Semantically Adaptive Image-to-image Translation for Domain Adaptation of Semantic Segmentation (Supplementary Material)

Luigi Musto
luigi.musto@studenti.unipr.it
Andrea Zinelli
andrea.zinelli1@studenti.unipr.it
University of Parma
Parma, IT

1 Representation of the semantic input

The image synthesis network of SPADE takes as input a *one-hot* encoding of the ground truth semantic segmentation. Here, instead, we use the unnormalized output of M for every translation that we perform. This is a consequence of the cycle consistency constraints.

As explained in the main article, we have to perform both the $S \rightleftarrows T$ and $T \rightleftarrows S$ cycles, which is why we have to train both G_S and G_T by feeding them semantic maps aligned with the input images. In UDA problems, we do not have access to Y_T, which is why we use $M(X_T)$ for the $T \rightleftarrows S$ cycle.

However, we note that the refined output classes predicted by M are far from the ground truth and cannot give an accurate conditioning, especially in the target domain when the segmentation is still in the initial training phases. Because of this, we choose to use as semantic guidance the unnormalized output of M. This representation has the advantage of carrying the confidence of the prediction, which could potentially be used by the SPADE layers to avoid denormalizing a region with the incorrect class (*e.g.* on the borders of objects, where the segmentation tends to fail more easily).

In the $S \rightleftarrows T$ cycle, we could use Y_S as semantic guidance, but this would lead to inconsistent input distributions for the SPADE layers, which is why we adopt $M(X_S)$ as semantic guidance in this case too.

© 2020. The copyright of this document resides with its authors.
It may be distributed unchanged freely in print or electronic forms.

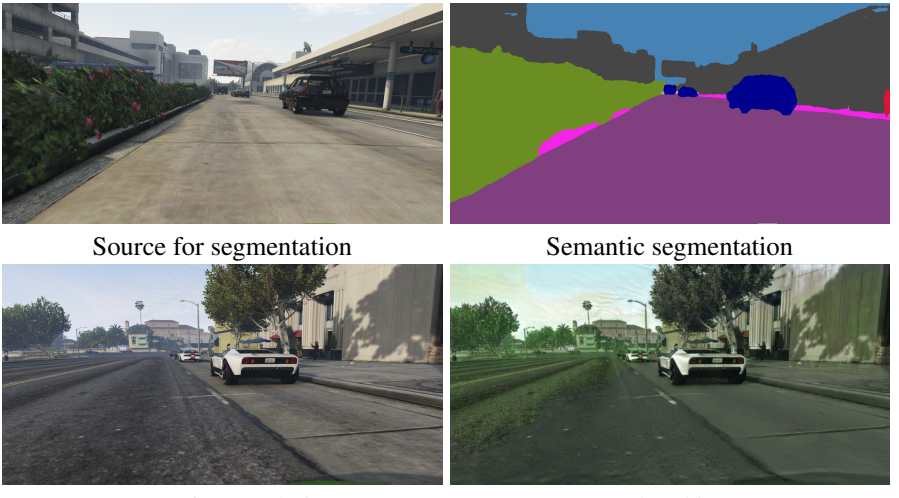
2 Detailed architecture

Encoder								
Kernel size	Stride	Input channels	Output channels	Output upsampling	Residual	Activation function	Normalization	Spectral normalization
7	1	3	64	-	-	ReLU	IN	✓
4	2	64	128	-	-	ReLU	IN	✓
4	2	128	256	-	-	ReLU	IN	✓
3	1	256	256	-	✓	ReLU	IN	✓
3	1	256	256	-	✓	ReLU	IN	✓
3	1	256	256	-	✓	ReLU	IN	✓
3	1	256	256	-	✓	ReLU	IN	✓
Generator								
Kernel size	Stride	Input channels	Output channels	Output upsampling	Residual	Activation function	Normalization	Spectral normalization
3	1	256	256	-	✓	ReLU	IN+SPADE	✓
3	1	256	256	-	✓	ReLU	IN+SPADE	✓
3	1	256	256	-	✓	ReLU	IN+SPADE	✓
3	1	256	256	✓	✓	ReLU	IN+SPADE	✓
5	1	256	128	✓	-	ReLU	LN	✓
5	1	128	64	-	-	ReLU	LN	✓
7	1	64	3	-	-	Tanh	-	✓
Discriminator (x3)								
Kernel size	Stride	Input channels	Output channels	Output upsampling	Residual	Activation function	Normalization	Spectral normalization
4	2	3	64	-	-	$\text{LReLU}_{0.2}$	-	✓
4	2	64	128	-	-	$\text{LReLU}_{0.2}$	-	✓
4	2	128	256	-	-	$\text{LReLU}_{0.2}$	-	✓
4	2	256	512	-	-	$\text{LReLU}_{0.2}$	-	✓
1	1	512	1	-	-	-	-	✓

Table 1: **Detailed architecture of encoders, generators and discriminators in the image-to-image translation step.** The architectures follow the schemes adopted by CycleGAN and UNIT. *Output upsampling* indicates that we use a $2\times$ nearest-neighbor upsampling of the output feature maps. *Residual* indicates that the layer is actually a residual block, not a simple convolutional one. $\text{LReLU}_{0.2}$ indicates the Leaky Rectified Linear Unit with slope $\alpha = 0.2$.

3 Fake segmentation

The effect of using the SPADE layers in the image-to-image translation model can be seen better when there is a mismatch between the source image and the semantic guidance. To show this effect, we feed the SPADE layers with a segmentation map extracted from an image that is different from the one being translated. In Figure 1, we can see how the denormalization wrongly creates some features in the region of the image they do not belong to (*i.e.* green on the road).



Source for segmentation | Semantic segmentation

Source for translation | Translated image

Figure 1: **Fake segmentation for image-to-image translation.** We take two different samples X_S^1 (a) and X_S^2 (c) from GTA5. We then use M to get the predicted segmentation $M(X_S^1)$ (b) and use it as semantic guidance for the translation of X_S^1 to get $X_{S\to T} = F_{S\to T}(X_S^1, M(X_S^2))$. The result (d) emphasizes the effect of the semantic guidance in our image-to-image translation method.

4 Additional results

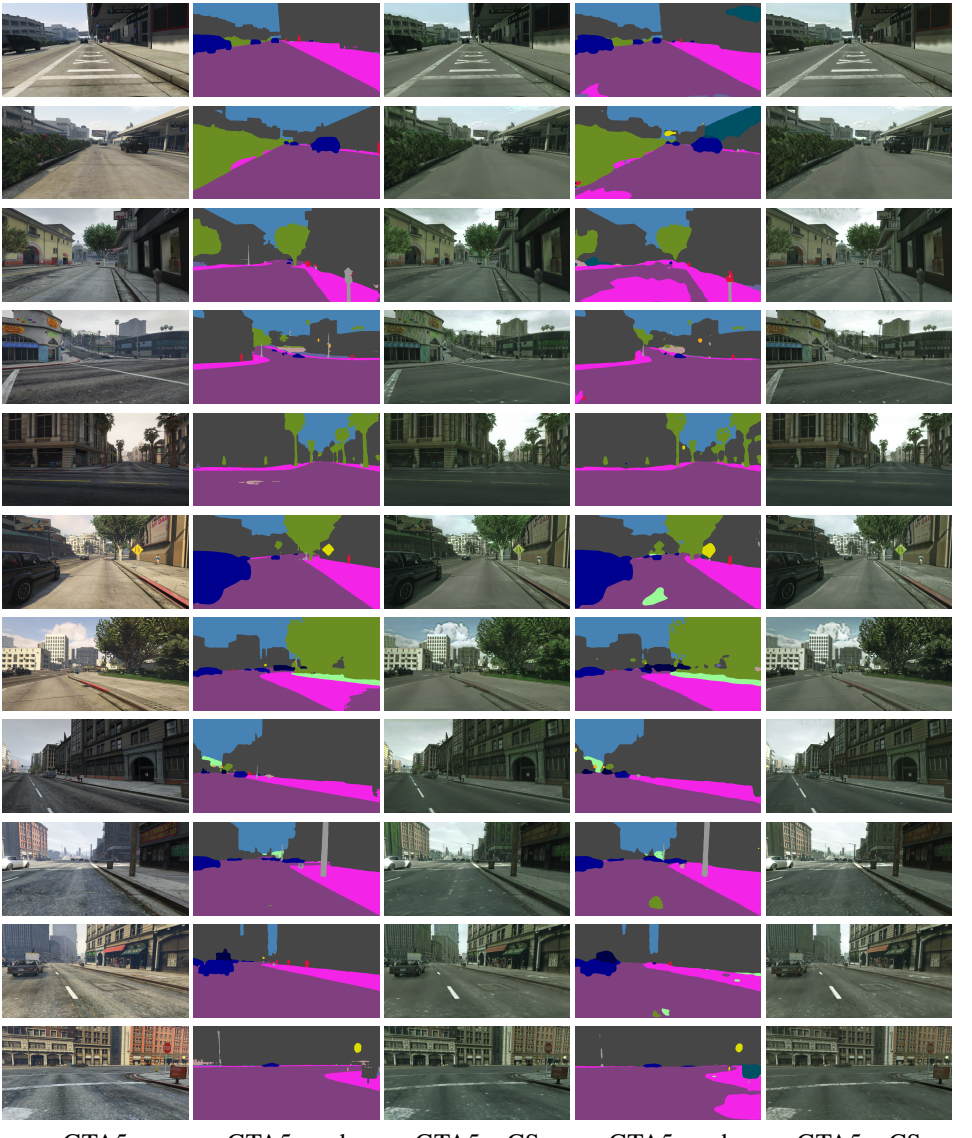


Figure 2: **Additional translations from GTA5 to Cityscapes.** We take a sample X_S from GTA5, get the predicted segmentation using M, and generate $X_{S\to T}$. We present the results obtained with both DeepLabV2 and FCN8s used as semantic guidance.

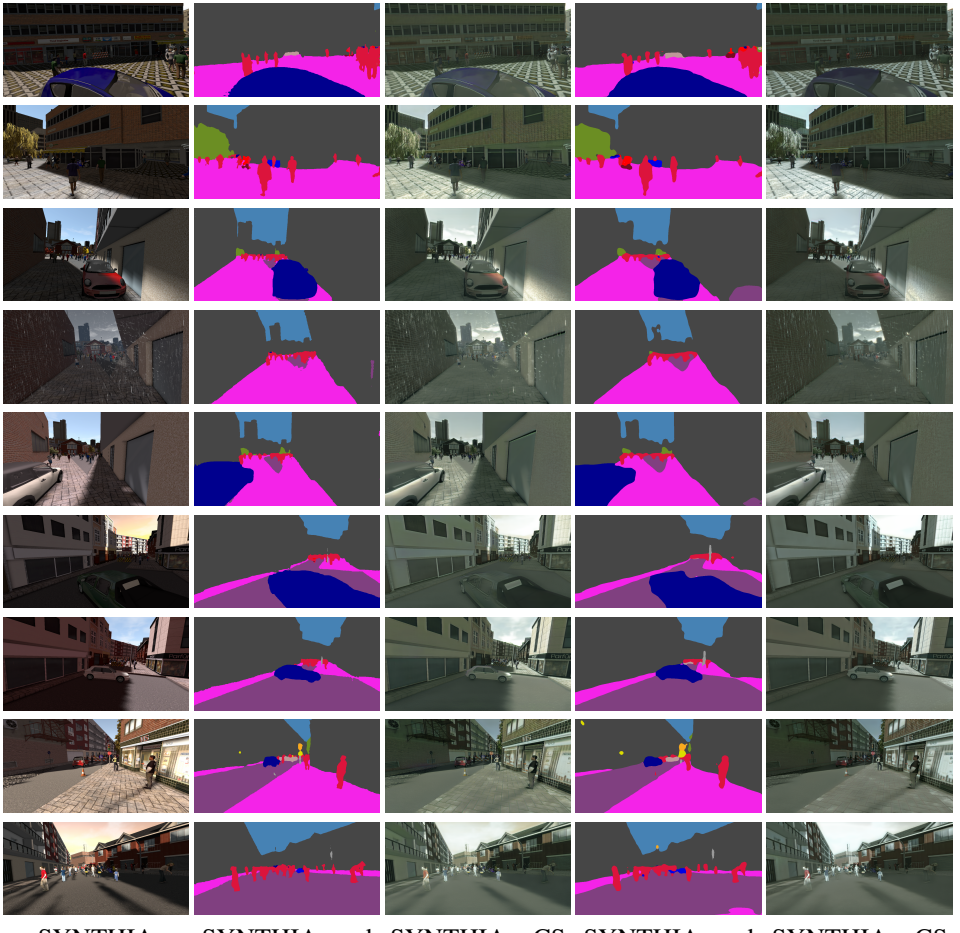


Figure 3: **Translations from SYNTHIA to Cityscapes.** We take a sample X_S from SYNTHIA, get the predicted segmentation using M, and generate $X_{S \to T}$. We present the results obtained with both DeepLabV2 and FCN8s used as semantic guidance.

Figure 4: **Additional segmentation results.** We take a sample X_T from the Cityscapes validation set and get the predicted segmentation using M. Here we show the different results obtainable with M being DeepLabV2. First we show the results obtained with M trained with no adaptation on GTA5, then the results obtained by adapting GTA5 and SYNTHIA.