Supplementary material for Learning Across Tasks and Domains

Pierluigi Zama Ramirez, Alessio Tonioni, Samuele Salti, Luigi Di Stefano
Department of Computer Science and Engineering (DISI)
University of Bologna, Italy
{pierluigi.zama, alessio.tonioni, samuele.salti, luigi.distefano }@unibo.it

1. Additional Experimental Results

We report here additional experiments to asses the contribution of the different components of AT/DT. In Sec. 1.1 we conduct a study on the performance achievable on the training domain $\mathcal{A}$ with $G_{1\to2}$. In Sec. 1.2 we show ablation studies to confirm the key role of $G_{1\to2}$ in our formulation. In Sec. 1.3 we perform tests considering different approaches to build a shared feature representation for $N_1^{A\cup B}$. In Sec. 1.4 we provide additional details and results on the integration of AT/DT with existing domain adaptation techniques. In Sec. 1.5 we report qualitative results using normal estimation as target task ($\mathcal{T}_2$). Finally, in Sec. 2 and Sec. 3 we provide additional details about the training and evaluation processes.

1.1. Train domain performance of $G_{1\to2}$

Our framework has to overcome two nuisances to effectively address the lacking of supervision in the target task and domain: translation of features between tasks and change of domain. In this section, we are interested in isolating the impact of the first nuisance, which will also provide some hints on the importance of the second one. In other words, we are trying to answer the question: *How well are we effectively learning to translate deep representations?*

To focus only on the effectiveness in transferring representations, we consider a test set of images from $\mathcal{A}$ and compare AT/DT and N_2^A (the network trained on domain $\mathcal{A}$ for $\mathcal{T}_2$). As the test data are sampled from the same domain as the training data, we do not have errors due to the domain shift and can use the gap in performance between the two algorithms as a measure of the effectiveness of our framework in transferring representations. As we wish to evaluate both semantic segmentation and depth estimation, we select the Synthia domain as $\mathcal{A}$, for which we have all labels available, and Cityscapes as $\mathcal{B}$. In Tab. 1 we report the results when transferring deep representations in the $Dep. \to Sem.$ scenario, while in Tab. 2 in the $Sem. \to Dep.$ scenario.

Tab. 1 shows how transferring deep representations from $\mathcal{T}_1$ to $\mathcal{T}_2$ with AT/DT results in a small loss in performance compared to N_2^A. In particular, the largest performance drops are related to classes dealing with small objects, like 'Fence', 'Poles' and 'Traffic Sign', that might get lost transferring features at the smallest spatial resolution in the network. These results suggest that a multi-scale transfer strategy would be a direction worth exploring in future work to better recover small details upon transferring representations. Nevertheless, the comparisin between the final pixel accuracy (Acc.) highlights that AT/DT loses only 1% though relying on a feature extractor trained for a different task.

In Tab. 2 AT/DT obtains again performance close to N_2^A. For some metrics, it even delivers better performance than N_2^A. This somewhat surprising result can be explained by the difference between the training sets: AT/DT uses as feature extractor $N_1^{A\cup B}$, which has been trained with samples from both $\mathcal{A}$ and $\mathcal{B}$, *i.e.* with a larger and more varied training set than that used by N_2^A. Therefore, the encoder of $N_1^{A\cup B}$ might learn a more general feature extractor than that of N_2^A, this resulting in better performance when applied on unseen data. AT/DT can successfully leverage on this better feature extractor and obtain slightly better performance when transferring them to $\mathcal{T}_2$.

The same reasoning may be applied to the results of Tab. 1. However, in this case, the shared encoder of $N_1^{A\cup B}$ has been partially trained with noisy ground truth depth labels on samples from $\mathcal{B}$. The introduction of noise in the training process might harm the learning of $N_1^{A\cup B}$ and explain the small gap in performance. Moreover, as stated above, due to the transferring of features at low resolution, AT/DT might struggle to transfer small image structures (*e.g.*, 'poles', 'traffic sign'...). However, wrong predictions on this kind of small structures do not arm much the depth estimation metrics (*i.e.*, few pixels are considered), though they have a larger impact on the mIoU metric considered for semantic segmentation. Finally, as stated in [2], the advantages yielded by semantic information to depth estimation are larger than the gains attainable going in the other direction, thus motivating the slight difference in performance across the two scenarios.

$\mathcal{A}$	Method	Road	Sidewalk	Walls	Fence	Person	Poles	Vegetation	Vehicles	Tr. Signs	Building	Sky	**mIoU**	**Acc**
Synthia	N_2^A	**99.23**	**87.16**	**92.67**	**28.62**	**48.53**	**63.54**	**85.02**	**88.92**	**52.67**	**96.91**	**98.39**	**76.52**	**98.45**
Synthia	AT/DT	98.34	76.09	84.99	1.06	29.25	45.57	80.15	85.72	25.31	95.53	97.45	65.41	97.53

Table 1: Experimental results of $Dep. \rightarrow Sem.$ scenario using as domain $\mathcal{A}$ the Synthia dataset. Best results highlighted in bold.

		Lower is better				Higher is better		
$\mathcal{A}$	Method	Abs Rel	Sq Rel	RMSE	RMSE log	δ_1	δ_2	δ_3
Synthia	N_2^A	0.138	**1.212**	**4.759**	0.825	**0.864**	0.952	0.970
Synthia	AT/DT	**0.135**	1.271	5.061	**0.634**	0.863	**0.958**	**0.977**

Table 2: Experimental results of $Sem. \rightarrow Dep.$ scenario using as domain $\mathcal{A}$ the Synthia dataset. Best results highlighted in bold.

Overall, the results reported in Tab. 2 and Tab. 1 show that our framework is indeed learning to transfer deep representations effectively and that it is possible to approximate $G_{1\rightarrow 2}$ by a neural network like that we propose in this work. This is further validated in Fig. 1, where we report two t-SNE[3] plots of deep features extracted by $N_1^{A\cup B}$ (in pink), N_2^A (in blue) alongside with the features transformed by $G_{1\rightarrow 2}$ (in red). All features are computed on image samples from the test set described above, *i.e.* samples unseen at training time. Therefore, $G_{1\rightarrow 2}$ takes as input pink points and produces red points that should be as close as possible to the blue points. Indeed, the two plots show how our task transfer network can successfully produce features suitable for $\mathcal{T}_2$.

1.2. Importance of $G_{1\rightarrow 2}$

We report results of additional tests to further assess the importance of $G_{1\rightarrow 2}$ in our cross tasks and domains adaptation. Purposely, we consider a single network made out of one encoder, $E_{1,2}^{A\cup B}$ and two decoders, $D_1^{A\cup B}$ and D_2^A. $D_1^{A\cup B}$ is trained with samples from $\mathcal{A}$ and $\mathcal{B}$ for $\mathcal{T}_1$. D_2^A is trained with samples from $\mathcal{A}$ for $\mathcal{T}_2$. Finally, $E_{1,2}^{A\cup B}$ is trained together with the two heads with both tasks and domains. Therefore we consider a single feature extractor which yields a shared representation for both tasks and domains without the need to learn a transfer function between tasks. We will refer to this configuration as the *No Transfer* setting.

We evaluate *No Transfer* for both $Dep. \rightarrow Sem.$ and $Sem. \rightarrow Dep.$ settings from Carla to Cityscapes and compare it to AT/DT and the transfer learning baseline of the main paper. Tab. 3 and Tab. 4 report results for $Dep. \rightarrow Sem.$ and $Sem. \rightarrow Dep.$ settings respectively. For $Sem. \rightarrow Dep.$ our method outperforms *No Transfer* for all metrics, and indeed this alternative is even worse than the baseline for Sq. Rel. and RMSE. On the other hand, for $Dep. \rightarrow Sem.$ our method achieves better performances in the majority of the classes and for the mIoU, while *No Transfer* provides the best pixel accuracy. We ascribe this result to *No Transfer* providing the highest IoU for the *road* class, which represents the vast majority of pixels in an autonomous driving scenario. However this good performance does not translate to other classes such that *No transfer* achieves the worst mIoU, even less than the baseline. These results confirm the importance of learning a mapping function (*e.g.*, $G_{1\rightarrow 2}$) between features to transfer representations between tasks.

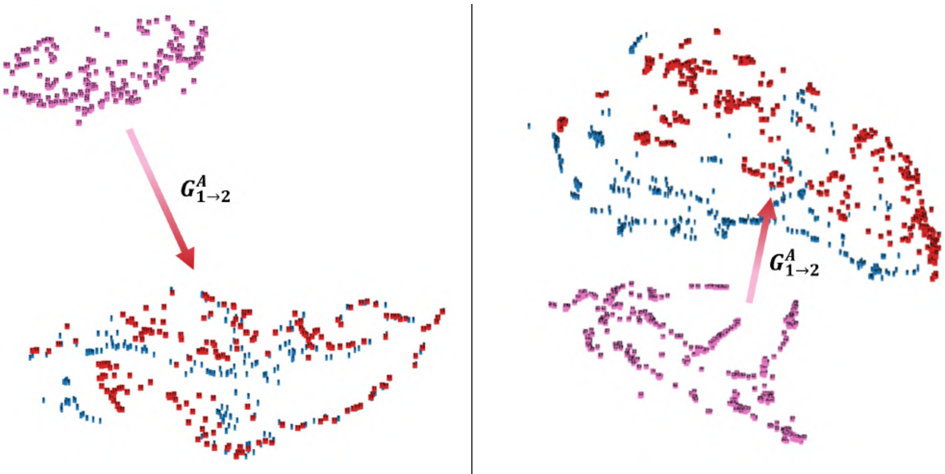
1.3. Shared Decoder and Separate Encoders for N_1

In Sec. 6.2 we highlighted how learning a common representation for $\mathcal{T}_1$ is crucial to learn a transfer function which generalize across domains. In this additional test we show that to learn a good shared representation across domains for one task, we need to share both encoders and decoders in $N_1^{A\cup B}$. For this reason we train a different version of $N_1^{A\cup B}$ with a shared decoder but two encoders, one trained only on $\mathcal{A}$ and the other only on $\mathcal{B}$. Tab. 5 compares this architecture to AT/DT for Synthia to Cityscapes in the $Dep. \rightarrow Sem.$scenario. Indeed training a shared encoder allows the representation to be more closely related resulting in better performance.

1.4. Integration with Domain Adaptation

We report here additional details on how we have used CycleGAN [5] to address domain adaptation.

We train CycleGAN to transform images from Carla ($\mathcal{A}$)



$Sem. \to Dep.$ $Dep. \to Sem.$

Figure 1: t-SNE [3] plots of deep features computed on $\mathcal{A}$. Pink denotes the features extracted for $\mathcal{T}_1$, *i.e.* $E_1^{A \cup B}(x_a)$. Blue features extracted for $\mathcal{T}_2$, *i.e.* $E_2^A(x_a)$. Red the prediction obtained by the feature transfer network $G_{1\to2}(E_1^{A\cup B}(x_a))$. Therefore, the red points are the transformations of the pink points according to $G_{1\to2}$. With an ideal $G_{1\to2}$ red and blue points would perfectly overlap, here we can see that unfortunately this is not the case. Nevertheless our transfer function successfully transform pink features to make them closer to blue ones.

	$\mathcal{A}$	$\mathcal{B}$	Method	Road	Sidewalk	Walls	Fence	Person	Poles	Vegetation	Vehicles	Tr. Signs	Building	Sky	**mIoU**	**Acc**
(c)	Carla	Cityscapes	Baseline	71.87	**36.53**	3.99	**6.66**	24.33	22.20	66.06	**48.12**	7.60	60.22	69.05	37.88	74.61
(c)	Carla	Cityscapes	No Transfer	**84.82**	33.15	1.00	1.79	6.30	14.26	**69.91**	40.32	1.84	65.67	73.49	35.69	**79.53**
	Carla	Cityscapes	AT/DT	76.44	32.24	**4.75**	5.58	**24.49**	**24.95**	68.98	40.49	**10.78**	**69.38**	**78.19**	**39.66**	76.37

Table 3: Experimental results of $Dep. \to Sem.$ scenario. Best results highlighted in bold.

to Cityscapes ($\mathcal{B}$) and vice-versa. The network is trained using the original author implementation[1] for 200k steps on random image crops of 400×400 pixels. We use the same hyper-parameters settings as proposed in the original paper.

Once trained, we transform the Cityscapes dataset into the Carla style generating a new $CityscapesLikeCarla$ dataset which we will call $\mathcal{B}like\mathcal{A}$ domain (see Fig. 2). The baseline is then obtained by testing N_2^A with the validation set of $\mathcal{B}like\mathcal{A}$. To integrate AT/DT with CycleGAN, we train a $N_1^{A\cup\{BlikeA\}}$ on both $\mathcal{A}$ and $\mathcal{B}like\mathcal{A}$ at step 1 of AT/DT. Then, at step 4, to infer the predictions for $\mathcal{T}_2$ on $\mathcal{B}$, we employ the validation set of $\mathcal{B}like\mathcal{A}$ as done for the baseline. To summarize we train the shared source network on samples obtained from $\mathcal{A}$ and $\mathcal{B}like\mathcal{A}$, then we test all networks on the test set of $\mathcal{B}like\mathcal{A}$ (*i.e.*, Cityscapes images transformed to look like those from Carla).

In Fig. 3 we show some qualitative results obtained when combining AT/DT together with the pixel level domain adaptation obtained through CycleGAN. Comparing the results in the $Sem. \to Dep.$ scenario (first row) with those obtained in a $Dep. \to Sem.$ scenario (second row) we can see how CycleGAN is very effective when targeting the semantic segmentation tasks, much less effective when targeting a depth estimation task. AT/DT, instead, consistently produce better predictions than the baseline in both the considered tasks.

1.5. Additional tasks

In Fig. 4 we report additional qualitative results when using as $\mathcal{T}_1$ semantic segmentation and as $\mathcal{T}_2$ normal estimation, with Carla as $\mathcal{A}$ and Cityscapes as $\mathcal{B}$. The results confirm the findings of the semantic to depth scenario, with AT/DT producing clearly better prediction than the baseline network. We report only qualitative results due to the lack of annotations to validate normal estimation on the real Cityscapes data.

[1] https://github.com/junyanz/pytorch-CycleGAN-and-pix2pix

	$\mathcal{A}$	$\mathcal{B}$	Method	Lower is better Abs Rel	 Sq Rel	 RMSE	 RMSE log	Higher is better δ_1	 δ_2	 δ_3
	Carla	Cityscapes	Baseline	0.667	13.500	16.875	0.593	0.276	0.566	0.770
(b)	Carla	Cityscapes	No Transfer	0.615	17.578	19.924	0.533	0.284	0.646	0.845
(b)	Carla	Cityscapes	AT/DT	**0.394**	**5.837**	**13.915**	**0.435**	**0.337**	**0.749**	**0.899**

Table 4: Experimental results of $Sem. \rightarrow Dep.$ scenario. Best results highlighted in bold.

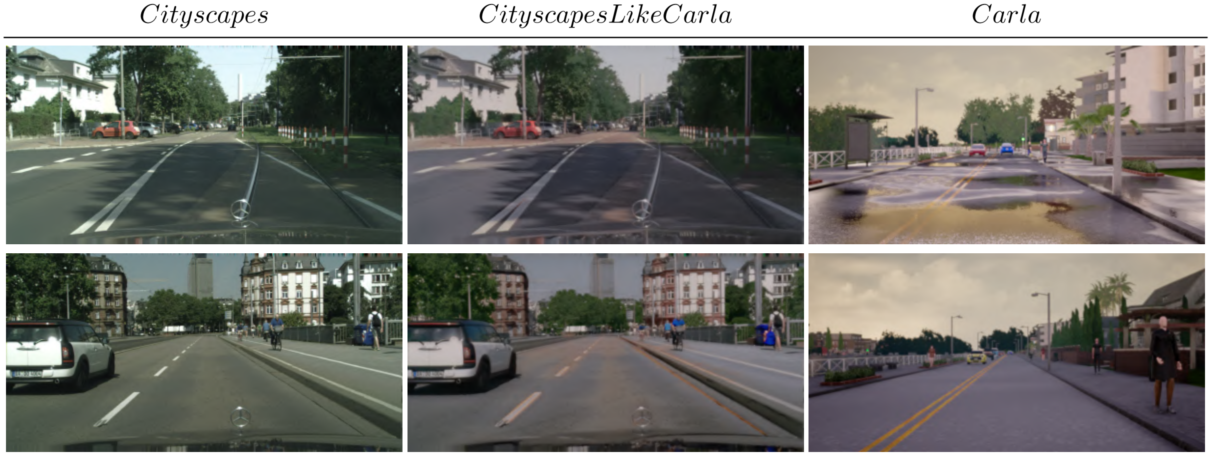


Figure 2: Images obtained applying CycleGAN to make Cityscapes samples similar to those of Carla. From left to right: samples from Cityscapes, corresponding image from $CityscapesLikeCarla$ obtained by CycleGAN, similar samples from Carla

Shared Encoders	mIoU	Acc.
✗	11.55	56.79
✓	23.24 **(+11.69)**	64.03 **(+7,24)**

Table 5: Study on Shared Decoder with Non Shared Encoders for $N_1^{A \cup B}$. We show a $\mathcal{A}$: Synthia to $\mathcal{B}$: Carla and $Dep. \rightarrow Sem.$ scenario. Performance improvement highlighted in bold.

2. Details on the training process

Our task networks consist of a ResNet50 as encoder and a stack of 3 series of bilinear upsampler followed by one convolution as the decoder. Our ResNet50 use dilated convolution with rate 2 and 4 in the last two residual blocks, similarly to DRN [4]. We trained our $N_1^{A \cup B}$ and N_2^A until the loss stabilizes with batch size 8 and crop 512x512.

We use Adam [1] as optimizer with a linear decaying learning rate 10^{-4} and $\beta_1 = 0.9$.

Our $G_{1 \rightarrow 2}$ consists in a stack of 6 convolutional layers with kernel size 3x3 going down to a quarter of the input resolution and then upsampling back to original resolution. We train this network for 100k iterations with batch size 1 and random crops of 512×512 pixels. We use Adam [1] as optimizer with learning rate 10^{-5}.

3. Details on the evaluation process

We perform all the evaluation at the original image resolution for Cityscapes, Carla and Synthia. Instead, for Kitti, we consider a central crop with size 320×1216 due to the varying size of images.

Semantic Segmentation We train and evaluate the semantic segmentation task on 11 classes, the 10 defined by the Carla framework[2] plus the additional 'Sky' class that we define as the set of points at infinite depth. To evaluate the network on Cityscapes we collapse some of the available classes to make them compatible with Carla: *car* and *bicycle* collapse into *vehicle* and *traffic sign* and *traffic light* into *traffic sign*. We ignore the other labels in Cityscapes which do not have a corresponding class in Carla.

Depth We trained and evaluate the depth networks clipping the max predictable depth to 100m and then normalizing between 0 and 1. At inference time we scale the predictions back to the 0m-100m range before computing the different metrics.

[2] https://github.com/carla-simulator/carla/releases/tag/0.8.4

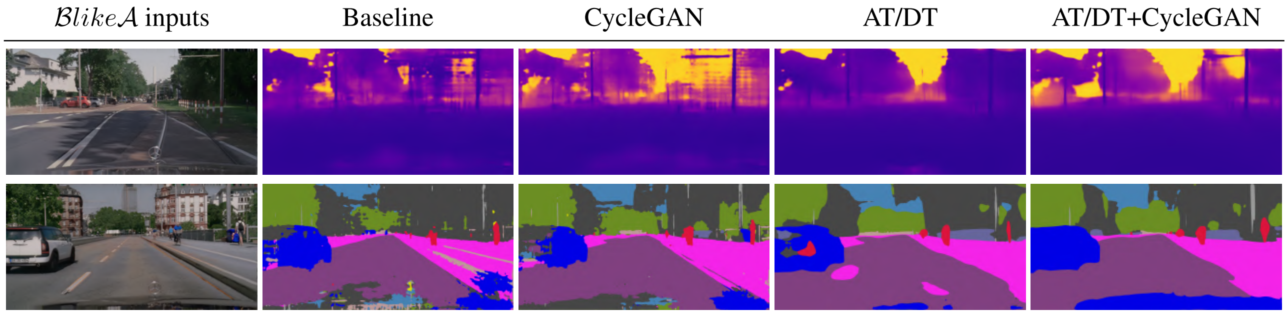


Figure 3: Qualitative results on the Cityscapes dataset in a $Sem. \rightarrow Dep.$ scenario (first row) and $Dep. \rightarrow Sem.$ scenario (second row). From left to right: $\mathcal{B}like\mathcal{A}$ inputs, predictions obtained by a transfer learning baseline, by a domain adaptation baseline (CycleGAN[5]), by our framework (AT/DT) and by our framework aided by domain adaptation (AT/DT+CycleGAN).

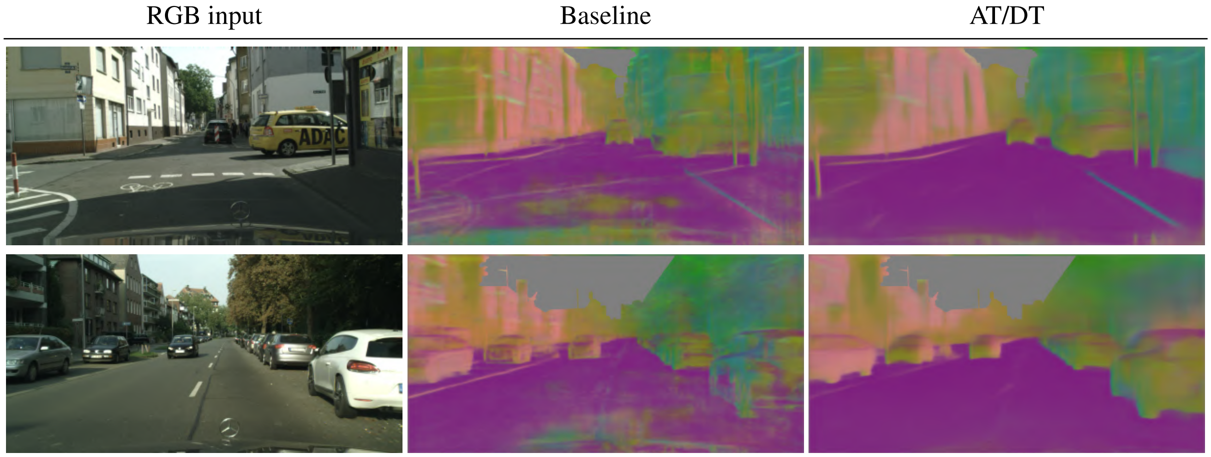


Figure 4: Qualitative results on Cityscapes dataset in a $Sem. \rightarrow Norm.$ scenario. From left to right: RGB input, prediction obtained by a transfer learning baseline and by our framework (AT/DT).

References

[1] Diederik P. Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2015. 4

[2] Pierluigi Zama Ramirez, Matteo Poggi, Fabio Tosi, Stefano Mattoccia, and Luigi Di Stefano. Geometry meets semantics for semi-supervised monocular depth estimation. *arXiv preprint arXiv:1810.04093*, 2018. 1

[3] Laurens van der Maaten and Geoffrey Hinton. Visualizing data using t-SNE. *Journal of Machine Learning Research*, 9:2579–2605, 2008. 2, 3

[4] Fisher Yu, Vladlen Koltun, and Thomas Funkhouser. Dilated residual networks. In *Computer Vision and Pattern Recognition (CVPR)*, 2017. 4

[5] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *The IEEE International Conference on Computer Vision (ICCV)*, Oct 2017. 2, 5