Unsupervised Domain Adaptation Solving Class-Imbalance Problem on Semantic Segmentation

[1]SeungHoon Lee, [1]Donghyeon Cho, *Dongil Han
[1]bluesky3036@sju.ac.kr, [1]hyeon9698@sju.ac.kr, *dihan@sejong.ac.kr
[1,*]Sejong Univ. [1]Equal contribution *Corresponding Author

Abstract

Deep learning models, which have been intensively studied in recent years, require a large quantity of data and paired labels in order to perform well. However, manually obtaining and labeling substantial amounts of data is quite expensive. In addition, even if the model is trained with such collected data, the model may not operate as expected due to domain shift, and it is difficult to expect good performance for the class if the amount of data collected per class is insufficient. In this paper, we propose applying a target domain style to the source domain, expecting the trained model without a target label to operate smoothly in the target domain, and using the Likely Location copy-paste method to solve the class imbalance problem for minor classes.

Ⅰ. Introduction

To achieve good performance in semantic segmentation which is one of the most important tasks of computer vision, a sufficient amount of labeled training data for the target domain is essential. However, it is expensive to continuously collect new data on the target domain. Recently the field of Unsupervised Domain Adaptation (UDA) has been widely studied to solve this problem, which employs source domain data (e.g. GTA5) collected from virtual reality to perform well in the target domain. However, the GTA5 dataset [1] has a critical class-imbalance problem, shown in Fig. 2. As a consequence, CyCADA [2] utilizing the GTA5 dataset [1] demonstrates that the performance of a particular class that has critical class-imbalance problems is comparatively poor. In order to address this issue, DAFormer [3] improves the performance for minority classes by sampling images with minority classes more frequently and with a high probability in the training phase. However, in this case, the model mistakenly presumes that the minority classes being sampled on the same scene, which may cause the minority classes being learned to be overfit for a specific scene. This paper employed the copy-paste method [4] in which only the region for the minority class is extracted and attached to the image with the other scene in order to increase the scene's variation, as opposed to merely sampling images with the minority class more frequently. When a simple copy-paste method is used, however, edge information is highlighted if the colors of the image class area to be copied and the image area to be pasted are distinct. Due to the CNN architecture's sensitivity to edge information, the model can only be trained with simple edge information in this instance. Therefore, we applied Gaussian Blur to the edge region to prevent pasted objects' edge information from being highlighted. When using copy-paste, the environment of the copied image and the pasted image may differ. For instance, if you copy a bus from a bright daytime scene and paste it into a dark tunnel scene, the bus will stand out due to the difference in light conditions between the copied and pasted images. To make the pasted object's region blend more naturally into the source image, the AdaIN [5] method is applied to the copy-pasted source domain image. Not only that but also, AdaIN [5] style transfer can reduce the domain shift between the source and target domains by transferring the style of the target domain to the source domain. Finally, as stated in Copy-Paste [4] of the previous methodology, location information was not important when paste was performed. However, the semantic segmentation task in the driving scene has similar object regions (ex. vehicles in the center and buildings on the edges). Thus, we determined that it is essential to place objects in a position where they can be located [6]. Therefore, the Likely Location (LL) method was applied.

Ⅱ. Method

2.1. Style Transfer for Unsupervised Domain Adaptation

As one of the attributes of UDA, the model trained in the source domain should also perform well in the target domain. In order to accomplish these attributes, our approach employs the AdaIN [5] technique for performing style transfer using Instance Normalization [7]. Batch Normalization (BN) [8], commonly employed for normalization techniques, normalizes the mean and standard deviation for each individual feature channel on a mini-batch. In Instance Normalization, unlike BN layers [8], μ(x) and σ(x) are individually computed across spatial dimensions for each channel and sample, as shown in Eq. 1 and Eq. 2 below.

$$\mu_{nc}(x) = \frac{1}{HW}\sum_{h=1}^{H}\sum_{w=1}^{W} x_{nchw} \quad \textbf{(1)}$$

$$\sigma_{nc}(x) = \sqrt{\frac{1}{HW}\sum_{h=1}^{H}\sum_{w=1}^{W}(x_{nchw} - \mu_{nc}(x))^2 + \epsilon)} \quad \textbf{(2)}$$

Here *n* is the index of the batch image and *c* is the feature channel. As shown in Eq. 3, AdaIN [5] simply replaces the scale and shift parameters of the source image with μ(y) and σ(y) derived from the target image.

$$\mathrm{AdaIN}(x, y) = \sigma(y)(\frac{x - \mu(x)}{\sigma(x)}) + \mu(y) \quad \textbf{(3)}$$

Where *x* is an input image for content information of the source domain and *y* is an input image for style information of the target domain. However, if the method proposed in the existing AdaIN [5] is adopted as-is, style transfer occurs at the high-level layer, and we confirmed that semantic information is ruined significantly and gets noised. Fig. 1. (b). Since the model makes a prediction per pixel in semantic segmentation tasks, the loss of semantic information is critical. Accordingly, we minimize the loss of semantic information by applying AdaIN (Eq. 3) in the low-level layer, which has been proposed in Mixstyle [9].

$$T(c, s, \alpha) = g((1-\alpha)f(c) + \alpha\mathrm{AdaIN}(f(c), f(s))) \quad \textbf{(4)}$$

In addition, the loss of semantic information has been minimized by setting the α value to 0.1 in Eq. 4 proposed in AdaIN [5] for the weaker style transfer as shown in Fig. 1. (b). The results of these methods are shown in Table 1.

Fig. 1. Comparison with AdaIN in (a) source image (GTA5), (b) low-layer, (c) high-layer.

Table 1. Result Comparison between Source-only and AdaIN (Eq. 3) on high and low layers. Low-layer AdaIN performed a higher mIoU score than high-layer AdaIN.

	layer	mIoU
Source Only		31.36
AdaIN	high-layer(*relu-4*)	39.3
AdaIN	low-layer(*relu-1,2*)	**40.33**

2.2. Class-imbalance Problem

Looking solely at the mIoU value for each source class in Table 2, it is evident that the performance of certain classes is particularly poor. As depicted in Fig. 2, we figure that this is attributed to the class-imbalance problem in the GTA5 dataset [1]. To tackle this problem, DaFormer [3] uses the Rare Class Sampling (RCS) method. This solves the class-imbalance problem by obtaining the pixel count of labels per class for all training datasets and sampling the images containing classes with a small number of pixels more frequently. Another research method [4, 6] employs a copy-and-paste method.

2.2.1. Rare Class Sampling (RCS)

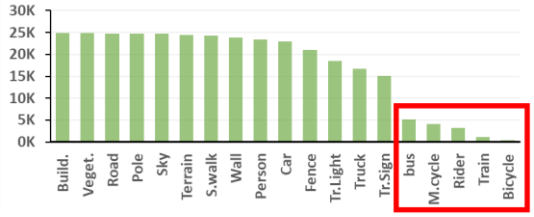

Fig. 2. The y-axis indicates the number of images containing specific classes in the GTA5 dataset. Note that five minor classes (bus, motorcycle, rider, train, and bicycle) have critical class imbalance problems.

Rare Class Sampling (RCS), a method for avoiding these problems in DaFormer [3], samples images frequently that have a low total number of labels per class. However, since this method samples full images containing the minor classes, we assume that the model will be overfitted on small scene variations of the minor classes. To increase the scene variation of the minor classes, we utilized the copy-and-paste technique.

2.2.2. Copy-Paste in Likely Location(LL)

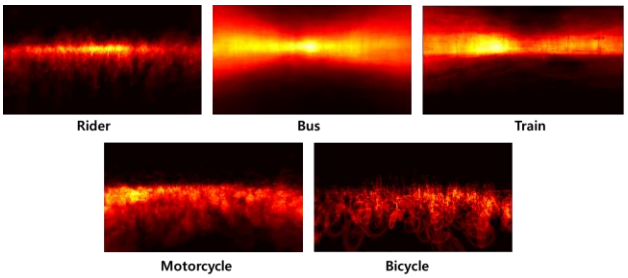

Fig. 3. Probability of class-wise Likely Location (LL) Heatmap

We focus on five classes (rider, bus, train, m.bike, and bike) which are suffering from data imbalance problems. Our method randomly extracts one image containing one of them, masks only the corresponding class label, and copy-paste. Then, the corresponding modifications are used to ground-truth annotations. Before proceeding, apply the Gaussian blur to the image to soften its edges.

Images of driving scenes depict essentially the same scene (ex. road in middle location, sidewalk on edge). Fig. 3 reveals that the majority of vehicles are located in the image's center, as opposed to its bottom. Therefore, we calculated the probability of appearing on the image by class (Fig. 3) and used these probability values to copy-paste at the Likely Location (LL). In Table 2, the performance of the LL and the copy-paste in a random location are compared, and it can be seen that the copy-paste method with the LL is superior in terms of both visual appearance and mIoU score.

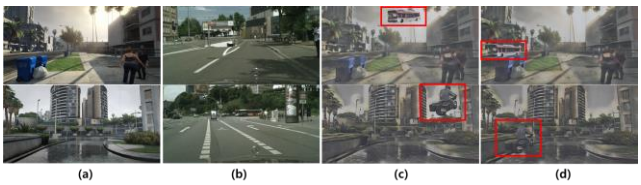

Fig. 4. Model results (a) source image (GTA5), (b) target image (CityScapes), (c) ours (w/o LL), (d) ours (w/ LL). The red boxes indicate the location of the copy-pasted object

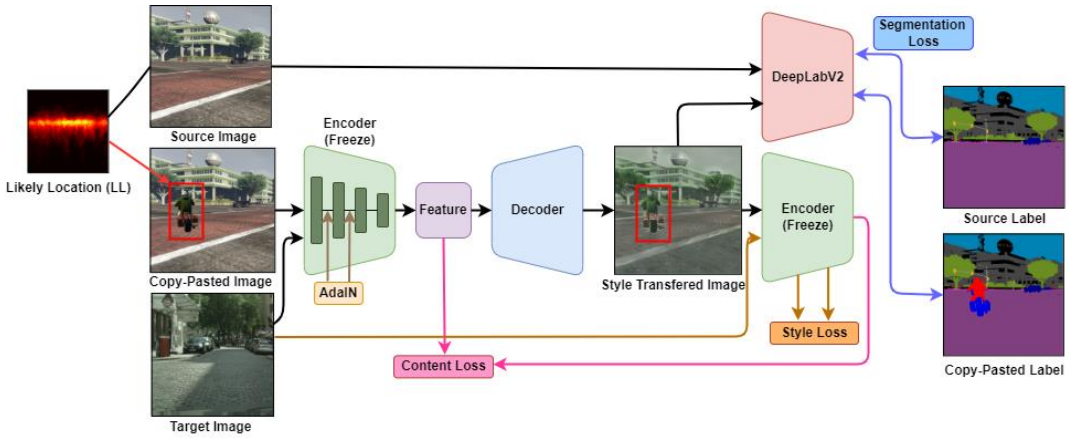

Fig. 5. Proposed Our model architecture on Training phase

Table 2. Result Comparison between source-only (row 1.), source-only+AdaIN (row 2.), RCS+AdaIN (row 3.), Ours without Likely Location (LL) (row 4.) and Ours with LL (row 5.). We report the performance of ours with LL (row 5.) shows a higher score on both the total mIoU score and the minor class-wise (Rider, Bus, Train, M.bike, Bike) mIoU score compared with RCS+AdaIN (row 3.).

	AdaIN	Road	S.walk	Build.	Wall	Fence	Pole	Tr.Light	Tr.Sign	Veget.	Terrain	Sky	Person	Rider	Car	Truck	Bus	Train	M.bike	Bike	mIoU
Source Only		22.15	15.14	59.07	7.69	16.24	17.78	26.21	16.62	76.03	8.70	74.24	48.15	20.46	35.72	14.85	12.15	1.67	17.99	22.82	31.36
Source Only	✓	33.50	19.56	65.04	18.10	15.15	23.58	30.93	28.29	77.04	27.00	78.60	54.72	21.81	40.61	15.98	15.91	2.74	20.69	31.49	36.97
Rare Class Sampling	✓	38.50	21.59	59.99	11.25	18.42	23.97	31.22	29.07	81.54	21.87	79.42	54.53	23.69	45.94	14.79	13.43	5.43	19.55	32.93	35.84
Ours (without LL)	✓	41.22	19.23	60.72	14.79	17.21	20.42	28.34	25.27	79.45	20.30	76.70	51.13	22.49	56.92	13.87	24.42	3.30	21.32	36.01	37.95
Ours (with LL)	✓	42.10	20.57	59.78	15.07	16.86	21.95	29.32	25.74	80.67	21.40	79.02	54.64	**23.89**	56.92	18.97	**28.05**	**5.69**	**22.75**	**38.40**	**40.33**

Ⅲ. Training and Evaluation

We trained our network using 24,966 images of the GTA5 dataset [1] for the source domain, 2,975 images of the CityScapes dataset [10] for style transfer, and 500 images of CityScapes images for evaluation on the target domain. We used the same encoder and decoder proposed in AdaIN [5] (note that only the decoder is trainable) and DeeplabV2 [11] for segmentation. For style and content loss, we used the same loss term proposed on AdaIN [5]. After the training phase, we only used DeeplabV2 in the evaluation phase.

Ⅳ. Conclusion

In this paper, we used two methods AdaIN on low-layer and Copy-paste on Likely Location and found that our model performed well on both minor class-wise mIoU and total mIoU compared with RCS and source-only shown in Table 2. From this experiment, we concluded that using AdaIN for style transfer in the lower layer performed well on the Unsupervised Domain Adaptation semantic segmentation task, and copy-paste on Likely Location (LL) worked well on both minor class-wise mIoU and total mIoU compared with Rare Class Sampling (RCS).

ACKNOWLEDGMENT

This work was supported by the National Research Foundation of Korea (NRF) grant funded by the Korea government (MSIT) (No. 2021R1F1A106168711), and by the Rural Development Administration joint research project (No. PJ015686).

REFERENCES

[1] Stepahn R. Richter, Vibhav Vineet, Stefan Roth and Vladlen Koltun: Playing for Data: Ground Truth from Computer Games. In ECCV, 2016

[2] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alyosha Efros, Trevor Darrell : CyCADA : Cycle-Consistent Adversarial Domain Adaptation. In ICLR, 2018

[3] Lukas Hoyer, Dengxin Dai, and Luc Van Gool. DAFormer: Improving network architectures and training strategies for domain-adaptive semantic segmentation. In CVPR, 2022.

[4] Golnaz Ghiasi, Yin Cui, Aravind Srinivas, Rui Qian, TsungYi Lin, Ekin D. Cubuk, Quoc V. Le, and Barret Zoph: Simple copy-paste is a strong data augmentation method for instance segmentation. In CVPR, 2021.

[5] Xun Huang and Serge Belongie: Arbitrary style transfer in real-time with adaptive instance normalization. In ICCV, 2017.

[6] Nikita Dvornik, Julien Mairal, and Cordelia Schmid: Modeling visual context is key to augmenting object detection datasets.In ECCV, 2018

[7] D. Ulyanov, A. Vedaldi, and V. Lempitsky. Improved texture networks: Maximizing quality and diversity in feed-forward stylization and texture synthesis. In CVPR, 2017

[8] S. Ioffe and C. Szegedy. Batch normalization: Accelerating deep network training by reducing internal covariate shift. In JMLR, 2015

[9] Kaiyang Zhou, Yongxin Yang, Yu Qiao and Tao Xiang: Domain Generalization with Mixstyle. In ICLR, 2021

[10] 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 CVPR, 2016

[11] 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. In IEEE Trans. 2017.