Real-time Domain Adaptation in Semantic Segmentation

Claudio Tancredi
Politecnico di Torino
Turin, Italy
s292523@studenti.polito.it

Francesca Russo
Politecnico di Torino
Turin, Italy
s287935@studenti.polito.it

Matteo Moschelli
Politecnico di Torino
Turin, Italy
s290143@studenti.polito.it

***Abstract*—Semantic segmentation has become more and more used in many fields in computer vision. The problem is that it requires a large amount of pixel-wise labeled data that can be obtained with a big human effort. In this paper, we explore the task of Domain Adaptation applied to Real-time Semantic Segmentation Networks. We exploit the BiSeNet network, that is first trained on source data, and then used as generator in an Adversarial Domain Adptation algorithm. In the last steps, we compare two different techniques, FDA and LAB, applied on source data with the goal of reducing the domain gap between source data and target data. A SSL framework is also implemented and further enhanced with LAB transformation applied on source images. Experimental results show that the domain shift can be partially reduced, but more work is needed to actually close the gap between source and target domains.**

I. Introduction

Semantic segmentation's [1], [3] objective is to assign a semantic label, e.g., person, car, to each pixel of an image. It is useful in a wide range of applications, such as autonomous driving, robotic navigation, etc. Recently, convolutional neural networks (CNNs) have been employed for this task, achieving significant results. These performances can only be reached by using a large number of images along with their fine annotated labels.

Unfortunately, the human effort needed to collect and annotate large datasets at a pixel-level is enormous. For this reason, recent works have focused on the usage of a synthetic dataset with computer-generated annotations, so that the labelling process is automatized and fine annotated labels on real images are not needed anymore. We'll refer to the domain of the synthetic dataset as "source" domain. We're interested in making predictions on real images, whose domain we'll call "target" domain. Source and target domains show significant differences, e.g., light conditions, weather variability, etc., that lead the model, trained on synthetic data, output inaccurate predictions for the target domain. A Domain Adaptation (DA) phase is needed to reduce the substantial gap between source and target domains.

Different strategies can be exploited in order to reduce the domain gap.

One possibility is to employ an Adversarial Domain Adaptation algorithm [6], that makes the predicted label distributions close to each other across source and target domains.

Another common strategy, that can also be combined with Adversarial DA to boost performances, consists in transforming the images of the source domain so that they match the style of the images of the target domain [7], [8], e.g., similar light conditions after transformation. Usually, these transformations are fast and parameterless, since they only change the visual appearance of source images.

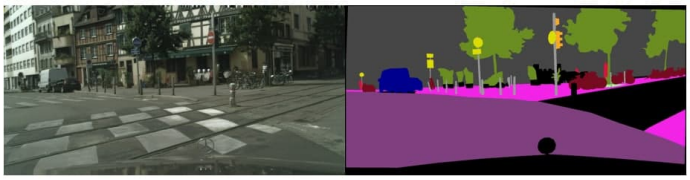

Fig. 1. *Left: image from target domain. Right: its semantic label.*

Lastly, Self-Supervised Learning (SSL) [9] can further enhance performances. While training, pseudo-labels for the target domain can be generated and the DA framework can exploit their availability to better align source and target domains, leading to more accurate predictions.

In this paper, we propose:

- an implementation of a real-time semantic segmentation network, BiSeNet (Bilateral Segmentation Network) [2], that can exploit two different backbones, ResNet-101 or ResNet-18;
- an implementation of an unsupervised adversarial domain adaptation algorithm;
- a variation of the unsupervised adversarial domain adaptation algorithm with lightweight depthwise-separable convolutions [10] for the adversarial discriminator, which significantly reduce the total number of parameters and the total number of Floating Point Operations (FLOPS) of the model, making it suitable for mobile mobile and/or embedded devices;
- two image-to-image transformations, FDA and LAB [7], [8], to improve domain adaptation;
- generation of pseudo labels for the target domain, to further enhance domain adaptation [9].
- a combination of LAB transformation and pseudo labels generation for the target domain.

The source code is available at https://github.com/claudiotancredi/Advanced-Machine-Learning.

II. Related work

A. Semantic Segmentation

Many advances in semantic segmentation are based on the progresses of deep neural networks. A classification CNN can be transformed to a fully convolutional network (FCN) to perform semantic segmentation [13]. Starting from this, several works have been proposed to enhance performances by utilizing context information [14], [15] or enlarging receptive field [16], [17]. The state-of-art results can be achieved by these methods only with a substantial amount of pixel-wise annotation, that are very difficult to obtain in the real world. For this reason, many synthetic datasets have been proposed, such as GTA5, in which the labeling process is partially automated. On one hand synthetic datasets helped at reducing the labeling cost but on the other hand, models trained on synthetic datasets suffer from domain shift. Besides the domain shift problem, many of the state-of-art models have a huge number of parameters but real-time applications require fast models, so several research focus on lightweight models. Poudel *et al.* [11] introduce fast segmentation convolutional neural network (Fast-SCNN), suited to efficient computation on embedded devices with low memory. This network is based on two-branches methods as [17].

B. Domain adaptation

To address the domain shift problem, domain adaptation methods have been developed. Many of them are based on the adversarial training techniques [12] that consists into the generator model that tries to fool the discriminator whose objective is to distinguish real images from those generated by the generator. To further explore UDA (Unsupervised Domain Adaptation) and enhance the model, Tsai *et al.* [6] construct a multi-level adversarial network to perform domain adaptation at different levels. Jia *et al.* [9] propose to leverage the unlabeled target domain by additionally constraining the outputs of multiple adaptation models with pseudo labels generated by an ensembled model. Furthermore, image-to-image translation techniques have been proposed [7], [8] that don't need any additional training but try to align source domain to target domain.

III. Method

A. BiSeNet

Our baseline model is BiSeNet, that is composed by two parts: Spatial Path (SP) and Context Path (CP). The two components address two issues, loss of spatial information and shrinkage of receiptive field, respectively.
The Spatial Path contains three layers, each including a convolution with stride = 2 followed by batch normalization and ReLU. As a consequence, this path extracts an output feature map that is 1/8 of the original image.
On the other hand, the Context Path is needed to provide an adequate receptive field. This path uses lightweight model and global average pooling to provide large receptive field. In the lightweight model, an incomplete U-shape structure is used to fuse the features of the last two stages.
Additional modules are also employed:

- Attention Refinement Module (ARM), used to refine the features of each stage in the Context Path. It employs global average pooling and computes an attention vector to guide the feature learning;
- Feature Fusion Module (FFM), used to fuse the features of the two paths because they are different in level of feature representation (Spatial Path is low level, Context Path is high level), therefore they can not simply be summed up. Output features of the two paths are first concatenated, then batch normalization is applied and, at the end, concatenated features are pooled to a vector for computing a weight vector, that allows feature selection and combination.

Figure 2 shows the entire network architecture.
To supervise the training, we employ a principal loss l_p and two auxiliary losses for the Context Path, l_i, which are all cross entropy losses. The joint loss can be expressed as:

$$\mathcal{L}(X;Y) = l_p(X;Y) + \alpha \sum_{i=2}^{K} l_i(X_i;Y) \tag{1}$$

where X_i is the output feature from stage i of the ResNet model, X is the final prediction of the BiseNet model, Y is the ground truth and K is to 3 as in [2].
The parameter α used to balance the weight of the auxiliary losses of the Context Path is equal to 1, allowing us to simplify the joint loss that results to a simple sum of the three losses, as in [2].

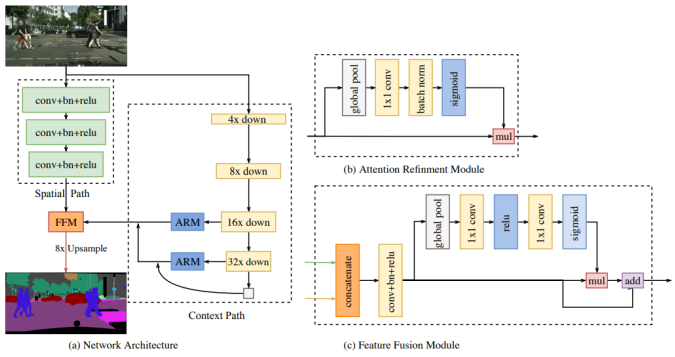

Fig. 2. *Architecture of BiSeNet. The image is taken from [2]. (a) Network Architecture. The length of block indicates the spatial size, while the thickness represents the number of channels. (b) Components of the Attention Refinement Module (ARM). (c) Components of the Feature Fusion Module (FFM). The read line represents a step taken only when testing.*

B. Unsupervised Adversarial Domain Adaptation

This framework [6] is based on a Generative Adversarial Network (GAN). It is composed by a segmentation model G to predict output results and a discriminator D to distinguish whether the input is from the source or target domain.

First, a source image I_s and its label Y_s is forwarded to the segmentation network for the optimization of G. Then, the segmentation softmax output P_t for the target image I_t (without label) is predicted. The key point of this DA framework is to make predictions of source and target images (i.e., P_s and P_t) as close as possible to each other. For this purpose, the discriminator D receives the two predictions and tries to distinguish whether the input is from the source or target domain. An adversarial loss on the target prediction allows the network to backpropagate gradients from D to G, encouraging G to generate similar segmentation distributions in the target domain to the source prediction.
Figure 3 shows the overview of this algorithm.
The joint loss for the adaptation task is defined as:

$$\mathcal{L}(I_s, I_t, Y_s) = \mathcal{L}_{seg}(I_s, Y_s) + \lambda_{adv}\mathcal{L}_{adv}(I_t) \tag{2}$$

where $\mathcal{L}_{seg}$ is the cross-entropy loss using source ground truth labels, $\mathcal{L}_{adv}$ is the adversarial loss and λ_{adv} is the weight used to balance the two losses.
Specifically, $\mathcal{L}_{seg}$ is defined as:

$$\mathcal{L}_{seg}(I_s) = -\sum_{h,w}\sum_{c\in C} Y_s^{(h,w,c)} \log(P_s^{(h,w,c)}) \tag{3}$$

where Y_s is the ground truth annotations for source images, $P_s = G(I_s)$ is the segmentation output, C is the number of classes and h, w refer to a specific pixel of the image with dimensions HxW.
Then, the images in the target domain are forwarded to G to obtain the prediction $P_t = G(I_t)$. The adversarial loss $\mathcal{L}_{adv}$ in (2) is defined as:

$$\mathcal{L}_{adv}(I_t) = -\sum_{h,w} \log(D(P_t)^{(h,w,1)}). \tag{4}$$

This loss is designed to train the segmentation network and fool the discriminator by maximizing the probability of the target prediction being considered as the source prediction, making the distribution of P_t closer to P_s.
The discriminator is trained by giving it as input the segmentation softmax output $P = G(I)$ and using a cross-entropy loss $\mathcal{L}_d$ for the two classes (i.e., source and target). This loss can be written as:

$$\mathcal{L}_d(P) = -\sum_{h,w}(1-z)\log(D(P)^{(h,w,0)}) + z\log(D(P)^{(h,w,1)}, \tag{5}$$

where $z = 0$ if the sample is drawn from the target domain, and $z = 1$ for samples taken from the source domain.

C. Lightweight depthwise-separable convolutions

The adversarial discriminator D largely uses 2D convolutions as its convolutional layers, which are performed over all input channels.
To improve efficiency and make the discriminator's model lightweight, 2D convolutions of D can be substituted with Depthwise Separable Convolutions (DSC) [10], where each channel is kept separate.

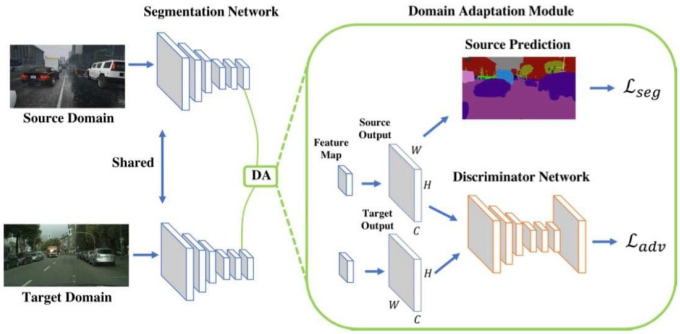

Fig. 3. *Algorithmic overview for unsupervised adversarial domain adaptation. The image is adapted from [6]. Given source and target images with size H x W, they are passed to the segmentation network to get output predictions. For source predictions, a segmentation loss is computed based on the source ground truth. Both source and target predictions are passed to the discriminator that tries to distinguish whether the input is from the source or target domain. An adversarial loss is computed on the target prediction and is back-propagated to the segmentation network.*

The approach of DSC consists in, first, splitting the input tensor of 3 dimensions into separate channels, convolving each channel with a 2D filter and stacking together the output of each channel to get the output of the entire 3D tensor, then the 3 color channels are combined to form 'n' number of channels, with n as desired.
Figure 4 presents the DSC approach.
Since DSC simplify computations by employing just 1D convolutions across all channels, leading to fewer parameters and Floating Point Operations (FLOPS), its usage is especially suggested for models that need to be run on devices that are not computationally powerful.
Due to the simplifications, the model that uses DSC usually achieves a lower accuracy. However, the tradeoff in accuracy is so small that the computational benefits make its employment effective most of the time.

D. Image-to-image transformations

We observe that images of source and target domains significantly differ in appearance, with color, texture and other attributes mismatches that make the domain adaptation task harder to solve. These discrepancies can be reduced by means of an effective style transfer, so that the style of images in the source domain is adapted to the style of images in the target domain. For this reason, two different image-to-image transformations are applied, FDA [7] and LAB [8].

1) FDA: The idea behind this transformation is to align low-level statistics between the source and target distributions. Given an image in source domain, the transformation requires to compute its Fast Fourier Transform (FFT). Similarly, FFT is computed for a randomly selected image from target domain. Then, the low-level frequency of the source image is replaced with the low-level frequency of the target image. Finally, the source image is reconstructed for training by means of the

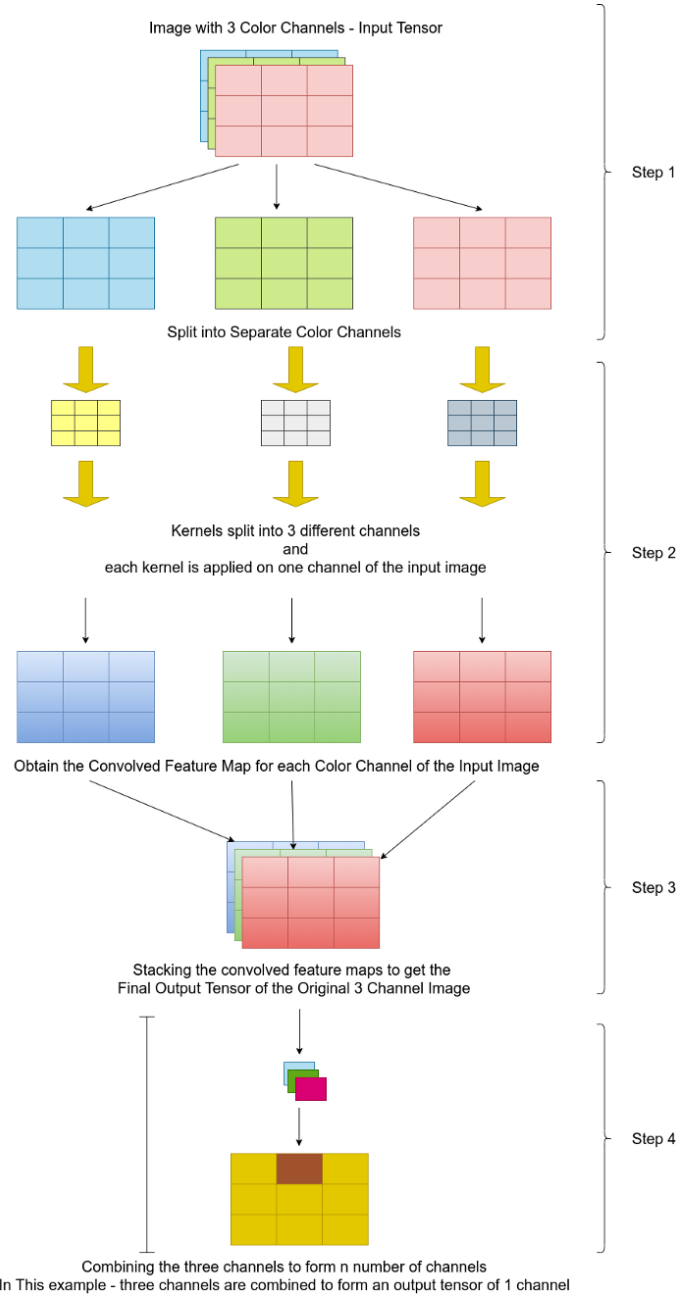

Fig. 4. *Graphical explanation of Depthwise Separable Convolutions. The image is taken from this Towards Data Science's article.*

inverse Fast Fourier Transform (iFFT).

For a single channel image x, its FFT can be computed as follows:

$$\mathcal{F}(x)(m,n) = \sum_{h,w} x(h,w) e^{-j2\pi(\frac{h}{H}m + \frac{w}{W}n)}, j^2 = -1, \quad (6)$$

where H is the image's height and W is the image's width. The amplitude and phase components of the Fourier transform $\mathcal{F}$ are $\mathcal{F}^A$ and $\mathcal{F}^P$, respectively, both with dimensions H x W x 3.

$\mathcal{F}^{-1}$ is the iFFT that maps spectral signals (phase and amplitude) back to image space.

Further, we denote with M_β a mask, whose value is zero except for the center region where $\beta \in (0,1)$:

$$M_\beta(h,w) = \mathbb{1}_{(h,w)\in[-\beta H:\beta H, -\beta W:\beta W]}, \quad (7)$$

assuming that the center of the image is (0,0).

Given two randomly sampled images x^s, x^t from source and target domains, Fourier Domain Adaptation can be formalized as:

$$x^{s\to t} = \mathcal{F}^{-1}([M_\beta \circ \mathcal{F}^A(x^t) + (1 - M_\beta) \circ \mathcal{F}^A(x^s), \mathcal{F}^P(x^s)]), \quad (8)$$

where the low frequence part of the amplitude of the source image $\mathcal{F}^A(x^s)$ is replaced by that of the target image x^t. Then, the modified spectral representation of x^s, with its phase component unaltered, is mapped back to the image $x^{s\to t}$, whose content is the same as x^s, but will resemble the appearance of a sample from the target domain.

Fourier domain adaptation requires selecting one free parameter, β, the size of the spectral neighbourhood to be swapped. Fig 5 shows the effectiveness of this style transfer method.

Fig. 5. *Left: image in source domain. Center: image in target domain. Right: image in source domain with target style after FDA is applied.*

2) LAB: This LAB-based transformation involves the translation of images from RGB color space to LAB color space, since the LAB color space is larger than RGB color space and the style of source images in LAB color space better aligns with the style of target images.

Given a RGB image $\mathcal{X}_S^{RGB}$ in source domain, it is firstly converted to LAB color space to generate the LAB image $\mathcal{X}_S^{LAB}$ through $\mathcal{X}_S^{LAB} = rgb2lab(\mathcal{X}_S^{RGB})$. The same operation is performed on a randomly selected image from target domain, converting it to LAB color space. Then, we compute the mean μ and the standard deviation σ of each channel of the two generated LAB images, $\mathcal{X}_S^{LAB}$ and $\mathcal{X}_T^{LAB}$, obtaining μ_S, σ_S, μ_T and σ_T. After this preparation step, the converted LAB image from source domain is converted to the style of target domain by shifting the distribution of pixel values to the image of target domain, according to:

$$\widehat{\mathcal{X}}_S^{LAB} = \frac{(\mathcal{X}_S^{LAB} - \mu_S)}{\sigma_S} * \sigma_T + \mu_T \quad (9)$$

After the alignment is successfully completed in LAB color space, we convert the translated LAB image $\widehat{\mathcal{X}}_S^{LAB}$ from source domain back to the RGB color space $\widehat{\mathcal{X}}_S^{RGB}$ through $\widehat{\mathcal{X}}_S^{RGB} = lab2rgb(\widehat{\mathcal{X}}_S^{LAB})$, to train the model on.

Figure 6 shows the effectiveness of this style transfer method.

Fig. 6. *Left: image in source domain. Center: image in target domain. Right: image in source domain with target style after LAB is applied.*

E. SSL

If we had labels also for the target domain, we could implement a fully supervised model. However, as pointed out in the introduction, labels for target domain are often costly to obtain. For this reason, we adopt a self-supervision technique and generate pseudo labels for the target domain to help the domain adaptation algorithm achieve better results.
This Self-Supervised Learning (SSL) approach [9] can be divided in two steps:

- at the beginning, pseudo labels are not available and the training is performed as usual until the segmentation model performs good enough to produce confident labels for the target domain;
- when the segmentation model is sufficiently accurate, starting from its predictions we obtain the pseudo labels $\widehat{Y}_t$ for the target domain that can be used to modify the overall loss function of G (2) as:

$$\mathcal{L}(I_s, I_t, Y_s, \widehat{Y}_t) = \mathcal{L}_{seg}(I_s, Y_s) + \lambda_{adv}\mathcal{L}_{adv}(I_t) + \mathcal{L}_{seg}(I_t, \widehat{Y}_t) \tag{10}$$

We need to define how to produce pseudo labels. We choose the "max probability threshold" (MPT) [9] to filter the pixels with high prediction confidence in I_T. Thus we can define the mask map $\widehat{y}_T$ as $m_T = \mathbb{1}_{[argmaxG(I_T) > threshold]}$.

IV. Experimental results

A. Datasets

The chosen target dataset is a subset of **Cityscapes** [4], with 500 images and their fine annotated labels for training and 250 other images without labels that are used both as validation set and as test set.
The source synthetic dataset is a subset of **GTA5** [5], with 500 images and their fine annotated labels for training.
For both training and evaluation, only the 19 classes shared by Cityscapes and GTA5 have been used.

B. Implementation details

Our baseline model is BiSeNet. For the first domain adaptation experiment, the discriminator used for the domain adaptation phase is a Fully Convolutional Discriminator (FCD) with 5 convolutional layers with kernel size 4x4, channel numbers {64,128,256,512,1}, stride 2 and padding 1. For all the following experiments, a lightweight discriminator is implemented by replacing each 2D convolution with a depthwise convolution of kernel size 4x4, each followed by a pointwise convolution, with kernel size 1x1. The channel numbers, stride and padding remain unaltered. Each convolutional layer is followed by a Leaky ReLU with a negative slope of 0.2.

Our general setting for the next experiments will be, where not specified differently:

- number of epochs, 50;
- mini-batch Stochastic Gradient Descent (SGD) with batch size 4, momentum 0.9, weight decay $1e^{-4}$, initial learning rate $2.5e^{-2}$ that progressively decreases according to the "poly" learning rate strategy in which the initial rate is multiplied by $\frac{1-iter}{max\_iter}^{power}$ at each iteration, with power 0.9;
- cropping on images, (1024,512);
- mean for normalization of images, (104.00698793, 116.66876762, 122.67891434).

C. BiSeNet

The first experiment we conduct is to train the BiSeNet model. We use both ResNet-101 and ResNet-18 as backbone, to compare their performances.
The results on the evaluation set (same as validation set) reported in table 1 will be the upper bound that we will try to reach in the domain adaptation phase. Since we have not encountered any issues in training the more complex model, ResNet-101, and its performances surpass the ones of ResNet-18, both at a global level (mIoU) and at a class-level (IoU of each class), our work will focus only on ResNet-101 from now on.
Moreover, as it can be seen from figure 7, at the beginning the model is greatly improving, but from the $\sim 20^{th}$ epoch the mIoU reaches a saturation point. For this reason, all the evaluations will be performed according to the model parameters of the best epoch.

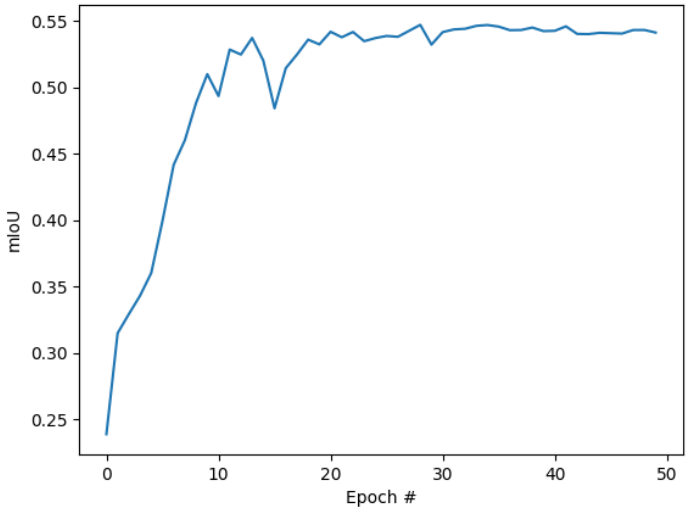

Fig. 7. *mIoU at each epoch during BiSeNet training.*

D. Unsupervised Adversarial Domain Adaptation

We now implement an unsupervised adversarial training using BiSeNet as segmentation model (generator G).
To train the discriminator, we use the Adam optimizer with

TABLE I
BISENET EVALUATION RESULTS ON CITYSCAPES

Backbone	road	sidewalk	building	wall	fence	pole	light	sign	vegetation	terrain	sky	person	rider	car	truck	bus	train	motocycle	bicycle	**mIoU (%)**
ResNet-101	96.5	73.7	86.4	33.5	27.2	37.7	40.1	54.8	87.9	51.5	89.7	61.8	35.9	88.7	29.9	40.3	14.3	31.3	58.4	**54.7**
ResNet-18	95.3	66.3	84.7	25.2	22.6	31.4	30.9	45.4	86.1	47.0	89.2	55.5	27.1	84.7	16.0	31.3	9.6	25.7	53.5	48.8

Note: for classes, the value represents the IoU (%)

TABLE II
COMPUTATION ANALYSIS FOR THE SEGMENTATION NETWORK, THE STANDARD DISCRIMINATOR NETWORK AND THE DSC VARIATION OF THE DISCRIMINATOR

Network	Total parameters	FLOPS [GFlops]
BiSeNet (segmentation network)	48,693,786	91.91
FC Discriminator	2,781,121	15.46
Discriminator DSC	**189,424**	**1.07**

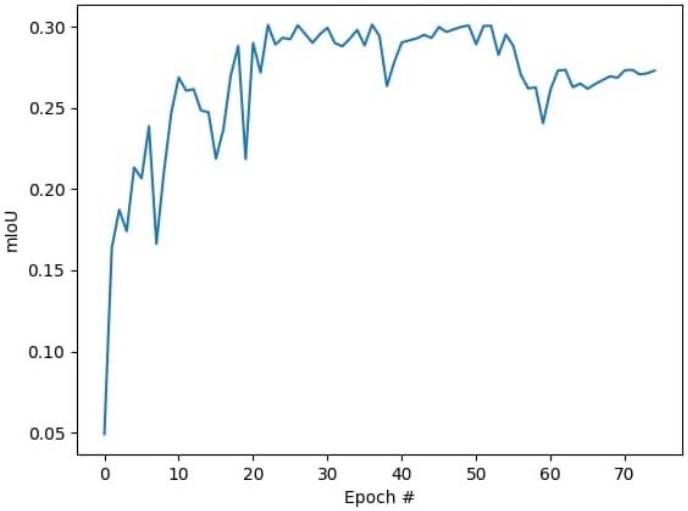

Fig. 8. *mIoU at each epoch for adversarial DA algorithm with DSC, pseudo labels and LAB.*

Fig. 9. *Left: target image. Center: its semantic label. Right: generated pseudo-label for the target image.*

learning rate $1e^{(-4)}$ and the same polynomial decay employed in the segmentation network. The betas parameter of Adam optimizer is set to (0.9, 0.99). The value of λ_{adv} is set to 0.001 as suggested in [6] for the single level setting.
Then, each 2D convolution of the discriminator is replaced with a DSC and training is performed again.
To evaluate the computational advantages of this variation, torchstat library will be used. The total number of parameters and the total numbers of FLOPS for the segmentation network, the standard discriminator network and the DSC variation of the discriminator are reported in table 2. It is evident that the DSC version of the discriminator is lighter with respect to its FC counterpart and low-level devices can benefit from it.
The results on the evaluation set for both DA variations are collected in table 3.
We would have expected the mIoU for the DSC variation to be slightly lower if compared to the mIoU of its FC counterpart, but in our case this does not happen. This is probably due to data shuffling performed during training.

E. Extensions

We implemented three variations of the adversarial DA algorithm with DSC to further reduce the domain gap between source and target domains: FDA, LAB and an implementation of SSL with the generation of pseudo labels for the target domain.

1) FDA: First we use FDA, which requires a parameter β to be set. Following the reasoning in [7], we choose to use $\beta = 0.01$ to avoid generating artifacts during the transformation process.
The results on the evaluation set are collected in table 3.
The mIoU shows an improvement of only 0.4%. A possible approach to obtain better results would be to fine-tune the parameter β, finding a good tradeoff between good quality of images (low values of β) and good style transfer (high values of β).

2) LAB: As a second extension we implemented LAB, which requires no parameters to be set.
The results on the evaluation set are collected in table 3.
The mIoU shows an improvement of 1.5%, if compared with the adversarial DA algorithm with DSC.

3) SSL: In this step, where pseudo labels are generated, the number of epochs is 75. Up to the 50^{th} epoch, the

TABLE III
ADVERSARIAL DA RESULTS ON GTA5 → CITYSCAPES

Model	road	sidewalk	building	wall	fence	pole	light	sign	vegetation	terrain	sky	person	rider	car	truck	bus	train	motocycle	bicycle	mIoU (%)
ResNet-101 (upper bound)	96.5	73.7	86.4	33.5	27.2	37.7	40.1	54.8	87.9	51.5	89.7	61.8	35.9	88.7	29.9	40.3	14.3	31.3	58.4	**54.7**
Adv DA	85.5	30.0	76.0	21.9	6.9	17.6	10.0	2.9	80.3	21.9	72.1	36.3	7.9	66.1	14.7	4.2	0	5.8	3.2	29.6
Adv DA with DSC	85.5	31.0	75.7	21.4	8.5	17.8	14.1	3.2	80.2	25.5	72.8	37.3	6.3	66.1	14.8	5.4	0	4.6	0	30.0
Adv DA with DSC and FDA	83.2	34.7	77.4	25.5	8.8	18.3	13.3	7.3	79.7	30.7	72.6	34.8	0.3	72.6	14.7	1.7	0	2.2	0	30.4
Adv DA with DSC and LAB	82.7	29.5	78.2	27.3	9.4	19.2	14.0	6.6	81.1	30.8	76.0	34.7	8.8	72.4	15.9	6.3	0	5.6	0.3	31.5
Adv DA with DSC and pseudo labels	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	25.3
Adv DA with DSC, pseudo labels and LAB	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	-	30.1

Notes: 1) for classes, the value represents the IoU (%). 2) for Adv DA with DSC and pseudo labels, and Adv DA with DSC, pseudo labels and LAB, IoU for classes are not available.

experimental setting is the same one used for DA with DSC, while starting from the 51^{st} epoch pseudo labels are generated by the segmentation network and used to improve predictions. The results on the evaluation set are collected in table 3.
For this extension, IoU for classes are not available. Indeed, the best epoch is before the generation of pseudo labels and, because of this, we do not consider its evaluation results, we just resort to the best mIoU obtained in the validation phase between epochs 51-75.
The figure 10 shows the mIoU at each epoch. Starting from the 50^{th} epoch, where pseudo labels are generated and then used for training, the mIoU significantly drops. This behavior is due to the limited number of samples at our disposal and to the segmentation model, which is not able to accurately define high confidence areas.

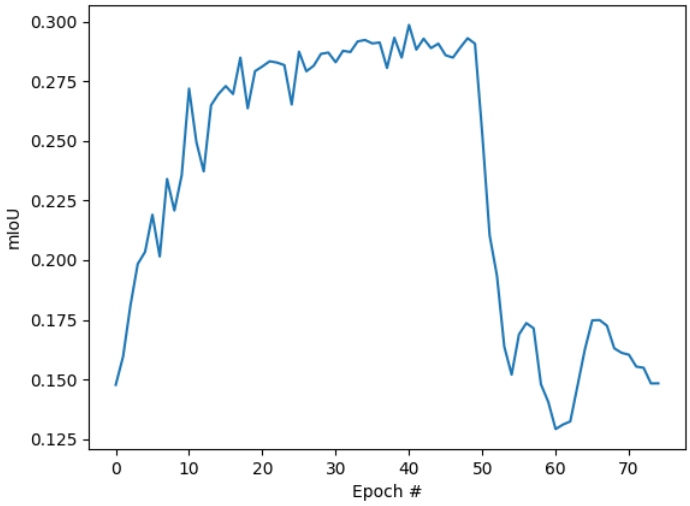

Fig. 10. *mIoU at each epoch for adversarial DA algorithm with DSC and pseudo labels.*

F. Additional considerations

As evident from table 3, all previous extensions fail or provide poor performances at predicting labels for classes train and bicycle. This can be explained by analyzing the source dataset, where objects from these two classes rarely appear. Predictions of all previous models on two target images are shown in figure 11.
The problem with bicycles is well visible in the domain adaptation algorithms. For example, DA provides low-level predictions, mixing predictions of bicycles and motocycles. Other extensions, instead, completely miss bicycles, thinking they are motocycles.

G. LAB and SSL

To address the last issue emerged in SSL implementation, we decide to combine LAB and SSL in a single variation. Our observations on the predicted labels of the LAB extension lead us to think that this model can better identify margins of objects. This also explains why LAB succeeds in achieving the best performances. Since pixels that are on a boundary region between two objects are the most critical ones for the pseudo labels generation, because their confidence level may not be high enough, we decide to employ LAB transformation to verify whether SSL performances can benefit from it.
The results on the evaluation set are collected in table 3.
Also for this extension, IoU for classes are not available for the same reasons they were not included in SSL analysis.
The figure 8 shows the mIoU at each epoch. The mIoU values obtained after the 50^{th} epoch confirm our hyphotesis. Even though the performances are far from the ones of the LAB extension, the SSL algorithm is now stable and provides sufficiently accurate predictions.
An example of a generated pseudo label is available in figure 9.

V. CONCLUSION

Our work proves the effectiveness of adversarial domain adaptation algorithms for semantic segmentation, with a focus on real-time applications and on different variations that try to reduce the gap between synthetic and real images. Some problems have been encountered, like SSL poor performances, but, step by step, models' performances have been pushed to their limits. We even combined different strategies, adding the LAB image transformation process in the SSL framework. Still, there is large space for improvements in this field. Future works may focus on SSL with more confident segmentation models, which we believe can outperform our implementation. Moreover, due to the limited computational power at our

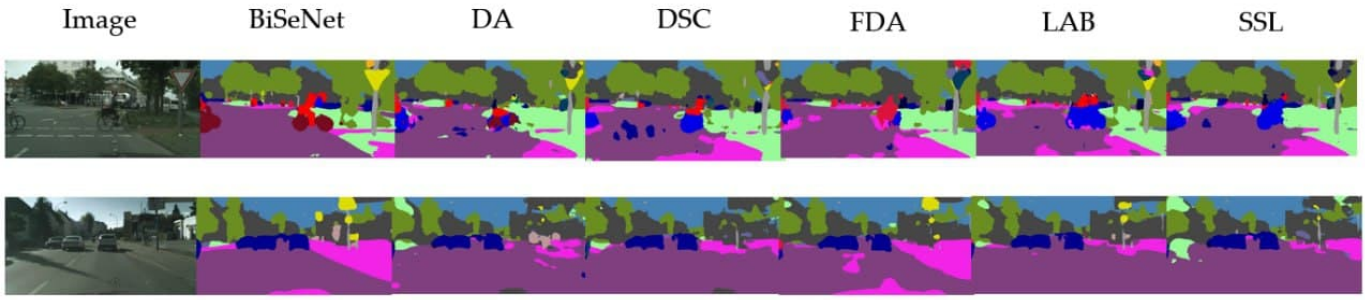

Fig. 11. *Predictions for all previous models on two target images.*

disposal, we could not work with large datasets, leading us into overfitting models. Hopefully, new analysis on large or properly augmented datasets will lead to better segmentation predictions.

ACKNOWLEDGMENT

We would like to thank Antonio Tavera for his support.

REFERENCES

[1] Shijie Hao, Yuan Zhou and Yanrong Guo, “A Brief Survey on Semantic Segmentation with Deep Learning”, Neurocomputing, vol. 406, pp. 302-321, September 2020.

[2] Changqian Yu, Jingbo Wang, Chao Peng, Changxin Gao, Gang Yu and Nong Sang, “BiSeNet: Bilateral Segmentation Network for Real-Time Semantic Segmentation’,’ Computer Vision – ECCV 2018, 2 August 2018.

[3] Sicheng Zhao, Xiangyu Yue, Shanghang Zhang, Bo Li, Han Zhao, Bichen Wu, Ravi Krishna, Joseph E. Gonzalez, Alberto L. Sangiovanni-Vincentelli, Sanjit A. Seshia and Kurt Keutzer, “A Review of Single-Source Deep Unsupervised Visual Domain Adaptation”, IEEE Transactions on Neural Networks and Learning Systems, 19 September 2020.

[4] M. Cordts, M. Omran, S. Ramos, T. Rehfeld, M. Enzweiler, R. Benenson, U. Franke, S. Roth, and B. Schiele, “The Cityscapes Dataset for Semantic Urban Scene Understanding”,2016 IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2016.

[5] Stephan Richter, Vibhav Vineet , Stefan Roth and Vladlen Koltun, “Playing for Data: Ground Truth from Computer Games”, Computer Vision – ECCV 2016.

[6] 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 IEEE/CVF Conference on Computer Vision and Pattern Recognition, June 2018.

[7] Yanchao Yang and Stefano Soatto, “FDA: Fourier Domain Adaptation for Semantic Segmentation”, 2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition, June 2020.

[8] Jianzhong He, Xu Jia, Shuaijun Chen, and Jianzhuang Liu, “Multi-Source Domain Adaptation with Collaborative Learning for Semantic Segmentation”, 2021 IEEE/CVF Conference on Computer Vision and Pattern Recognition, June 2021.

[9] Yunsheng Li, Lu Yiuan, and Nuno Vascolncelos, “Bidirectional Learning for Domain Adaptation of Semantic Segmentation”, 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition, June 2019.

[10] Francois Chollet, ”Xception: Deep Learning With Depthwise Separable Convolutions”, 2017 Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), 2017, pp. 1251-1258.

[11] Rudra P K Poudel, Stephan Liwicki, Roberto Cipolla, ”Fast-SCNN: Fast Semantic Segmentation Network”, 2019, CVPR, 2019.

[12] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky, ”Domain adversarial training of neural networks”, JMLR, 2016.

[13] J. Long, E. Shelhamer, and T. Darrell, ”Fully convolutional networks for semantic segmentation”, CVPR, 2015.

[14] W.-C. Hung, Y.-H. Tsai, X. Shen, Z. Lin, K. Sunkavalli, X. Lu, and M.-H. Yang, ”Scene parsing with global context embedding”, ICCV, 2017.

[15] H. Zhao, J. Shi, X. Qi, X. Wang, and J. Jia, ”Pyramid scene parsing network”, CVPR, 2017.

[16] F. Yu and V. Koltun, ”Multi-scale context aggregation by dilated convolutions”, ICLR, 2016.

[17] L.-C. Chen, G. Papandreou, I. Kokkinos, K. Murphy, and A. L. Yuille, ”Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs”, CoRR, abs/1606.00915, 2016.