Real-time Domain Adaptation in Semantic Segmentation

Lorenzo Nikiforos
Politecnico di Torino
s317616@studenti.polito.it

Alessio Cappello
Politecnico di Torino
s309450@studenti.polito.it

Claudio Macaluso
Politecnico di Torino
s317149@studenti.polito.it

Abstract

Semantic segmentation has become widely diffused for different applications, especially in its real-time version. This task requires massive amounts of annotated data, which translates to human effort to label each pixel of the images. In this paper, we take a look at overcoming this problem by exploiting a synthetic dataset: this means that a Domain Adaptation framework has to be embraced and, more specifically, we focused on an adversarial approach. Different image transformation techniques can also be exploited, such as FDA, to further reduce the gap between the target and source domain. It's also possible to proceed in a self-supervised manner, letting a model generate target labels to use in the training process. This work represents a first step towards the right direction, but more expedients need to be taken to get impressive performance.

1 Introduction

Semantic segmentation's objective [2] is to categorize all the pixels of an image, assigning each of them a class label. It is a fundamental task for different applications, such as autonomous driving, disease diagnosis, etc. Convolutional Neural Networks (CNNs) have been recently employed to perform semantic segmentation, achieving notable results. One struggle with CNNs is finding the balance between performance and computational speed: a way to address this challenge has been proposed with the BiSeNet architecture [10]. One trouble with the whole process, instead, is related to the effort required to humanly annotate large amount of data: this strongly depends on the task and on how much is already available. So, what if not enough data are available? An approach that made its way to become an object of study is to use synthetic datasets, where data is computer generated and annotated, and then adapt the model to the domain of interest. In other terms, this can be easily reduced to a Domain Adaptation (DA) problem, where synthetic data represents the source part and the one we want to achieve acceptable performance on represents the target part. Different approaches can be followed to perform the task.
Domain adaptation algorithms, such as the one proposed in [7], can be implemented to enhance the alignment between the feature distributions of the two domains.
It is possible to further improve the performance via classical data augmentation techniques, such as color jitter, adjusting brightness, randomly cropping the images etc. Another common strategy is to bring closer the target and the source domains by transforming the images belonging to the source domain such that these match the style of the target ones: this can be achieved, for instance, via Fourier Domain Adaptation (FDA) [9], that consists in injecting the low-level frequencies of the target images into the source ones. After having trained the model with this technique, it is possible to generate pseudo-labels in a self-supervised way for the source images processed, and this has been proven to be beneficial.
In this paper, we propose:

- an implementation of BiSeNet for real-time semantic segmentation;
- an implementation of the unsupervised domain adaptation algorithm proposed in [7], with some slight variations;

- application of FDA to enhance the performance of the DA;
- use of pseudo-labels to further enhance the performance.

The source code is available at: https://github.com/AlessioCappello2/AML_Semantic_DA.

2 Related Work

2.1 Semantic Segmentation

Interest and research in semantic segmentation have been boosted by the rise of deep learning techniques: as an example, architectures realised to perform classification can be turned into fully convolutional networks and fine-tuned to perform semantic segmentation [2][3]. The increase in performance led to questions about how to make progress also in the real-time field: a high number of parameters and subsequent floating point operations contribute to the computational complexity. Some early works moved towards this direction, such as [2][4] which combined an encoder-decoder architecture with early downsampling, brought different improvements, as the one proposed in [2][6] to further speed up the process. Wanting to achieve high performance requires a large amount of annotated data, and this can't always be an option: for this reason, training models over synthetic datasets has become widely diffused, but it is needed to address the domain shift problem.

2.2 Domain Adaptation

Domain shift is a critical hurdle to overcome because a model trained on synthetic data may not generalize correctly to real data, resulting in decreased performance in real-world scenarios. This disparity can come from differences in noise levels, lighting conditions, backgrounds, object geometries, and various other features between synthetic and real data. Different algorithms and methods have been proposed to address the problem of domain shift: for example, Tuan-Hung *et al.*'s ADVENT [8] proposed to minimize an entropy loss and an adversarial loss jointly, or Tsai *et al.*'s algorithm [7] performs adversarial domain adaptation at multiple levels. Other techniques have been developed, that don't require additional training, such as Fourier Domain Adaptation [9], and match the style of the two domains' images.

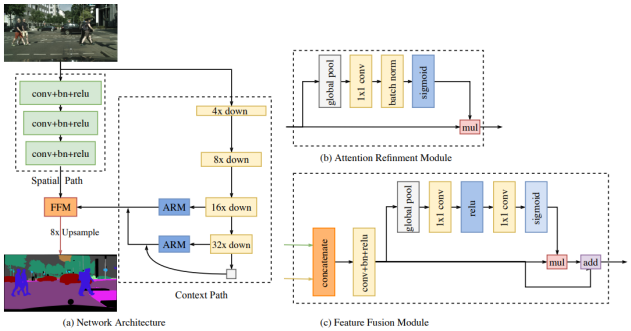

Figure 1: Overview of the BiSeNet architecture [10]. (a) Architecture (b) Attention Refinement Module (c) Feature Fusion Model.

3 Method

3.1 BiSeNet

Our baseline model is BiSeNet: the novel approach proposed in [10] is to decouple the demand for spatial information and receptive fields. Two main components are used to pursue these two needs simultaneously:

- *Spatial Path*: following this path leads to large feature maps that preserve the original input image size and encode rich spatial information, and it's based on three layers of convolution, batch normalization and ReLU;
- *Context Path*: this path provides large receptive fields, crucial for achieving significant performances, and it's based on a lightweight model with a global average pooling at the end to get a receptive field with global context information.

There are another two components that figure in the proposal: *Attention Refinement Module* (ARM), used to refine the features at different stages along the Context Path exploiting the global context information without any need of upsampling, and *Feature*

Fusion Module (FFM), used to join the features of the two paths, that are coming from different levels of features representation.

3.2 Unsupervised Adversarial Domain Adaptation

We used the framework proposed in [7] as our Adversarial Domain Adaptation network. This implementation is based on a Generative Adversarial Network (GAN) that aims to adapt pixel-level predictions (instead of features) between the source and target domain. This is beneficial for two reasons: 1) predictions are lower-dimensional than features, and 2) predictions themselves retain a good amount of spatial and local information. Since final predictions are used to perform DA, lower-level features may not be well adapted. For this reason, a multi-level strategy has been developed, extracting segmentation outputs at different network levels and feeding them to a specialized discriminator.
The framework consists of a segmentation network $\mathbf{G}$, used as a generator, and a discriminator $\mathbf{D}_i$ (one for each level). The training phase consists in:

- forwarding the source image (with annotation) to the segmentation network to optimize $\mathbf{G}$
- predicting the segmentation softmax output for the target image (without annotation)
- use the two predictions as input in the discriminator

The discriminator uses the segmentation softmax output $P = \mathbf{G}(I) \in \mathbb{R}^{H \times W \times C}$ as input and a cross-entropy loss for classes $\{source, target\}$. The loss is defined as:

$$\mathcal{L}_d = -\sum_{h,w}(1-z)\log(\mathbf{D}(P)^{(h,w,0)}) + z\log(\mathbf{D}(P)^{(h,w,1)}) \quad (1)$$

where $z = 0$ for target samples and $z = 1$ for source samples.
For the generator, we first define the segmentation loss as a cross-entropy loss in the source domain:

$$\mathcal{L}_{seg}(I_s) = -\sum_{h,w}\sum_{c \in C} Y_s^{(h,w,c)} \log(P_s^{(h,w,c)}) \quad (2)$$

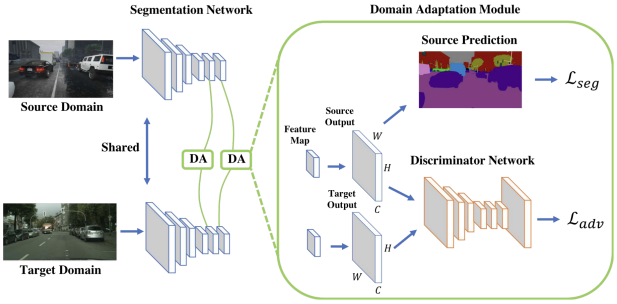

Figure 2: Overview of the Unsupervised Adversarial Domain Adaptation architecture in [6][7].

Then the adversarial loss adapts the predicted segmentation of target images to the distribution of source prediction:

$$\mathcal{L}_{adv}(I_t) = -\sum \log(\mathbf{D}(P_t)^{(h,w,1)}) \quad (3)$$

Finally, the generator uses both losses combined:

$$\mathcal{L}(I_s, I_t) = \mathcal{L}_{seg}(I_s) + \lambda_{adv}\mathcal{L}_{adv}(I_t) \quad (4)$$

3.3 Fourier Domain Adaptation

This technique aims to reduce the domain shift by bringing closer the source images to the target ones: the idea behind it is that even minor changes among the low-level features lead to deterioration in performance. Fourier Domain Adaptation (FDA) [9] aligns the low-level statistics between the two domains by replacing the low-level frequencies in the source image with the target ones while maintaining its semantic meaning. It's possible to do so by computing the Fast Fourier Transform (FFT), performing the replacement and recomputing the source image in the target style by computing the Inverse Fast Fourier Transform (iFFT).
We denote with: $\mathcal{F}$ the FFT, $\mathcal{F}^A$ its amplitude, $\mathcal{F}^P$ its phase and $\mathcal{F}^{-1}$ its inverse.
The following represents the mask M_β, zero-filled except for the central region with $\beta \in (0, 1)$:

$$M_\beta(h,w) = \mathbb{1}_{(h,w) \in [-\beta H:\beta H, -\beta W:\beta W]} \quad (5)$$

Figure 3: Left: Source image. Center: Target image. Right: Source image after FDA application.

In this way, it is possible to formalize the FDA with two randomly sampled images x^t target and x^s source, obtaining the $x^{s \to t}$ source image that matches the target style:

$$x^{s \to t} = \mathcal{F}^{-1}([M_\beta \circ \mathcal{F}^A(x^t) + (1 - M_\beta) \circ \mathcal{F}^A(x^s), \mathcal{F}^P(x^s)]) \quad (6)$$

The hyper-parameter β can be fine-tuned: the higher its value the more the amount of target style transferred. An example with $\beta = 0.01$ is provided in Figure 3.

3.4 Self-Supervised Learning

Another approach to the DA problem is Self-Supervised Learning (SSL): in this way, it's possible to generate highly-confident pseudo-labels, predicted as if they were the ground truth. The model must be trained before starting to emit such pseudo-labels: at the beginning, the configuration is the usual (unsupervised), while after having trained it to get acceptable performance, it is possible to generate pseudo-labels for the target domain. However, since working with a single model would be self-referential, using more than one model has a regularization effect in the learning process [9]. That said, for each target image we take the average of the softmax output given by each model.
The SSL term is integrated into the loss (we denote with $\hat{Y}^t$ the pseudo-labels for the target images):

$$\mathcal{L}(I_s, I_t, Y_s, \hat{Y}^t) = \mathcal{L}_{seg}(I_s, Y_s) + \lambda_{adv}\mathcal{L}_{adv}(I_t) + \mathcal{L}_{seg}(I_t, \hat{Y}^t) \quad (7)$$

4 Experiments

4.1 Datasets

In our analysis, the target dataset is a subset of *Cityscapes* [1]. There are respectively 1572 and 500 images (2048x1024) for training and validation, provided with their annotated labels. The source dataset is a subset of *GTA5* [5]. The dataset, composed of 2500 images (1954x1052) and computer-generated labels, was provided without distinction in training and validation, so we opted for an 80/20 split. Both test sets have also been used for validation, and only the 19 classes shared between the two datasets have been considered through the evaluation process.

4.2 Implementation details

As previously said, our baseline model is BiSeNet, using as backbone STDCNet813. For our experiments with augmentation techniques, we defined for each sample a probability of 0.5 to be processed with some of them. For the unsupervised adversarial domain adaptation, we deployed a Fully Convolutional Discriminator (FCD), with 5 convolutional layers (number of channels per each layer: 64, 128, 256, 512, 1), kernel size equal to 4x4, stride 2 and padding 1. The activation function used after each convolution is the Leaky ReLU with a negative slope of 0.2. Our predefined settings for the experiments, if not specified otherwise, will be:

- number of epochs equal to 50;
- mini-batch Stochastic Gradient Descent (SGD) with batch size equal to 8, momentum 0.9, weight decay $1e^{-4}$, initial learning rate $1e^{-2}$ which decreases according to the poly strategy: for each iteration the value of the learning rate is $\text{init\_lr} * \left(\frac{1-\text{iter}}{\text{max\_iter}}\right)^{\text{power}}$;
- random cropping on images to reduce their size to 1024x512;
- normalization for images, with mean (0.485, 0.456, 0.406) and standard deviation (0.229, 0.224, 0.225).

Setting	road	sidewalk	building	wall	fence	pole	light	sign	vegetation	terrain	sky	person	rider	car	truck	bus	train	motorcycle	bicycle	**mIoU %**
$\beta = 0.01$	82.4	23.0	68.9	12.1	6.3	28.5	5.1	8.2	75.9	14.1	47.8	43.5	5.5	66.7	7.0	11.5	3.2	1.4	0.0	**26.9**
$\beta = 0.05$	79.8	17.1	68.2	6.7	2.1	24.0	7.7	10.6	70.9	11.9	55.5	39.9	2.1	58.4	12.8	9.1	3.3	1.2	0.0	**25.3**
MBT	82.2	19.1	71.1	11.1	2.7	27.1	6.1	10.1	75.6	15.0	55.0	43.8	3.6	67.9	13.1	13.2	5.3	1.1	0.0	**27.5**

Table 1: IoU values for each class and mIoU in the FDA setting.

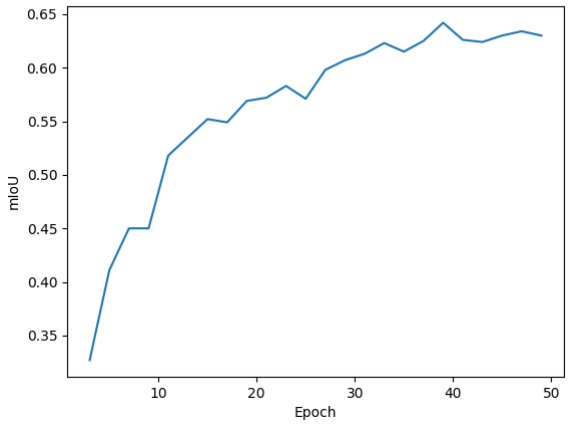

Figure 4: mIoU for the BiSeNet Cityscapes training.

4.3 Baseline evaluation

We trained BiSeNet on Cityscapes to define the upper bound for all our experiments. It's possible to see in Figure 4 how it reaches a high value of mIoU in some epochs. Then, we trained BiSeNet on GTA5 and evaluated on the Cityscapes test set to notice the effects of the domain shift. The first attempt we made to enhance the performance was to use some combination of data augmentation techniques; we got an improvement of a few percentage points with the augmentation pipeline constituted by (contrast, saturation, and horizontal flip). The results of this phase are reported in Table 2.

4.4 Unsupervised Adversarial Domain Adaptation

Moving to the DA framework, we implemented the approach using BiSeNet as segmentation model (generator G) and only one discriminator D. We first developed a model with three discriminators, but due to our limited computational resources, moved to use only one discriminator. Furthermore, according to our consideration that the datasets are not huge, we wouldn't have noticed impressive improvements in training the model with three discriminators.
We used the Adam optimizer to train the discriminator with an initial learning rate equal to $1e^{-4}$, betas equal to (0.9, 0.99) and the same poly strategy previously seen. Since we are working in a single-level setting, we set λ_{adv} to $1e^{-3}$ as advised in [7]. We trained using a batch size equal to 2. The results are reported in the fourth row of Table 2: the approach effectively leads to some performance improvements (for instance, the 'train' class has a non-zero value of IoU).

4.5 Extensions

We tried to push the performance even further by applying some extensions to the adversarial framework: we implemented FDA and then tried to enhance it with the generation and use of pseudo-labels for the target domain via self-supervised learning.

4.5.1 Fourier Domain Adaptation

The whole training process is the same; the only thing to take into account is that, according to [9], it's possible to notice worse performance considering just single models. To overcome this possibility, we trained two models with two different values of β (0.01 and 0.05) and then evaluated their fusion (Multi-Band Transfer). The two models were trained for a lower number of epochs because we noticed an earlier improvement and we evaluated the fusion over their best checkpoints (for both models at epoch 25). We report the results for the two models and their fusion in Table 1 to point out how the fusion effectively improves performance. The fusion is also reported in

Setting	road	sidewalk	building	wall	fence	pole	light	sign	vegetation	terrain	sky	person	rider	car	truck	bus	train	motorcycle	bicycle	**mIoU %**
CS (upper bound)	97.0	78.6	88.9	42.3	46.2	47.4	46.0	64.9	90.3	57.5	91.9	73.8	46.5	91.6	44.9	54.7	34.4	38.9	69.7	**63.4**
GTA → *CS*	37.3	2.1	57.4	2.4	2.2	9.8	9.1	9.9	70.5	2.6	48.5	30.8	1.9	19.2	1.5	1.9	0.0	0.0	0.0	**16.2**
GTA+augm.→ *CS*	80.9	6.6	68.1	6.8	7.3	0.3	14.6	4.9	76.8	13.4	56.2	21.8	0.0	55.3	10.2	7.9	0.0	0.0	0.0	**22.6**
Adversarial	78.3	29.7	71.0	8.9	1.8	26.2	9.7	14.1	74.2	6.9	64.4	33.9	0.1	60.3	7.5	10.1	3.8	0.0	0.0	**26.4**
Adv.+FDA	82.2	19.1	71.1	11.1	2.7	27.1	6.1	10.1	75.6	15.0	55.0	43.8	3.6	67.9	13.1	13.2	5.3	1.1	0.0	**27.5**
Adv.+FDA+SSL	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	**23.3**

Table 2: IoU values for each class and mIoU. IoU values for Adversarial+FDA+SSL are not reported.

Figure 5: Left: Target image. Center: Ground truth label. Right: Generated pseudo-label.

the fifth row of Table 2: we gained one percentage point on mIoU and only the 'bicycle' class now has a zero value.

4.5.2 FDA + Self-Supervised Learning

Exploiting the two checkpoints obtained for the FDA, we used them to generate pseudo-labels for the target domain: we gave as input the Cityscapes validation set to obtain labels generated in a self-supervised way (an example is provided in Figure 5). We didn't get the results we hoped: we tried different options by adjusting the learning rates, but no improvements were noticed. In Table 2 we reported the best value found for mIoU (using learning rates equal to half of their original value), but decided to not report the classes' values since it wasn't obtained any improvement. This could be explainable since we have a limited amount of data and the segmentation model can't accurately define high confidence areas.

5 Conclusions

Our work proves the effectiveness of the adversarial framework in going through the domain adaptation problem for semantic segmentation. We got overall fine results, considering the limited amount of data and computational resources. Surely, it's possible to further reduce the gap between synthetic and real images, probably using different segmentation models to generate more accurate pseudo-labels and/or combining different image transformation techniques.

References

[1] 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, 2016. 4

[2] Shijie Hao, Yuan Zhou, and Yanrong Guo. A brief survey on semantic segmentation with deep learning, 2020. 1, 2

[3] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation, 2015. 2

[4] Adam Paszke, Abhishek Chaurasia, Sangpil Kim, and Eugenio Culurciello. Enet: A deep neural network architecture for real-time semantic segmentation, 2016. 2

[5] Stephan R. Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games, 2018. 4

[6] Michael Treml, José Arjona-Medina, Thomas Unterthiner, Rupesh Durgesh, Felix Friedmann, Peter Schuberth, Andreas Mayr, Martin Heusel, Markus Hofmarcher, Michael Widrich, Bernhard Nessler, and Sepp Hochreiter. Speeding up semantic segmentation for autonomous driving, 2016. 2

[7] Yi-Hsuan Tsai, Wei-Chih Hung, Samuel Schulter, Kihyuk Sohn, Ming-Hsuan Yang, and Manmohan Chandraker. Learning to adapt structured output space for semantic segmentation, 2018. 1, 2, 3, 5

[8] Tuan-Hung Vu, Himalaya Jain, Maxime Bucher, Matthieu Cord, and Patrick Pérez. Advent: Adversarial entropy minimization for domain adaptation in semantic segmentation, 2018. 2

[9] Yanchao Yang and Stefano Soatto. Fda: Fourier domain adaptation for semantic segmentation, 2018. 1, 2, 3, 4, 5

[10] Changqian Yu, Jingbo Wang, Chao Peng, Changxin Gao, Gang Yu, and Nong Sang. Bisenet: Bilateral segmentation network for real-time semantic segmentation, 2018. 1, 2