Real-time Domain Adaptation in Semantic Segmentation

Advanced Machine Learning

1st David Azoulay
Pôle Léonard de Vinci - ESILV, Paris-La Défense
Erasmus in Politecnico di Torino
Paris, France
s293596@studenti.polito.it

2nd Andrea Cavallo
Politecnico di Torino
Torino, Italy
s287965@studenti.polito.it

3rd Lise Cottin
Ecole Polytechnique de Bruxelles
Erasmus in Politecnico di Torino
Torino, Italy
s293662@studenti.polito.it

***Abstract*—This work is concerned with the computer vision task of semantic segmentation and addresses a main issue related to it: the need for large datasets containing pairs of images and pixelwise labelling maps representing the results of the task. As a matter of fact, manually generating these labels is extremely expensive, since every pixel needs to be properly associated to its class. To overcome this problem, it is possible to train the network on synthetic images, which are automatically labelled. To this end, the domain adaptation paradigm can be applied: not only is the segmentation network trained on the labelled synthetic images, but it should also fool a discriminator into believing that images from the source domain belong to the target domain. In this way, the segmentation network is forced to learn a hidden feature representation for images that is common to both domains. Moreover, an extension of the domain adaptation paradigm is proposed, in which pseudo-labels are generated for the target domain and are used in the following training iterations. The code for this work is available in the following repository: https://github.com/andrea-cavallo-98/Advanced-Machine-Learning.**

***Index Terms*—real-time semantic segmentation, domain adaptation, adversarial methods**

I. Introduction

In this project, the concept of real-time domain adaptation is studied in the context of semantic segmentation.

Semantic segmentation, a challenging and fundamental task in the field of computer vision, consists in assigning a semantic class to each pixel in an image. In Figure 1, it is possible to see an image with the corresponding output of the semantic segmentation task, called label, i.e. a constructed image where each pixel value has been set to a specific level according to the class to which the pixel belongs. This kind of task is particularly helpful in the context of augmented reality devices, autonomous driving, and video surveillance for which a fast response or interactivity is needed [2], justifying the need for real-time techniques to infer the results.

Today, the most efficient and widely used approaches to semantic segmentation are deep learning techniques. These deep neural networks are trained on images already associated with their labels to learn the mapping between a semantic label and its diversified visual appearances [1].

Fig. 1: Image associated with its corresponding semantic segmentation

However, to attain reasonable performances, the dataset fed to the network should be large enough, which constitutes a problem since, in many applications, it is difficult to obtain a large amount of labels, as manual annotation is expensive and time-consuming [3]. This issue constitutes one of the main challenges in semantic segmentation. To avoid this problem, a solution would be to train the model on a large-scale source domain related to the desired one but for which labels are easy to obtain (e.g. a simulation domain) and then to apply it to the desired unlabeled target domain (e.g. a real-world domain). Unfortunately, this approach suffers from domain shift, which causes the estimation of the expected loss on the test to be biased and might result in poor performances. This is where the concept of domain adaptation comes in, which allows to address the problem of domain shift. This framework is a specialized form of transfer learning that uses as training set a labelled source domain that can generalize well to a different (but related) unlabelled or sparsely labelled target domain [3]. In this work, the selected approach for domain adaptation is an unsupervised adversarial technique, which consists in pairing the network to a discriminator, whose role is to distinguish whether the extracted features belong to the source domain or

to the target domain. In this way, the segmentation network is forced to learn a feature representation that fits both source and target domain and can more easily transfer the knowledge acquired on one domain to the other.

In this work, the performance of the domain adaption technique is investigated using two well-known and commonly used datasets representing complex urban street scenes: GTA5, a dataset of synthetic images used as source domain (pair of images and labels) and Cityscapes for the target domain (images only). Actually, the Cityscapes dataset also provides labels to its images, which allow the evaluation of the domain adaptation procedure: a first model will be trained on the Cityscapes dataset (with images and labels) and its performance at test time (which represents an upper bound for the domain adaptation method) will constitute a baseline to evaluate the other models that will be trained on GTA5 and tested on Cityscapes.

Furthermore, the adversarial method will be enhanced with the generation of pseudo-labels for the images belonging to the target domain during training. These pseudo-labels will then be used to train the model in the following epochs.

II. Related work

Deep learning methods have allowed a great step forward in the field of semantic segmentation [1], explaining their extensive utilisation for this task and the emergence of a large number of new techniques. As precursors, Long et al. proposed the Fully Convolutional Network (FCN) [8], which allowed to substantially increase the segmentation accuracy. Among the most commonly used deep architectures in semantic segmentation we can mention ResNet (deep residual network), which allowed the use of much deeper network structures.

The various research approaches on semantic segmentation can be divided into 3 main categories, depending on their supervision level: fully-supervised methods, weakly-supervised methods and semi-supervised methods [1].

A. Fully-supervised methods

The first category of techniques concerns situations where sufficient labelled training data are available. Among all the methods existing in this category, we will briefly discuss some real-time methods, as the implementation of our baseline network is based on one of those.

In real-time models, the critical point is to accelerate the execution without losing in the accuracy of the results. Among the existing real-time methods we can mention E-Net [9], one of the earliest works focusing on real-time semantic segmentation, which designs a lightweight network from scratch offering extremely high speed. An extended review of other real-time methods can be found in [1]. Although these methods have improved greatly in terms of accuracy and effectiveness, there is still a long way to go before they achieve the results obtained by other techniques. According to [2], the key points to achieve good results in semantic segmentation are rich spatial information and sizeable receptive field (related to context information). The information about the context is usually obtained by enlarging the receptive field or fusing different context information. However, a lot of approaches, like CNN [10] for example, compromise spatial resolution to achieve real-time inference speed, which leads to poor performance. To address this loss of spatial information, the U-shape structure is widely utilised [2]. In this context the Bilateral Segmentation Network (BiSeNet) architecture was proposed. It is composed of two parts: the Spatial Path (SP), a shallow but wide network designed with a small stride to preserve the spatial information from original images and the Context Path (CP), a lightweight model with a fast downsampling strategy employed to obtain sufficient receptive field. This architecture has been shown to provide a right balance between the speed and segmentation performance [2].

B. Weakly-supervised methods

As specified in I, the need for large number of labels within the dataset is costly and thus wants to be avoided. This is why methods that only use weakly supervised annotations have been developed. Those annotations, such as tags or scribbles require less effort to produce than the pixelwise labels. A full discussion about the related implementations can be found in in [1].

C. Semi-supervised methods

In the same spirit of reducing the time/cost needed to obtain labels, the semi-supervised techniques work with a reduced number of image-label pairs. Within this category, we can find the methods based on domain adaptation, which rely on the idea of, as briefly explained in I, using a model trained on a source domain on another similar target domain. These techniques indeed fit in the semi-supervised framework since the target domain only contains few or no labels. To perform domain adaptation, several categories of methods exist [3]:

- Discrepancy-based methods: measure the discrepancy between the source and target domains
- Adversarial discriminative models: aim to confuse a discriminator regarding the domain images belong to
- Adversarial generative models: combine the domain discriminative model with a generative component, consisting in a discriminator and a generator, that generates images and tries to confuse the discriminator associated with it
- Self-supervision-based methods: incorporate auxiliary self-supervised learning tasks into the original task network

All these categories are widely discussed in [3], with the mention of the main implementations.
Since this work adopts adversarial discriminative models, we will report some more details about this approach. The underlying idea of this procedure is to train a segmentator to produce accurate labels and then to fool a discriminator in distinguishing whether the images come from the source or target domain. In this context, the Domain-Adversarial Neural Networks (DANN) algorithm was proposed by [12]. To go further, [6] proposed a multi-level adversarial network

in order to perform domain adaptation at different feature levels, which showed competitive accuracy in regards to the state-of-the-art methods. Furthermore, in order to increase the segmentation accuracy of the adversarial model, pseudo-labels can be generated from the target images using self-supervised learning and then be used for training, as discussed in [7].

III. Method

A. Real-time semantic segmentation network

The first task of this project consists in implementing BiSeNet, a real-time semantic segmentation neural network [2]. The peculiarity of this network is that it involves two different paths to decouple spatial information preservation and receptive field offering. In particular, the Spatial Path (SP) consists of three convolutional layers, followed by batch normalization and ReLU, whereas the Context Path (CP) is, in this case, the pretrained ResNet-101 model. Moreover, the network contains two additional modules to improve performances: the Attention Refinement Module employs global average pooling to capture global context and computes an attention vector to guide the feature learning, and the Feature Fusion Module fuses the features coming from the two different paths. Figure 2 provides a visual sketch of the architecture of BiSeNet.

In the first part of the project, the BiSeNet network is trained for 50 epochs on a labelled subset of Cityscapes, using only 19 classes. The loss function is per-pixel cross-entropy loss, the initial learning rate is set to $2.5 \cdot 10^{-5}$ and it is decayed using the "poly" strategy, as described in [2]. Random horizontal flipping is performed as image augmentation strategy.

B. Unsupervised adversarial domain adaptation

The second part of the project focuses on implementing adversarial domain adaptation. In particular, the goal is to train BiSeNet on the synthetically generated images of GTA5 (very easily annotated) and test it on Cityscapes. To achieve this, the network is paired with a discriminator, whose role is to distinguish whether the extracted features belong to the source domain (GTA5) or to the target domain (Cityscapes). The discriminator consists of four convolutional layers followed by leaky ReLU and a final convolutional layer that outputs probability scores.

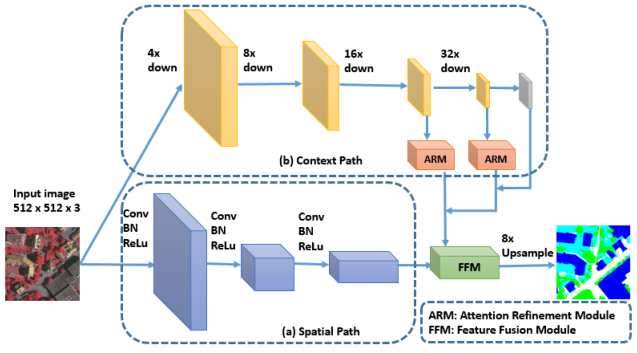

Fig. 2: Sketch of the BiSeNet architecture [2]

A lightweight version of the classifier is also implemented, where every convolutional layer is substituted with a depthwise-separable convolutional layer, processing channels separately, which has significantly less parameters.

Both modules are trained according to the following strategy. To begin with, the BiSeNet is trained on a labelled image from the source domain, minimizing the per-pixel cross-entropy loss. Then, the BiSeNet is fed an image from the target domain and is trained to fool the discriminator into classifying that image as belonging to the source domain. In particular, the segmentation classes predicted by BiSeNet are fed to the discriminator and a binary cross-entropy loss is computed between the output of the discriminator and the label of the source domain. After that, the discriminator is trained to predict whether the output of the BiSeNet corresponds to an image coming from the source or from the target domain, by optimizing a binary cross-entropy loss. This procedure is summarized in Figure 3.

The training is performed for 50 epochs using 19 semantic classes. The initial learning rate is set to $2.5 \cdot 10^{-5}$ and it is decayed using the "poly" strategy, as in the first task. Image augmentation is applied to the GTA5 dataset in the form of random horizontal flip.

C. Pseudo labelling of target domain

The third part of the project consists in extending the adversarial domain adaptation training scenario to improve the final performances. In particular, the selected strategy consists in generating pseudo-labels for the images in the target domain during training. The generated pseudo-labels are used as training data in the following epochs. A detailed description of the strategy is reported in [7]. An issue with this approach consists in choosing a proper threshold to discriminate which predicted pixels to include in the pseudo-labels. This choice should represent a tradeoff between the desire to have as many labelled pixels as possible in the following training epoch and the risk of degrading training performances by including too much noise into the pseudo-labels. Following the extensive explanations and experiments in [7], we tested two different approaches.

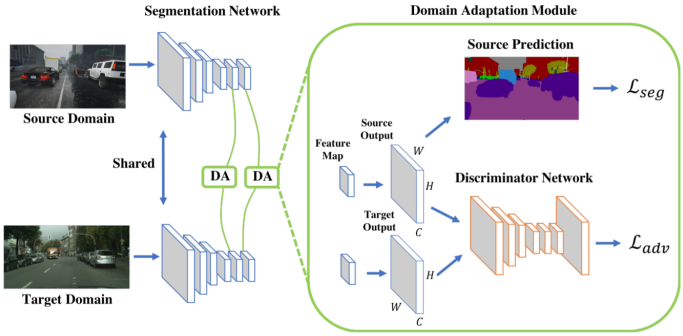

Fig. 3: Sketch of the procedure used for domain adaptation training [6]

In the first one, we only select as pseudo-labels pixels that are predicted with an accuracy higher than a fixed threshold (0.9 or 0.8). In the second case, we scale the threshold depending on the distribution of confidence of predictions for each different class, such that we select at least half of all the classified pixels for each class (more than half if there are more than 50% of pixels predicted with confidence higher than 0.9).
Regarding the training phase, the only difference with respect to the second task consists in the loss used to train the segmentation network using an image from the target domain. As a matter of fact, if pseudo labels are available, the loss becomes

$$\ell_{tot} = \lambda_{adv} \cdot \ell_{adv} + \ell_{seg}$$

where ℓ_{adv} is the adversarial binary cross-entropy loss described before, ℓ_{seg} is the per-pixel cross-entropy loss on the pseudo-labels and λ_{adv} is an hyperparameter set to 0.001 as specified in [7]. All the other hyperparameters for training are the same as in the previous section.

IV. Experiments, results and discussion

A. Datasets

a) Cityscapes: The original Cityscapes dataset was designed for urban scene understanding [4]. It contains 20,000 coarse-annotated images and 5000 fine-annotated images. In this work, we will only use a subset of these images: 500 as train data and 250 for test and validation (same subsets). There are in total 34 classes available but we will only use 19, corresponding to the ones that are compatible with GTA5.

b) GTA5: The original dataset, collected in the computer game GTA-V with pixel-wise semantic labels, contains 24,966 images (video frames) but again in this work only a subset is used. The 19 classes that are compatible with Cityscapes are considered.

B. Evaluation metrics

To evaluate the performances of our various models in terms of accuracy, we will use 2 metrics: the pixel accuracy (PA) and the mean intersection over union (mIoU). In the following, we will represent by k the total number of different classes and by Y the pixel number, i.e., Y_{ii}, Y_{ij}, and Y_{ji} denote the number of True Positive (TP), False Positive (FP), and False Negative (FN), respectively.

a) Pixel accuracy (PA): This metric characterises the ratio between the number of correctly classified pixels and the total number (i.e., $\frac{TP}{TP+FN}$). It can be computed using (1):

$$PA = \frac{\sum_{i=0}^{k-1} Y_{ii}}{\sum_{i=0}^{k-1} \sum_{j=0}^{k-1} Y_{ij}} \tag{1}$$

b) Mean intersection over union (mIoU): This metric characterises the average of the per-class IoU, which represents the rate between intersection and union (i.e., $\frac{TP}{TP+FP+FN}$). It can be computed using (2):

$$mIoU = \frac{1}{k} \sum_{i=0}^{k-1} \frac{Y_{ii}}{\sum_{j=0}^{k-1} Y_{ij} + \sum_{j=0}^{k-1} Yji - Y_{ii}} \tag{2}$$

The efficiency of some models will also be evaluated in the second experiment via the model complexity, which is characterised by the number of parameters and the Floating Point Operations (FLOPs). Models with large number of parameters and FLOPs tend to have low implementation efficiency [1].

C. Experiment 1: Real-time segmentation network

As the implementation of the BiSeNet was already provided for this first experiment, the main thing left to do was the implementation of the dataset class for Cityscapes. This has been done in the file `cityscapes_dataset.py`. As detailed in III-A, the settings of this experiment are the following: the BiSeNet network is trained for 50 epochs on a labelled subset of Cityscapes, using only 19 classes.

The evolution of the loss during the training is reported in Figure 4. This training loss shows a reasonable decrease confirming that the network is learning and that the performances are improving.

The final results are displayed in the Table I and show that the model performs well with a very good PA. These results will be used as a baseline to compare the performances of the models trained with domain adaptation.

Figure 10 reports some examples of the segmentation output provided by the network. By comparing them to the ground truth (Figure 9), it is possible to observe that the predictions are also visually quite good: all classes are identified, even though some errors are evident.

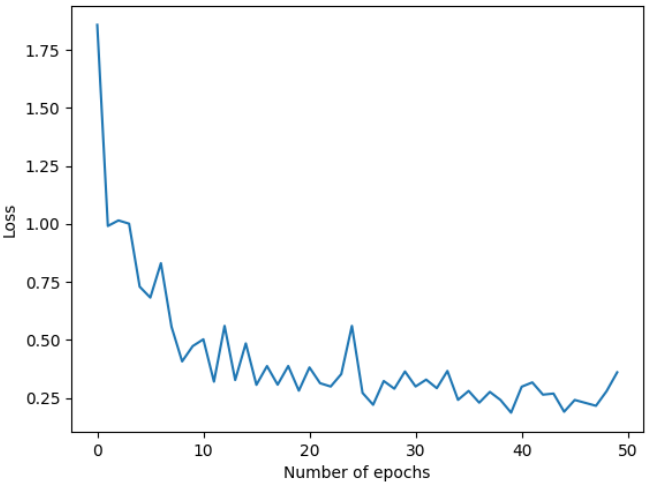

Fig. 4: Loss during the training of the baseline network

TABLE I: Results of the first experiment

Experiment 1	Evaluation metrics	
	PA	***mIoU***
Baseline	0.926	0.566

D. Experiment 2: Unsupervised adversarial domain adaptation

For the second experiment, the implementation of the discriminator has been based on the following repository: https://github.com/wasidennis/AdaptSegNet. The GTA5 dataset class has also been implemented in the file `GTA5_dataset.py`. As for the first experiment, the training is done for 50 epochs but on a subsets of both GTA5 (labelled) and Cityscapes (unlabelled) datasets using 19 classes. The experiments are performed using both the standard and the lightweight discriminator.

It is possible to observe the evolution of the losses during training in Figure 5. In particular, it can be noticed that the adversarial and the discriminator loss remain approximately constant during the training, as expected in the adversarial training setting. The segmentation loss, conversely, decreases at a reasonable rate, showing that the segmentation performances of the network are improving.

The final results are displayed in the Table II. Looking at the first row, which shows the results obtained with the basic version of the discriminator, as initially described in III-B, we can see a clear gap between the performances of the baseline model and the adversarial one, with a mIoU that is less than half of the one obtained in the first case. We can thus see that the adversarial model designed to perform domain adaptation in order to allow unsupervised learning in the context of semantic segmentation still needs a fair amount of improvements before being able to compete with the supervised techniques. A positive consideration, on the other hand, concerns the performance of the lightweight discriminator (second row in Table II), with respect to the standard one. Indeed, the first outperforms the latter while having way less parameters and FLOPs, implying that it is at the same time more efficient. For the following experiment, the lightweight version of the discriminator will be used.

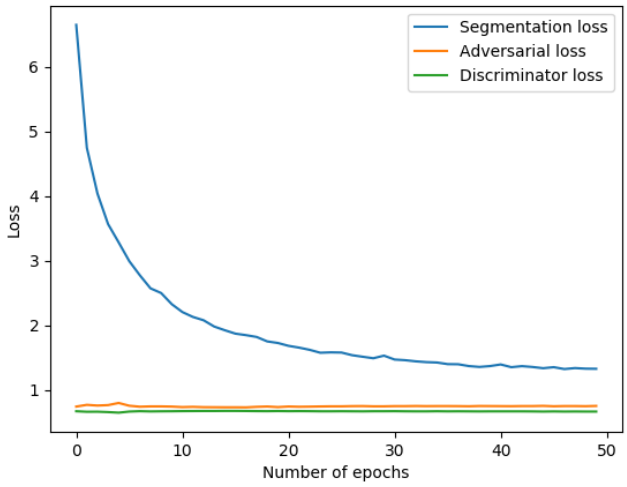

Fig. 5: Losses evolution during the training of the adversarial network

TABLE II: Results of the second experiment

Experiment 2 Discriminator	Evaluation metrics			
	PA	***mIoU***	***Number of parameters***	***FLOPs***
Normal	0.709	0.211	51,474,907	527.50 G
Lightweight	0.738	0.233	48,885,150	370.04 G

Figures 11 and 12 report some examples of predictions deriving from the networks trained with domain adaptation. In this case, it can be noticed that some classes are not identified at all (e.g. road signs), whereas for the most common classes (e.g. road, sidewalk) the predictions are better, even though road and sidewalk get confused in some cases.

E. Experiment 3: Pseudo labelling of target domain

For the pseudo-labels implementation, we based our code on the following repository: https://github.com/liyunsheng13/BDL/. Again, the training was performed for 50 epochs, and pseudo-labels were generated at every epoch. In particular, pseudo-labels were created by considering as labels all the pixels that are predicted by the network with a confidence higher than a given threshold. Multiple options are tested for this threshold: two fixed values (0.9 and 0.8) and a variable value, chosen to guarantee that at least 50% of the classified pixels for each class are selected. Figure 6 reports an example of a pseudo-label: pixels predicted with a high confidence are selected, whereas the others (not colored in the figure) are neglected.

The evolution of the losses during the training can be observed in Figure 7. It can be noticed that, with respect to the standard adversarial training setting reported in Figure 5, the adversarial loss increases and the discriminator loss decreases, even though their variations are small. This is due to the fact that the main contribution to the training loss is given by the accuracy of the segmentation of the pseudo-labels, whose loss significantly decreases during training. Therefore, the segmentation network is mainly trained with pseudo-labels, and the discriminator can more easily distinguish between the two different domains.

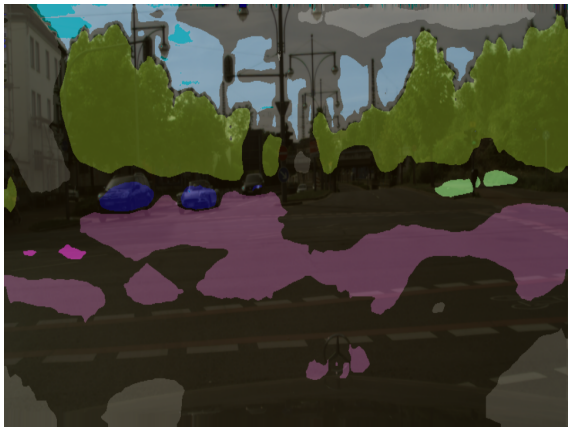

Fig. 6: Example of a pseudo label: only colored pixels are selected

The generation of pseudo-labels for images in the target domain aims at improving the performance of the adversarial model by providing labelled training data for the target domain, which are initially missing. However, as demonstrated in Table III where the results are displayed, we do not get any boost in the performance of the model. This is probably due to the limited number of images accessible for training, which is not sufficient to appreciate the benefit of generating pseudo labels.

The comparison between the different thresholds brings results that are consistent with the expectations based on the experiments in [7]. As a matter of fact, the best performances are achieved with a fixed threshold of 0.9, which represents a good tradeoff between the number of selected pixels and the noise of wrongly classified pixels. Lowering this threshold to 0.8 worsens the performances, as more of the selected pseudo-labels are wrongly classified. The same consideration holds for a variable threshold, since selecting at least 50% of the pixels at each epoch causes the presence of a huge amount of noise in the pseudo-labels.

Figures 13, 14 and 15 report some examples of the predictions of the networks trained with pseudo-labels. It is possible to notice that the confusion between the classes road and sidewalk, which was already present in Figures 11 and 12, becomes even more apparent in the case of the fixed 0.8 threshold and the variable one. Moreover, the networks are not able to recognize all classes (e.g. road signs or bicycles are never identified).

To further investigate the correctness of the segmentation predictions, Figure 8 reports recall, precision and IoU for each different class. It can be observed that some non-frequent classes get very low or zero IoU, and this affects the final result, since all classes weigh the same in the computation of the final mIoU.

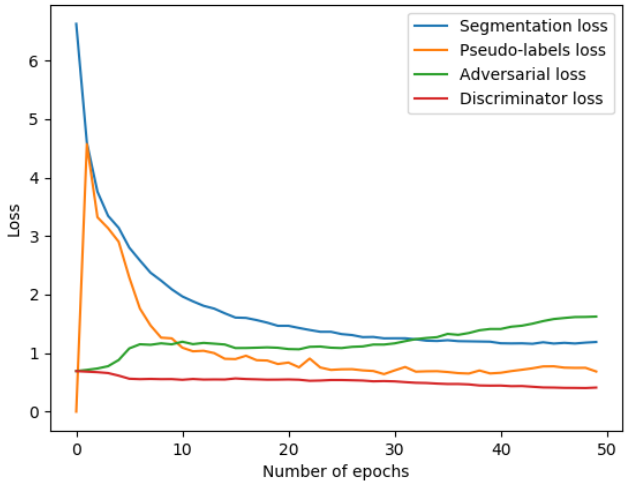

Fig. 7: Losses evolution during the training of the adversarial network using pseudo labels and a variable threshold

TABLE III: Results of the third experiment

Experiment 3 Threshold	Evaluation metrics	
	PA	***mIoU***
Fixed (0.9)	0.717	0.203
Fixed (0.8)	0.595	0.200
Variable	0.462	0.142

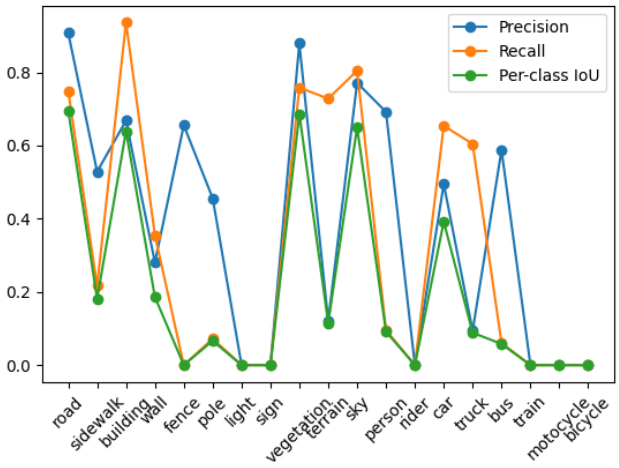

Fig. 8: Precision, recall and per-class IoU on test dataset for the model trained on domain adaptation with pseudo labels

V. Conclusion

To conclude, in this work, we were interested in semantic segmentation which aims at producing pixelwise semantic labels for images. For this task, we investigated the performance of an adversarial method for domain adaptation which allows to proceed with the semantic segmentation in an unsupervised setting, where the model is trained on a dataset for which semantic labels are easy to obtain and then tested on a dataset for which no labels exist. In comparison to the performance of a model trained in a supervised setting where the same dataset, for which labels and images exist, is used at training and test time, we must unfortunately admit the inferior performances of the domain adaptation technique. These results were expected but the gap between the performances is quite big, indicating that there is still a lot of improvement to be done to get unsupervised adversarial models that are competitive with the ones trained with supervised learning.

Concerning the adversarial model, two discriminators have been implemented and we are pleased to report that the lightweight version, where depthwise-separable convolutional layers replace the convolutional layers of the standard version, shows really good results, with better performances in terms of accuracy and efficiency.

Lastly, the pseudo-labelling method did not prove to be successful in our case, due to the small number of training images. Still, this technique could be very effective on a larger

database to improve the performance of the domain adaptation method, which is crucial in order to reduce the gap with the performance of the baseline model and be able to perform semantic segmentation on data that are not especially the same as the ones used for the training of the model.

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," Augustus 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," Augustus 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,".

[5] Stephan Richter, Vibhav Vineet , Stefan Roth and Vladlen Koltun, "Playing for Data: Ground Truth from Computer Games," 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,".

[7] Y. Li, L. Yuan and N. Vasconcelos, "Bidirectional Learning for Domain Adaptation of Semantic Segmentation," 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), 2019, pp. 6929-6938, doi: 10.1109/CVPR.2019.00710.

[8] J. Long, E. Shelhamer and T. Darrell, "Fully convolutional networks for semantic segmentation,", Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition, pp. 3431–3440, 2015.

[9] A. Paszke, A. Chaurasia, S. Kim and E. Culurciello, "Enet: A deep neural network architecture for real-time semantic segmentation," arXiv, 2016.

[10] A. Krizhevsky, I. Sutskever and G.E. Hinton, "Imagenet classification with deep convolutional neural networks.", Neural Information Processing Systems, 2012.

[11] V. Badrinarayanan, A. Kendall and R. Cipolla, "SegNet: A deep convolutional encoder-decoder architecture for image segmentation", IEEE Transactions on Pattern Analysis and Machine Intelligence, vol. 39(12), pp. 2481–2495, 2017.

[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, vol. 17, no. 1, pp. 2096–2030, 2016.

Fig. 9: Ground truth

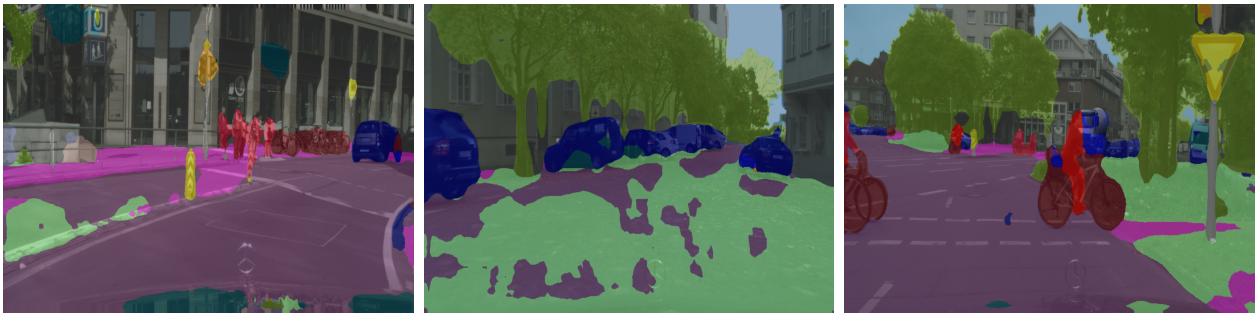

Fig. 10: Baseline

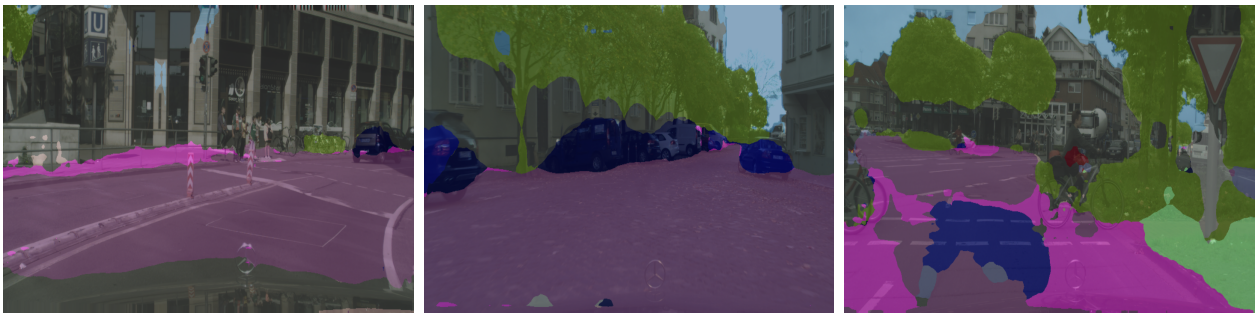

Fig. 11: DA standard

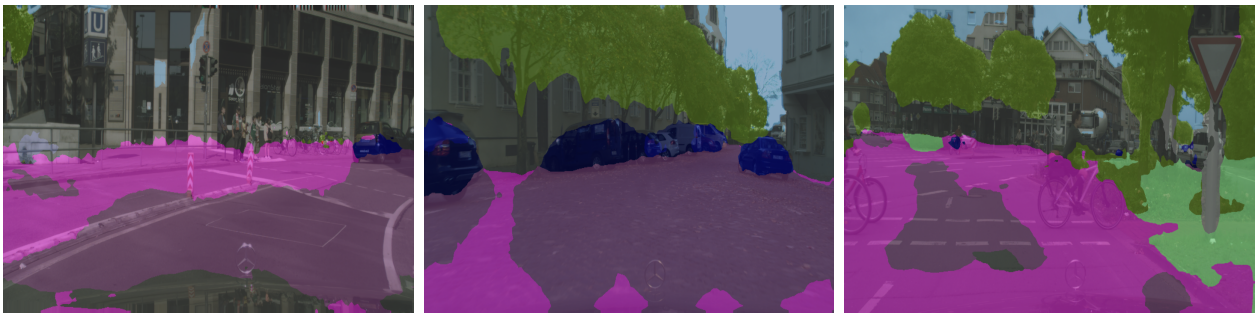

Fig. 12: DA lightweight

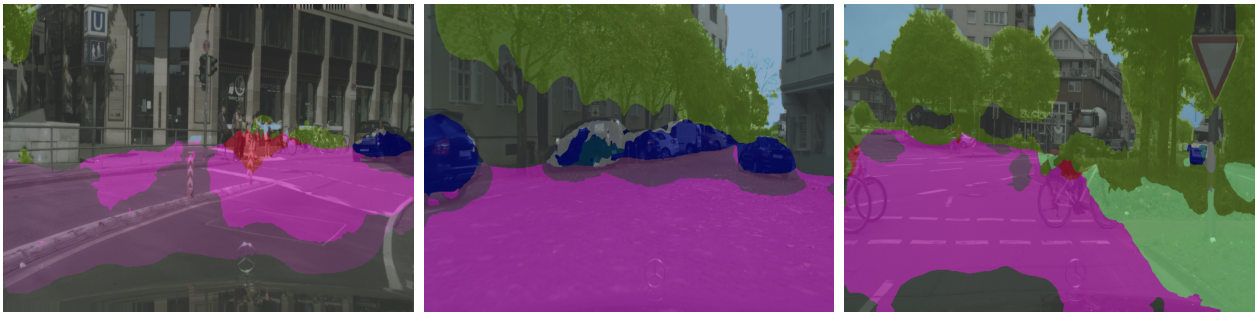

Fig. 13: DA PL fixed threshold (0.8)

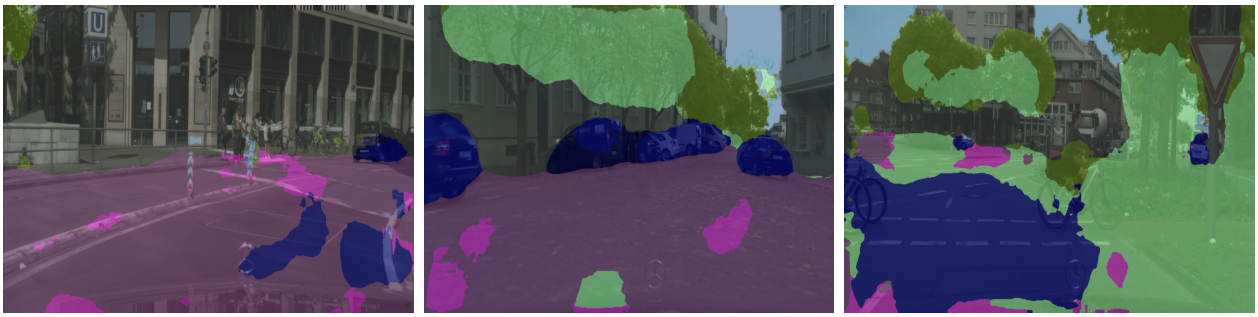

Fig. 14: DA PL fixed threshold (0.9)

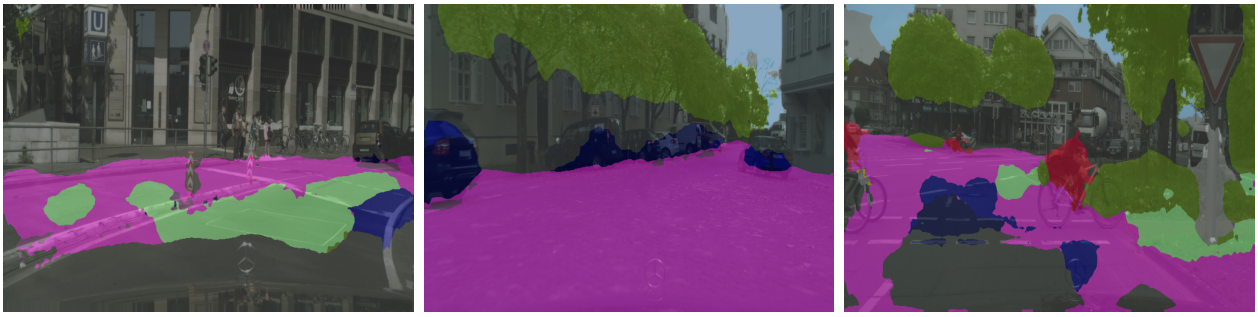

Fig. 15: DA PL variable threshold