The Visual Computer
https://doi.org/10.1007/s00371-024-03373-8

RESEARCH

Per-class curriculum for Unsupervised Domain Adaptation in semantic segmentation

Roberto Alcover-Couso[1] · Juan C. SanMiguel[1] · Marcos Escudero-Viñolo[1] · Pablo Carballeira[1]

Accepted: 15 March 2024

Abstract
Accurate training of deep neural networks for semantic segmentation requires a large number of pixel-level annotations of real images, which are expensive to generate or not even available. In this context, Unsupervised Domain Adaptation (UDA) can transfer knowledge from unlimited synthetic annotations to unlabeled real images of a given domain. UDA methods are composed of an initial training stage with labeled synthetic data followed by a second stage for feature alignment between labeled synthetic and unlabeled real data. In this paper, we propose a novel approach for UDA focusing the initial training stage, which leads to increased performance after adaptation. We introduce a curriculum strategy where each semantic class is learned progressively. Thereby, better features are obtained for the second stage. This curriculum is based on: (1) a class-scoring function to determine the difficulty of each semantic class, (2) a strategy for incremental learning based on scoring and pacing functions that limits the required training time unlike standard curriculum-based training and (3) a training loss to operate at class level. We extensively evaluate our approach as the first stage of several state-of-the-art UDA methods for semantic segmentation. Our results demonstrate significant performance enhancements across all methods: improvements of up to 10% for entropy-based techniques and 8% for adversarial methods. These findings underscore the dependency of UDA on the accuracy of the initial training. The implementation is available at https://github.com/vpulab/PCCL.

Keywords Semantic Segmentation · Unsupervised Domain Adaptation · Curriculum learning · Synthetic data

1 Introduction

Semantic Segmentation (SS) refers to the task of classifying each pixel in a given image to its semantic category. Convolutional Neural Networks (CNN) have achieved significant advances in SS [1–5]. However, effective training of CNN-based architectures requires a large number of labeled images. The annotation process is a time-consuming task, taking an average of 90 min per image [6], hindering the development of large-scale datasets. Furthermore, standard deep learning models are prone to overfit to small-scale datasets with few labeled images [7]. To confront these limitations, abstracting knowledge from automatically annotated synthetic images to real images has become popular in recent years [8–13].

However, a model trained from a source domain typically presents a significant drop in performance when evaluated on a target domain. This is commonly known as domain gap. To minimize the effects of domain gap, Unsupervised Domain Adaptation (UDA) considers knowledge transfer from a large set of annotated source data (e.g., synthetic) to target data where labels are not available (e.g., real). Typically, UDA in SS considers the synthetic-to-real adaptation and follows a two-stage training [10–15], where first the architecture of choice is trained using only synthetic images and their ground truth labels until convergence. Secondly, an alignment to real images is performed. Many works focus on the alignment of the learned features from synthetic images to real images, as this stage drives the final performance of the model. However, we argue that if better and more general features are

✉ Roberto Alcover-Couso
roberto.alcover@uam.es

Juan C. SanMiguel
juancarlos.sanmiguel@uam.es

Marcos Escudero-Viñolo
marcos.escudero@uam.es

Pablo Carballeira
pablo.carballeira@uam.es

[1] Video Processing and Understanding Lab, Universidad Autónoma de Madrid (UAM), 28049 Madrid, Spain

Published online: 15 April 2024

learned from synthetic data in the first stage, the subsequent alignment process will inherently be more effective.

To that end, we propose a curriculum-based learning (CL) strategy [16]. We divide the training set into subsets for a gradual introduction of semantic classes in terms of increasing difficulty. The per-class difficulty is based on self-taught learning (defined by a *scoring function*), which is obtained from the performance of the same model trained in a standard manner employing source data. Moreover, our proposal also includes a *pacing function* for easy-to-hard introduction of classes in training while maintaining comparable training times. Finally, the *per-class training loss* is adapted from the traditional image segmentation loss for operating at class level while maintaining the spatial context of each pixel. Defining this incremental difficulty order is unattempted for SS, probably due to the additional complexity of having multiple labels per image, as this precludes estimating the difficulty for the whole image (as in classification [17]). The validity of the proposal is demonstrated by improving the performance of the main adversarial-based and entropy-based UDA methods [10–13] on the GTAV-to-Cityscapes synthetic-to-real setup. The performance gain is specially notable for the less represented semantic classes in the source annotated data.

Our contribution is threefold:

- Highlighting the significance of training for the first stage in UDA. We shed light on a crucial but often overlooked aspect in UDA research: the importance of the initial model weights that will be used for feature alignment (second stage in UDA). This contribution underscores the impact of well-tailored initial weights on UDA performance, a topic that has not received adequate attention in the existing literature.
- Introducing a novel curriculum learning framework. We propose to leverage curriculum learning for improving the initial training with synthetic data. This proposal is decomposed into a scoring function, pacing strategy and a per-class loss, showcasing improvements across multiple UDA methods.
- Extensive empirical validation. Through our experiments, we not only validate the effectiveness of our proposed framework but also achieve better-performing models from entropy-based methods than the usually top-performing adversarial-based methods. Furthermore, we propose a unified analysis between methods with homogeneous hyperparameter settings.

The paper is organized as follows. Section 2 reviews the state-of-the-art on UDA and CL. Section 3 describes the proposed per-class curriculum. Section 4 presents the experimental results, including a comparison with the state of the art. Finally, conclusion remarks are described in Sect. 5.

2 Related work

In this section, we review different techniques for UDA in SS and CL. As per-class curriculum methods, to the best of our knowledge, have not yet been proposed for SS, we discuss the general aspects on other related tasks in Sect. 2.2.

2.1 Unsupervised Domain Adaptation for semantic segmentation

CNN-based UDA follows a two-stage scheme [10–13], where first the model is solely trained employing source images in a standard manner. The so-trained weights are later refined by a second stage of alignment to learn domain-invariant features so that the discrepancy across source and target domains is reduced. To quantify the discrepancy between the source and the target feature distributions, different alignment metrics are proposed to drive this second stage [10–13, 18]. Minimizing a defined metric aims at producing domain-agnostic features, which can be used to train a classifier capable of generalizing to both domains indistinguishably [15]. Two main methods exist for this second stage based on Adversarial Networks and Entropy minimization. To the best of our knowledge, the initial training has not been the aim of study of previous research.

Adversarial UDA Adversarial training follows the typical adversarial scheme of Generative Adversarial Networks (GANs) [12, 19, 20]. This scheme defines a min-max game where the segmentation network has two tasks: performing a good segmentation on source images and fooling the discriminator into believing that its output on the target images comes from source images. Meanwhile, the discriminator tries to differentiate if the output of the segmentation results from the analysis of an image from the source or the target dataset. Domain alignment is facilitated by rendering the features indiscernible to a discriminator, thereby assisting the classifier in making decisions that are extrapolable across domains. The pioneering adversarial approach for UDA was introduced by [19], with [21] extending this methodology to SS. They achieved this by leveraging the segmentation's final features within the discriminator. This was later extended by including a pixel-level domain classification [12]. By employing the output of the segmentation head instead of the final feature, the adaptation is expected to be performed across all levels of the segmentation, thus improving the results. Additionally, visual-style classifier has been proposed to produce a closer alignment on visually similar images [15]. Currently, it is believed that the domain discrimination power is deeply correlated to the semantic category a pixel represents [11]. Therefore, proposals include the predicted class into the discriminator loss [11, 22]. ***Entropy minimization UDA*** Visual discrepancies present on source and target data hinder the confidence of the model on the tar-

get data, promoting the learning of *close-to-uniform* output probability distributions per sample [10]. This is alleviated through entropy minimization by sharpening the probability distribution of the model on the target data so high probabilities are predicted for fewer classes. Recent efforts employ the output probabilities of the model to modify the loss to speed up training [13], or reduce the impact of high confidence mis-classifications [11]. MaxSquare [13] demonstrated that the entropy loss gradient decreases the more uniform a probability distribution is, thus making the initial stages of the training sub-optimal. Therefore, they proposed to employ a quadratic formulation to increase the slope of the gradient in the initial stages. Current works propose to minimize the relative entropy of the predicted probabilities to a smoothed version of the source semantic labels. This smoothing provides better generalization to the target dataset by penalizing mis-classifications and overly confident classifications [23].

The efficacy of both domain alignment mechanisms is intricately linked to the model's initial performance. Entropy minimization approaches focus on refining the model's classifications on the target dataset, indicating that a superior initial model can lead to more effective final training outcomes. However, should the model's initial predictions on the target data be largely inaccurate, it risks reinforcing these misclassifications, significantly hindering performance. Conversely, adversarial methods strive to make the feature representations from different domains indistinguishable. This approach necessitates a feature extractor that is inherently robust and capable of comprehending both source and target datasets effectively.

Concluding our comparative analysis, while entropy-based methods offer valuable insights within probability-based frameworks, their applicability remains limited to scenarios where probabilistic interpretations are feasible. In contrast, adversarial methods emerge as more versatile, extending their utility beyond mere probabilistic contexts to encompass a broader spectrum of machine learning paradigms, including clustering and regression.

2.2 Curriculum learning

CL is conceived as a protocol to gradually increase the complexity of the data employed during training [16]. It requires to define subsets of the training set that present similar complexity and include them into the training based on this complexity. Curriculum-based learning can be defined by two functions for scoring the data employed and pacing how the data are fed for training [24]. The following subsections describe these functions.

2.2.1 Scoring function

The scoring function estimates the learning difficulty of the samples. Methods employing scoring functions can be considered as continuation methods [25]. A continuation method is an established non-convex optimization protocol, where the optimization fine tunes from a simpler, smooth objective function to the goal non-convex objective function. For the context of deep learning, one expects that employing easy training samples would result in smaller training losses compared to training with the whole training set. Formally this expectation is because the objective function (typically cross-entropy) should be smoother on those easy samples [16]. As the complexity of the samples is increased, the objective function becomes less smooth, thus making CL a more effective and efficient learning framework compared to standard training. CL has proved its efficacy in various machine learning tasks [26, 27]. In order to measure the difficulty, existing scoring functions can be categorized into the so-called *self-taught* scoring, which is employing the experience of training the network in a standard manner [17, 24, 28], or scoring based on data transformations [27, 29].

Self-taught scoring These functions are based on the results of training the model employing random mini-batches from the full dataset. This is a particularization of scoring based on the objective function. For example, [24] proposed to train a vanilla network to define a scoring function. Then, the loss of that network on each image defines the scoring: the lower the loss, the easiest an image is considered. However, this direct association is not straightforward in SS, as each pixel in an image has its individual loss. Therefore, there is no such assignment like the one proposed for image classification [24].

Scoring based on data transformations Differently, some works define a manual scoring based on data transformations to increasingly generate more complex versions of the images. As an example for object localization, [30] proposed to employ increasingly smaller scales and bigger rotation angles of 3D models of objects to implicitly define an easy-to-hard curriculum. Alternatively, the features produced by the model can be the subject of the transformation. One can define a curriculum by progressively increasing the learning capacity of the model. For instance, this can be achieved by blurring the convolutional activation maps during training (e.g., at each epoch) with Gaussians of progressively smaller variance [31].

Our scoring proposal builds upon the self-taught scoring strategy. By gradually learning the classes, we impose a smooth version of the loss which is gradually sharpened as more classes are included based on the previously estimated class difficulties. Therefore, as more classes are included, the loss is gradually sharpened from a few-classes problem to the complete multi-class one. However, our proposal separates

from previous work as we estimate difficulty at pixel level instead of globally for the whole image, which allows us to apply the proposal for semantic segmentation.

2.2.2 Pacing function

The pacing function determines the curriculum's update frequency, dictating when a new subset should be introduced into the training process. These functions can be classified into fixed-step and variable-step pacing [24].

Fixed-step pacing function This function defines a fixed amount of iterations to train on every subset. Current class-incremental methods in image classification propose to iteratively train a model G by adding a new class every T epochs, where T is the number of epochs to fully train the model G [32]. This process takes $T \cdot C$ epochs, where C is the number of classes. In SS, this protocol requires a huge amount of computational resources and longer training times, which may be unfeasible for resource-constrained settings. Typically, self-taught scoring and scoring based on data transformations proposals must manually define when to update the pacing function, thus reducing the pacing function to a scheduled update [24, 30, 31].

Variable-step pacing function This function dynamically selects when to sample from the next subset in the curriculum order, therefore defining an individual training time for each training set based on the evolution of the loss [33, 34] or by gradually increasing a fixed initial training time [24].

Our proposal lays between fixed and variable step pacing functions. A fixed training time is defined for each subset. However, prior knowledge on the amount of training samples available on each subset is used to filter out samples in each subset, thereby reducing the total training time without harming and even improving performance.

3 Per-class curriculum for semantic segmentation

In this section, we introduce the proposed curriculum-based learning for SS, and Fig. 1 provides an overview. We first discuss the SS framework, and later we focus on the key elements of the proposal: scoring function, pacing function and per-class loss.

3.1 Semantic segmentation

Let us consider training data for SS as the set $\mathbb{X} = \{(\mathbf{X}_i, \mathbf{Y}_i)\}_{i=1}^{N}$, where N is the number of images, $\mathbf{X}_i \in \mathbb{R}^{H\times W\times 3}$ is a $H \times W$ color image, $\mathbf{Y}_i \in \{1, C\}^{H\times W\times 1}$ is the associated pixel-level label map and C is the number of semantic classes.

Moreover, let $G(\mathbf{X}_i; \theta)$ be the segmentation model defined by the parameters θ and fed by an image $\mathbf{X}_i$, which predicts a C-dimensional probability map by virtue of a final softmax layer $P(G(\mathbf{X}_i); \theta) \in [0, 1]^{H\times W\times C}$, so that $\sum_{c=1}^{C} P(G(\mathbf{X}_i))^{h,w,c} = 1$ for any $h \in [1, H]$ and $w \in [1, W]$. $G(\mathbf{X}_i; \theta)$ is trained by optimizing a loss L_{seg} on $\mathbb{X}$, where typically the cross-entropy loss is employed to minimize the difference between the predicted probabilities and the pixel-level labels for each image $\mathbf{X}_i$ [35].

3.2 Curriculum

Our proposed per-class curriculum is based on a self-taught approach using a teacher–student framework [24]. First, a teacher model provides a *scoring function* to estimate the learning difficulty of each semantic class. Later, a student model is trained using the proposed per-class curriculum strategy, guided by the teacher model and a *pacing function* that determines how classes are injected in the iterative learning process. Moreover, our proposal employs a *per-class loss* that minimizes error impact on difficult classes, and progressively amplifies their importance as the model advances in curriculum learning. For simplicity, we consider that both the student and teacher have the same architecture, yet this has no implications on the generality of the proposed solution.

Scoring function We estimate the learning difficulty of each semantic class y_j as the corresponding Intersection-over-Union (IoU) performance obtained by a pre-trained model, which acts as our class-scoring function $f : [1, C] \rightarrow \mathbb{R}$. Specifically, the higher the performance score $f(c)$, the easier the class is considered. Later, we determine the curriculum learning order by sorting the classes from easiest to hardest estimated class difficulty (i.e., ordered by decreasing IoU performance), where the sorted classes are identified as $\{\varphi_i\}_{i=1}^{C}$ and φ_1 is the easiest and first class to be learned.

To define our scoring function $f(\cdot)$, two self-taught scoring functions are proposed (see Fig. 2): *Single-teacher* and *Multi-teacher*. For *Single-teacher*, a unique teacher model $M(\cdot)$ is obtained by training the model for the C-class segmentation problem with uniformly sampled batches from the entire source data. Then, we employ the so-trained model $M(\cdot)$ as the unique scoring function $f(\cdot)$. *Multi-teacher* refers to training C models $M_c(\cdot)$, where each one corresponds to a binary segmentation problem (i.e., one semantic class versus all the other classes). Therefore, each $M_c(\cdot)$ acts as the score function for the c-class. For both Single-teacher and Multi-teacher, the lower the performance, the higher the class difficulty.

Pacing function

We propose a strategy to incrementally include classes while maintaining a bounded training time. Formally, our pacing strategy considers a number steps $t \in T$, where one or mul-

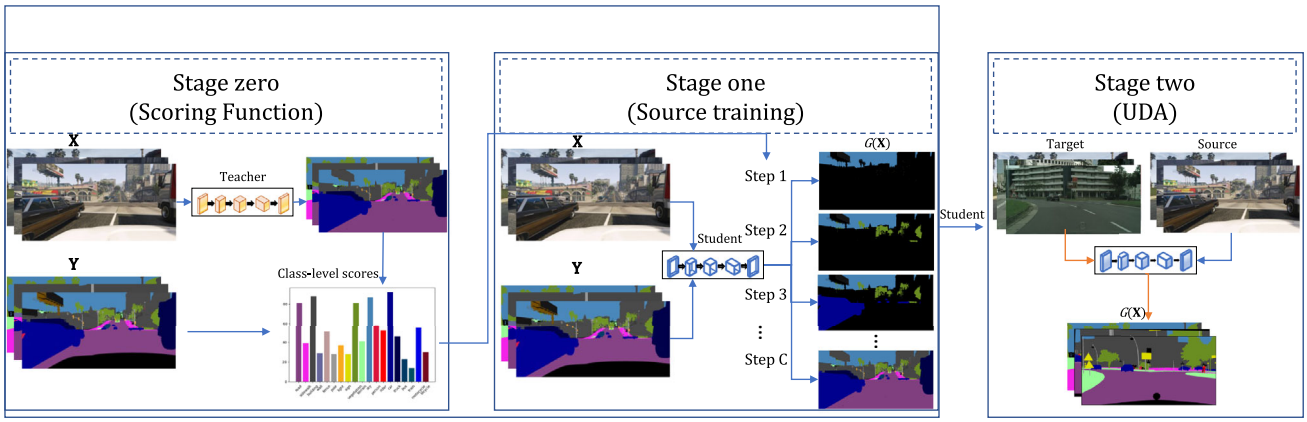


Fig. 1 Overview of the proposed per-class curriculum learning (PCCL) for Unsupervised Domain Adaptation in Semantic Segmentation. First, we define our scoring function in a self-taught framework to obtain the class-level scoring (scoring function). Second, this class-level scoring is employed to train gradually the student model (iterative curriculum-based learning). Finally, stage two performs adaptation to unlabeled real data by employing the weights obtained from stage one

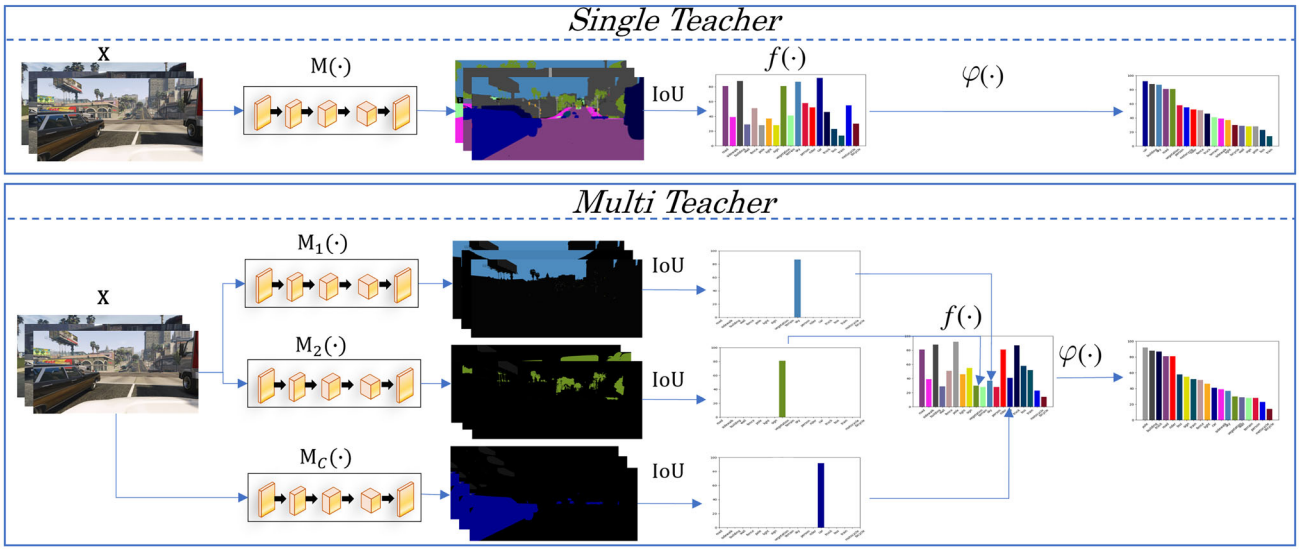


Fig. 2 Graphical illustration of the proposed scoring functions $f(\cdot)$, which act as teacher models in the proposed approach to define the class-ordered list $\{\varphi_i\}_{i=1}^{C}$ from easiest to hardest classes. We propose two strategies for the score function: In *Single-teacher*, we train a single model to predict all classes (top row), and in *Multi-teacher* we train one model for each class in a binary classification manner (bottom row)

tiple classes can be introduced for each learning step. To maintain simplicity and validate the utility of the scoring function, we assume that one class is introduced in each learning step, except during the initialization phase where one may consider multiple classes to speed up and increase the accuracy of the learning process.

For each step t, let us assume that class $\varphi_{i=t}$ is chosen for introduction into the learning process, requiring the collection of data associated with this class. A direct application of the resource-intensive standard CL [32] may progressively utilize sets of increasing size Φ_t, each one including every pixel for all images with annotated classes $\{\varphi_i\}_{i=1}^{t}$. As the sizes of these subsets Φ_t can grow substantially during the steps, our approach focuses on constraining the number of selected images while facilitating the learning of the new class. In particular, our proposal only utilizes subsets of images $\tilde{\Phi}_t$ from images $\mathbf{X}_k$ where there exist pixels (x_l, y_l) related to the new class $\varphi_{i=t}$ for learning:

$$(\mathbf{X}_k, \mathbf{Y}_k) \in \mathbb{X} \mid \exists (x_l, y_l) \in (\mathbf{X}_k, \mathbf{Y}_k) \wedge y_l = \varphi_{i=t}. \quad (1)$$

As for the labels $\mathbf{Y}_k$ associated with the chosen images $\mathbf{X}_k$ at step t, we include labels corresponding to class $\varphi_{i=t}$ as well as those belonging to classes learned in prior training

steps (i.e., $\{\varphi_i\}_{i=1}^{t-1}$) to prevent forgetting them. It is important to emphasize that labels from previously learned classes are exclusively considered in the selected images $\mathbf{X}_k$, and this selection may not encompass all images containing these classes, as used in previous iterations. Specifically, label sets are incremental, but image sets are not.

Figure 3 illustrates the process of gathering different subsets Φ_t and $\tilde{\Phi}_t$ for a four-image source dataset, where the subsets for the three first steps are displayed. The first step $t = 1$ considers the data for the easiest class g_1 (e.g., sky). The second step $t = 2$ accounts for the data of the two classes with lower learning difficulty, being g_1 and g_2 (e.g., sky and vegetation). The last step $t = T$ includes the data for all classes.

After every pixel in $\tilde{\Phi}_t$ has been employed for training, the learning process for the t step is completed and the pacing function moves to the next step $t + 1$. Therefore, a new subset $\tilde{\Phi}_{t+1}$ is gathered for the next training step, similarly as previously described.

In this proposal, the initial step may be a critical design choice to optimize the learning process, so we may consider the simultaneous learning of several classes. As previously stated in Sect. 2.2, the benefit of employing a scoring function is to simplify the learning task. Thus, if the initial number of classes is too large, the initial associated optimization task may be too complex, thus diminishing the potential benefit compared to not applying any curriculum. Moreover, a small initial number of classes will require longer training times as compared to start with one single class. For example, the number of selected pixels is expected to change when starting with one or two classes, as the number of images $\mathbf{X}_k$ containing one or two classes may differ. Then, we consider that the curriculum strategy can start with the first n classes, so the training subset for the first step is defined as

$$\tilde{\Phi}_1 = \bigcup_{i=1}^{n} (\mathbf{X}_k, \mathbf{Y}_k) \in \mathbb{X} \mid \exists (x_l, y_l) \in (\mathbf{X}_k, \mathbf{Y}_k) \wedge y_l = \varphi_i, \tag{2}$$

and the labels $\mathbf{Y}_k$ apply to all images where the classes $\{\varphi_i\}_{i=1}^{n}$ appear. The rest of subsets $\tilde{\Phi}_{t=2\ldots C-n}$ in the curriculum correspond to the learning of the classes $\{\varphi_i\}_{i=n+1}^{C}$, and they are selected as previously described in this subsection. This initial number of classes $\tilde{\Phi}_1$ enables to control the trade-off effect between efficiency and efficacy; a smaller-set initial step will theoretically provide better final performances at the cost of additional computational overhead compared to larger-set initial steps. In Sect. 4.2, different values for n are analyzed.

Per-Class curriculum loss

The training of SS architectures needs to employ full images, due to the impossibility of identifying the semantic class of a given pixel without its context. However, the proposed curriculum strategy relies on learning from pixels of selected classes instead of all the classes in the images of the source data. It is noteworthy to mention that the spatial context of selected classes is maintained as we consider the full RGB content of the selected images $\mathbf{X}_k$ at each step of the curriculum and learned incrementally by controlling the chosen labels $\mathbf{Y}_k$ as more classes are added to the curriculum. Hence, the cross-entropy loss cannot be directly applied to our curriculum using its standard definition [35]:

$$\mathcal{L}^{seg} = -\sum_{c=1}^{C} \hat{\mathbf{Y}}_{\mathbf{i}}{}^{c} \cdot log(P(G(\mathbf{X}_i))^c, \tag{3}$$

where $\hat{\mathbf{Y}}_{\mathbf{i}} \in \{0, 1\}^{H \times W \times C}$ is the one-hot encoding of the pixel-level labels $\mathbf{Y}_i$ corresponding to $\mathbf{X}_i$. Therefore, we need to modify the classical cross-entropy loss, to account for individual pixel information while maintaining the structure of feeding the full image to the training process. To do so for each step t, we define a per-class weighting factor $\mathbf{v}_t = \{v_t^c\}_{c=1}^{C}$, where each element v_t^c selects the class to consider ($v_t^c = 1$) or discards ($v_t^c = 0$) for the computation of the loss. This $\mathbf{v}_t$ is updated depending on the scoring function and the current step as

$$\mathbf{v}_{t+1} = \mathbf{v}_t + \mathbf{e}_{g(t)}, \tag{4}$$

where $\mathbf{e}_{g(t)}$ is a zero vector of length C with a single one value in the position $c = \varphi_{i=t}$. Our proposal for the per-class loss at pixel level is defined as:

$$\mathcal{L}_t^{PCCL} = -\frac{\sum_{c=1}^{C} v_t^c \cdot \hat{\mathbf{Y}}_i{}^{c} \cdot log(P(G(\mathbf{x}))^c}{\sum_{c=1}^{C} v_t^c \cdot \hat{\mathbf{Y}}_i{}^{c}}. \tag{5}$$

Building from the cross-entropy loss in Eq. 3, we marginalize the learning to the selected classes. Furthermore, with the denominator, we normalize the loss to present a stable slope as new classes are included.

Figure 4 illustrates how the learning process advances with the curriculum first classifying the whole image as a single label to then sequentially refine the prediction with different semantic instances. Please note how it first learns coarsely different areas where the first classes can be located to later enhance the prediction with each new class. This allows the network to learn that the sky is usually on top, vegetation in

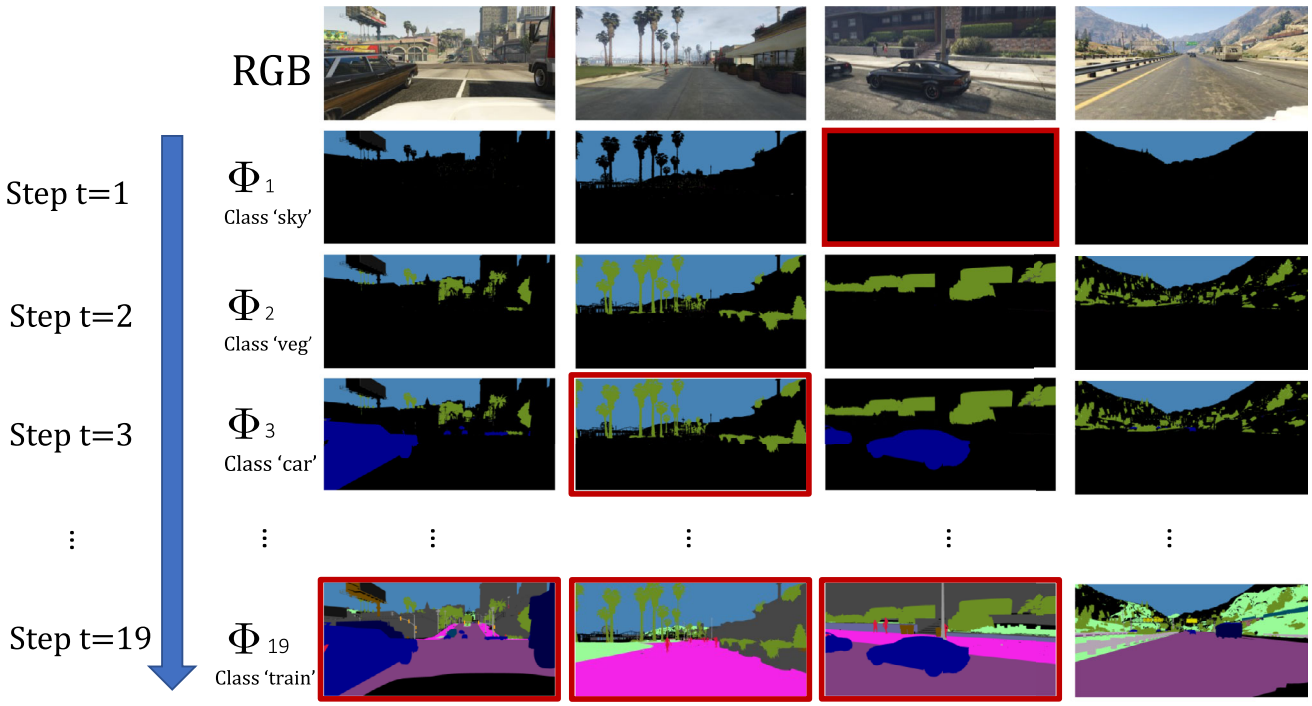


Fig. 3 Illustrative example of the sets of pixels Φ_t selected for each step t. The columns correspond to the selected pixels for four sample images using the Cityscapes color codes [6] (sky:light blue, vegetation:green and car:dark blue), the non-selected pixels are marked in black. The rows correspond to the sets for the first (Φ_1), second (Φ_2), third (Φ_3) and last (Φ_{19}) training steps for the $C = 19$ classes of the Cityscapes dataset. Our pacing proposal $\tilde{\Phi}_t$ filters out the images that do not contain pixels representing the target class to learn. The filtered images are highlighted in red. Excluding the highlighted images, the second row represents $\tilde{\Phi}_1$ composed of the pixels represented as the sky (i.e., the images containing pixels labeled as the sky; the third image is excluded). The third row represents $\tilde{\Phi}_2$, composed of the pixels represented as sky or vegetation of images containing pixels labeled as vegetation. Therefore, the four images are included. Subsequentially, the process is repeated for the 19 classes. Note that the same image can be included in several updates, but its knowledge contribution increases by expanding the previous one with that of the new classes

the lateral margins and cars are usually located around the center of the image— to name just some of the inductive biases characteristic of the Cityscapes dataset. Then, as the global position of previous semantic classes has already been learned, the focus of the training is to refine with the new classes rather than globally learning the specifics of each class.

4 Experiments

In this section, we present the experimental procedure and results for our proposal (PCCL): First, we introduce the datasets, evaluation metrics and implementation details. Second, we present the results of training with only synthetic data (stage one): validating the effectiveness of the pacing strategy and comparing the performance achieved with other alternatives. Finally, we discuss the downstream effects of our proposal when employing different state-of-the-art UDA methods (training with annotated synthetic data and unlabeled real color images).

4.1 Setup

Datasets To evaluate our proposal, we employ two popular UDA datasets for SS: GTAV [36] and Synthia [37] as source synthetic datasets and the Cityscapes [6] dataset as the target dataset.

GTAV is a synthetic dataset with urban scenes composed of 25K images from the game Grand Theft Auto V. GTAV shares 19 semantic classes with Cityscapes and is used as source training data.

Synthia is a collection of different synthetic urban scenes datasets for semantic segmentation; we pick its subset SYNTHIA-RAND-CITYSCAPES. Synthia is composed of 9,5K images rendered from a virtual environment. Synthia shares 16 semantic classes with Cityscapes and is employed as source training data.

Cityscapes is a real dataset with urban scenes generated by filming with a camera inside of a car while driving through different German cities. It consists of 3K images for training and 0.5K images for validation. ***Evaluation metrics*** The main metric employed for evaluation is consistent

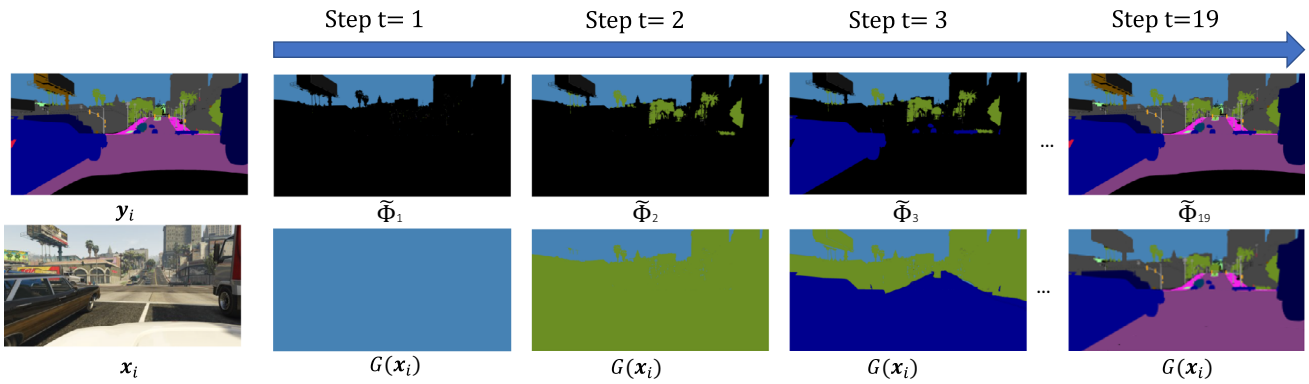


Fig. 4 Illustrative example of expected output (top) and output (bottom) of the network at the first three and last curriculum stages. The first column represents the GT and the color image, respectively. The following columns represent the pixels from the sample image belonging to each of the pacing sets $\tilde{\Phi}_t$ and the prediction of the network at the end of each curriculum step (taking as input the color image represented in the first column). Note how for a single image the knowledge of previous classes has been already acquired; thus, the learning is targeted toward understanding the new semantic class

Table 1 Performance comparison of different scoring functions and with the analyzed pacing training sets for the proposed PCCL approach (with $n = 3$ for the initial stage)

Scoring	Pacing set	Iterations (K)	mIoU	Δ_{mIoU} (%)	$\Delta_{\text{precision}}$ (%)	Δ_{recall} (%)	Δ_{F1} (%)
ST	Φ_t	80	35.7	0	0	0	0
	$\tilde{\Phi}_t$	32	**40.2**	12.6	10.3	14.5	13.1
MT	Φ_t	80	34.6	0	0	0	0
	$\tilde{\Phi}_t$	32	**37.0**	6.9	3.4	9.5	6.5
Borders [45]	Φ_t	80	33.9	0	0	0	0
	$\tilde{\Phi}_t$	32	**36.1**	6.4	2.8	9.0	5.8
# Images [46]	Φ_t	80	33.7	0	0	0	0
	$\tilde{\Phi}_t$	32	**35.0**	3.6	2.6	3.7	3.1

Our proposed scoring functions are the *Single-teacher* and *Multi-teacher* scoring functions. The compared training sets denote the employment of $\tilde{\Phi}_t$, i.e., our proposal for constrained data training, or Φ_t for training, i.e., the baseline pacing set selection [24]. All models are trained with source data without any domain adaptation (i.e., only GTAV dataset) and evaluated on the target validation set (Cityscapes dataset). This evaluation setup ensures that there is no dependency on the UDA strategy. (KEY. ST: *Single-teacher*, MT: *Multi-teacher*, Borders: number of shared Borders of each semantic class, # Images: number of images including each semantic class)

with the SS task: the PASCAL VOC per-class intersection over union (IoU) [38], between the model prediction and the ground-truth label. IoU measures at pixel-level the relationship between True Positives (TP), False Positives (FP) and False Negatives (FN): $IoU = \frac{TP}{TP+FP+FN}$. As a global performance evaluation metric, the mean IoU (mIoU) over all classes is employed. For the GTAV-to-Cityscapes problem, classes are shared between both datasets, whereas for the Synthia-to-Cityscapes problem, we evaluate the mIoU across the shared classes following the standard protocol [10, 12, 39]. Additionally, we present the precision: $Pr = \frac{TP}{TP+FP}$, recall: $Rc = \frac{TP}{TP+FN}$ and F1-score: $F1 = \frac{2RcPr}{Rc+Pr}$ results on the Cityscapes validation set.

Implementation details We employ Deeplabv2 [40] as the SS architecture, consistent with the state-of-the-art UDA methods [10–13, 15]. For the stage one of UDA in Sect. 4.2, we employ the GTAV dataset for training and evaluate with the Cityscapes validation set. For the stage two of UDA in Section 4.3, we employ the GTAV dataset or the Synthia dataset as the labeled source domain and the training images of Cityscapes as the unlabeled target domain. We evaluate the trained models on the Cityscapes validation set. All experiments employ up to four Titan RTX GPU with 24GB memory. Our models are trained using stochastic gradient descent optimizer with momentum of 0.9 and weight decay of 10^{-4} for a fair comparison with the selected state-of-the-art methods [10, 12, 21, 41–44]. For each of these methods, the training hyper-parameters are those reported in their respective papers [10, 42, 43].

Table 2 Performance comparison at class level for different scoring functions in our PCCL proposal

Scoring function	Pacing set	mIoU	per class																
			Road	Sidewalk	Building	Wall	Fence	Pole	Light	Sign	Vegetation	Terrain	Sky	Pedestrian	Rider	Car	Truck	Bus	Train	Motorcycle	Bicycle
ST	Φ_t		**81.8**	27.5	78.2	21.9	13.9	19.9	12.3	0.0	80.2	28.6	80.8	47.3	10.5	80.4	30.8	45.3	0.8	15.5	1.8
	$\tilde{\Phi}_t$		81.2	30.2	79.9	**25.4**	19.4	20.9	23.1	8.9	**83.1**	31.7	**81.9**	52.3	19.3	**81.6**	**31.6**	**43.5**	2.8	**28.2**	15.6
MT	Φ_t		77.3	25.2	78.7	20.6	20.3	19.1	10.5	0.0	79.9	29.7	80.9	48.3	14.8	80.5	28.8	24.8	0.0	17.3	1.2
	$\tilde{\Phi}_t$		74.1	30.0	**81.9**	21.6	**21.2**	25.2	21.6	6.0	82.2	**33.2**	78.8	**54.8**	**22.8**	65.0	30.6	7.9	**8.0**	20.7	**18.6**
Borders	Φ_t		73.4	23.7	78.6	21.6	19.9	20.8	9.1	0.1	79.9	29.4	81.4	46.9	0.0	78.6	29.5	42.3	0.9	8.5	0.0
	$\tilde{\Phi}_t$		74.0	31.9	81.5	24.7	20.0	26.4	18.9	**12.2**	81.3	30.3	78.9	39.3	12.7	60.6	29.5	41.3	0.9	10.8	4.3
# Images	Φ_t		68.1	33.1	66.7	18.6	23.4	33.2	28.7	5.3	82.2	25.7	70.3	46.9	17.1	74.8	10.8	24.2	0.0	12.2	0.0
	$\tilde{\Phi}_t$		76.6	**33.8**	81.5	23.0	19.3	28.0	21.1	6.4	81.2	30.4	79.1	41.4	13.1	62.3	22.6	18.0	5.3	11.8	5.1

All models are trained with source data without any domain adaptation (i.e., only GTAV dataset) and evaluated on the target validation set (Cityscapes dataset). This evaluation setup ensures that there is no dependency on the UDA strategy. Results represent the percentage change of employing $\tilde{\Phi}_t$ or Φ_t for training each round of the iterative learning process. Positive (negative) values indicate performance gain (loss). (KEY. ST: *Single-teacher*; MT: *Multi-teacher*; Borders: number of shared Borders of each semantic class; # imgs: number of images including each semantic class)

4.2 Results for synthetic training (stage one)

Scoring and Pacing functions First, we want to validate the effectiveness of our scoring functions: *Multi-teacher* and *Single-teacher*. To that end, we adopt two alternative scoring functions based on the number of shared Borders [45] and the number of images including each semantic class [46]. Table 1 compares the global performance and total training iterations of different scoring functions and pacing sets employed. Across both pacing functions, our scoring functions *Multi-teacher* and *Single-teacher* present better performance than the adapted scoring functions. Similarly, the filter-out pacing function $\tilde{\Phi}_t$ provides improvements to all of the studied scoring functions of up to 12% of mIoU. This enhancement is primarily attributed to an increase in the model's recall, suggesting a reduction in False Positives and more accurate classification of less prevalent classes.

These results are disentangled at a per-class level in Table 2. We find that *Multi-teacher* scoring function presents a greater improvement in some classes such as *fence, pole* and *train*, at the cost of worse performance in coarser classes such as *road, sky* and *car*. We believe this to be because the *Multi-teacher* assumes the semantic class *bus* to be one of the easiest classes to be abstracted. However, it is one of the hardest classes to generalize from synthetic data [10, 42]. We attribute this drop in performance to the introduction of a hard class in an initial curriculum stage disrupting the curriculum principle, thus resulting in worse global performance, as described by the so-called anti-curriculum [24]. On the other hand, *Single-teacher* defines a sorting seemingly more aligned with the difficulty of the network to learn each class, thus presenting a greater performance in 11 out of the 19 classes. Across all scoring functions, our pacing function notably enhances model accuracy for *train*, *bicycle*, *light*, *rider* and *sign*—classes that typically present significant adaptation challenges in state-of-the-art methods.

In order to compare the pacing proposals, global performance is not the only measurement. A good CL scheme should mainly drive performance gains to harder subsets [24], (i.e. subsets added last in the training). In Fig. 5, we compare the per-class performances obtained from the two pacing functions sorted by the place in the curriculum, that is, employing $\tilde{\Phi}_t$ with respect to employing Φ_t across the different scoring functions. It can be seen how the position in the scoring function correlates with the improvement in our proposal, having a greater impact on the last classes added to the curriculum. Therefore, $\tilde{\Phi}_t$ seems to perform a better CL scheme as the main goal is to improve the performance of "hard classes."

Initial number of classes

As mentioned in Sect. 3, our PCCL proposal depends on the initial number of classes n. This hyper-parameter affects

Table 3 Analysis of the hyper-parameter n representing the initial number of classes

Method	Initial classes	mIoU	Recall	Precision	F1-score	Iterations (K)
NCL [11]-reported	19	37.0	47.8	65.4	55.2	20
NCL [11]-retrained	19	37.7	49.4	66.2	56.6	32
PCCL	1	40.2	53.7	68.2	60.1	40
PCCL	2	40.2	52.3	68.7	59.4	36
PCCL	3	40.2	52.1	68.9	59.3	32
PCCL	4	39.7	50.8	67.6	58.0	29

The performance is measured by the stage one training of only synthetic data with a *Single-teacher* scoring function. The "NCL" corresponds to the results of stage one in FADA [11] a state-of-the-art method that does not employ curriculum. Moreover, we also retrain [11] to increase the number of training iterations, while keeping the setup as [11]. All models are trained with source data without any domain adaptation (i.e., only GTAV dataset) and evaluated on the target validation set (Cityscapes dataset). This evaluation setup ensures that there is no dependency on the UDA strategy

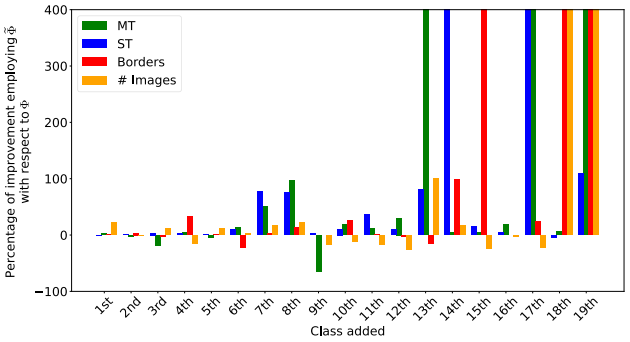


Fig. 5 Relative performance comparison for different scoring functions in our PCCL proposal. All models are trained with source data without any domain adaptation (i.e., only GTAV dataset) and evaluated on the target validation set (Cityscapes dataset). This evaluation setup ensures that there is no dependency on the UDA strategy. Results represent the percentage change of employing $\tilde{\Phi}_t$ or Φ_t for training each round of the iterative learning process. Positive (negative) values indicate performance gain (loss). Notably, each scoring function defines a different order—wherein the n^{th} class differs based on the scoring criteria—thus the lack of specific class names. (KEY. ST: *Single-teacher*, MT: *Multi-teacher*, Borders: number of shared Borders of each semantic class, # Images: number of images including each semantic class)

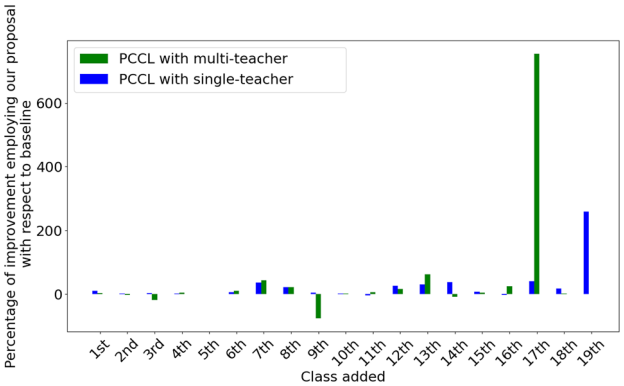


Fig. 6 Relative performance improvement in the proposed scoring functions in the PCCL approach as compared to the reported results of stage one in FADA [11]. All models are trained with source data without any domain adaptation (i.e., only GTAV dataset) and evaluated on the target validation set (Cityscapes dataset). This evaluation setup ensures that there is no dependency on the UDA strategy. Positive (negative) values indicate performance gain (loss). Notably, each scoring function defines a different order—wherein the n^{th} class differs based on the scoring criteria—thus the lack of specific class names

both the training time and the performance (see Table 3). There is a significant gain in performance regardless of the initial number of classes. However, less or equal training time is only achieved with $n \geq 3$. Therefore, for the following experiments, $n = 3$ is selected as the initial number of classes.

Comparison with the absence of curriculum Figure 6 compares the per-class performance of the two proposed scoring functions to the current state-of-the-art for synthetic-only trained models FADA (stage one) [11]. We outperform in most classes the reported results for stage one. This improvement is especially notable in classes included in the final stages of the curriculum. In Fig. 7, we present a comparison of some class losses for the model trained with our proposed *Single-teacher* PCCL and employing $\tilde{\Phi}_t$ as our training sets against the losses obtained from the reported synthetic only model [11] on the GTAV dataset. Note that both models are frozen during computation and both losses are computed equally. Our guided learning allows the model to better learn small (*pole*) and low-populated (*wall*) semantic classes (in the source training set) while maintaining similar or even better performances on coarse and highly populated semantic classes such as *sky*, *sidewalk* and *vegetation*. For the remaining experiments, *Single-teacher* is selected as the default scoring function.

Impact on the backbone employed To conclude this subsection, Table 4 showcases the enhancement in performance afforded by our PCCL across a variety of segmentation architectures (Deeplabv2 and FCN as segmentation head and VGG, ResNet50 and ResNet101 as backbones) when trained on the GTAV dataset and validated on the Cityscapes val-

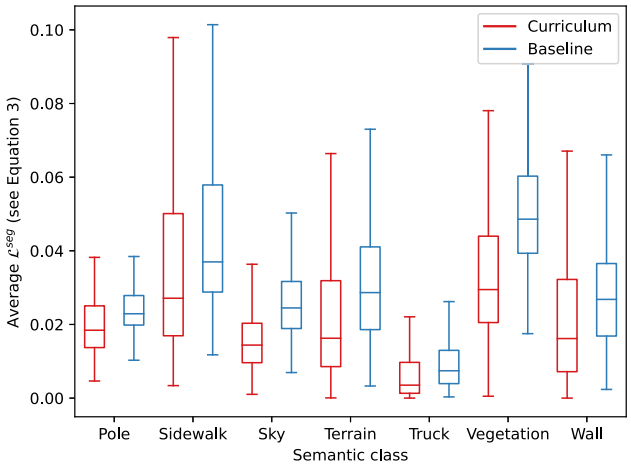


Fig. 7 GTAV per-class loss comparison of our PCCL proposal against the retrained loss of [11] for the initial training stage where only synthetic data is employed (GTAV train set [36]). Loss computed, CE Loss (see Eq. 3) employing the frozen models on all the GTAV dataset

idation set. This improvement across different backbones indicates that PCCL boosts the model's ability to adapt from source domain training to the target data, despite the model not having direct exposure to target images during training. On a detailed level, while performance gains are observed across all semantic classes, they are particularly pronounced for low-represented categories. However, FCN-based models, tend to misclassify smaller objects such as *Rider* or *Bicycle* as coarser classes like *Road*. Therefore, improving the identification of vehicles and *Rider* contributes to reducing False Positives, thereby elevating the *Roads* mIoU.

4.3 Results for Unsupervised Domain Adaptation (stage two)

Exploring source performance transfer to target As the source datasets (GTAV and Synthia) do not present a validation set, we generate a partition 80–20 from the GTAV training set (20k Images for train and 5k images from validation) preserving the original class distribution. This split was used for training two models, one with a state-of-the-art baseline semantic segmentation learning strategy [11] and another with our proposed PCCL framework (Both models are trained on a Deeplabv2 with a ResNet101 backbone). Table 5 compiles the experimental results for the source validation set (GTAV dataset), the target validation set at stage 1 (Cityscapes) and the target validation set at stage 2 using both labeled source data and unlabeled target data (GTAV-to-Cityscapes). Our experimental results suggest that our PCCL globally obtains better performance in the analyzed domains: source (67.6 vs 68.1), target without UDA (32.2 vs 38.7) and target after UDA (39.3 vs 42.8). We significantly improve the least represented classes in the source dataset (e.g., train and

Table 4 Performance comparison of different SS architectures

Method	mIoU per class																		mIoU
	Road	Sidewalk	Building	Wall	Fence	Pole	Light	Sign	Vegetation	Terrain	Sky	Pedestrian	Rider	Car	Truck	Bus	Train	Motorcycle	Bicycle	
FCN-VGG8	26.0	14.9	65.1	5.5	12.9	8.9	6.0	2.5	70.0	2.9	47.0	24.5	0.0	40.0	12.1	1.5	0.0	0.0	0.0	17.9
PCCL+FCN-VGG8	**30.9**	**15.1**	**66.8**	**1.6**	**15.7**	**9.3**	**9.2**	**3.1**	**73.7**	**4.1**	**51.2**	**26.4**	**0.2**	**46.4**	**13.3**	**5.1**	**0.3**	**2.1**	**1.3**	**20.1**
FCN-RN50	25.3	13.7	56.8	2.7	17.2	21.2	20.0	8.7	75.3	11.2	72.0	45.7	4.9	42.2	14.2	20.2	0.4	19.5	0.0	24.8
PCCL+FCN-R50	**70.4**	**15.2**	**58.0**	**6.1**	**20.8**	**22.4**	**22.5**	**8.0**	**77.0**	**11.2**	**75.3**	**47.0**	**10.0**	**51.4**	**14.8**	**31.8**	**0.5**	**20.1**	**3.7**	**30.2**
DeeplabV2-R101	73.4	23.7	78.6	22.9	19.0	20.5	8.7	0.5	79.9	29.4	81.4	46.9	3.0	78.6	29.5	42.3	0.9	3.0	5.5	37.0
PCCL+DeeplabV2-R101	**81.2**	**30.2**	**79.9**	**25.4**	**19.4**	**20.9**	**23.1**	**8.9**	**83.1**	**31.7**	**81.9**	**52.3**	**19.3**	**81.6**	**31.6**	**43.5**	**2.8**	**28.2**	**15.6**	**40.2**

All models are trained with source data without any domain adaptation (i.e., only GTAV dataset) and evaluated on the target validation set (Cityscapes dataset). This evaluation setup ensures that there is no dependency on the UDA strategy. The name of the architecture is formated as: classification head-backbone. PCCL-trained models employ *Single-teacher* as the scoring function and the initial number of classes is set to 3. Improvements are indicated with bold characters

bicycle). Conversely, more prevalent classes are introduced earlier in our PCCL curriculum, and we observed a slight decrease in performance as compared to the model trained without PCCL. Moreover, our PCCL shows a more even performance across all classes. This helps to avoid overfitting to dominant classes in the source domain, which could make the model to wrongly classify pixels in unseen/target domains as the dominant classes in the source domain.

Application to popular Unsupervised Domain Adaptation methods

Table 6 compiles the results for employing our PCCL proposal in popular alignment methods based on adversarial training (Advent [10], AdaptSegNet [12] and FADA [11]) and entropy (MinEnt [10] and MaxSquare [13]). In particular, we compare the performance of the selected methods with and without our PCCL pretraining, that is, employing as initial weights for the stage two our *Single-teacher* model, or employing weights from training without CL. Results show a performance improvement for all methods, thus demonstrating the effectiveness of the initial training stage. While adversarial training tends to be unstable and less reliable [10], entropy minimization techniques rely largely on the weights obtained on stage one. Therefore, our proposal greatly benefits the latter.

Setting equal training hyper-parameters for all methods

A major drawback found in existing literature is the disparity in the experimental conditions (e.g., hyperparameters). Therefore, a significant inconsistency is present when comparing these methods, due to different batch sizes (Advent = 1, FADA=8), different data augmentations (Advent= None, FADA= Random Horizontal Flip and Color jitter, MaxSquare= Random Horizontal Flip, Color jitter and Random Blurr), different amount of iterations for stage two training (FADA=40k, MaxSquare= 80K, AdaptSegNet=150K) and different amount of iterations for stage one training (FADA=20k, MaxSquare= 80K). Note that all the selected methods employ the same network architecture and these parameters are not part of their proposal; therefore, the methods should be compared in a similar matter. However, we acknowledge that larger training times may hinder the performance of the models; therefore, we always compare the best performance obtained throughout the training, not the final performance. For fair comparisons, we propose to retrain these methods using the provided source codes to homogenize the hyper-parameters, with a batch size of 4 and 120K total training iterations and accuracy effective data augmentation (color jitter and random horizontal flip). As our stage one requires 12K additional iterations, we train 12K less iterations in the stage two to compensate the initial computational overhead. All our retrained methods employ for the stage two the weights obtained from the retrained model of stage one for [11], see Table 3.

Table 5 Performance comparison of source and target performance of stage 1 training (synthetic only) with a random subset of the GTAV dataset

Method	mIoU per class																		mIoU
	Road	Sidewalk	Building	Wall	Fence	Pole	Light	Sign	Vegetation	Terrain	Sky	Pedestrian	Rider	Car	Truck	Bus	Train	Motorcycle	Bicycle	
Source performance (GTAV dataset)																				
w/o PCCL	94.4	71.3	93.4	51.6	46.8	48.5	55.9	61.0	91.5	40.4	92.0	72.1	38.3	91.8	74.3	77.9	59.1	49.2	47.4	67.6
PCCL	91.1	69.5	88.9	**55.2**	**49.9**	**52.6**	**58.6**	**64.4**	88.4	**44.3**	89.7	**73.4**	**44.5**	89.5	**77.3**	**80.5**	**66.4**	**54.9**	**55.3**	**68.1**
Target performance (Cityscapes dataset)																				
w/o PCCL	71.1	23.0	78.1	22.0	17.2	12.3	5.3	0.0	79.7	27.6	**81.4**	45.3	1.1	79.3	22.3	41.3	0.0	2.1	0.0	32.2
PCCL	**80.3**	**27.8**	**79.8**	**24.6**	**19.0**	**16.8**	**20.6**	**7.3**	**82.1**	**30.5**	80.1	**51.6**	**17.6**	**80.3**	**31.1**	**42.1**	**2.1**	**26.3**	**14.8**	**38.7**
UDA performance [10] (GTAV-to-Cityscapes setup)																				
w/o PCCL	84.4	23.5	83.8	23.0	19.7	15.2	21.1	14.4	84.3	30.9	85.5	55.1	17.2	81.7	30.8	41.0	0.0	20.6	15.3	39.3
PCCL	**88.1**	**31.6**	**82.8**	**25.7**	**27.3**	**18.7**	**23.7**	**16.3**	**84.4**	**35.7**	82.7	55.0	**20.0**	**82.1**	**40.2**	**46.9**	**0.6**	**26.5**	**25.0**	**42.8**

Target performance evaluated on the Cityscapes validation set. PCCL-trained model employs *Single-teacher* as the scoring function, and the initial number of classes is set to 3. Improvements are indicated with bold characters

Table 6 Performance comparison of the Unsupervised Domain Adaptation setup from GTAV-to-Cityscapes against the reported results of the 6 proposed methods

Method	mIoU per class																	mIoU
	Road	Sidewalk	Building	Wall	Fence	Pole	Light	Sign	Vegetation	Terrain	Sky	Pedestrian	Rider	Car	Truck	Bus	Train	Motorcycle	Bicycle	
AdaptSegNet [12]	86.5	36.0	79.9	23.4	23.3	23.9	35.2	14.8	83.4	33.3	75.6	58.5	27.6	73.7	32.5	35.4	3.9	30.1	28.1	42.4
PCCL+AdaptSegNet	**90.7**	**48.1**	**82.2**	**37.7**	19.4	**27.2**	28.9	**16.2**	83.3	**34.3**	**82.6**	53.4	16.0	**84.2**	32.3	**45.8**	2.0	**35.7**	1.6	**43.2**
Advent [10]	89.9	36.5	81.6	29.2	25.2	28.5	32.3	22.4	83.9	34.1	77.1	57.4	27.9	83.7	29.4	39.1	1.5	28.4	23.3	43.8
PCCL+Advent	89.8	**43.5**	**82.4**	**36.1**	24.5	**30.1**	32.1	21.4	83.4	**35.5**	76.8	55.7	14.4	**84.3**	**42.6**	**48.5**	**6.1**	**32.0**	6.8	**44.5**
FADA [11]	86.9	38.0	83.4	36.9	24.6	31.1	35.4	22.0	82.8	36.3	82.9	58.1	34.0	82.88	31.5	35.7	23.3	33.5	31.6	46.9
PCCL+FADA	**90.2**	**44.8**	82.9	28.0	**26.0**	29.7	**40.8**	**30.5**	**83.4**	**36.5**	**83.3**	**62.7**	30.3	**86.7**	**37.4**	34.7	11.6	30.9	27.9	**47.3**
MinEnt [10]	84.4	18.7	80.6	23.8	23.3	28.4	36.9	23.4	83.2	25.2	79.4	59.0	29.9	78.5	33.7	29.6	1.7	29.9	33.7	42.3
PCCL+MinEnt	**88.7**	**34.5**	**82.4**	**35.9**	18.3	21.4	28.9	17.9	**83.7**	**38.8**	**83.8**	55.7	20.1	**83.2**	**47.4**	**50.1**	0.5	29.5	20.9	**44.3**
MaxSquare [13]	89.4	43.0	82.1	30.5	21.3	30.3	34.7	24.0	85.3	39.4	78.2	63.0	22.9	84.6	36.4	43.0	5.5	34.7	33.5	46.4
PCCL+MaxSquare	76.4	32.7	**85.6**	**41.0**	**27.5**	**30.6**	**37.1**	**26.0**	**86.0**	**41.1**	**84.8**	59.7	20.6	83.0	**55.1**	**55.5**	2.7	**35.1**	27.1	**47.8**

mIoU is computed on the *Cityscapes* validation set. Improvements are indicated with bold characters

Table 7 summarizes the obtained results for the equal training setup. We also include the originally reported result as a reference. First, we would like to address that our PCCL-trained models outperform their respective reported baselines in the range of 1.8−10.5%. Note that our proposal surpasses the reported results of [11] with less memory requirements (half batch size). Second, it is worth addressing how entropy minimization methods (MinEnt and MaxSquare) benefit greater from our proposed initial pre-training than adversarial methods (Advent, AdaptSegNet, FADA), as we discuss later. Finally, on the same conditions, the alignment metric employed has little impact when compared to the batch size and the data augmentations employed. This can be seen by the relatively comparable performance across methods, while the reported results differ drastically (e.g., MinEnt present better results than FADA while having a reported result of less than 10%). Note that our results from Table 6 differ from the ones in Table 7 as in the former the hyper-parameters employed are the ones from each specific paper and in the latter are the unified ones.

Figure 8 compares the per-class results of models trained with and without our PCCL. For FADA (the best reported result) and MinEnt, we outperform the results for all classes in the range 0.25–150%. In line with the previous discussion, the better initial weights obtained from our proposal for stage one yield better performances for every class in the case of entropy minimization (MinEnt). Compared with MaxSquare, another entropy minimization framework, the nature of the proposed alignment imposes a strong bias to the low-represented classes in the source training set. As our initial PCCL training amplifies drastically the classification capabilities of those classes, our final model tends to expand the sets of samples predicted as *bike, train* and *rider*, therefore reducing the number of TP in surrounding coarser areas (*road*) and increasing the number of FP of low-represented classes (*truck, wall, bus* and *train*). However, overall, PCCL provides an improvement to 14 out of the 19 classes in the range of 5–50% mIoU. Regarding the adversarial methods (Advent and FADA), using PCCL for stage one yields final models with better performances for low-represented classes of the source dataset (*truck, wall, bus* and *train*), in the range of 40-200%, while presenting a consistent performance for the other 15 classes.

Figure 9 presents a qualitative comparison of reported models trained with adversarial adaptation (Advent) and entropy adaptation (MinEnt) against our models trained first with PCCL and then employing the same adaptation. Models trained without our PCCL training present similar mistakes due to the domain gap: the increased light reflection of white surfaces in the real domain (see the white building in the first column of Fig. 9), the reflection of the windows in vehicles (see the back window of the white car in the second column of Fig. 9) and the shape and size discrepancy to the source

Table 7 Performance comparison of the five selected UDA methods trained with and without our PCCL on the same experimental conditions in terms of pretraining time, data augmentations, batch size and testing protocol

Compared methods		Training setting			mIoU		Δ_{mIoU}	
Stage two alignment	Stage one synthetic training	BS	DA	Iterations (K)	Single scale	Multi scale	$\Delta_{\text{mIoU}_{\text{Single}}}$	$\Delta_{\text{mIoU}_{\text{Multi}}}$
AdaptSegNet [12]	NCL [11]-reported	2	0	150	42.4	–	0%	–
	NCL-retrained	4	2	120	45.1	46.0	6.5%	0%
	PCCL	4	2	120	46.0	46.6	8.5%	1.2%
Advent [10]	NCL [11]-reported	2	0	120	43.8	–	0%	–
	NCL-retrained	4	2	120	44.8	45.3	2.3%	0%
	PCCL	4	2	120	46.4	46.9	6.1%	3.6%
FADA [11]	NCL [11]-reported	8	2	60	–	46.9	–	0%
	NCL-retrained	4	2	120	45.5	46.3	0%	−1.2%
	PCCL	4	2	120	46.7	47.2	1.8%	0.58%
MinEnt [10]	NCL [11]-reported	2	0	120	42.3	–	0%	–
	NCL-retrained	4	2	120	45.6	46.2	7.8%	0%
	PCCL	4	2	120	46.7	47.3	10.5%	2.3%
MaxSquare [13]	NCL [11]-reported	4	3	160	45.2	46.4	0%	0%
	NCL-retrained	4	2	120	44.7	45.2	−1.2%	−2.6%
	PCCL	4	2	120	47.0	47.6	4.0%	2.6

(KEY. Reported: paper published results, Retrained: retrained results under homogeneous conditions. PCCL: proposed per-pixel curriculum learning, NCL: no curriculum learning, BS: batch size, DA: data augmentations (2 = horizontal flip + color jitter, 3 = 2 + Gaussian Blur). mIoU computed using single-scale testing ($mIoU_{\text{Single}}$) or multi-scale testing ($mIoU_{Multi}$), following respective papers testing protocol. In this vein, performance for validation setups not reported in the papers is indicated with an "–"

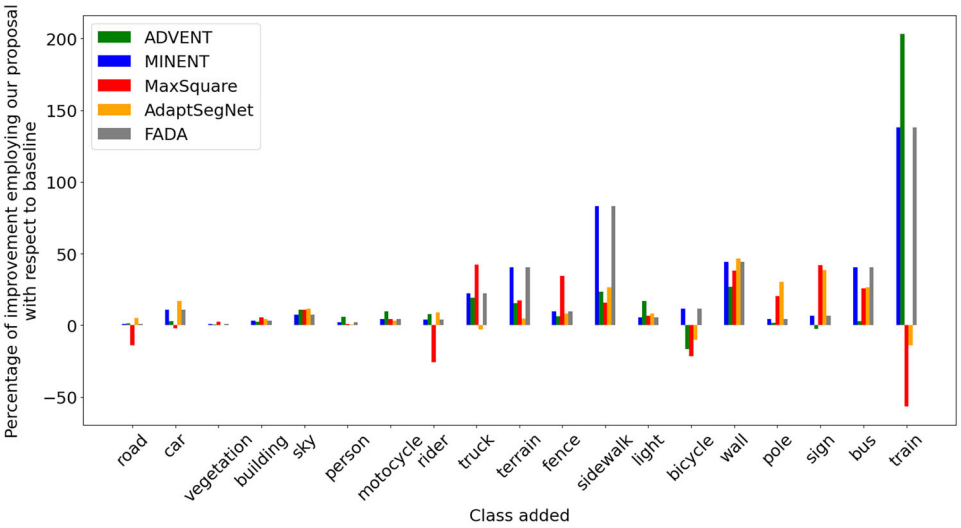


Fig. 8 Per-class performance comparison on the GTAV-to-Cityscapes problem of models trained with and without our PCCL training under the same experimental conditions (batch size of 4, horizontal flip and color jitter as data augmentation and multi-scale testing)

data (see the bus in the third column of Fig. 9). PCCL especially benefits both baseline models for small classes (*poles*, *traffic signs* and *traffic lights*) and low-represented classes in the source dataset (*rider*, *bus* and *terrain*)

Comparison with the state-of-the-art UDA methods In Table 8, we provide a comprehensive comparison between our PCCL-trained models and state-of-the-art methods utilizing DeepLabV2. We categorize these methods into two distinct groups: entropy-based [12, 13, 47–49] and adversarial-based [8, 10, 12, 15, 22, 39, 50] methods.

Notably, our PCCL-trained models consistently improve all evaluated methods, with particularly impressive results observed for the entropy-based approaches across the two studied source datasets. Even when taking the worst-performing framework, MinEnt [47], as the baseline, our PCCL-trained models achieve superior performance compared to significantly better-performing methods. This outcome strongly underscores the substantial benefits our PCCL pretraining approach brings to the model performance.

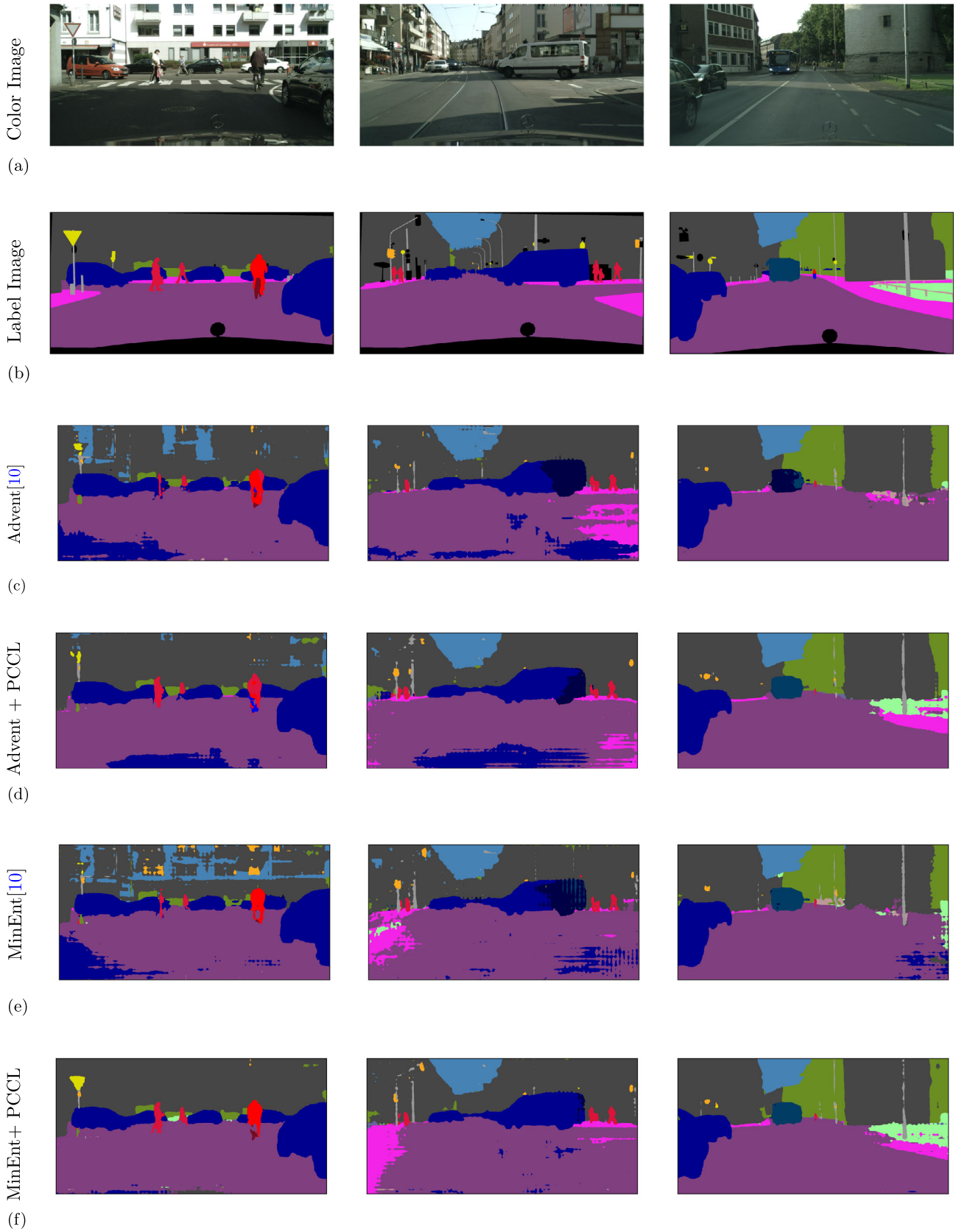


Fig. 9 Qualitative comparison of reported models and our models trained with PCCL on the Cityscapes validation set. First and second rows present the color and label images, respectively. Third and fifth rows present the reported models for Advent and MinEnt, respectively. Fourth and sixth rows present our results of applying PCCL for the initial training (only synthetic data) and then Advent and MinEnt, respectively, for the adaptation stage (labeled synthetic data and unlabeled real data)

Table 8 State-of-the-art performance comparison of the GTAV-to-Cityscapes (G2C) and Synthia-to-Cityscapes (S2C) problems of methods employing DeeplabV2

Alignment type	Framework	mIoU (G2C)	mIoU (S2C)
Adversarial	AdaptSeg [12]	42.4	46.7
	ADVENT[10]	43.8	47.6
	DAOT [51]	44.3	48.8
	CBST [50]	45.9	48.4
	CRA [39]	46.7	51.2
	LSR$^+$ [8]	46.9	–
	ASDM [15]	46.9	–
	FADA [11]	46.9	52.5
	CRST [22]	47.1	48.7
	FADA+PCCL	47.2	53.7
Entropy	MinEnt [10]	42.3	44.2
	SAN-SAW [47]	45.3	51.8
	WildNet [48]	45.8	–
	MaxSquare [13]	46.4	48.2
	SHADE[49]	46.7	–
	MinEnt+PCCL	47.3	53.9

mIoU is computed on the *Cityscapes* validation set. Results not reported in their original papers are marked with "–"

We posit that the initial weights of the model play a pivotal role in Unsupervised Domain Adaptation (UDA) and should be given due consideration in UDA research. Specifically, the superior performance of our PCCL-trained models, especially within entropy-based methods, suggests that starting with better-suited initial weights contributes to more effective training and better predictive capabilities during the training process. In contrast, adversarial-based methods rely on auxiliary models that are less affected by our PCCL initial training, resulting in comparatively smaller performance gains.

5 Conclusion

In this paper, we tackle the challenge of Unsupervised Domain Adaptation for Semantic Segmentation by introducing a novel approach called per-class curriculum learning (PCCL). Our method is founded on the principle that enhancing initial model weights through the use of synthetic data alone can lead to improved overall model performance for inter-domain feature alignment. We demonstrate the effectiveness of PCCL by showcasing superior performance across all semantic classes compared to conventional initialization methods. These advantages are subsequently transferred to the adapted models when applied to real-world data.

We validate our approach by a series of experiments and ablation studies as well as evaluating it under five state-of-the-art methods for Unsupervised Domain Adaptation. Our results consistently show significant performance enhancements across all selected methods. To ensure a fair and robust evaluation, we conduct additional experiments where we retrain all selected methods under uniform training conditions, including batch size, training duration, data augmentations and testing protocols. In these experiments, we observe that our proposal consistently delivers substantial improvements, particularly for semantic classes introduced in the later stages of the curriculum. Remarkably, these classes often correspond to underrepresented categories, underscoring the robustness and versatility of our approach.

In summary, our work offers a novel and effective solution for enhancing the performance of semantic segmentation models in Unsupervised Domain Adaptation scenarios, with a particular focus on improving underrepresented semantic classes.

Acknowledgements This work has been partially supported by the Spanish Government through its TED2021-131643A-I00 (HVD) and the PID2021-125051OB-I00 (SEGA-CV) projects.

Funding Open Access funding provided thanks to the CRUE-CSIC agreement with Springer Nature.

Data availability The datasets analyzed during the current work are publicly available in their respective repositories. The code will be publicly available upon acceptance in the PCCL repository: http://www-vpu.eps.uam.es/publications/PCCL/.

Declarations

Conflict of interest The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

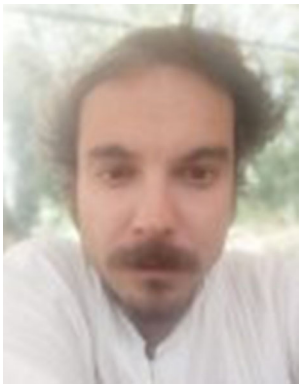
References

1. Feng, Q., Li, F., Li, H., Liu, X., Fei, J., Xu, S., Lu, C., Yang, Q.: Feature reused network: a fast segmentation network model for strip steel surfaces defects based on feature reused. Vis. Comput. (2023). https://doi.org/10.1007/s00371-023-03056-w
2. Bayoudh, K., Knani, R., Hamdaoui, F., Mtibaa, A.: A survey on deep multimodal learning for computer vision: advances, trends, applications, and datasets. Vis. Comput. (2022). https://doi.org/10.1007/s00371-021-02166-7
3. Long, J., Shelhamer, E., Darrell, T.: Fully convolutional networks for semantic segmentation. In: IEEE Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3431–3440 (2015). https://doi.org/10.1109/CVPR.2015.7298965
4. Chen, L.-C., Zhu, Y., Papandreou, G., Schroff, F., Adam, H.: Encoder–decoder with atrous separable convolution for semantic image segmentation. In: IEEE European Conference on Computer Vision (ECCV), pp. 833–851 (2018). https://doi.org/10.1007/978-3-030-01234-2_49
5. Huang, Y., Shi, P., He, H., He, H., Zhao, B.: Senet: spatial information enhancement for semantic segmentation neural networks. Vis. Comput. (2023). https://doi.org/10.1007/s00371-023-03043-1
6. Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B.: The cityscapes dataset for semantic urban scene understanding. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3212–3223 (2016). https://doi.org/10.1109/CVPR.2016.350
7. Zhao, X., Vemulapalli, R., Mansfield, P.A., Gong, B., Green, B., Shapira, L., Wu, Y.: Contrastive learning for label efficient semantic segmentation. In: Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV), pp. 10603–10613 (2021). https://doi.org/10.1109/ICCV48922.2021.01045
8. Barbato, F., Michieli, U., Toldo, M., Zanuttigh, P.: Road scenes segmentation across different domains by disentangling latent representations. Vis. Comput. (2023). https://doi.org/10.1007/s00371-023-02818-w
9. Liu, H., Li, C., Lei, D., Zhu, Q.: Unsupervised video-to-video translation with preservation of frame modification tendency. Vis. Comput. (2020). https://doi.org/10.1007/s00371-020-01913-6
10. Vu, T.-H., Jain, H., Bucher, M., Cord, M., Pérez, P.: Advent: adversarial entropy minimization for domain adaptation in semantic segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2512–2521 (2019). https://doi.org/10.1109/CVPR.2019.00262
11. Wang, H., Shen, T., Zhang, W., Duan, L., Mei, T.: Classes matter: a fine-grained adversarial approach to cross-domain semantic segmentation. In: IEEE European Conference on Computer Vision (ECCV), pp. 642–659 (2020). https://doi.org/10.1007/978-3-030-58568-6_38
12. Tsai, Y.-H., Hung, W.-C., Schulter, S., Sohn, K., Yang, M.-H., Chandraker, M.: Learning to adapt structured output space for semantic segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVF), pp. 7472–7481 (2018). https://doi.org/10.1109/CVPR.2018.00780
13. Chen, M., Xue, H., Cai, D.: Domain adaptation for semantic segmentation with maximum squares loss. In: IEEE Conference on Computer Vision (ICCV), pp. 2090–2099 (2019). https://doi.org/10.1109/ICCV.2019.00218
14. Shen, W., Peng, Z., Wang, X., Wang, H., Cen, J., Jiang, D., Xie, L., Yang, X., Tian, Q.: A survey on label-efficient deep image segmentation: bridging the gap between weak supervision and dense prediction. IEEE Trans. Pattern Anal. Mach. Intell. **45**(8), 9284–9305 (2023). https://doi.org/10.1109/TPAMI.2023.3246102
15. Luo, X., Chen, W., Liang, Z., Li, C., Tan, Y.: Adversarial style discrepancy minimization for unsupervised domain adaptation. Neural Netw. (2023). https://doi.org/10.1016/j.neunet.2022.10.015
16. Bengio, Y., Louradour, J., Collobert, R., Weston, J.: Curriculum learning. In: The International Conference on Machine Learning, pp. 41–48 (2009). https://doi.org/10.1145/1553374.1553380
17. Escudero-Viñolo, M., López-Cifuentes, A.: Ccl: class-wise curriculum learning for class imbalance problems. In: IEEE International Conference on Image Processing (ICIP) (2022). https://doi.org/10.1109/ICIP46576.2022.9897273
18. Wael, H.G., Aly, F.: A survey of text similarity approaches. Int. J. Comput. Appl. **68**(13), 13–18 (2013)
19. Ganin, Y., Ustinova, E., Ajakan, H., Germain, P., Larochelle, F.L.H., Marchand, M., Lempitsky, V.: Domain-adversarial training of neural networks. J. Mach. Learn. Res. **17**, 2096–2030 (2016)
20. Goodfellow, I., Pouget-Abadie, J., Mirza, M., Xu, B., Warde-Farley, D., Ozair, S., Courville, A., Bengio, Y.: Generative adversarial nets. In: Advances in neural information processing systems, pp. 2672–2680 (2014). https://doi.org/10.5555/2969033.2969125
21. Hoffman, J., Wang, D., Yu, F., Darrell, T.: Fcns in the wild: Pixel-level adversarial and constraint-based adaptation. arXiv:1612.02649 (2016) arXiv:1612.02649 [cs.CV]
22. Zou, Y., Yu, Z., Liu, X., Kumar, B.V.K.V., Wang, J.: Confidence regularized self-training. In: IEEE Conference on Computer Vision (ICCV), pp. 5981–5990 (2019). https://doi.org/10.1109/ICCV.2019.00608
23. Park, S., Kim, J., Heo, Y.S.: Semantic segmentation using pixel-wise adaptive label smoothing via self-knowledge distillation for limited labeling data. Sensors (2022). https://doi.org/10.3390/s22072623
24. Hacohen, G., Weinshall, D.: On the power of curriculum learning in training deep networks. In: International Conference on Machine Learning (ICML), pp. 2535–2544 (2019). https://doi.org/10.48550/arXiv.1904.03626.
25. Soviany, P., Ionescu, R.T., Rota, P., Sebe, N.: Curriculum learning: a survey. Int. J. Comput. Vis. (2022). https://doi.org/10.1007/s11263-022-01611-x
26. Qin, W., Hu, Z., Liu, X., Fu, W., He, J., Hong, R.: The balanced loss curriculum learning. IEEE Access (2020). https://doi.org/10.1109/ACCESS.2020.2970726
27. Ionescu, R.T., Alexe, B., Leordeanu, M., Popescu, M., Papadopoulos, D.P., Ferrari, V.: How hard can it be? estimating the difficulty of visual search in an image. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2157–2166 (2016). https://doi.org/10.1109/CVPR.2016.237
28. Gong, C., Tao, D., Maybank, S.J., Liu, W., Kang, G., Yang, J.: Multi-modal curriculum learning for semi-supervised image classification, pp. 3249–3260 (2016). https://doi.org/10.1109/TIP.2016.2563981
29. Pentina, A., Sharmanska, V., Lampert, C.H.: Curriculum learning of multiple tasks. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5492–5500 (2015). https://doi.org/10.1109/CVPR.2015.7299188
30. Hinterstoißer, S., Pauly, O., Heibel, T.H., Marek, M., Bokeloh, M.: An annotation saved is an annotation earned: using fully synthetic

training for object detection, pp. 2787–2796 (2019). https://doi.org/10.1109/ICCVW.2019.00340
31. Sinha, S., Garg, A., Larochelle, H.: Curriculum by smoothing. In: Larochelle, H., Ranzato, M., Hadsell, R., Balcan, M.F., Lin, H. (eds.) Advances in Neural Information Processing Systems, pp. 21653–21664 (2020). https://doi.org/10.5555/3495724.3497541
32. Cheng, H., Lian, D., Deng, B., Gao, S., Tan, T., Geng, Y.: Local to global learning: gradually adding classes for training deep neural networks. In: 2019 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), pp. 4743–4751 (2019). https://doi.org/10.1109/CVPR.2019.00488
33. Jiang, L., Meng, D., Zhao, Q., Shan, S., Hauptmann, A.G.: Self-paced curriculum learning. In: AAAI Conference on Artificial Intelligence, pp. 2694–2700 (2015). https://doi.org/10.5555/2886521.2886696
34. Ma, F., Meng, D., Xie, Q., Li, Z., Dong, X.: Self-paced co-training. In: International Conference on Machine Learning, vol. 70, pp. 2275–2284 (2017)
35. Hoyer, L., Dai, D., Van Gool, L.: HRDA: context-aware high-resolution domain-adaptive semantic segmentation. In: IEEE European Conference on Computer Vision (ECCV) (2022). https://doi.org/10.1007/978-3-031-20056-4_22
36. Richter, S.R., Vineet, V., Roth, S., Koltun, V.: Playing for data: ground truth from computer games. In: IEEE European Conference on Computer Vision (ECCV), pp. 102–118 (2016). https://doi.org/10.1007/978-3-319-46475-6_7
37. Ros, G., Sellart, L., Materzynska, J., Vazquez, D., Lopez, A.M.: The synthia dataset: a large collection of synthetic images for semantic segmentation of urban scenes. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) 2016-Dec, pp. 3234–3243 (2016) https://doi.org/10.1109/CVPR.2016.352
38. Everingham, M., Gool, L.V., Williams, C.K.I., Winn, J., Zisserman, A.: The PASCAL Visual Object Classes (VOC) challenge (2010)
39. Wang, Z., Liu, X., Suganuma, M., Okatani, T.: Unsupervised domain adaptation for semantic segmentation via cross-region alignment. Comput. Vis. Image Underst. (2023). https://doi.org/10.1016/j.cviu.2023.103743
40. Chen, L.-C., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L.: Deeplab: semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE Trans. Pattern Anal. Mach. Intell. (2018). https://doi.org/10.1109/TPAMI.2017.2699184
41. Sankaranarayanan, S., Balaji, Y., Jain, A., Lim, S.N., Chellappa, R.: Learning from synthetic data: addressing domain shift for semantic segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 3752–3761 (2018). https://doi.org/10.1109/cvpr.2018.00395
42. Zhang, Y., David, P., Foroosh, H., Gong, B.: A curriculum domain adaptation approach to the semantic segmentation of urban scenes. IEEE Trans. Pattern Anal. Mach. Intell. (2020). https://doi.org/10.1109/TPAMI.2019.2903401
43. Wang, Q., Gao, J., Li, X.: Weakly supervised adversarial domain adaptation for semantic segmentation in urban scenes. IEEE Trans. Image Process. (2019). https://doi.org/10.1109/TIP.2019.2910667
44. Biasetton, M., Michieli, U., Agresti, G., Zanuttigh, P.: Unsupervised domain adaptation for semantic segmentation of urban scenes. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops (2019). https://doi.org/10.1109/CVPRW.2019.00160
45. Borse, S., Wang, Y., Zhang, Y., Porikli, F.: Inverseform: a loss function for structured boundary-aware segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 5897–5907 (2021). https://doi.org/10.1109/CVPR46437.2021.00584
46. Tsvetkov, Y., Faruqui, M., Ling, W., MacWhinney, B., Dyer, C.: Learning the curriculum with Bayesian optimization for task-specific word representation learning. In: Computational Linguistics, pp. 130–139 (2016). https://doi.org/10.18653/v1/P16-1013
47. Peng, D., Lei, Y., Hayat, M., Guo, Y., Li, W.: Semantic-aware domain generalized segmentation. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 2584–2595 (2022). https://doi.org/10.1109/CVPR52688.2022.00262
48. Lee, S., Seong, H., Lee, S., Kim, E.: Wildnet: learning domain generalized semantic segmentation from the wild. In: IEEE Conference on Computer Vision and Pattern Recognition (CVPR), pp. 9926–9936 (2022). https://doi.org/10.1109/CVPR52688.2022.00970
49. Zhao, Y., Zhong, Z., Zhao, N., Sebe, N., Lee, G.H.: Style-hallucinated dual consistency learning for domain generalized semantic segmentation. In: IEEE European Conference on Computer Vision (ECCV), pp. 535–552 (2022). https://doi.org/10.1007/978-3-031-19815-1_31
50. Zou, Y., Yu, Z., Kumar, B.V.K.V., Wang, J.: Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In: IEEE European Conference on Computer Vision (ECCV), pp. 289–305 (2018). https://doi.org/10.1007/978-3-030-01219-9_18
51. Guo, Y., Wang, X., Li, C., Ying, S.: Domain adaptive semantic segmentation by optimal transport. Fundam. Res. (2023). https://doi.org/10.1016/j.fmre.2023.06.006

Roberto Alcover-Couso was born in Madrid, Spain, in 1996. He received the B.S. and M.S. degrees in computer science from the Autonomous University of Madrid and currently pursuing a Ph.D. degree in domain adaptation for semantic segmentation. From 2019 to 2021, he worked as a data scientist in Magiquo. Since 2021, he is a research intern in the Video Processing and Understanding Lab (VPULab) in the Autonomous University of Madrid (UAM). His research focuses on domain adaptation, semantic segmentation and synthetic data.

Juan C. SanMiguel received the Ph.D. degree in computer science and telecommunication from the Autonomous University of Madrid, Madrid, Spain, in 2011. He was a Postdoctoral Researcher with the Queen Mary University of London, London, UK, from 2013 to 2014, under a Marie Curie IAPP Fellowship. He is currently an Associate Professor at the Autonomous University of Madrid and a Researcher with the Video Processing and Understanding Laboratory. He has authored over 55 journals and conference papers. His research interests include computer vision with a focus on reliability estimation and multicamera activity understanding for video segmentation and tracking.



Marcos Escudero-Viñolo is a researcher and university teacher co-author of more than 25 papers published in international high-quality peer-reviewed journals and international conferences. His research was funded by national grants and European and national competitive projects from the public and the private sectors, including four projects in which he has been the Principal Investigator. Regarding research topics, he has defined strategies for driving the analysis of vision signals based on regional and contextual constrains, especially for semantic segmentation, scene recognition and medical image analysis. His current research deals with the creation of strategies to provide interpretability, assessment and profiling gates to the knowledge encoded by deep-learning visual models. These are used to untap the reasons that preclude these models from being reliable, trustworthy and fair. He is a recurrent reviewer of top journals (e.g., TCSVT, TIP) and conferences (e.g., CVPR) and has recently accepted to be evaluator of the Spanish AEI.

Pablo Carballeira received the Telecommunication Engineering degree (five-year engineering program), Communications Technologies and Systems Master degree (two-year MS program) and the Ph.D. degree in Telecommunication from the Universidad Politécnica de Madrid (UPM) in 2007, 2010 and 2014, respectively. From 2008 to 2017, he has been a member of the Grupo de Tratamiento de Imágenes (Image Processing Group) at the UPM. Since 2017, he is a member of the Video Processing and Understanding Lab, at the Universidad Autónoma de Madrid (UAM), and Associate Professor at UAM since 2022. His research interests include computer vision, video coding and quality of experience evaluation for immersive visual media. He has been actively involved in European projects, national projects and standardization activities from ISO's Moving Picture Experts Group (MPEG) related to lightfield and free-navigation video, technologies.