Supplementary Materials for Exploiting Inter-pixel Correlations in Unsupervised Domain Adaptation for Semantic Segmentation

Inseop Chung Jayeon Yoo Nojun Kwak
Seoul National University
{jis3613,jayeon.yoo,nojunk}@snu.ac.kr

Method	GTA5→CS	SYNTHIA→CS
No Pseudo	40.6	36.3
Pseudo-Only	42.8	39.6
Ours (8) & (10)	**43.1**	**40.0**
Ours (9) & (10)	42.8	39.7
Ours only on Target (11)	42.6(8)	39.7(8)

Table A.1. Results of Ablation Studies. The numbers are mIoU of 19 classes and 16 classes for GTA5→CS and SYNTHIA→CS respectively.

	GTA5→CS		SYNTHIA→CS	
Gen	Pseudo-Only	Ours	Pseudo-Only	Ours
Gen1	42.8	43.1	39.6	40.0
Gen2	43.0	43.9	40.7	41.5
Gen3	43.3	44.2	41.0	42.0
Gen4	44.1	44.7	42.1	42.2
Gen5	44.2	45.7	**42.4**	42.8
Gen6	**44.3**	**45.7**	42.3	**43.1**

Table A.2. Results of Iterative Training. The best results are in bold. mIoU 19 and mIoU 16 are used for GTA5→CS and SYNTHIA→CS respectively.

A. Experiments on FCN-8s with VGG16 backbone

A.1. Training details

We also conduct experiments using a FCN-8s with VGG16 backbone which is another segmentation network that is widely used in unsupervised domain adaptation for semantic segmentation. We train FCN-8s with VGG16 backbone by ADAM optimizer with an initial learning rate of 1×10^{-5} and the momentums of 0.9 and 0.99. The learning rate is decayed by 'step' learning rate policy with a step size of 50,000 and a decay rate of 0.1. The hyper-parameter λ is set to 0.001 empirically. Other details are the same as the DeepLabV2 with ResNet101 backbone. Different from DeepLabV2, FCN-8s uses transposed convolution instead of bilinear interpolation for upsampling z.

A.2. Ablation Study

Tab. A.1 shows our ablation study on FCN-8s with VGG16 backbone. As shown in the table, applying our self-attention loss improves the performance for both domains, but its performance gain is somewhat lower than that of DeepLabV2. Moreover, in contrast to DeepLabV2, using (8) instead of (9) and applying our self-attention loss on both domains rather than only on the target domain achieve better performance when using FCN-8s. We conjecture this difference comes from the different architecture of FCN-8s which uses transposed convolution for upsampling z instead of bilinear interpolation used in DeepLabV2. Since our proposed self-attention loss is computed on z and not on $U(z)$, it can not train the transposed convolutional layer of FCN-8s. This could be the possible reason why ours does not show as much performance gain as it shows in the DeepLabV2 experiment.

A.3. Iterative Training

We also conduct iterative training analogous to DeepLabV2. As it can be seen in the Tab. A.2, Both 'Pseudo-Only' and 'Ours' show improved performance as the generation goes on. However, 'Ours' shows better performance improvement than 'Pseudo-Only' between the generations. The performance of 'Pseudo-Only' gets saturated around a certain generation while 'Ours' keeps showing noticeable performance gain even in the later generation.

A.4. Comparison with other methods

Tab. A.3 and Tab. A.4 show performance comparison with other methods using FCN-8s with VGG16 backbone on GTA5→Cityscapes and SYNTHIA→Cityscapes tasks respectively. Some methods are not included in the table be-

Method	road	side.	build.	wall	fence	pole	light	sign	vege.	terrain	sky	person	rider	car	truck	bus	train	motor	bike	mIoU
CrDoCo [3]	89.1	33.2	80.1	26.9	25.0	18.3	23.4	12.8	77.0	29.1	72.4	55.1	20.2	79.9	22.3	19.5	1.0	20.1	18.7	38.1
CrCDA [5]	86.8	37.5	80.4	30.7	18.1	26.8	25.3	15.1	81.5	30.9	72.1	52.8	19.0	82.1	25.4	29.2	10.1	15.8	3.7	39.1
BDL [7]	89.2	40.9	81.2	29.1	19.2	14.2	29.0	19.6	83.7	35.9	80.7	54.7	23.3	82.7	25.8	28.0	2.3	25.7	19.9	41.3
FDA-MBT [11]	86.1	35.1	80.6	30.8	20.4	27.5	30.0	26.0	82.1	30.3	73.6	52.5	21.7	81.7	24.0	30.5	**29.9**	14.6	24.0	42.2
Kim et al. [6]	**92.5**	**54.5**	**83.9**	**34.5**	**25.5**	**31.0**	30.4	18.0	84.1	39.6	**83.9**	53.6	19.3	81.7	21.1	13.6	17.7	12.3	6.5	42.3
SIM [8]	88.1	35.8	83.1	25.8	23.9	29.2	28.8	28.6	83.0	36.7	82.3	53.7	22.8	82.3	26.4	**38.6**	0.0	19.6	17.1	42.4
Label-driven[9]	90.1	41.2	82.2	30.3	21.3	18.3	33.5	23.0	84.1	37.5	81.4	54.2	24.3	83.0	27.6	32.0	8.1	**29.7**	26.9	43.6
MaxCos [4]	90.3	42.6	82.2	29.7	22.2	18.5	32.8	26.8	84.3	37.1	80.2	55.2	26.4	83.0	**30.3**	35.1	7.0	29.6	28.9	44.3
CADA [10]	90.1	46.7	82.7	34.2	25.3	21.3	33.0	22.0	**84.4**	41.4	78.9	55.5	25.8	**83.1**	24.9	31.4	20.6	25.2	27.8	44.9
Ours	87.4	40.8	81.8	31.7	19.3	26.3	**36.3**	**34.1**	83.9	**43.2**	79.9	**56.1**	**27.0**	81.8	26.4	38.3	4.1	29.4	**39.9**	**45.7**

Table A.3. Comparison results with other methods on GTA5→Cityscapes. The numbers in bold are the best score for each column.

Method	road	side.	build.	wall	fence	pole	light	sign	vege.	sky	person	rider	car	bus	motor	bike	mIoU*	mIoU
CrCDA [5]	74.5	30.5	78.6	6.6	0.7	21.2	2.3	8.4	77.4	79.1	45.9	16.5	73.1	24.1	9.6	14.2	41.1	35.2
ROAD-Net [2]	77.7	30.0	77.5	9.6	0.3	25.8	10.3	15.6	77.6	79.8	44.5	16.6	67.8	14.5	7.0	23.8	41.7	36.2
GIO-Ada [1]	78.3	29.2	76.9	**11.4**	0.3	26.5	10.8	17.2	**81.7**	**81.9**	45.8	15.4	68.0	15.9	7.5	30.4	43.0	37.3
Kim et al. [6]	**89.8**	**48.6**	78.9	—	—	—	0.0	4.7	80.6	81.7	36.2	13.0	74.4	22.5	6.5	32.8	43.8	—
CrDoCo [3]	84.9	32.8	**80.1**	4.3	0.4	**29.4**	14.2	21.0	79.2	78.3	50.2	15.9	69.8	23.4	11.0	15.6	44.3	38.2
BDL [7]	72.0	30.3	74.5	0.1	0.3	24.6	10.2	25.2	80.5	80.0	54.7	23.2	72.7	24.0	7.5	44.9	46.1	39.0
FDA-MBT [11]	84.2	35.1	78.0	6.1	0.44	27.0	8.5	22.1	77.2	79.6	55.5	19.9	74.8	24.9	**14.3**	40.7	47.3	40.5
CADA [10]	73.0	31.1	77.1	0.2	0.5	27.0	11.3	27.4	81.2	81.0	**59.0**	**25.6**	75.0	26.3	10.1	47.4	48.1	40.8
Label-driven[9]	73.7	29.6	77.6	1.0	0.4	26.0	14.7	26.6	80.6	81.8	57.2	24.5	76.1	27.6	13.6	46.6	48.5	41.1
MaxCos [4]	73.6	30.6	77.5	0.8	0.4	26.7	14.1	**29.3**	80.9	80.6	57.9	24.7	**76.5**	27.2	10.8	**47.8**	48.6	41.2
Ours	85.6	43.7	77.9	7.0	**0.8**	26.3	**21.4**	25.4	80.8	80.5	58.6	21.2	74.7	**29.1**	12.5	44.3	**50.5**	**43.1**

Table A.4. Comparison results with other methods on SYNTHIA→Cityscapes. The numbers in bold are the best score for each column. mIoU* and mIoU denote mIoU of 13 classes and 16 classes respectively.

Loss function		Layer	
$L1$	50.7	Prediction	50.7
KL-Div	49.2	Feature map	50.4
Cosine	48.5	Both	50.2
Pseudo-Only		48.9	

Table A.5. Experimental results of using different loss functions and computing the loss at different layers of the segmentation network.

cause they only conduct experiments using the DeepLabV2. Our method achieves the highest performance compared to other state-of-the-art methods.

B. Loss function and Layer Analysis

We conduct experiment of computing the proposed self-attention loss utilizing different loss functions and at different layers of the segmentation network. The experiment is done on GTA5→Cityscapes task using the DeepLabV2 with ResNet101 backbone. '$L1$', 'KL-Div' and 'Cosine' refer to $L1$ loss, Kullback-Leibler divergence loss and cosine similarity loss respectively. 'KL-Div' and 'Cosine' are computed between each logit (row) of z and z''. The cosine similarity is calculated between $z_i \in \mathbb{R}^C$ and $z''_i \in \mathbb{R}^C$ which are the logits of z and z'' respectively. It trains the segmentation network by maximizing the computed cosine similarity or minimizing the negative cosine similarity equivalently. 'KL-Div' loss function tries to minimize the KL-divergence between z_i and z''_i. We first apply softmax on z_i and z''_i and then calculate the KL-divergence between them, $D_{KL}(\sigma(z''_i)||\sigma(z_i))$, here σ refers to the softmax. We train the segmentation network with three different loss functions independently and compare the results. As shown in the Tab. A.5, we could observe that $L1$ loss shows the best score compared to other two loss functions while 'Cosine' shows the worst score which is even lower than 'Pseudo-Only'.

We also test about on which layer of the segmentation network the self-attention loss would work the best. A segmentation network mainly consists of two parts, a feature extractor, $\mathcal{F}$, and a classification head, $\mathcal{H}$, hence $\mathcal{G}(x) = \mathcal{H}(\mathcal{F}(x))$. 'Prediction' and 'Feature map' in the Tab. A.5, refer to applying our proposed self-attention loss on z and

the feature map, $\mathcal{F}(x) = f \in \mathbb{R}^{h \times w \times k}$, where k refers to the number of channels. 'Both' is applying our method on both z and f. We find that computing self-attention loss only on the prediction z achieves the best performance. We conjecture this result is due to the fact that applying our method only on 'Feature map' can not train the classification head and applying on both layers regularizes the network excessively more than necessary. The layer analysis experiments are done using the $L1$ loss.

C. Selecting the τ^c for pseudo label generation

τ^c is set differently for each class as mentioned in the main paper. We basically follow the pseudo label generation process of [7]. Pseudo labels are generated using a pre-trained segmentation network. The pre-trained network $\mathcal{G}$ makes inference on all images in the training set of the target domain to obtain the prediction results. Then, for each class, we collect all the pixels that are classified as the class from the entire predicted results and add the confidence score of each pixel to a list. Therefore, there is one list for each class. We sort each list and choose the median value of the list as the τ^c for the corresponding class, hence τ^c is set by the confidence score of top 50% of each class. If the chosen median value is higher than 0.9, τ^c is set as 0.9.

D. Discussion about why different settings work for different source domains

From the Tab. 1 of the main paper, we could observe that different settings are suitable for different source domains. It is difficult to prove why this tendency happens exactly, but we can conjecture by analysis. Fig. D.1 is the same entropy analysis figure from the main paper (Fig.4). We think we can get a little hint from it. Ours (8) shows higher entropy than Ours (9) generally on both UDA tasks. However, for SYNTHIA→Cityscapses, Ours (8) show much higher entropy than Ours (9) compared to GTA5→Cityscapses. We can clearly observe the large gap between Ours (8) and Ours (9) on SYNTHIA task. This gap is much larger than that of GTA5 task, especially for incorrect pixels. Ours (8) is defined as minimizing the $L1$ loss between z and z'', where $z'' = z + z'$. On SYNTHIA task, adding the output of the segmentation network, z to the output of the SAM, z' somehow brings noisy and incorrect information and eventually make z'' corrupted and under-perform.

We guess this is because of the larger domain gap between SYNTHIA and Cityscapes than GTA5 and Cityscapes. In fact, SYNTHIA has very different data distribution from Cityscapses and GTA5. Cityscapses and GTA5 are both collected under driving scenario but SYNTHIA is not, it has more images taken from higher position such as traffic surveillance cameras. Also its number of classes shared with Cityscapses (16 classes) is less than GTA5 (19 classes). Therefore, due to this larger domain gap, z itself does not contain well-represented domain-invariant information that can improve the performance but rather deteriorates performance when combined with z'. On the other hand, z' which is refined version of z with the help of the SAM, contains domain-invariant information than can further boost the performance. For this reason, we conjecture that it is better to just follow z' instead of z'' for SYNTHIA task.

We think the same reason applies to why using the self-attention loss only on the target domain works better than using it on both domains for SYNTHIA task. If the self-attention loss is applied on both the source domain and the target domain, the network could be more overfitted and trained towards the source domain. This is not desirable especially when there is a large domain gap between the source and the target domains, such as SYNTHIA and Cityscapes.

E. More Qualitative Results

E.1. Attention and Prediction visualization

Fig. E.1-E.3 show more qualitative results of attention and prediction visualization introduced in Sec 5.6. of main paper. Each figure shows the attention and prediction visualizations of an image on GTA5→Cityscapses and SYNTHIA→Cityscapses tasks. Each row of attention visualization refers to a different class.

E.2. Pixel-wise similarity visualization

In Fig. E.4, we show visualization of pixel-wise similarity. We visualize how each logit of predicted pixel ($z \in \mathbb{R}^{hw \times C}$) is similar to other pixels. It is computed as follows:

$$M = ReLU(\frac{z \cdot z^{\mathsf{T}}}{\|z\|_2 \cdot \|z\|_2^{\mathsf{T}}}) \in \mathbb{R}^{hw \times hw} \tag{1}$$

It is basically an attention map of z itself. We take ReLU on the attention map to visualize the difference in positive correlation between pixels more prominently. We visualize this for both 'Ours' and 'Pseudo-Only'. For the ground truth, we use the nearest interpolation to resize the ground truth label to the spatial size of $h \times w$ from $H \times W$, hence $y_t \in \mathbb{R}^{h \times w \times C}$ where each pixel is a C dimensional one-hot vector. We flatten y_t in the spatial dimension ($y_t \in \mathbb{R}^{hw \times C}$) and compute M by inserting y_t instead of z in (1).

The experiment is conducted using DeepLabV2 with ResNet101 backbone segmentation network that is trained on GTA5→Cityscapses task. We sample several images from validation set of Cityscapses and visualize the pixel-wise similarity described as above. In the figure, bluer means higher similarity. Since 'GT' is visualized using the one-hot vectors of y_t, its each element is either 0 or 1 while elements of 'Ours' and 'Pseudo-Only' are between 0 and 1.

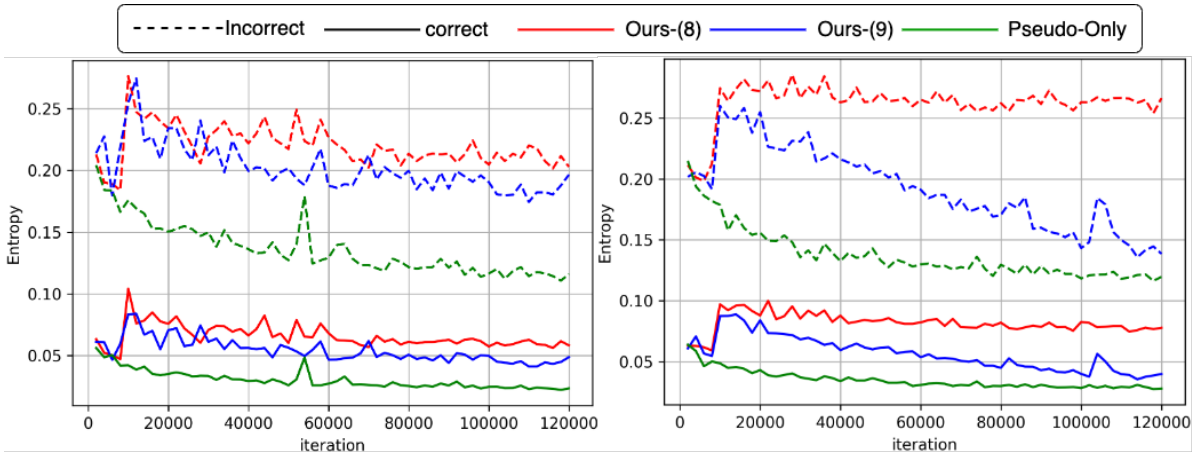

Figure D.1. Entropy analysis of GTA5→Cityscapes (Left) and SYNTHIA→Cityscapes (right).

As shown in the figure, 'Ours' show much similar results to 'GT' compared to 'Pseudo-Only'. Also, a clear difference can be observed between 'Ours' and 'Pseudo-Only'. It means that each pixel of z which is a C dimensional logit, is more attended well with the correct pixels corresponding to the same class and dissimilar to other pixels of different classes. On the other hand, pixels of 'Pseudo-Only' are attended with even irrelevant pixels showing high similarity.

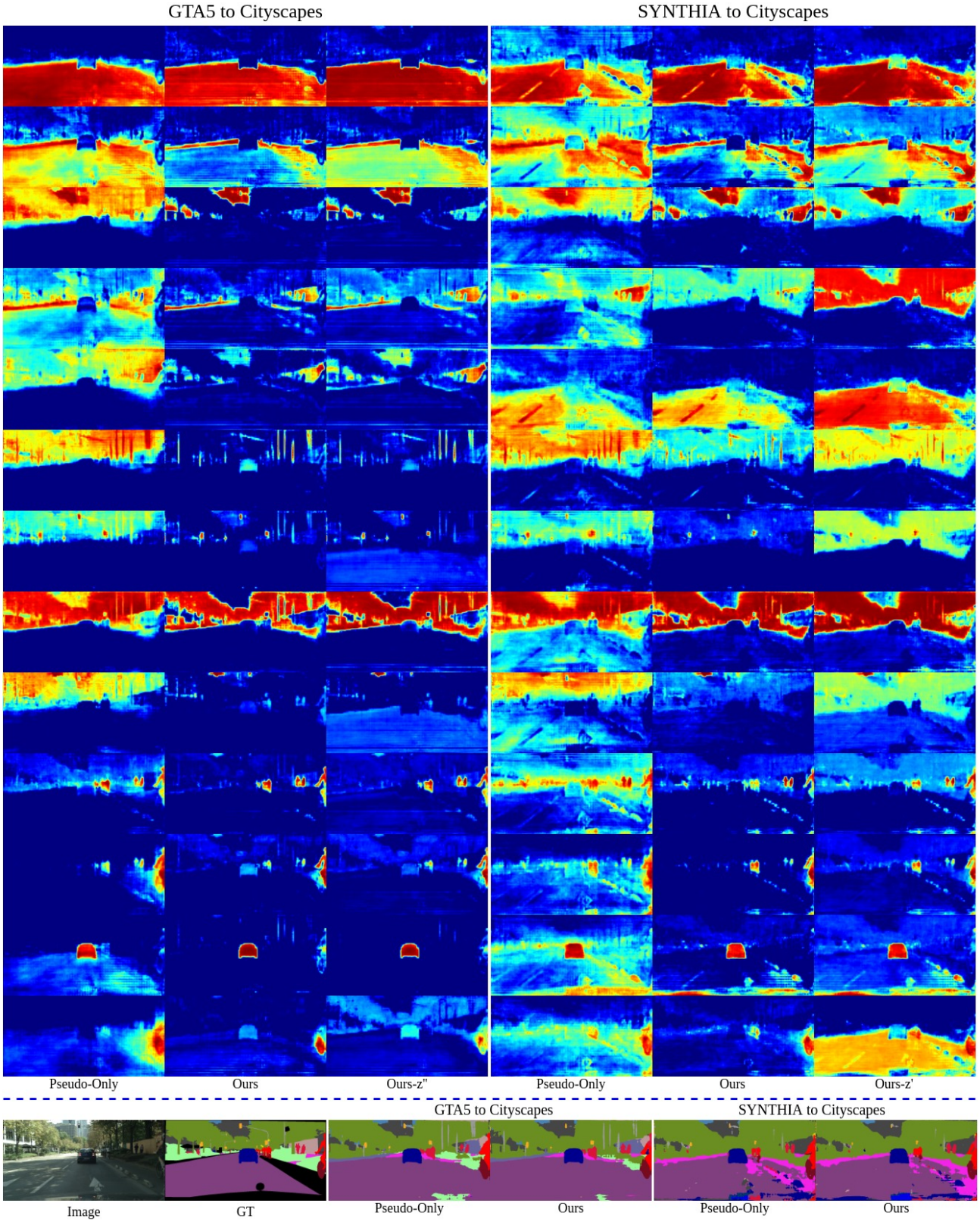

Figure E.1. More attention and prediction visualization.

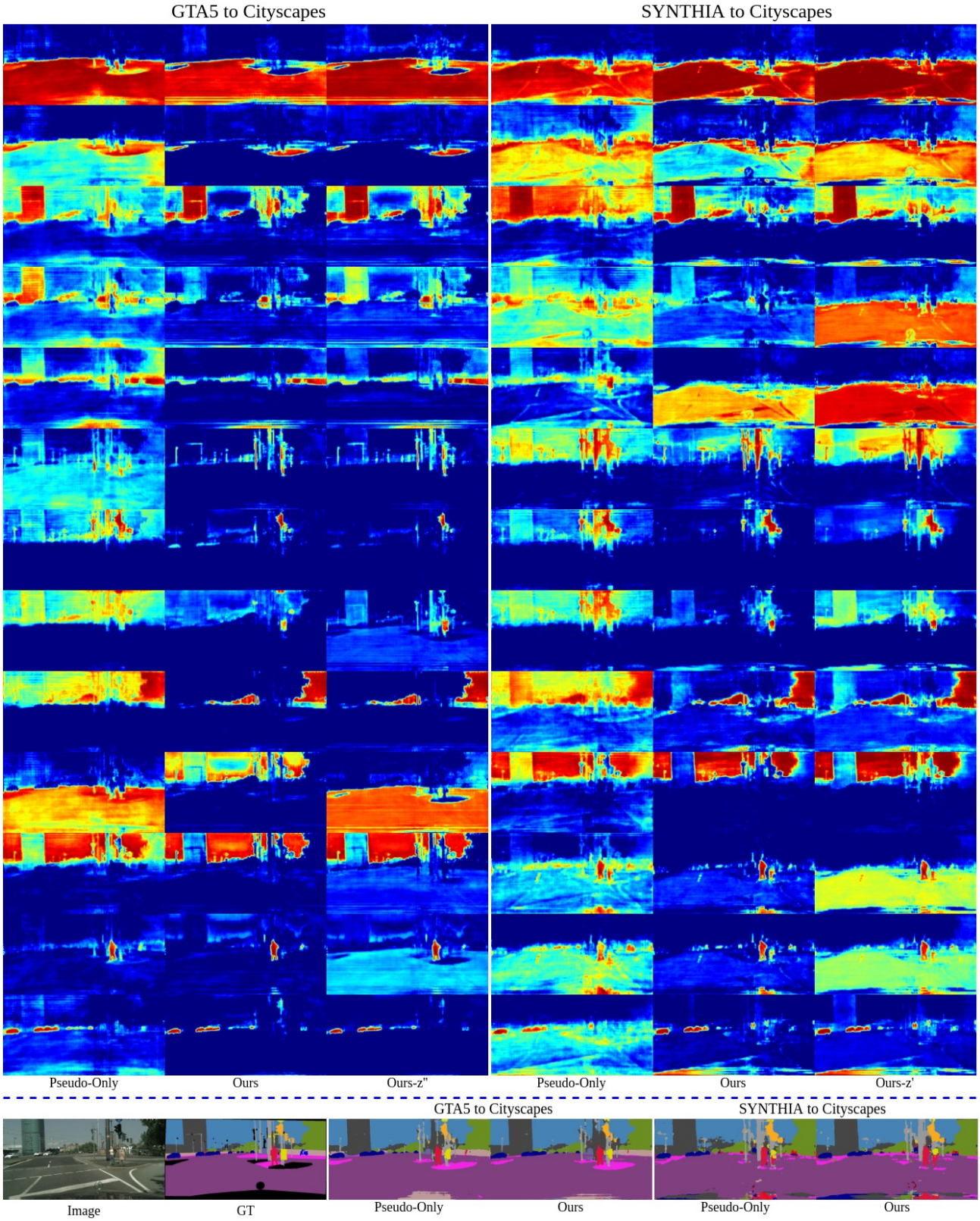

Figure E.2. More attention and prediction visualization.

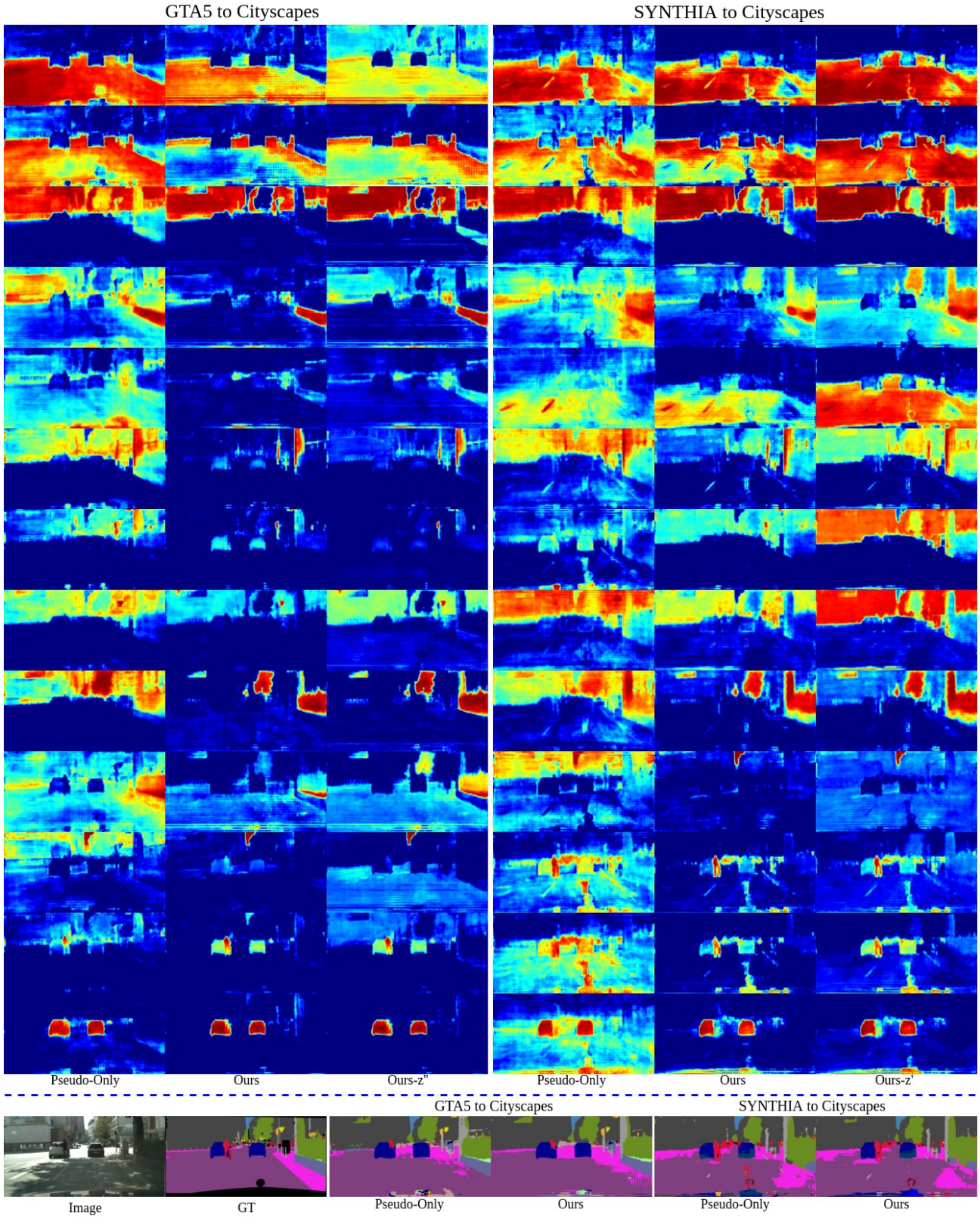

Figure E.3. More attention and prediction visualization.

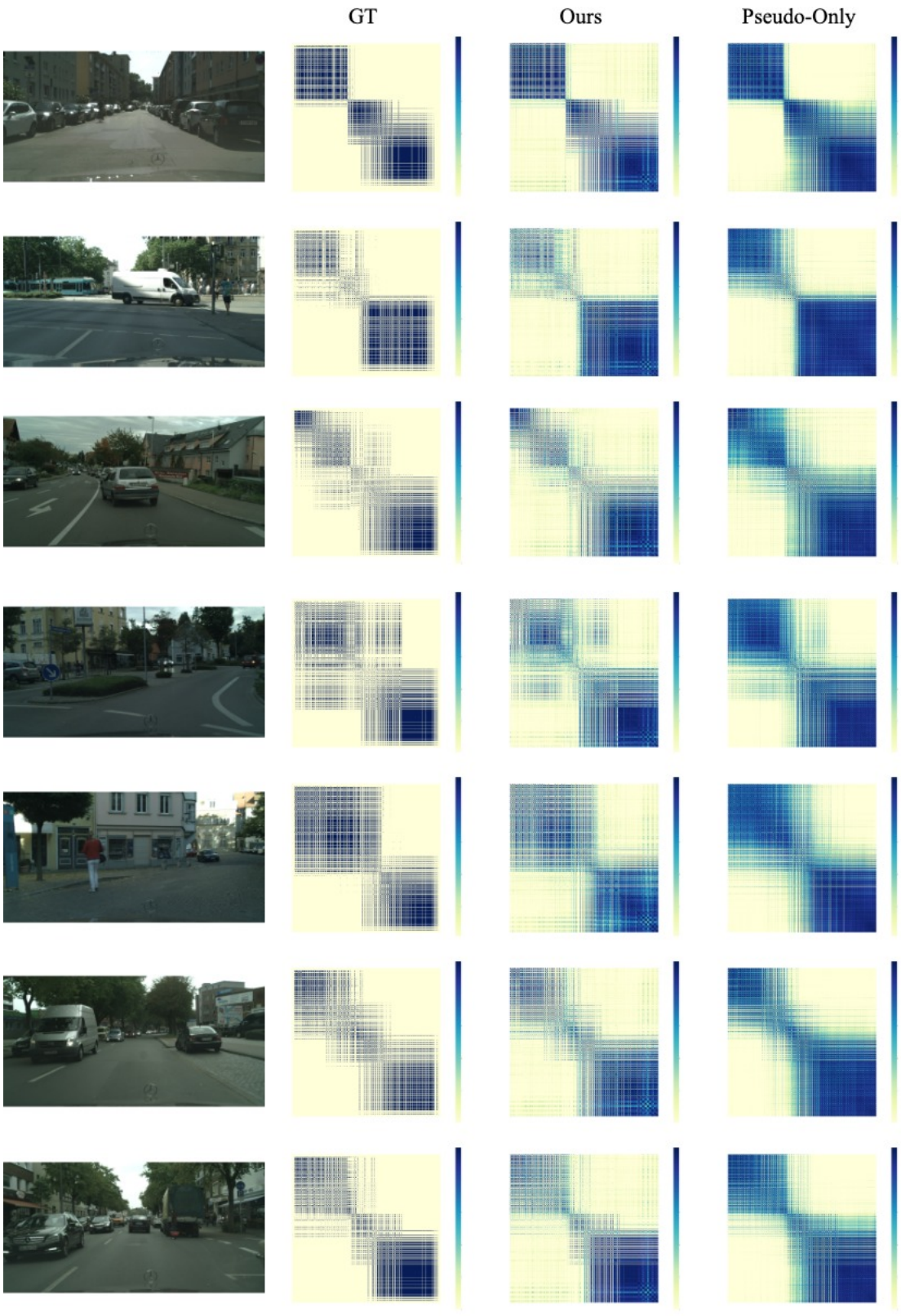

Figure E.4. Comparison of pixel-wise similarity between ‘Ours’ and ‘Pseudo-Only’.

References

[1] Yuhua Chen, Wen Li, Xiaoran Chen, and Luc Van Gool. Learning semantic segmentation from synthetic data: A geometrically guided input-output adaptation approach. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1841–1850, 2019.

[2] Yuhua Chen, Wen Li, and Luc Van Gool. Road: Reality oriented adaptation for semantic segmentation of urban scenes. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 7892–7901, 2018.

[3] Yun-Chun Chen, Yen-Yu Lin, Ming-Hsuan Yang, and Jia-Bin Huang. Crdoco: Pixel-level domain transfer with cross-domain consistency. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 1791–1800, 2019.

[4] Inseop Chung, Daesik Kim, and Nojun Kwak. Maximizing cosine similarity between spatial features for unsupervised domain adaptation in semantic segmentation. *arXiv preprint arXiv:2102.13002*, 2021.

[5] Jiaxing Huang, Shijian Lu, Dayan Guan, and Xiaobing Zhang. Contextual-relation consistent domain adaptation for semantic segmentation. In *European Conference on Computer Vision*, pages 705–722. Springer, 2020.

[6] 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.

[7] Yunsheng Li, Lu Yuan, and Nuno Vasconcelos. Bidirectional learning for domain adaptation of semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 6936–6945, 2019.

[8] Zhonghao Wang, Mo Yu, Yunchao Wei, Rogerio Feris, Jinjun Xiong, Wen-mei Hwu, Thomas S Huang, and Honghui Shi. Differential treatment for stuff and things: A simple unsupervised domain adaptation method for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 12635–12644, 2020.

[9] Jinyu Yang, Weizhi An, Sheng Wang, Xinliang Zhu, Chaochao Yan, and Junzhou Huang. Label-driven reconstruction for domain adaptation in semantic segmentation. *arXiv preprint arXiv:2003.04614*, 2020.

[10] Jinyu Yang, Weizhi An, Chaochao Yan, Peilin Zhao, and Junzhou Huang. Context-aware domain adaptation in semantic segmentation. In *Proceedings of the IEEE/CVF Winter Conference on Applications of Computer Vision*, pages 514–524, 2021.

[11] Yanchao Yang and Stefano Soatto. Fda: Fourier domain adaptation for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4085–4095, 2020.