NANYANG TECHNOLOGICAL UNIVERSITY SINGAPORE

Nanyang Technological University

School of Computer Science and Engineering

Project: Image Translation and UDA

Yuan Xinnan G2202162G

Pan Boyu G2202284K

Lin Zhiyuan G2203036B

An Assignment submitted for the NTU:

22S1-AI6121-Computer Vision

March 13, 2023

1 Introduction

1.1 Generative Adversarial Networks

Generative adversarial network (GAN) has received more and more attention from academia and industry since it was proposed by Goodfellow I[7]. With the rapid development of GAN in theory and model, it has more and more in-depth applications in computer vision, natural language processing, human-computer interaction and other fields, and continues to extend to other fields.

As generative models, GANs achieve the synthesis realism by pairing a generator, which learns to produce the target output, with a discriminator, which learns to distinguish true data from the generated. The generator tries to fool the discriminator, and the discriminator tries to keep from being fooled. Both generator and discriminator are networks. The discriminator is a classifier, which tries to distinguish real data from the data generated by the generator G, and the generator learns to create fake data to fool the discriminator based on the discriminator feedback. In the training process, G and D are trained alternately, and GANs often use a minimax loss. The overall structure of basic GAN network and how it works are shown in Figure 1.

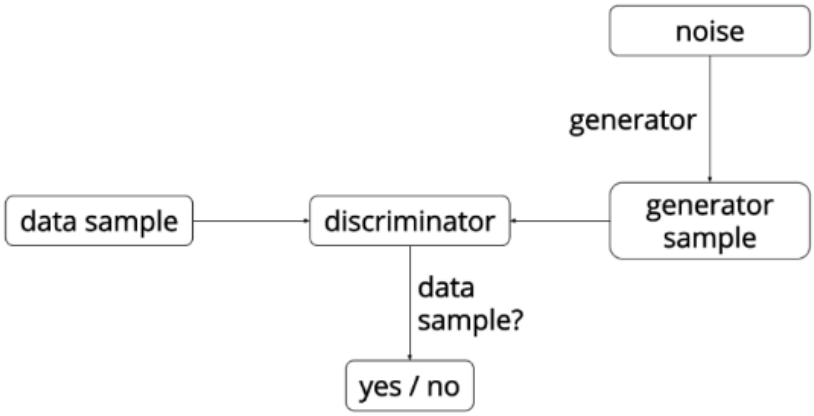

Figure 1: The overview of GAN network

In addition, GANs can take different architectures, e.g. direct architecture, that uses a single generator and discriminator, hierarchical architecture that uses a pair of generators and discriminators for different purposes, and iterative architecture that uses multiple generators and discriminators in cascade for generating from coarse to fine. Moreover, GANs can add some modules like additional task-specific classifiers to the discriminator, which allows to use pre-trained models and helps generate sharper images and other modules to implement different functions.

1.2 Image-to-image Translation

Image-to-image translation changes images in a specific or controlled way. l2l has the ability to convert the content of an image from one image domain X to another image domain Y. It can be regarded as removing a certain attribute X of the original image and re-giving it a new attribute Y. In general, l2l has three different training settings to modify image style and appearance, with aligned paired training images, with unpaired training images of similar domains and with unpaired training images of very distinctive domains.

- **With aligned paired training images:** Pix2pix[2] proposes a method to achieve image-to-image translation by mapping pixels to pixels. They use conditional generative adversarial networks to learn the mapping from input to output images. However, Pix2Pix requires that the training data must be paired, and in real life, it is quite difficult to find pairs of pictures in the two domains.
- **With unpaired training images of similar domains:** In order to be able to solve the problem of lack of paired training data, CycleGAN[3] is proposed. The basic idea of CycleGAN is to trans-

form an image from one domain to another, and then inversely transform it back, so the inversely transformed image should be the same as the original image.

- **With unpaired training images of very distinctive domains:** CoCosNet[8] requires no paired images or any alignment between conditional inputs and exemplars. It uses cross-domain correspondence to map the two domains images to an intermediate domain, find the matching relationship between the two on the intermediate domain, and then use the matching relationship to warp the example image. Then it uses translation network to generate high-quality target domain images.

1.3 Unsupervised Domain Adaptation

Unsupervised Domain Adaptation is a learning framework to transfer knowledge learned from source domains with a large number of annotated training examples to target domains with unlabeled data, to solve the problem that traditional supervised learning requires a large number of manual annotations.

UDA can be achieved in input space, feature space, and output space. In input space, UDA aims for cross-domain resemblance in image appearance by mitigating low-level image statistics. It's often achieved by style transfer or image translation to close the marginal distributions of source and target domains. However, it tends to hurt semantic consistency and content preservation without good regularization constraints, e.g. objects of one category may be translated to a different object category. In addtion, most of UDA methods involve complicated training processes with a number of hyper-parameters and losses.

2 I2I Translation with CycleGAN

2.1 CycleGAN[3]

CycleGAN has two mapping functions $G : (X \rightarrow Y)$ and $F : (Y \rightarrow X)$ and two adversarial discriminators D_Y and D_X. D_Y encourages G to translate X to outputs indistinguishable from domain Y, and vice versa for D_X and F. To further regularize the mappings, it involves two cycle consistency losses including a forward cycle-consistency loss: $x \rightarrow G(x) \rightarrow F(G(x)) \approx x$, and a backward cycle-consistency loss: $y \rightarrow F(y) \rightarrow G(F(y)) \approx y$ as shown in Figure 2. CycleGAN trains G and F with a structural assumption: The mappings between the two domains are bijections, so $G : (X \rightarrow Y)$ and $F : (Y \rightarrow X)$ should be inverses of each other.

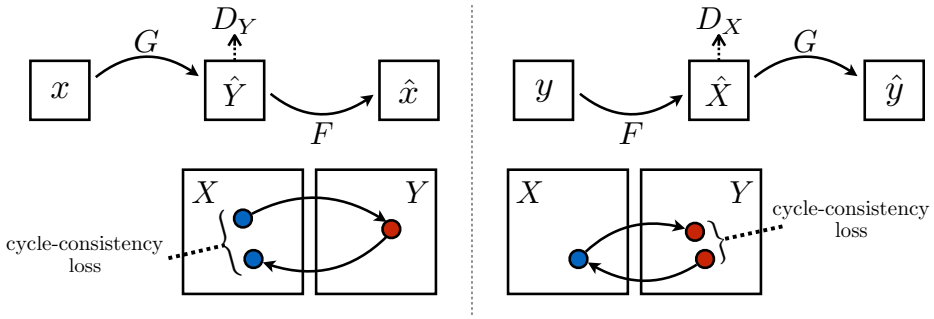

Figure 2: Cycle consistency loss

CycleGAN use the network architecture in [1] whose work has shown impressive results. The overall loss function consists of three parts, adversarial loss G, adversarial loss D, and cycle consistency Loss. To minimize the adversarial loss ensures that the pictures generated by the generator are more and more realistic (the style is more and more like another type of pictures). And to minimize the cycle consistency loss ensures that the content of the image generated by the generator is roughly unchanged.

$$\mathscr{L}(G,F,D_X,D_Y) = \mathscr{L}_{GAN}(G,D_Y,X,Y) + \mathscr{L}_{GAN}(F,D_X,Y,X) + \lambda \mathscr{L}_{cyc}(G,F) \tag{1}$$

2.2 Experiment and results

In the section of experiment, we attempt to implement the model of CycleGAN. We take GTA5 dataset and Cityscapes dataset as experimental datasets, and try to transfer GTA5 style to Cityscapes images and Cityscapes style to GTA5 images. In order to investigate the influence of the experimental parameters on the final results, we conducted several experiments, and the experimental environment is:

GPU: PG500-216 (V100-32GB); CPU: E5-2678 v3; Video memory: 46GB

Due to the huge amount of data (more than 60GB), our computing power and equipment cannot learn all the original images. Therefore, we first tried the training of small-scale samples, and selected 300 images in part1 and Cityscapes datasets of GTA5 dataset respectively (it is observed that the similarity of image scene is high) for the first experiment. All the selected samples are resized to 256×256. The experimental parameters are as follows:

batch_size is set to 2, epochs is 200, the initial learning rate is 0.0002 and reduced to 0 in a decreasing way.

Figure 3: The images in the first row are from GTA5 dataset, and those in the second row are the Cityscapes style transferred GTA5 images generated by the generator

It can be seen in the Figure 3 that the effect of the experiment is poor, especially in the front part of the cars. Many generated pictures do not have a clear boundary of the front of the car, and the texture of the picture is not good, though the tone of the picture is changed, the details are not in place. We speculate this is due to the randomness of the samples, because the number of photos containing the front of the car is limited during training, the discriminator cannot judge whether the front of the car is generated well or not. In addition, in the case of a small number of samples, the picture also appears to add extra objects (like trees, high-rise buildings and etc.) in the background blank, as the Figure 4 indicates:

It can be seen that many background objects that are not in the original image appear in the generated image. We speculate that this is also caused by the uneven distribution of samples when selecting Cityscapes data.

For the second experiment, the experimental environment is as follows:

GPU: RTX A6000; CPU: AMD EPYC 7742 ; Video memory: 48GB

The experimental parameters are as follows: batch_size is set to 8, epochs is 200, and the initial

Figure 4: The arrangement of displayed images are the same with the Figure 3. Some extra objects appear in the transferred images.

learning rate is 0.0002 and reduced to 0 in a decreasing manner.We selected 2250 images numbered 00001-02250 as the source domain in the GTA5 dataset, and took the images of cities *bochum*, *Bremen*, *cologne*, *Darmstadt*, *dusseldorf*, *Erfurt*, *hamburg*, *hanover*, *jena*, *krefeld*, *Monchengladbach*, and *strasbourg* in Cityscapes (2276 images in total) as the target domain, and the images are cropped to 256*256 pixels before input to the network.

During this experiment, which takes about 12 hours to run, we use the visdom library to visualize the process of training data, resulting in the following chart of loss curves:

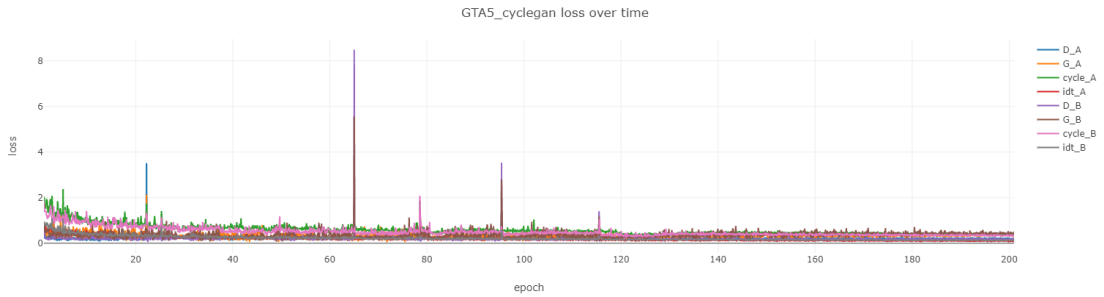

Figure 5: The loss curve during training. The loss functions converge rapidly after epoch 40.

It could be seen in the Figure 5 that different loss functions drop quickly and converge after epoch 40. Besides, it is obviously that the loss functions jump to a very high value occasionally, this is possibly due to the disturb of noise samples which leads to gradient explosion. These heap points gradually disappear, which indicates the good fit to the datasets of the model.

Here is the result of transferring the GTA5 style to the Cityscapes image:

Figure 6: The images in the first row are from Cityscapes dataset, and those in the second row are generated by the generator after style transfer to GTA5.

The Figure 6 shows that the details in the images are very clear, and the processing of the position of headstock has been significantly improved compared with the first experiment. In the blank space of the background, the generator reasonably fill it with white clouds and blue sky.

2.3 Constraints & Solutions

2.3.1 Constraints

CycleGAN performs well in so many cases, but it does not work in all scenarios. In Zhu's[3] original paper, they point out several shortcomings of CycleGAN, for example, the model is not really sensitive to geometric changes in the images. In addition, it may fail in the process of transfer due to the difference of the distribution characteristic of the training datasets. Apart from the defects mentioned by the authors themselves, CycleGAN is a one-to-one mapping model, which just links an input image with a single output image.

In our experiments, these shortcomings are also reflected. The pair of cycled gan is not equivalent effective: the style transfer of images from GTA5 to Cityscapes does not perform well, while that from Cityscapes to GTA5 is reasonable. We could observe in the Figure 6 that the color and texture changes processed by the model look good and make the transfer successful.

However, in the Figure 3, when it comes to the geometric changes, the front of the car is nearly eliminated but do not completely disappear, which make it unreasonable in vision. Besides, in the Figure 4, the distribution of characteristic of GTA5 and Cityscapes are different. There are many border trees and high-rise buildings in Cityscapes dataset, thus when transfering the style of Cityscapes to GTA5, there may occur some logically incorrect border trees and buildings in the background, which look strange. We could also explain the phenomenon by the one-to-one mapping character of CycleGAN, which means the image in GTA5 dataset could not learn the proper characteristic from its paired image in Cityscapes.

In addition, the output result of CycleGAN cannot restore the original image's color in few cases, which means color distortion. If the color style of the target domain and the source domain is various, CycleGAN cannot express the color style of the source data clearly. We think the reason for this phenomenon may be: the generator is trained on a large amount of data, and the color of the generated

picture or style is likely to be the average color of the training set data, therefore, for the special cases with different color styles, the pertinence is not strong enough, so the original color is converted into an average tone.

2.3.2 Solutions

After reading other papers, we found that the problem of different styles, textures and geometric features between two domains can be solved by attention mechanism and AdaLIN (Adaptive Layer-Instance Normalization)[6]. This article proposes the concept of auxiliary classifier, which is a classifier that can generate attention maps. If we add this classifier to CycleGAN during the process of unsupervised learning, it can guide the translation to focus on more important regions and ignore minor regions by distinguishing between source and target domains.At the same time, this paper proposes a new normalization function AdaLIN, which can help the attention-guided model learn and control the amount of texture variation without changing the model structure or hyperparameters, which may also benefit CycleGAN a lot.

To solve the problem of poor learning effect caused by the large gap between the two domains, a model called Augmented CycleGAN is mentioned in the paper by Amjad Almahairi et al[4]. His goal is to make the learning of CycleGAN The process can be extended to the multi-map level by adding random noise terms to the input. In CycleGAN network, a many-to-many mapping is learned by looping over the original domain with augmented auxiliary latent space, by marginalizing auxiliary variables, we can model many-to-many mappings between domains.

3 Unsupervised Domain Adaptation via I2I Translation

3.1 CyCADA[5]

In CyCADA, by training the model on multiple loss functions, the feature-level and pixel-level alignment is carried out at the same time. At the same time, compared with the previous adversarial adaptation methods, it pays more attention to preserving the semantic information in the data during the alignment process, while retaining in the alignment process. Semantic information is similar to previous methods, which takes into account the categorical information of the data.

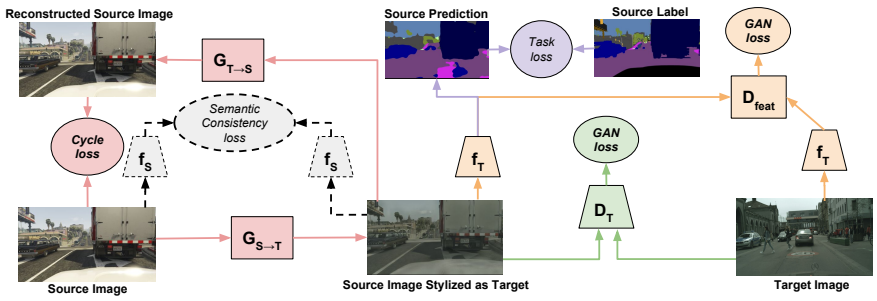

Figure 7: The overall structure of CyCADA

The network structure of CyCADA can be seen in Figure 7. By performing pixel-level reconstruction loss terms and semantic loss terms on the model to enforce consistency in local and global distribution structures, the reconstruction loss can encourage the model to preserve local feature information in inter-domain translation, while the semantic loss Semantic consistency can be enforced.

3.2 Experiment and results

In this part of the experiment, we investigate two segmentation models, the source-only model and the domain adaptive semantic segmentation model. The experimental environment is the same as in the second section.

Task1: The source-only model

First, we explore the source-only semantic segmentation model. With the image translation model from section 2, we use the GTA5 image and GTA5 label of the source domain for training, and evaluate the trained model on the Cityscapes dataset. The whole experiment process is shown in Figure 8.

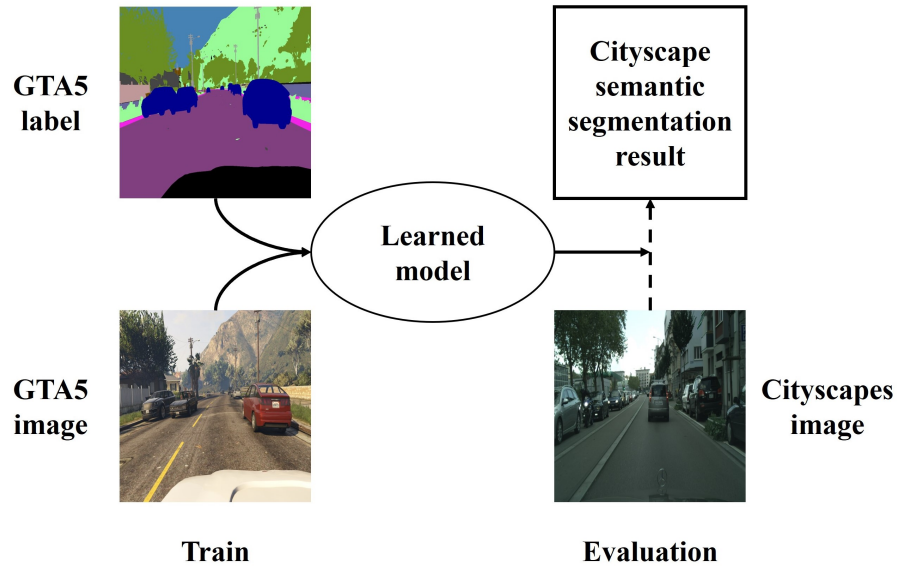

Figure 8: The source-only model

Then we put the model on the server for training, the experiment is carried out for 200 epochs, and the learning process curve is recorded as shown in Figure 9. It can be seen that convergence is reached after about 50 epochs and remains relatively stable during subsequent training.

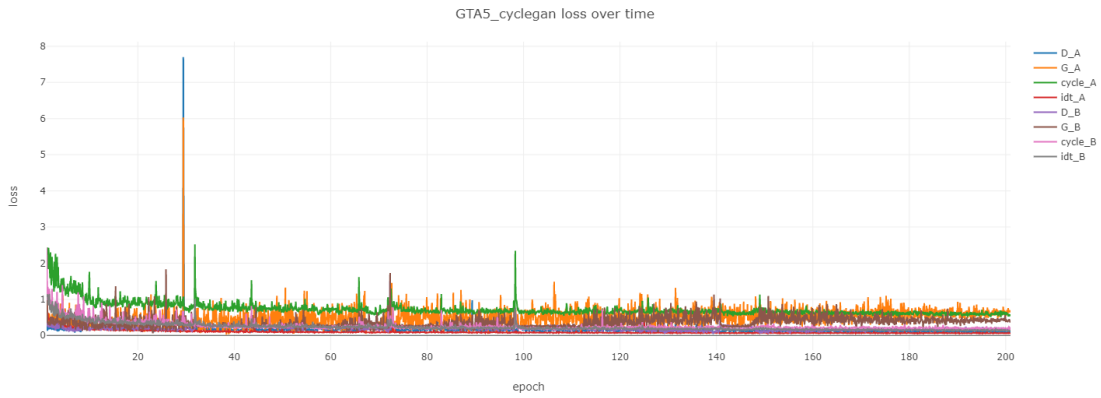

Figure 9: The loss curve during training.

As can be seen from the semantic segmentation results of the source-only model in Figure 10, the segmentation results in the second row are quite different from the labels of the Cityscapes dataset, such as the streets and tree outlines in the first column, the outlines of houses in the second and third columns are not well described. In addition, there are many identical objects that are misjudged as two or more objects, such as the houses in the third and fourth columns are confused with objects such as trees and cars, and the street lighting and shadows in the fourth column are are divided. Furthermore, none of the nearby cars are described in the four generated segmentation results.

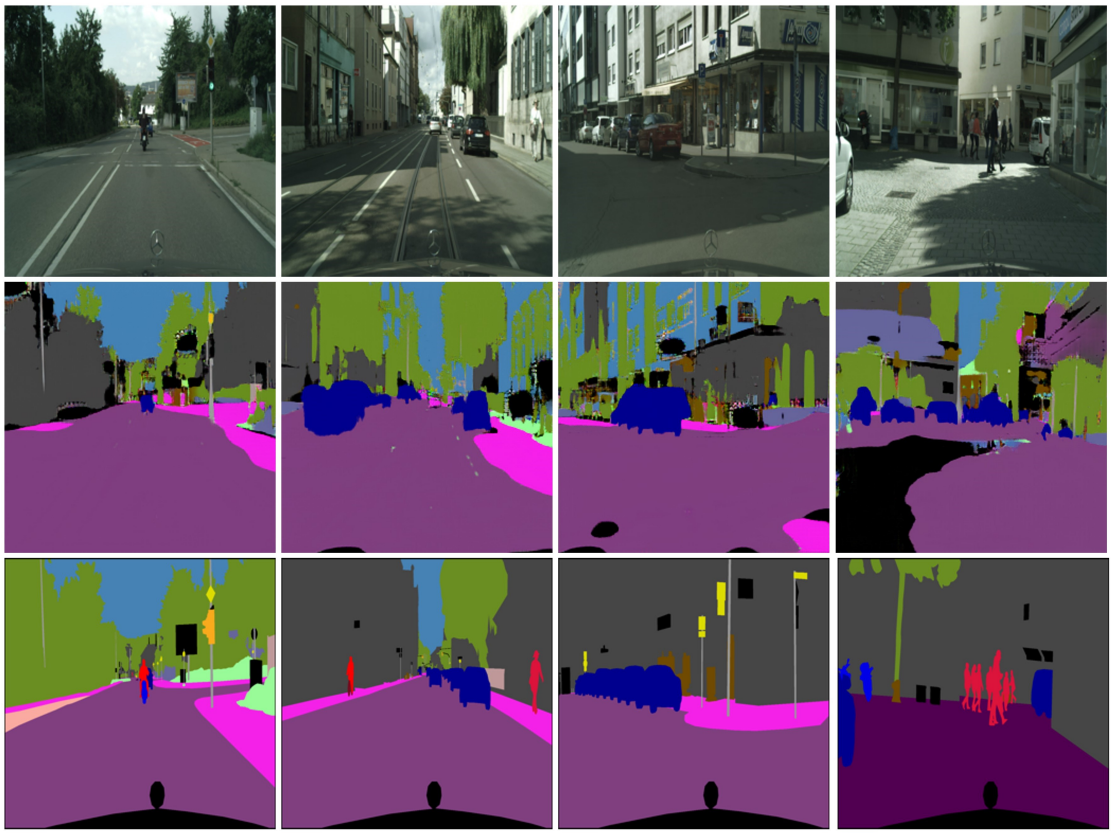

Figure 10: Semantic segmentation results. The images of the first row are from Cityscapes data, and the second row are generated semantic segmentation results, and the third row images are Cityscapes data label

Task2: The domain adaptive semantic segmentation model

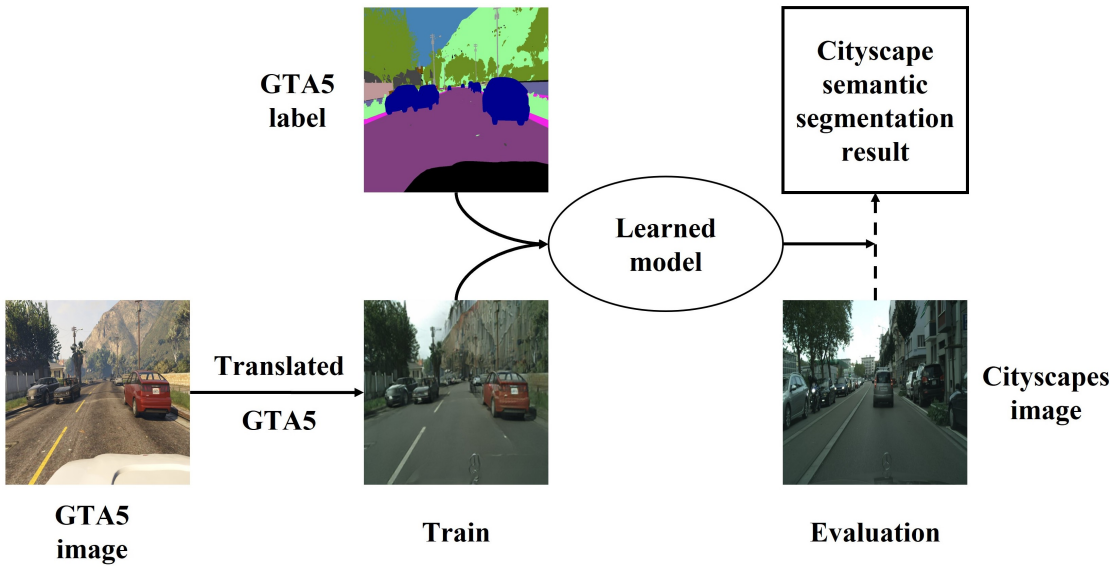

Figure 11: The domain adaptive semantic segmentation model

Then, we investigate the domain adaptive semantic segmentation model. In the task requirements, we need to implement UDA via input-space alignment. We already know that the GTA5 dataset is the source domain, the Cityscapes dataset is the target domain, and in the work of section 2, the style transfer from the GTA5 dataset to the Cityscapes dataset has been completed, and the game scene has been converted

into a real-world scene. We have realised the adaptation from the source domain to the target domain. Therefore, here we use translated GTA5 data and GTA5 label for training, then evaluate the trained model on the Cityscapes dataset. The whole experiment process is shown in Figure 11.

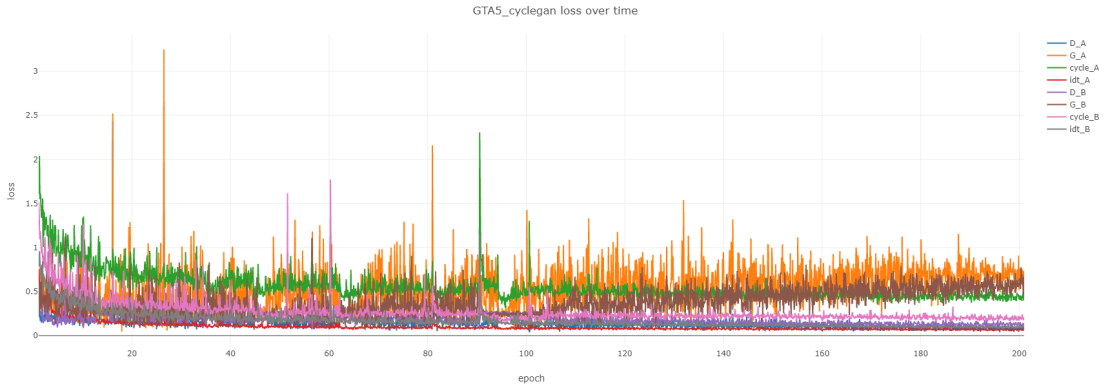

Figure 12: The loss curve during training.

Similarly, we select the GTA5 dataset part1 as the training set, train 200 epochs on the server, and use python visdom to supervise the training process. The entire training process is shown in the figure 12. It can be seen that the overall loss shows a downward trend during the overall training process. However, there are large fluctuations in the training process, and the generator loss increases slightly after the 100th epoch.

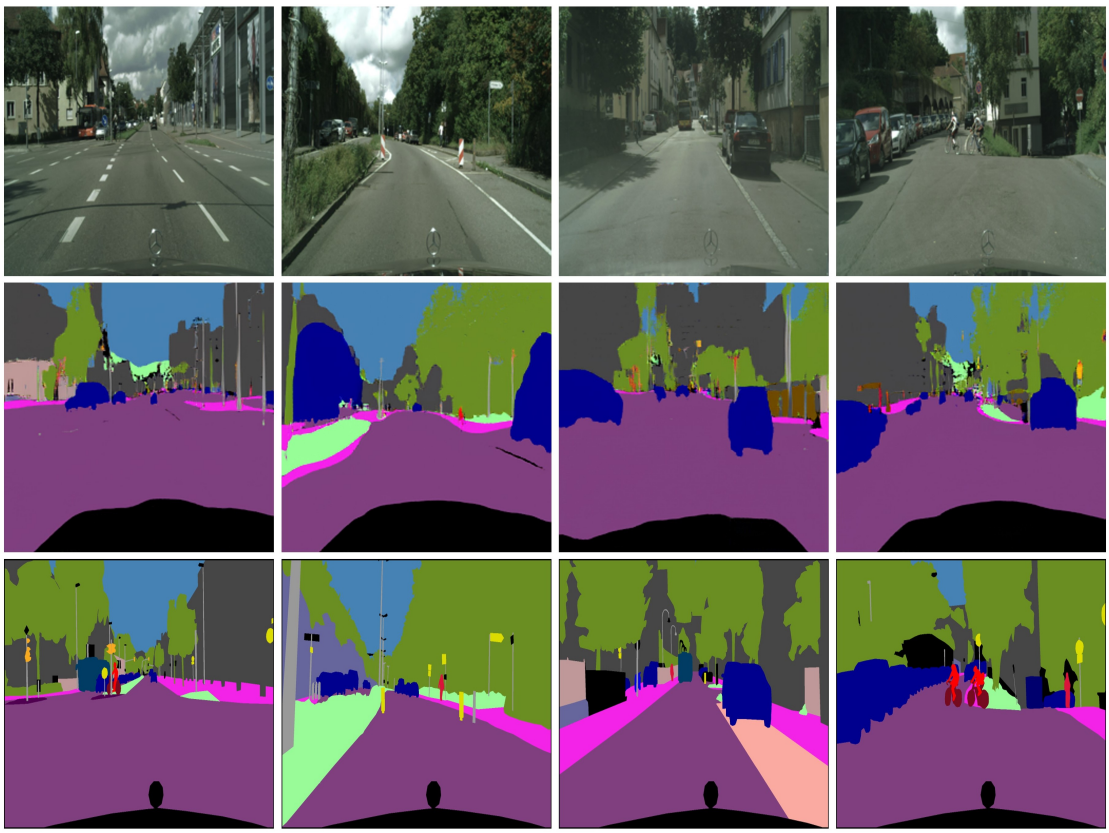

Figure 13: Semantic segmentation results. The images of the first row are from Cityscapes data, and the second row are generated semantic segmentation results, and the third row images are Cityscapes data label

Next, we test the trained model on the Cityscapes dataset, and the results are shown in Figure 13. It

can be seen that the general outline of most of the scene features in the Cityscapes dataset can be described, but the processing of the details is not very good. For example, in the first column of Cityscapes images, the rough outlines of streets, cars, and houses can be seen, but the boundaries between objects are not very clear. In addition, the semantic segmentation of objects with relatively small targets, such as people in the fourth column and cars in the distance, is not well processed, and images with large light changes such as tree shadows in the third column are treated as cars.

3.3 Comparison

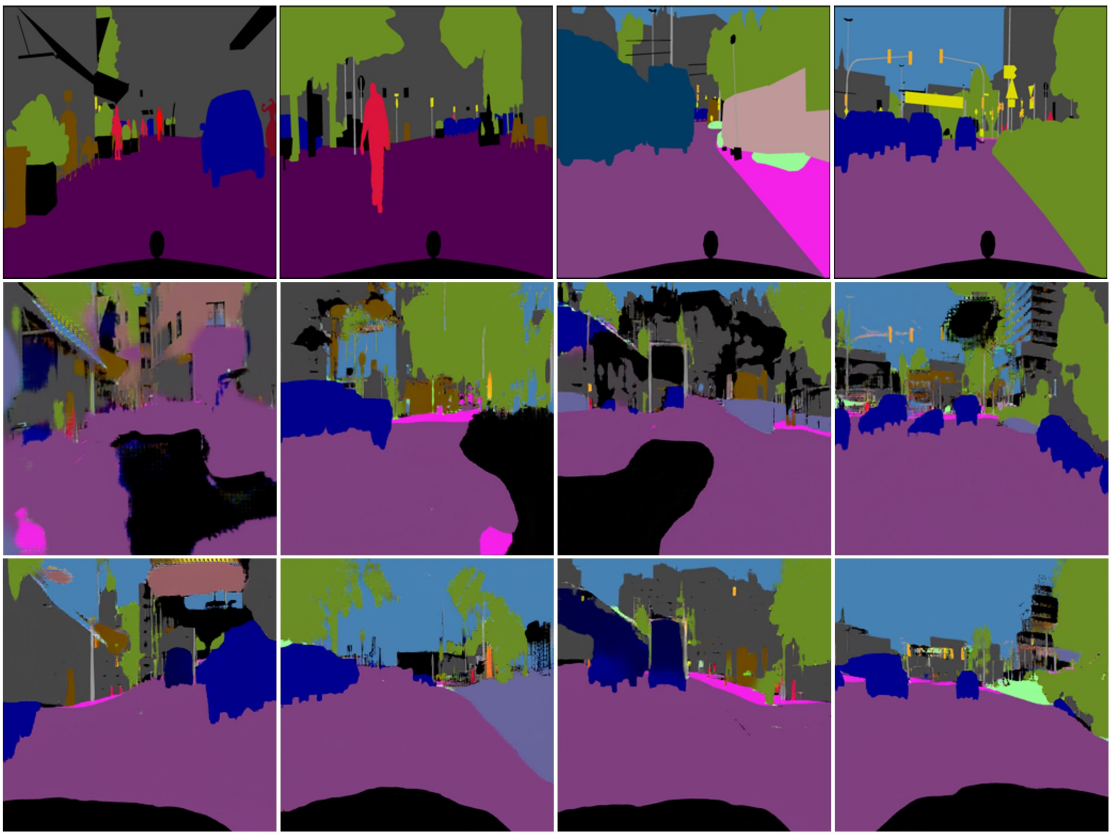

Figure 14: Semantic segmentation results comparison. The images of the first row are from Cityscapes data label, and the second row are source-only model results, and the third row images are domain adaptive model results

In our experiment, the source-only model performs bad in Cityscapes dataset. As the Figure 14 shows, the source-only model could not extract reasonable semantic information. For example, the semantic segmentation of the front of the car is completely distorted and become meaningless black block. Besides, the additional background semantic segmentation is added, which is significantly different from the corresponding Cityscapes data label.

In contrast with the source-only model, the domain adaptive model could remain the semantic information of the front of the car, and correctly express it. And the trees and the buildings are roughly similar with the Cityscapes data label, which indicates the translated GTA data really provides effective semantic information of Cityscapes during training process.

Apart from the points mentioned above, both models are not sensitive to the pedestrians. But in the source-only model results, the pedestrians are absolutely eliminated, or addressed as cars, while in the domain adaptive model results, a few pedestrians remain, which shows the domain adaptive model has a little better adaptability than source-only model.

Our experimental results illustrate that the source-only model just learn the feature of source domain, and only has the capability of extracting the semantics of source domain, thus it has poor generalization

ability in another domain (target domain). However, domain adaptation remains the semantic information of source domain during the alignment process. In other words, domain adaptation maintains the performance of the model in the target domain and the structure and content of the source domain are preserved, while the gap in the feature space between the source and target domains are reduced as much as possible. Therefore, the test result of domain adaptation model is obviously better than the result of source-only model.

4 Conclusion

In this assignment, we explore the field of image-to-image translation and unsupervised domain adaptation (UDA). First, we use the CycleGAN model to realise image translation, test the model on GTA5 and Cityscapes datasets, and analyze the results obtained from the experiments, and propose some possible improvement methods based on its constraints.

In addition, from CyCADA we have a good understanding of source-only model and domain adaptive semantic segmentation model. We implement UDA via input-space alignment, use the GTA5 dataset as the source domain and the Cityscapes dataset as the target domain to conduct two sets of experiments, and compare the results obtained by the two model experiments, and find that UDA via input-space alignment can effectively improve the semantic segmentation results.

References

[1] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. "Perceptual losses for real-time style transfer and super-resolution". In: *European conference on computer vision*. Springer. 2016, pp. 694–711.

[2] Phillip Isola et al. "Image-to-image translation with conditional adversarial networks". In: *Proceedings of the IEEE conference on computer vision and pattern recognition*. 2017, pp. 1125–1134.

[3] Jun-Yan Zhu et al. "Unpaired image-to-image translation using cycle-consistent adversarial networks". In: *Proceedings of the IEEE international conference on computer vision*. 2017, pp. 2223–2232.

[4] Amjad Almahairi et al. "Augmented CycleGAN: Learning Many-to-Many Mappings from Unpaired Data". In: *Proceedings of the 35th International Conference on Machine Learning*. Ed. by Jennifer Dy and Andreas Krause. Vol. 80. Proceedings of Machine Learning Research. PMLR, 2018, pp. 195–204. URL: https://proceedings.mlr.press/v80/almahairi18a.html.

[5] Judy Hoffman et al. "Cycada: Cycle-consistent adversarial domain adaptation". In: *International conference on machine learning*. Pmlr. 2018, pp. 1989–1998.

[6] Junho Kim et al. "U-gat-it: Unsupervised generative attentional networks with adaptive layer-instance normalization for image-to-image translation". In: *arXiv preprint arXiv:1907.10830* (2019).

[7] Ian Goodfellow et al. "Generative adversarial networks". In: *Communications of the ACM* 63.11 (2020), pp. 139–144.

[8] Pan Zhang et al. "Cross-domain correspondence learning for exemplar-based image translation". In: *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*. 2020, pp. 5143–5153.

A Image-to-image translation with CycleGAN

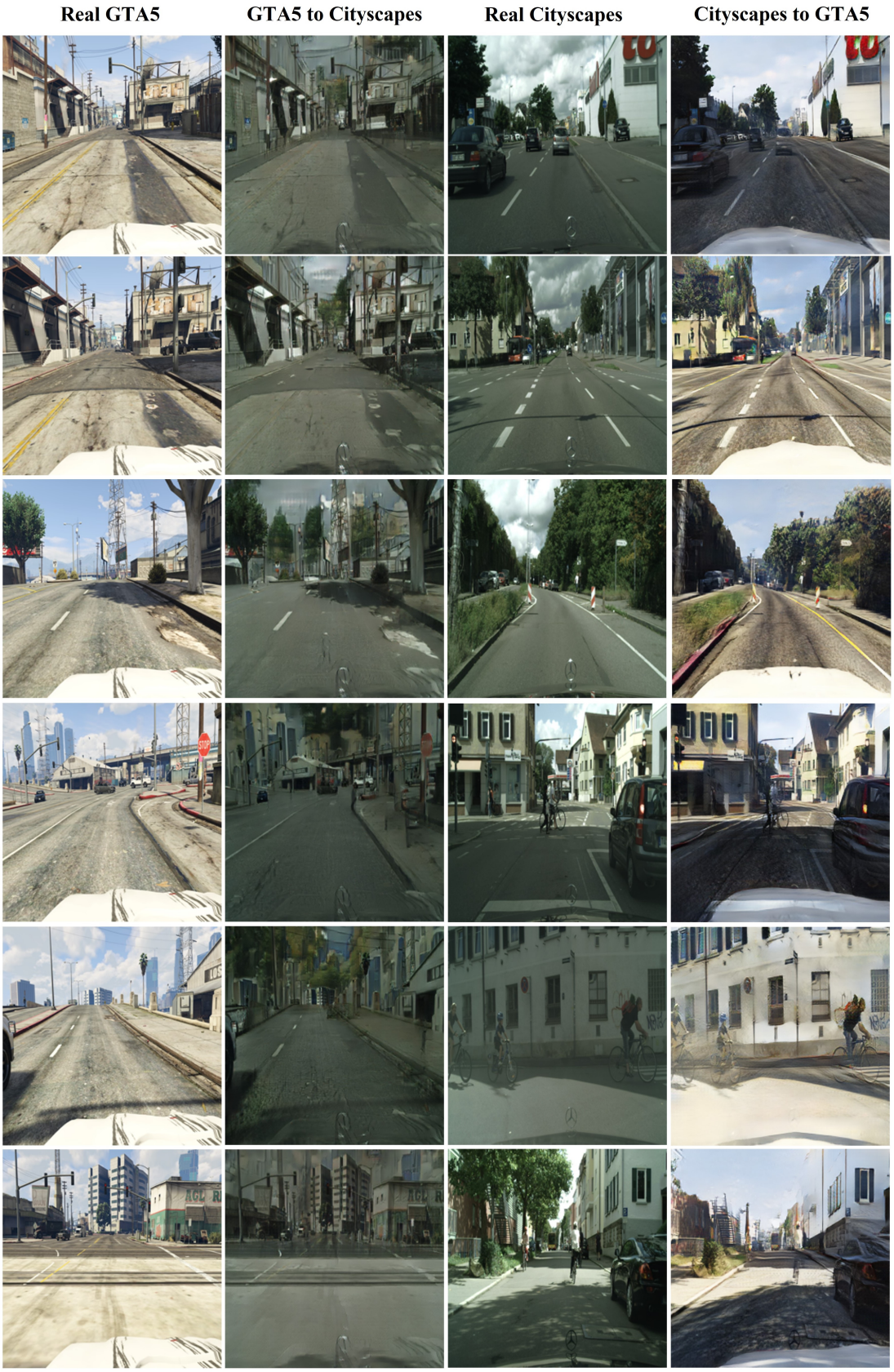

Figure 15: Additional image translation results

B Unsupervised domain adaptation via I2I translation

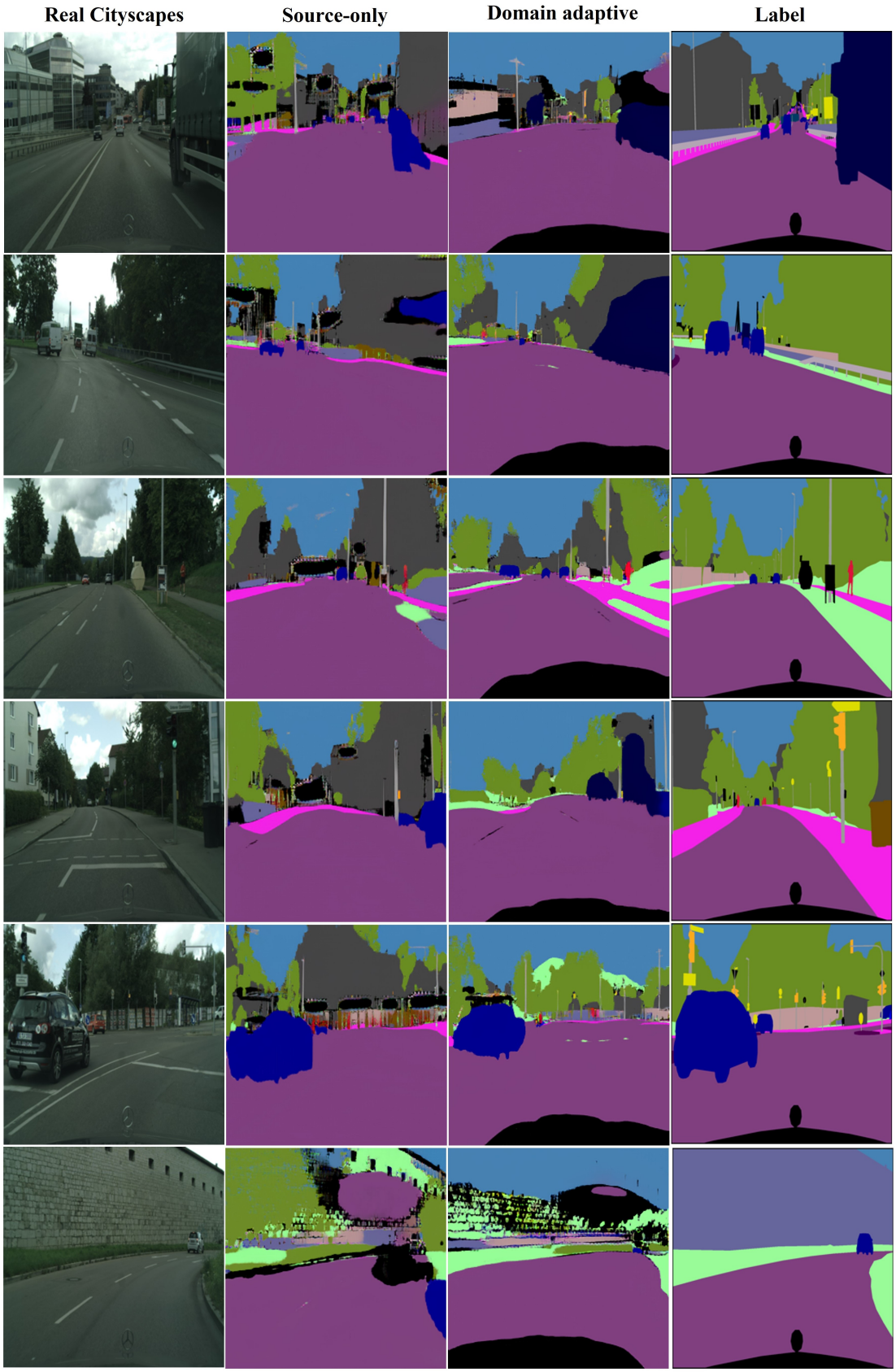

Figure 16: Additional UDA semantic segmentation results