattia Segu, and Davide Scaramuzza. A general framework for uncertainty estimation in deep learning. *IEEE Robotics and Automation Letters*, 5(2):3153–3160, 2020. 8

[47] Laurens Maaten, Minmin Chen, Stephen Tyree, and Kilian Weinberger. Learning with marginalized corrupted features. In *International Conference on Machine Learning*, pages 410–418, 2013. 2, 8

[48] Pranava Madhyastha and Rishabh Jain. On model stability as a function of random seed. In *Proceedings of the 23rd Conference on Computational Natural Language Learning (CoNLL)*, pages 929–939, Hong Kong, China, Nov. 2019. Association for Computational Linguistics. 7

[49] Elliot Meyerson and Risto Miikkulainen. Pseudo-task augmentation: From deep multitask learning to intratask sharing—and back. In *International Conference on Machine Learning*, pages 3511–3520, 2018. 3, 4, 8

[50] Zak Murez, Soheil Kolouri, David J. Kriegman, Ravi Ramamoorthi, and Kyungnam Kim. Image to image translation for domain adaptation. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, 2018. 8

[51] Tanya Nair, Doina Precup, Douglas L Arnold, and Tal Arbel. Exploring uncertainty measures in deep networks for multiple sclerosis lesion detection and segmentation. *Medical image analysis*, 59:101557, 2020. 8

[52] Mehdi Noroozi and Paolo Favaro. Unsupervised learning of visual representations by solving jigsaw puzzles. In *European Conference on Computer Vision*, pages 69–84. Springer, 2016. 8

[53] Marin Orsic, Ivan Kreso, Petra Bevandic, and Sinisa Segvic. In defense of pre-trained imagenet architectures for real-time semantic segmentation of road-driving images. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 12607–12616, 2019. 7

[54] Yassine Ouali, Céline Hudelot, and Myriam Tami. Semi-supervised semantic segmentation with cross-consistency training. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12674–12684, 2020. 2, 3, 8

[55] Yaniv Ovadia, Emily Fertig, Jie Ren, Zachary Nado, David Sculley, Sebastian Nowozin, Joshua Dillon, Balaji Lakshminarayanan, and Jasper Snoek. Can you trust your model's uncertainty? evaluating predictive uncertainty under dataset shift. In *Advances in Neural Information Processing Systems*, pages 13991–14002, 2019. 3

[56] Adam Paszke, Abhishek Chaurasia, Sangpil Kim, and Eugenio Culurciello. Enet: A deep neural network architecture for real-time semantic segmentation. *arXiv preprint arXiv:1606.02147*, 2016. 7

[57] Benjamin Recht, Rebecca Roelofs, Ludwig Schmidt, and Vaishaal Shankar. Do ImageNet classifiers generalize to ImageNet? volume 97 of *Proceedings of Machine Learning Research*, pages 5389–5400, Long Beach, California, USA,

09–15 Jun 2019. PMLR. 1

[58] Stephan R. Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, *Proc. of European Conference on Computer Vision (ECCV)*, volume 9906, pages 102–118. Springer International Publishing, 2016. 1, 4

[59] Eduardo Romera, José M Alvarez, Luis M Bergasa, and Roberto Arroyo. Erfnet: Efficient residual factorized convnet for real-time semantic segmentation. *IEEE Transactions on Intelligent Transportation Systems*, 19(1):263–272, 2017. 7

[60] 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 *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3234–3243, 2016. 1, 4

[61] Swami Sankaranarayanan, Yogesh Balaji, Arpit Jain, Ser Nam Lim, and Rama Chellappa. Learning from synthetic data: Addressing domain shift for semantic segmentation. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 3752–3761, 2018. 7

[62] Robert E Schapire, Yoav Freund, Peter Bartlett, Wee Sun Lee, et al. Boosting the margin: A new explanation for the effectiveness of voting methods. *The annals of statistics*, 26(5):1651–1686, 1998. 2

[63] Daniel L Silver, Qiang Yang, and Lianghao Li. Lifelong machine learning systems: Beyond learning algorithms. In *2013 AAAI spring symposium series*, 2013. 1

[64] Yu Sun, Eric Tzeng, Trevor Darrell, and Alexei A Efros. Unsupervised domain adaptation through self-supervision. *arXiv preprint arXiv:1909.11825*, 2019. 4, 8

[65] Yu Sun, Xiaolong Wang, Liu Zhuang, John Miller, Moritz Hardt, and Alexei A. Efros. Test-time training with self-supervision for generalization under distribution shifts. In *ICML*, 2020. 4

[66] 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 *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 7472–7481, 2018. 6, 7

[67] Yi-Hsuan Tsai, Kihyuk Sohn, Samuel Schulter, and Manmohan Chandraker. Domain adaptation for structured output via discriminative patch representations. In *Proc. of International Conference on Computer Vision (ICCV)*, pages 1456–1465, 2019. 7

[68] Pascal Vincent, Hugo Larochelle, Isabelle Lajoie, Yoshua Bengio, Pierre-Antoine Manzagol, and Léon Bottou. Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. *Journal of machine learning research*, 11(12), 2010. 8

[69] Tuan-Hung Vu, Himalaya Jain, Maxime Bucher, Matthieu Cord, and Patrick Pérez. Advent: Adversarial entropy minimization for domain adaptation in semantic segmentation. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2517–2526, 2019. 2, 6, 7, 8, 3

[70] Haoran Wang, Tong Shen, Wei Zhang, Lingyu Duan, and Tao Mei. Classes matter: A fine-grained adversarial approach to cross-domain semantic segmentation. *arXiv preprint arXiv:2007.09222*, 2020. 7

[71] Qizhe Xie, Eduard Hovy, Minh-Thang Luong, and Quoc V Le. Self-training with noisy student improves imagenet classification. *arXiv preprint arXiv:1911.04252*, 2019. 4

[72] Yanchao Yang and Stefano Soatto. FDA: fourier domain adaptation for semantic segmentation. *arXiv preprint arXiv:2004.05498*, 2020. 5, 8

[73] Fisher Yu, Haofeng Chen, Xin Wang, Wenqi Xian, Yingying Chen, Fangchen Liu, Vashisht Madhavan, and Trevor Darrell. Bdd100k: A diverse driving dataset for heterogeneous multitask learning. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 2636–2645, 2020. 1

[74] Fisher Yu, Vladlen Koltun, and Thomas A. Funkhouser. Dilated residual networks. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 636–644, 2017. 7

[75] Yang Zhang, Philip David, Hassan Foroosh, and Boqing Gong. A curriculum domain adaptation approach to the semantic segmentation of urban scenes. *IEEE Trans. on Pattern Analysis and Machine Intelligence (TPAMI)*, 2019. 8

[76] Yiheng Zhang, Zhaofan Qiu, Ting Yao, Dong Liu, and Tao Mei. Fully convolutional adaptation networks for semantic segmentation. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 6810–6818, 2018. 6

[77] Zhanpeng Zhang, Ping Luo, Chen Change Loy, and Xiaoou Tang. Facial landmark detection by deep multi-task learning. In *European conference on computer vision*, pages 94–108. Springer, 2014. 8

[78] Hengshuang Zhao, Xiaojuan Qi, Xiaoyong Shen, Jianping Shi, and Jiaya Jia. Icnet for real-time semantic segmentation on high-resolution images. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 405–420, 2018. 7

[79] Hengshuang Zhao, Jianping Shi, Xiaojuan Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Proc. of IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2881–2890, 2017. 7

[80] Yang Zou, Zhiding Yu, Xiaofeng Liu, B.V.K. Vijaya Kumar, and Jinsong Wang. Confidence regularized self-training. In *Proc. of International Conference on Computer Vision (ICCV)*, pages 5982–5991, 2019. 5, 6, 8

[81] Yang Zou, Zhiding Yu, BVK Vijaya Kumar, and Jinsong Wang. Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In *Proc. of European Conference on Computer Vision (ECCV)*, pages 289–305, 2018. 4, 6, 7

Uncertainty Reduction for Model Adaptation in Semantic Segmentation

Appendix

Prabhu Teja S
Idiap Research Institute & EPFL
prabhu.teja@idiap.ch

François Fleuret
University of Geneva & Idiap Research Institute
francois.fleuret@unige.ch

A. Toy example's network architecture

For the toy experiment presented in Section 3.1, we use a network from Kristiadi et al. [38]. The architectural details of the network are given in Table 5.

Table 5: Architecture of the network used for the toy experiment in Figure 3

Layer name	**Description**
Feature Extractor	Linear 2×20 BatchNorm ReLU Linear 20×2 ReLU
Classifier	Softmax(2)

B. Entropy measurements

In Figure 1 and in Section 2.1, we glossed over the details of the toy-problem. We provide the details here.

Referring to Figure 1, we are trying to reason scenarios that, in the absence of labeled target data, can be expected to result in good target performance. Our argument is that reducing the uncertainty of predictions on the target domain is the an effective strategy to improve performance on the target domain. In order to do so, we concoct toy scenarios, and analyse their uncertainties. For this illustration, we use entropy as a measure of the uncertainty.

Let us consider a two-class classification problem as shown in Figure 1. Let X be the feature random variable, and $Y \in \{0, 1\}$ be the labels. Let us assume that the class conditional distributions be normally distributed *i.e.* $X|Y = k \sim \mathcal{N}(\mu_k, \sigma^2)$. Thus $\mu_X(x) = \frac{1}{2}\left(\mathcal{N}(x; \mu_0, \sigma^2) + \mathcal{N}(x; \mu_1, \sigma^2)\right)$, assuming uniform prior on Y. Let the threshold random variable T with a density $\mu_T(t)$. This distribution is determined by a learning algorithm. Let $\hat{Y}$ be the random variable denoting the predictions whose probability is computed using a sigmoid on the feature and a threshold as

$$\rho(x; t) \equiv p(\hat{Y} = 1 | X = x; T = t) = \frac{1}{1 + e^{-(x-t)}}. \tag{7}$$

The entropy of the above defined categorical distribution is given by

$$\mathbb{H}(\hat{Y} | X = x, T = t) = -\left(\rho(x; t) \log(\rho(x; t)) + (1 - \rho(x; t)) \log(1 - \rho(x; t))\right) \tag{8}$$

The entropy of a classification over the entire domain X can be computed as

$$\mathbb{H}(\hat{Y} | T = t) = \int -\left(\rho(x; t) \log(\rho(x; t)) + (1 - \rho(x; t)) \log(1 - \rho(x; t))\right) \mu_X(x) dx \tag{9}$$

The above defined marginal entropy is defined for a specific choice of the threshold t. To see the how the feature distribution itself influences the generalization, the entropy over all possible thresholds is computed. The *total entropy* over all the thresholds possible is computed by integrating over the entire range of t.

$$\mathbb{H}(\hat{Y}) = \int\int -\left(\rho(x;t)\log(\rho(x;t)) + (1-\rho(x;t))\log(1-\rho(x;t))\right)\mu_X(x)\mu_T(t)dxdt \tag{10}$$

We emphasize that t is the set of thresholds that can be generated by a learning algorithm. However, we simplify it to using the domain X for this discussion.

We would like to train networks that result in low overall entropy in Equation (10). This coincides with out argument that the features that can be separated by several choices of threshold are likelier to generalize better. To visualize this, we run a few simulations that compute the marginal and total entropy for various settings of feature distributions. In Figure 4, we show the data distribution $\mu_X(x)$ for various μ_0, μ_1 values. Variance σ^2 is fixed to 1. In orange, we show marginalized entropy (Equation (9)). We scale it by a constant for plotting purposes.

It is very apparent that the higher is $|\mu_1 - \mu_2|$, lower the overall entropy. Given that the two Gaussians are the class conditional distributions, the Bayes optimal decision boundary, that can be computed to be $\frac{\mu_1+\mu_2}{2}$, coincides with the point where Equation (9) is minimized, for all scenarios except for the one with high overlap in Figure 4(a). Extending it, the lower is the Bayes risk, better is the generalization. Thus the least entropy separation is also the best separator we can achieve. We skip an important detail here: by extending the threshold search to a value that is on the extremities of the X-axis, we can get a threshold that results in an overall entropy that is very low. However, such a separator is practically useless. In our method, we avoid this by using the pseudo-labeling and the initializing the network with the source trained weights.

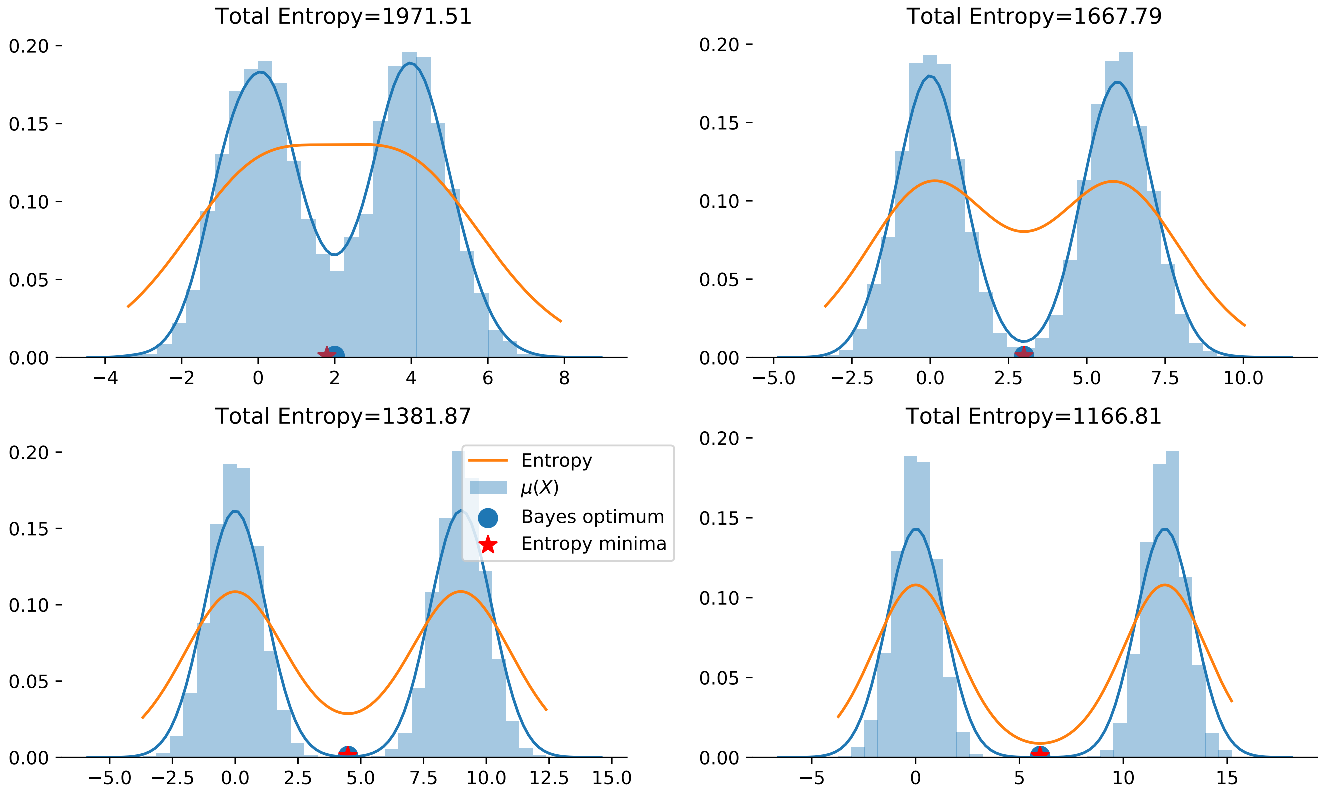


Figure 4: Illustration of the effect of the separation of the Gaussians on the entropy of the classification.

We can see that for this simple scenario that minimizing the entropy is a fairly good strategy in the absence of labels to find near optimal decision boundary.

C. Using squared error instead of entropy loss in Equation (2)

In Section 2.2, we proposed the uncertainty loss, and we use a squared error form instead of the standard entropy based uncertainty quantification. Here we provide a plausible explanation that follows the argument in Chen et al. [12]. Experimental evidence is provided in [54].

The traditional entropy regularizer has been used extensively in various applications like semi-supervised learning [29], domain adaptation [69]. However, the gradient of the entropy penalty with respect to the softmax output is not well behaved for probability is close to 1. We refer the reader to [12, Figure 1] for an illustration of this. However, squared loss and entropy loss can be viewed as a special case of f-divergence.

Let P and Q be two distributions with pdfs p and q their density functions. Then their f-divergence is defined as

$$D_f(P||Q) = \int_{\mathbb{R}} f\left(\frac{p(x)}{q(x)}\right) q(x)dx \tag{11}$$

We get KL-divergence by using $f(t) = t\log t$. Instead, using $f(t) = (t-1)^2$ gives the Max-squared loss formulation in [12]. They find that using this instead of $f(t) = t\log t$ results in better behavior for optimization. We find a similar empirical result that using the squared error form for Equation (2) results in better results than using posterior entropy.

D. Experimental ablations

D.1. Sensitivity to auxiliary decoders

In the numbers reported in Tables 1a and 1b, we use a dropout ratio of $p = 0.5$. We show an ablation on the dropout values in Table 6. Intuitively, this value indicates the level of noise resiliency that we expect in the network; a too low a value has very little use as it is equivalent to having a single decoder without dropout, and too high a value destroys too much information fed to the decoders. Thus an intermediate value like 0.5 is likely to be more appropriate for our use. We find that our experiments support this notion in Table 6. For this comparison, we use 5 auxiliary decoders. However, our proposed method is quite robust to this choice.

Dropout (p)	0.2	0.5	0.8
Performance (mIoU) %	44.44	45.07	44.14

Table 6: Optimal feature dropout proportions for GTA → Cityscapes experiment.

D.2. Number of auxiliary decoders

The auxiliary decoders play an important role in the level of feature stability induced. Indeed Gal and Ghahramani [25] show that larger the number of samples drawn from the dropout distribution, better is the approximation. In Table 7, we see that the number of decoders plays an important role, but the method is fairly stable in its performance over a range of decoder count.

# Decoders	1	3	4	5
Performance (mIoU)	43.5	44.20	44.67	45.07

Table 7: Performance variation to the number of auxiliary decoders for GTA → Cityscapes experiment.

D.3. Using single dropout decoder

In our method we propose the use of multiple decoders instead of one, and we show the effect of using a single decoder to decode the noisy features in Table 8. While it conceptually seems that using a single decoder should suffice, we see that using multiple auxiliary decoders is helpful. Using multiple decoders forces the feature extractor to be more robust, whereas using a single decoder forces the decoder to be more stable. However, the ASPP decoder is one layer deep, its representation capacity is limited and thus it is unable to do so, as evidenced in Table 8.

Method	mIoU %
Single decoder with dropout	43.29
Multiple auxiliary decoders	45.07

Table 8: Utility of using multiple decoders

E. Qualitative results for Cityscapes to Cross City Adaptation

In Figures 5 to 8 we show some qualitative improvements from our results for the *CS-CC* experiments.

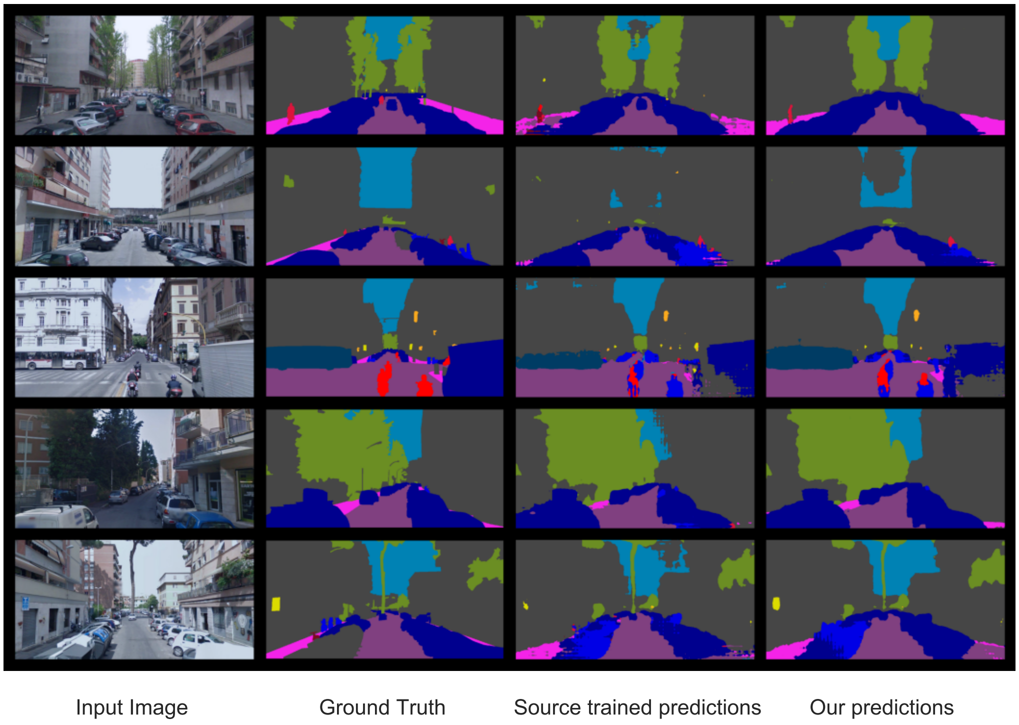


Figure 5: Best five case results of adaptation for the Cityscapes to Rome (Cross City). The first column is the test image, the second column is the ground truth, the third and fourth columns are the source trained model, and results of our proposed method.

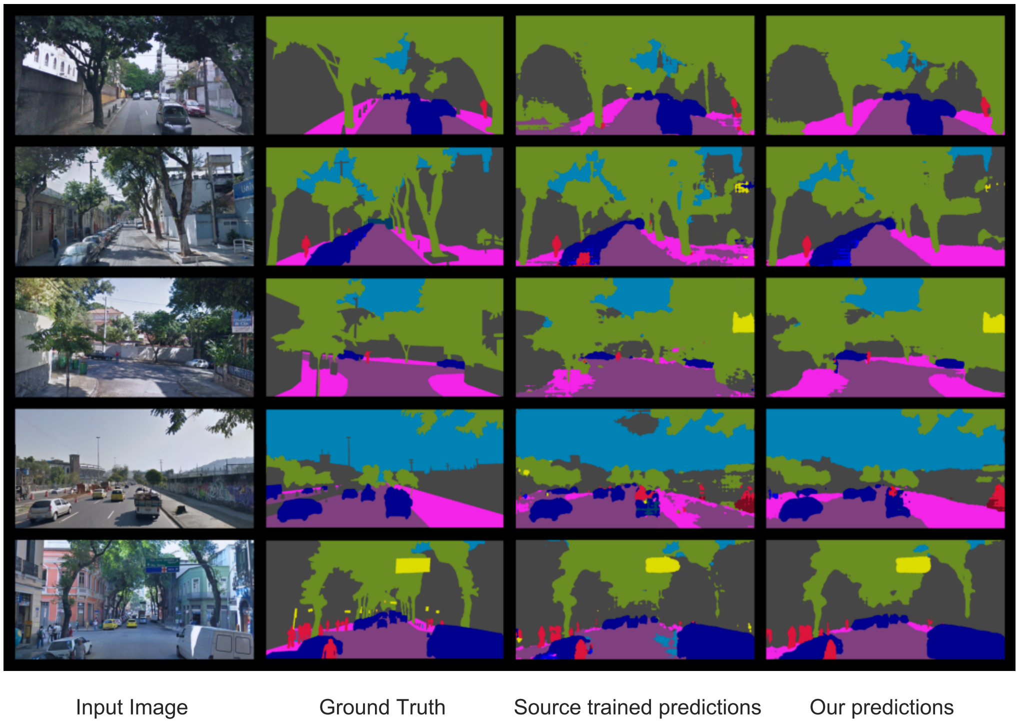


Figure 6: Best five results of adaptation for the Cityscapes to Rio (Cross City). The first column is the test image, the second column is the ground truth, the third and fourth columns are the source trained model, and results of our proposed method.

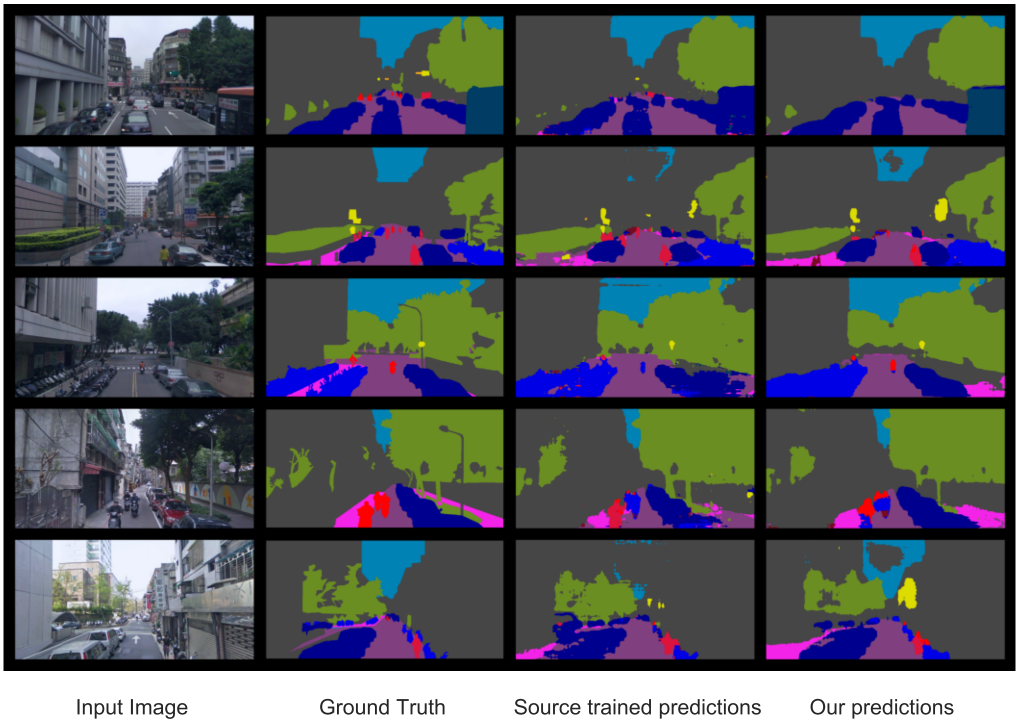


Figure 7: Best five results of adaptation for the Cityscapes to Taipei (Cross City). The first column is the test image, the second column is the ground truth, the third and fourth columns are the source trained model, and results of our proposed method.

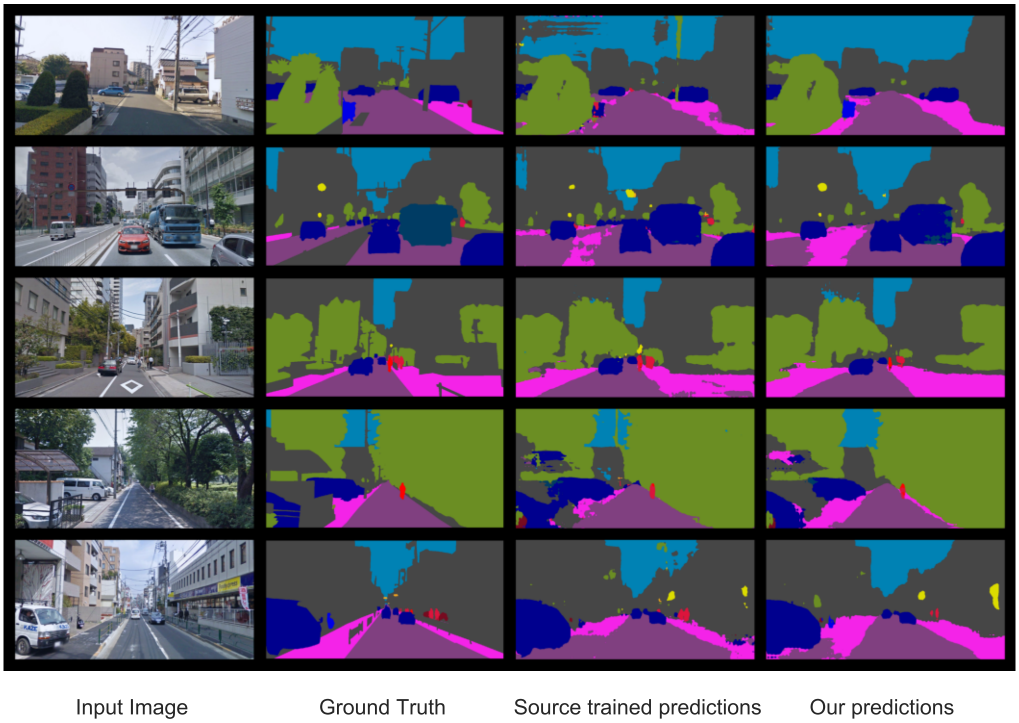


Figure 8: Best five results of adaptation for the Cityscapes to Tokyo (Cross City). The first column is the test image, the second column is the ground truth, the third and fourth columns are the source trained model, and results of our proposed method.

E.1. Failure cases

In Figures 9 to 12, we show the five images that have least improved over the adaptation process.

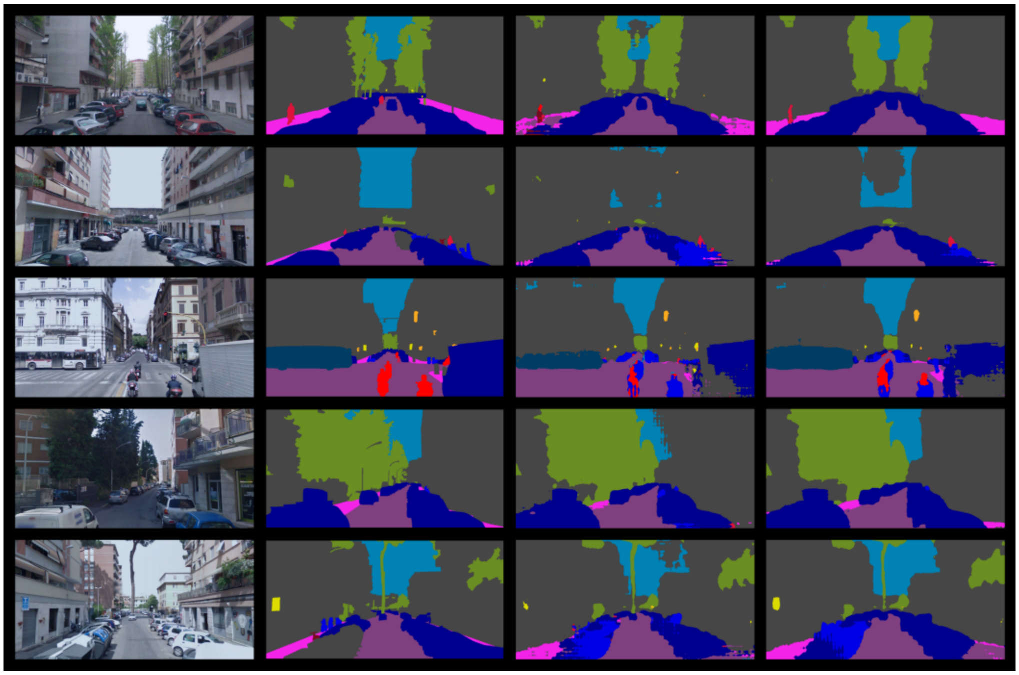


Figure 9: Worst five results of adaptation for the Cityscapes to Rome (Cross City). The first column is the test image, the second column is the ground truth, the third and fourth columns are the source trained model, and results of our proposed method.

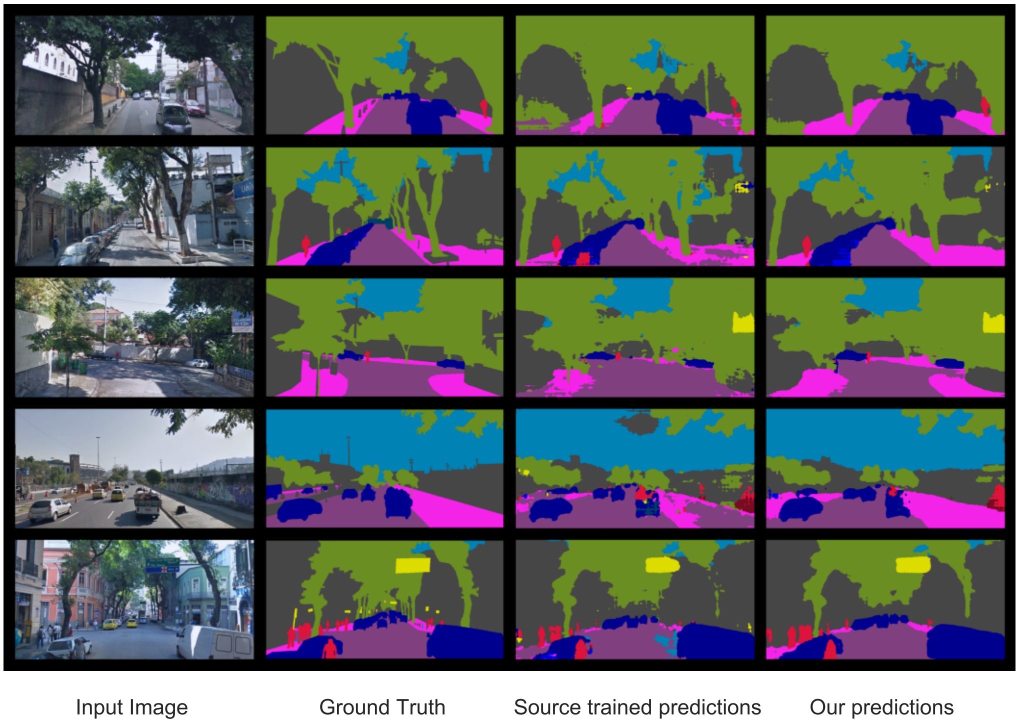


Figure 10: Worst five results adaptation for the Cityscapes to Rio (Cross City). The first column is the test image, the second column is the ground truth, the third and fourth columns are the source trained model, and results of our proposed method.

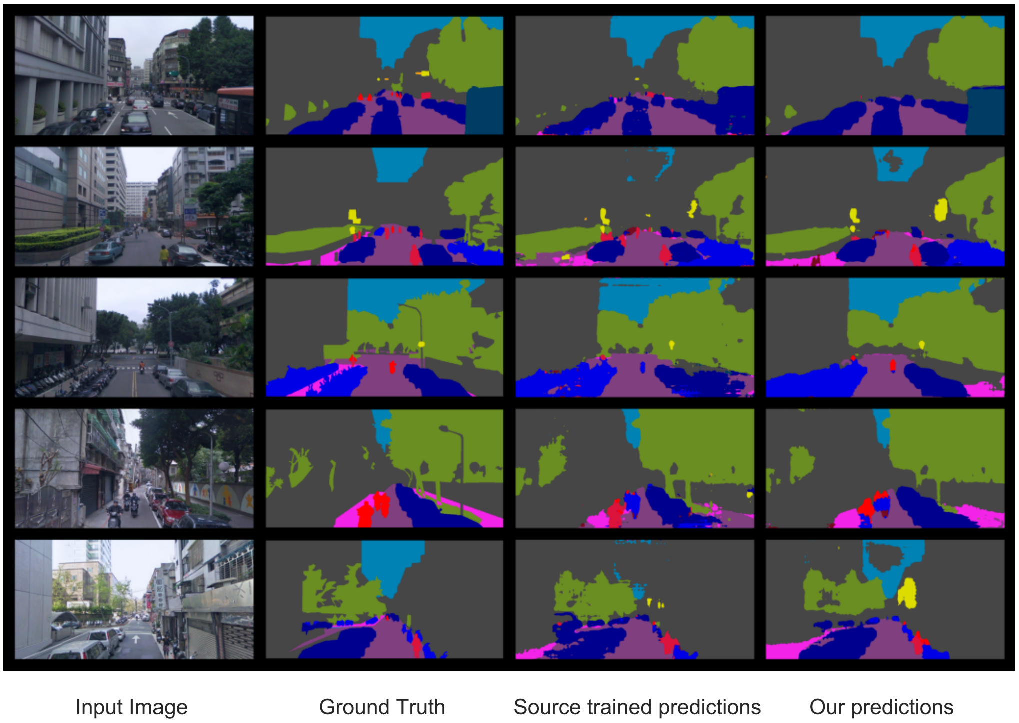


Figure 11: Worst five results adaptation for the Cityscapes to Taipei (Cross City). The first column is the test image, the second column is the ground truth, the third and fourth columns are the source trained model, and results of our proposed method.

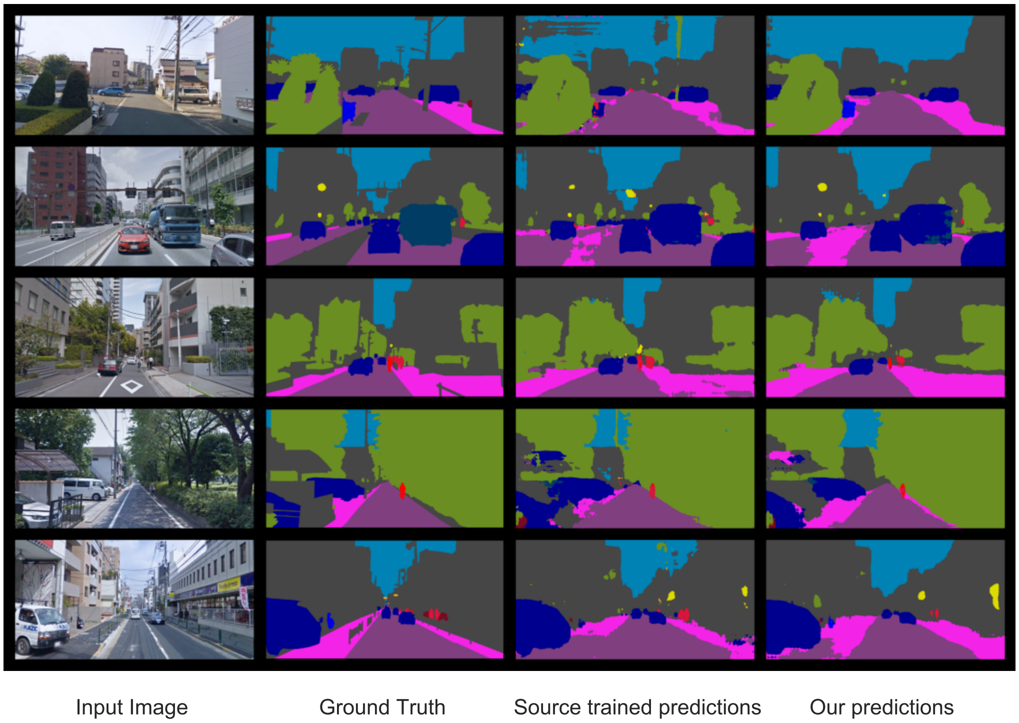


Figure 12: Worst five results adaptation for the Cityscapes to Tokyo (Cross City). The first column is the test image, the second column is the ground truth, the third and fourth columns are the source trained model, and results of our proposed method.