DACS: Domain Adaptation via Cross-domain Mixed Sampling

Wilhelm Tranheden*  Viktor Olsson*  Juliano Pinto  Lennart Svensson

ABSTRACT

Semantic segmentation models based on convolutional neural networks have recently displayed remarkable performance for a multitude of applications. However, these models typically do not generalize well when applied on new domains, especially when going from synthetic to real data. Unsupervised domain adaptation (UDA) attempts to train on labelled data from one domain (source domain), and simultaneously learn from unlabelled data in the domain of interest (target domain). Existing methods have seen success by training on pseudo-labels for these unlabelled images. Multiple techniques have been proposed to mitigate low-quality pseudo-labels arising from the domain shift, with varying degrees of success. We propose DACS: Domain Adaptation via Cross-domain mixed Sampling, which mixes images from the two domains along with the corresponding labels. These mixed samples are then trained on, in addition to the labelled data itself. We demonstrate the effectiveness of our solution by achieving state-of-the-art results for two common synthetic-to-real semantic segmentation benchmarks for UDA. Code is available at: https://github.com/vikoiss/DACS.

1 Introduction

Deep neural networks have significantly advanced the state of the art for the task of semantic segmentation [1,2,3], displaying remarkable generalization abilities. Results have in large been presented for datasets where training and test domains are similar or identical in distribution. In real-world scenarios, however, a domain shift may occur, where the training data (source domain) is significantly different to the data encountered in inference (target domain) for the intended application. A common practice when dealing with domain shift is to annotate some data from the domain of interest and re-train (fine-tune) the network on this new data. Additionally, Semi-Supervised Learning (SSL) methods for semantic segmentation have been proposed [4,5,6,7,8,9], effectively training on a small amount of labelled data by relying on complementary learning from unlabelled data. These approaches are not always feasible as sometimes no annotations at all are accessible in the target domain.

Unsupervised Domain Adaptation (UDA) deals with the problem where labelled data is available for a source domain, but only unlabelled data is available for the target domain. The field has generated a lot of interest due to its potential for effectively utilizing synthetic data in training deep neural networks. Semantic segmentation in particular has been explored in recent research [10,11,12,13,14,15,16,17,18,19,20,21,22] due to its potential benefits in applications such as autonomous driving, the associated high annotation costs, and the cheap generation of synthetic data.

A technique originally proposed for SSL is pseudo-labelling [23] (or self-training), training on artificial targets based on the class predictions of the network. Pseudo-labelling was later adapted to UDA [17,18,22], where certain modifications were introduced to compensate for the domain shift. One of the earlier works on pseudo-labelling for UDA [17] pointed out that pseudo-labelling naively applied to UDA tends to bias the predictions of the network to easy-to-predict classes, causing difficult classes to stop being predicted during training. To combat this collapse, they propose class balanced sampling of the pseudo-labels. Additional difficulties of erroneous pseudo-labels owing to the domain shift have prompted later research to add modules for uncertainty estimation [18,22]. We note that in existing methods for correcting erroneous pseudo-labels, certain images in the target domain are over-sampled, and low confidence pixels within images filtered out. Many pixels of low confidence are aligned with predictions at

*Equal contribution
semantic boundaries \[24, 25, 9\], thus leading to a diminished training signal there. Circumventing these issues offers an opportunity to better leverage available data in the target domain.

In this paper we propose Domain Adaptation via Cross-domain mixed Sampling, or DACS for short, which adapts the segmentation-based augmentation technique ClassMix, originally proposed for SSL \[9\]. We show that applying the naive implementations of pseudo-labelling and ClassMix as it is used for SSL causes classes to become conflated, i.e., certain classes are confused with others, similar to the previously mentioned findings in \[17\] for naive pseudo-labelling. One potential solution in adapting said approach would be to incorporate existing techniques, used for pseudo-labelling in UDA. We, however, instead propose to solve this problem by mixing images across domains. In particular, classes in the labelled source domain images are pasted onto the unlabelled target domain images. These classes are selected from the ground-truth semantic maps of the source domain images, which are also mixed with the corresponding pseudo-labels of the target domain images. Mixing across domains this way leads to certain parts of the pseudo-labels always being injected with ground-truth semantic maps, ensuring that over the course of training all classes are present. In doing so, we solve the issues of pseudo-labelling and using ClassMix for UDA. In contrast to existing methods for correcting erroneous pseudo-labels, we do not get rid of training-data from over-sampling by class or confidence, and are able to efficiently learn from the entire unlabelled target domain dataset during the course of training. We demonstrate the effectiveness of our method by applying it on two synthetic-to-real unsupervised domain adaptation benchmarks, GTA5 \(\rightarrow\) Cityscapes and SYNTHIA \(\rightarrow\) Cityscapes.

In summary, our main contributions are: (1) We apply an SSL method based on the data augmentation ClassMix \[9\] to UDA, illustrating its flaws in this setting and providing an analysis of the possible causes. (2) We introduce DACS, a simple framework that adapts ClassMix based on cross-domain mixing of samples. (3) We present improvements over the state of the art in UDA for two prevalent benchmarks.

### 2 Related Work

For semantic segmentation, predominant methods for UDA include adversarial-based \[14\], self-training \[17, 18, 22\] and consistency-based approaches \[26, 27\]. This section reviews these related UDA methods in more detail, as well as some of their counterparts in SSL, as the two tasks are closely related. For a more extensive review of UDA in semantic segmentation, see \[28\].

**Adversarial Learning.** Existing research in adversarial learning for UDA has in various aspects tried to bridge the gap existing between source and target domains, minimizing differences between the distributions. This can be targeted at different levels such as at the pixel level \[29, 30, 31\], feature map level \[32, 11, 33\] or semantic level \[14, 34\]. In particular, alignment on the semantic level has been explored with similar methods for both UDA \[14\] and SSL \[4\]. The key idea is viewing the segmentation network as a generator in a generative adversarial network setup, to encourage realistic semantic maps to be predicted. This approach is viable for UDA because even though the source and target domain images are quite different, it is often reasonable to assume that the corresponding semantic maps are similar in terms of spatial layout and local context. Our proposed method should also benefit from this similarity in output space, since mixing images across domains will then lead to semantic classes being placed in contexts with more similar semantics.

**Pseudo-labelling.** In contrast, methods based on pseudo-labelling \[23\], or self-training, directly train on the target domain data by the use of pseudo-labels, which are artificial targets for training, created from class predictions of the network. However, UDA problems characteristically suffer from large domain gaps, i.e., considerable differences in data distributions, which give rise to faulty pseudo-labels. One of the problems is a bias in the target domain towards initially easy-to-transfer classes \[17, 18\], where some classes are merged, meaning certain classes are never predicted. A similar phenomenon has been observed in UDA for classification \[35\], where trivial predictions for the target domain were identified to occur when applying entropy regularization. For semantic segmentation, entropy regularization methods have observed a bias towards easy-to-transfer classes \[15\] as well. The close connection between entropy regularization and pseudo-labelling was pointed out in the original proposal of pseudo-labelling \[23\]. As pseudo-labelling and entropy regularization strive to minimize entropy in the predictions of unlabelled data, and since predictions with conflated, i.e., merged, classes have lower entropy, entropy minimization is a reasonable explanation for why conflation of classes can occur for large enough domain shifts. To combat the problem of faulty pseudo-labels for UDA in semantic segmentation, existing works have suggested careful selection and adjustment procedures, accounting for the domain gap. Variants include specialised sampling \[17\] and handling of uncertainty \[18, 22\].

Our proposed solution also makes use of pseudo-labelling, while cross-domain mixing of samples offers a simple solution to the class conflation problem by injecting reliable entropy from the source domain ground truth labels into the pseudo-labels of the target domain.

\[DACS: \text{DOMAIN ADAPTATION VIA CROSS-DOMAIN MIXED SAMPLING}\]


3 Method

This section details our proposed approach for unsupervised domain adaptation: Domain Adaptation via Cross-domain mixed Sampling, or DACS for short. We start with a short review of the ClassMix augmentation and the pitfalls of applying it directly to UDA, followed by an explanation of the adjustments necessary in order to achieve good performance, resulting in the DACS algorithm. We then conclude with a description of the loss function and the training procedure used.

3.1 ClassMix

ClassMix is a recently proposed data augmentation technique, used to achieve state-of-the-art results in challenging semi-supervised semantic segmentation benchmarks [9]. It works by “mixing” two images, $A$ and $B$, from the unlabelled part of the dataset into an augmented image, while also generating a pseudo-label for it. The mixing is done by first using a segmentation network to make predictions for images $A$ and $B$ and creating pseudo-labels from them, resulting in the semantic maps $Y_A$ and $Y_B$, respectively. Then, half of the classes present in $Y_A$ are selected, and a binary mask $M$ is generated. This binary mask contains 1’s in the same positions as the pixels of the selected classes of $A$, and 0’s elsewhere. Lastly, this mask is used to mix the images and their semantic maps, resulting in the augmented image $X_M$ and its corresponding pseudo-label $Y_M$:

$$X_M = M \odot A + (1 - M) \odot B, \quad Y_M = M \odot Y_A + (1 - M) \odot Y_B,$$

where $\odot$ denotes element-wise multiplication. An illustration of example input images, their semantic maps, and the output from ClassMix is shown in Figure 1.

3.2 Naive ClassMix Adaptation to UDA

As stated previously, the original formulation from [9] uses ClassMix with samples from the unlabelled dataset to generate augmented images. In UDA the unlabelled samples are the ones from the target dataset, so the natural adaptation of ClassMix to this context is mixing target-domain images. This approach, henceforth referred to as “Naive ClassMix”, mixes target-domain samples to generate augmented images and corresponding pseudo-labels, then trains the network using both the augmented images and the source-domain images, as illustrated in Figure 2. Similarly to [9], we use the Mean-Teacher framework [40], where instead of using the current parameters of the segmentation network for predicting the semantic maps of the input images, we use an exponential moving average of the previous weights during the optimization, resulting in more stable predictions.

However, this intuitive adaptation of ClassMix performs poorly in practice, as shown in the experiments of Section 4. The resulting segmentation network conflates some of the classes when predicting the semantics of target-domain images. For instance, classes with fewer occurrences like “sidewalk” are confused with more frequent and semantically similar classes, like “road”. Similarly, the “rider” class is misclassified as “person”, and “terrain” as “vegetation”, among others. This seems to be a consistent pattern across different seeds of training, and impacts performance considerably. The problem occurs exclusively for the target domain images, not for the images in the source domain.

This problem, which we refer to as class conflation, is similar to one identified in early works applying pseudo-labelling to UDA for semantic segmentation tasks [177], where they point out a bias towards easy-to-transfer classes when...
applying pseudo-labelling naively to UDA. As the described method for SSL [9] relies on generation of pseudo-labels (applied on images mixed with ClassMix), it can be expected to inherit the same underlying issues. While existing works have proposed other improvements for how to correct erroneous pseudo-label generation arising due to the domain shift [17, 18, 22], we instead propose a change in the augmentation procedure, detailed in the next subsection.

### 3.3 DACS

In order to improve the aforementioned poor performance of ClassMix in unsupervised domain adaptation, we make an important change to the framework presented in the previous subsection. Instead of performing the mixing using only images from the target domain, we instead mix images across domains, as illustrated in Figure 3. The mix is performed in the same way as described in subsection 3.1 but now the source domain image $X_S$ is the one used to
DACS: Domain Adaptation via Cross-domain Mixed Sampling

Figure 3: The images $X_S$ and $X_T$ are mixed together, using $Y_S$ for the labels of $X_S$, instead of a predicted semantic map. The segmentation network is then trained on both batches of augmented images and images from the source dataset.

Figure 4: Example augmentation used in DACS: an image from the source domain (in this case synthetic data from GTA5) is mixed with an image from the target domain (Cityscapes), resulting in an augmented image which contains parts from both domains.

compute the mask $M$, instead of an image from the target dataset. Likewise, parts of the target will be from the available ground-truth label, $Y_S$, instead of the target just coming from predictions from the segmentation network. An example of the input images and the resulting augmentation is illustrated in Figure 3. Note that the resulting mixed images are not necessarily realistic. However, this is not critical for the functioning of our method, as shown in section 4.

In order to explain why this change is beneficial, we first go back to Naive ClassMix. Before introducing cross-domain mixing, we have two types of data: 1) source domain data with ground-truth labels, and 2) target domain data with potentially conflated pseudo-labels, where the gap between the domains may be large. Due to this potentially large gap, a network may implicitly learn to discern between the domains in order to perform better in the task, and (incorrectly) learn that the class distributions are very different in the two domains. With cross-domain mixing (the change we propose in this section), we introduce new data. Considering that the labels for these new images partly come from the source domain, they will not be conflated for entire images. Furthermore, the pixels that are pseudo-labelled (target-domain labels) and the pixels that have ground-truth labels (source-domain labels) may now be neighbors in an image, making the implicit discerning between domains unlikely, since it would have to be done at a pixel level.

Both of these aspects help the network to better deal with the domain gap, and effectively solve the class conflation problem, as shown in section 4, resulting in considerably better performance. The overall UDA algorithm that trains on source-domain images and cross-domain augmentations is what we refer to as DACS, Domain Adaptation via Cross-domain mixed Sampling.

The implementation of DACS is presented as pseudocode in Algorithm 1, where the source-domain and target-domain datasets are referred to as $D_S$ and $D_T$, respectively. A batch of images and labels, $X_S$ and $Y_S$, is sampled from $D_S$, and a batch of images, $X_T$, from $D_T$. The images in $X_T$ are then fed to the network $f_B$, which outputs their predicted semantic maps $\hat{Y}_T$. Then, the augmented images $X_M$ are created by mixing $X_S$ and $X_T$, and the pseudo-labels $Y_M$ by mixing the corresponding maps in $Y_S$ and $\hat{Y}_T$. From this point forward, the algorithm resembles a supervised learning approach: compute predictions, compare them with the labels (in our case using the cross-entropy loss) as explained in

\[1\] In the pseudocode and in our implementation we use $L$, the monte-carlo approximation to $\mathcal{L}$ of Section 3.4, computed in batches.
DACS: DOMAIN ADAPTATION VIA CROSS-DOMAIN MIXED SAMPLING

Algorithm 1 DACS algorithm

Require: Source-domain and target-domain datasets \( D_S \) and \( D_T \), segmentation network \( f_\theta \).
1: Initialize network parameters \( \theta \) randomly.
2: for \( i = 1 \) to \( N \) do
3: \( X_S, Y_S \sim D_S \)
4: \( X_T \sim D_T \)
5: \( \hat{Y}_T \leftarrow f_\theta(X_T) \)
6: \( X_M, Y_M \leftarrow \text{Augmentation and pseudo-label from mixing } X_S, Y_S, X_T \text{ and } \hat{Y}_T. \)
7: \( \hat{Y}_S \leftarrow f_\theta(X_S), \hat{Y}_M \leftarrow f_\theta(X_M) \) \( \triangleright \) Compute predictions.
8: \( \ell \leftarrow \frac{1}{B} \sum_{i=1}^{B} \ell(h_S, y^i_S, \hat{Y}_M^i, Y_M^i) \) \( \triangleright \) Compute average loss in the batch.
9: Compute \( \nabla_\theta \ell \) by backpropagation (treating \( \hat{Y}_M \) as constant.)
10: Perform one step of stochastic gradient descent on \( \theta \).
11: end for
12: return \( f_\theta \)

Section[3,4], perform backprogragation, and perform a step of gradient descent. This process is then repeated for a predetermined amount of iterations \( N \).

3.4 Loss Function

In DACS the network parameters \( \theta \) are trained by minimizing the following loss:

\[
\mathcal{L}(\theta) = \mathbb{E} \left[ H(f_\theta(x_S), y_S) + \lambda H(f_\theta(x_M), y_M) \right],
\]

where the expectation is over the random variables \( x_S, y_S, x_M, \) and \( y_M \). Here, \( x_S \) is an image sampled uniformly at random from the source-domain distribution, and \( y_S \) is its corresponding label. The random variables \( x_M \) and \( y_M \) are the mixed image and its pseudo-label, created by performing cross-domain mixing from an image sampled uniformly at random from the source domain and one from the target domain, as explained previously. Finally, \( H \) is the cross-entropy between the predicted semantic map and the corresponding label (ground-truth or pseudo) averaged over all pixels, and \( \lambda \) is a hyper-parameter that decides how much the unsupervised part of the loss affects the overall training. In line with [6] and [9], we use an adaptive schedule for \( \lambda \), where it, for each image, is the proportion of pixels where the predictions of \( f_\theta \) on that image have a confidence above a certain threshold. Training is performed by stochastic gradient descent on this loss using batches with the same number of source-domain images and augmented images.

4 Experiments

In order to validate the proposed DACS algorithm, we evaluate it in two popular datasets for UDA and compare to the state of the art for such tasks. This section details the experimental setup and provides the qualitative and quantitative results found.

Implementation Details. For all the experiments in this paper, we adopt the widely used [10][11][12][13][14][15][16][17][18][19][20][21][22] DeepLab-v2 framework [41] with a ResNet101 backbone [42] as our model. The backbone is pretrained on ImageNet [43] and on MSCOCO [44]. Most hyperparameters are identical to those used in [10]. We use Stochastic Gradient Descent with Nesterov acceleration, and an initial learning rate of \( 2.5 \times 10^{-4} \), which is then decreased using polynomial decay with exponent 0.9 as in [41]. Weight decay is set to \( 5 \times 10^{-4} \) and momentum to 0.9. Source images are rescaled to \( 760 \times 1280 \) and target images to \( 512 \times 1024 \), after which random crops of size \( 512 \times 512 \) are extracted. Apart from ClassMix we also apply Color jittering and Gaussian blurring on the mixed images. We train using batches with 2 source images and 2 mixed images for 250k iterations. The code was implemented using the PyTorch framework, and is available at [https://github.com/vikolss/DACS](https://github.com/vikolss/DACS) Experiments were performed using a GTX 1080 Ti GPU with 12 GB memory.

Datasets. We present results for two synthetic-to-real benchmarks common for UDA for semantic segmentation. Namely GTA5 \( \rightarrow \) Cityscapes and SYNTHIA \( \rightarrow \) Cityscapes. The target dataset Cityscapes has 2,975 training images taken from a car in urban environments and is labelled with 19 classes [45]. The source datasets GTA5 [46] and SYNTHIA [47] contain 24,966 and 9,400 synthetic training images respectively. Example images of all three datasets are shown in Figure 5, together with ground truth semantic maps. The GTA5 images are labelled with the same 19
We present our results for GTA5 → Cityscapes, as shown in Table 1, alongside the results from several existing works on the same task. All of our comparisons have results for the same DeepLab-v2 network, but for completeness we choose to include their best presented performance, regardless of backbone. In the table, “Source” refers to models only trained on the source data and then evaluated on the target data, which serve as our baseline, and “Naive ClassMix” is the result from using the method directly adapted from [9], as described in Section 3.1. “DACS” refers to the results from our proposed method. DACS achieves the strongest results for eight of the individual classes (if ignoring Naive ClassMix), as well as an overall mIoU of 52.14%, higher than all previous methods, effectively pushing the state of the art for this task. As can also be seen in the table, the performance is significantly stronger for DACS than it is for Naive ClassMix. As stated in Section 3.1, a major reason for this is that Naive ClassMix conflates several of the classes, which impacts the overall performance considerably. This is clear from the per-class IoU in Table 1 where 7 of the classes have scores below 1% for Naive ClassMix.

Additionally, qualitative results are shown in Figure 5 illustrating the predictions for the supervised baseline (trained only on the source domain), naive ClassMix, and DACS on a few Cityscapes frames. The figure shows the same performance ordering as the table, where naive ClassMix outperforms the baseline, and DACS outperforms naive ClassMix. Specifically, we can clearly see an illustration of the class conflation problem mentioned in Section 3.1, where for example the “sidewalk” class is not predicted by naive ClassMix in any of the frames, instead being classified as road. In contrast, DACS is able to correctly discern between these classes in all frames.

Figure 5: Images from the Cityscapes, GTA5, and SYNTHIA datasets along with their corresponding semantic maps.

Table 1: GTA5 to Cityscapes, our results are averages from three runs, and shown as per-class IoU and mIoU. We also compare to several previous works. We present results for both Naive ClassMix and DACS.
Figure 6: Qualitative results of validation images from Cityscapes, when training models on the GTA5 dataset.

Table 2: SYNTHIA to Cityscapes, our results are averages from three runs, and shown as per-class IoU and mIoU for 13 and 16 classes. We also compare to several previous works.

| Method         | Road  | SW    | Build | Wall 7 | Fence | Pole | Ht   | Ts    | Veg   | Sky   | Person | Rider | Car   | Bus   | MC    | Bk    | mIoU% |
|----------------|-------|-------|-------|--------|-------|------|------|------|-------|-------|--------|-------|-------|-------|-------|-------|-------|
| Source         | 55.6  | 23.8  | 74.6  | -      | -     | 6.1  | 12.1 | 74.8  | 79.0  | 55.3  | 19.1   | 39.6  | 23.3  | 13.7  | 25.0  | -     | 38.6  |
| AdaptSegNet [10]| 84.3  | 42.7  | 77.5  | -      | -     | 4.7  | 7.0  | 77.9  | 82.5  | 54.3  | 21.0   | 72.3  | 32.2  | 18.9  | 32.3  | 46.7  |
| SIBAN [11]     | 82.5  | 24.0  | 79.4  | -      | -     | 16.5 | 12.7 | 79.2  | 82.8  | 58.3  | 18.0   | 79.3  | 25.3  | 17.6  | 25.9  | 46.3  |
| CLAN [12]      | 81.5  | 37.0  | 80.1  | -      | -     | 16.1 | 13.7 | 78.2  | 81.5  | 53.4  | 21.2   | 73.0  | 32.9  | 22.6  | 30.7  | 47.8  |
| APODA [13]     | 86.4  | 41.3  | 79.3  | -      | -     | 22.6 | 17.3 | 80.3  | 81.6  | 56.9  | 21.0   | 84.1  | 49.1  | 24.6  | 45.7  | 53.1  |
| PatchAlign [14]| 82.4  | 38.0  | 78.6  | 8.7    | 0.6   | 26.0 | 3.9  | 11.1  | 75.5  | 84.6  | 53.5   | 21.6  | 74.1  | 32.6  | 19.3  | 46.5  | 40.0  |
| AdvEnt [15]    | 85.6  | 42.2  | 79.7  | 8.7    | 0.4   | 25.9 | 5.4  | 8.1   | 80.4  | 84.1  | 57.9   | 23.8  | 73.3  | 36.4  | 14.2  | 33.0  | 48.0  | 41.2  |
| CBST [17]      | 68.0  | 29.9  | 76.3  | 10.8   | 1.4   | 33.9 | 22.8 | 29.5  | 77.6  | 78.3  | 60.6   | 28.3  | 81.6  | 23.5  | 18.8  | 39.8  | 48.9  | 42.6  |
| MRKLD [18]     | 67.7  | 32.2  | 73.9  | 10.7   | 1.6   | 37.4 | 22.2 | 31.2  | 80.8  | 80.5  | 60.8   | 29.1  | 82.8  | 25.0  | 19.4  | 45.3  | 50.1  | 43.8  |
| CADASS [20]    | 82.5  | 41.4  | 81.4  | 8.7    | -     | 18.5 | 15.8 | 90.6  | 84.5  | 55.5  | 61.4   | 55.5  | 45.9  | 39.5  | 26.6  | 43.9  | 32.4  |
| Source         | 44.0  | 19.3  | 70.9  | 8.7    | 0.8   | 28.2 | 16.1 | 16.7  | 79.8  | 81.4  | 57.8   | 19.2  | 46.9  | 17.2  | 12.0  | 43.8  | 40.4  | 35.2  |
| MRNet [21]     | 82.0  | 36.5  | 80.4  | 4.2    | 0.4   | 33.7 | 18.0 | 13.4  | 81.1  | 80.8  | 61.3   | 21.7  | 84.4  | 32.4  | 14.8  | 45.7  | 50.2  | 43.2  |
| R-MRNet [22]   | 87.6  | 41.9  | 83.1  | 14.2   | 1.7   | 36.2 | 31.3 | 19.9  | 81.6  | 80.6  | 63.0   | 21.8  | 86.2  | 40.7  | 23.6  | 53.1  | 54.9  | 47.9  |
| Source         | 36.30 | 14.64 | 68.78 | 9.17   | 0.20  | 24.39| 5.59 | 9.05  | 68.96 | 79.38 | 52.45  | 11.34 | 49.77 | 9.53  | 11.03 | 20.66 | 33.65 | 29.45 |
| DACS           | 80.56 | 25.12 | 81.90 | 21.46  | 2.85  | 37.20| 22.67| 23.99 | 83.69 | 90.77 | 67.61  | 38.33 | 82.92 | 38.90 | 28.49 | 47.58 | 54.81 | 48.34 |

4.2 SYNTHIA → Cityscapes Results

For the SYNTHIA dataset, in the same way as for GTA5 → Cityscapes, we compare with the best reported performance of existing methods, regardless of network. We do not present results for Naive ClassMix, as DACS was the most successful method between the two when training on GTA5. Additionally, the SYNTHIA dataset only contains 16 of the 19 classes of Cityscapes, and some authors present results for these 16 classes while other present for only 13 of them. Because of these differences in benchmarking, we present results for both 13 and 16 classes for the DACS method. The results for the SYNTHIA dataset, computed with the same metrics (per-class IoU and mIoU) as the former dataset, are shown in Table 2 alongside results from several other existing works.

When evaluating on all the 16 classes, DACS again improves the state of the art, obtaining an mIoU of 48.34%, while also achieving the strongest per-class results for 8 out of the 16 classes. It achieves competitive results with the state of the art of the 13-class formulation, obtaining an mIoU of 54.81%.
4.3 Early Stopping

As mentioned previously, DACS obtains the highest results for both GTA5 \(\rightarrow\) Cityscapes and SYNTHIA \(\rightarrow\) Cityscapes. The reported results are those obtained by models trained for 250k iterations. However, many existing methods in Tables 1 and 2 report their results from when using early stopping based on the same validation set that is used for the final evaluation (there is no publicly available test set for Cityscapes). We believe that this is not a fair evaluation, as a high performance on the validation set does not necessarily mean high performance on all data, but could just mean that the model is performing well on those exact images. Using early stopping in our case would increase the results substantially: for GTA5 it would increase to 35.68% for the baseline just trained on the source data and to 53.84% for the DACS results. This means that our results for GTA5 \(\rightarrow\) Cityscapes, had we used early stopping, would be more than 3.5% above the previous state of the art [22], which does use early stopping.

For SYNTHIA, the source baseline would increase to 32.85%, and the results for 13 classes would increase to 55.98% and for 16 classes to 49.10%. Hence we would achieve better results than the previous state of the art for evaluation on both 13 and 16 classes. The reason for the considerable performance increase from early stopping is that the network’s performance on the test set fluctuates a lot over the course of training, rather than that the model is overfitting the training data.

4.4 Similarity Between Source and Target

We note that our results increase more in relation to previous works for GTA5 \(\rightarrow\) Cityscapes than it does for SYNTHIA \(\rightarrow\) Cityscapes. We hypothesize that this depends on the similarity between the source and target data, namely that GTA5 images are more similar to Cityscapes than SYNTHIA images are. This is clearly helpful in UDA and particularly so when mixing images between domains, since the mixed images will be more sensible if objects end up in locations that are reasonable. This can be quantified by the spatial distribution of classes: where in the images certain classes appear. Cityscapes and GTA5 are very similar in that road is always down, sky is always up and cars are always vertically near the center of the images. This is not the case for SYNTHIA, however, as the images are taken from different perspectives, including from the ground and from above. Hence, when pasting objects from SYNTHIA to Cityscapes, it is likely that the resulting images will be nonsensical, which we believe is detrimental for training.

4.5 Additional Experiments

Besides the aforementioned evaluations of DACS, we also performed smaller experiments for better understanding the source for the class conflation problem, while also investigating an alternative solution to it. Table 3 presents the results for these additional experiments, which are explained in detail further below.

No Mixing. Since Naive ClassMix is mixing images based on predictions, we investigated if the problem of class conflation could be related to, or made worse by, the mixing component. We observe that when just using pseudo-labelling (that is removing the mixing component), even more classes stop being predicted by the network, with overall performance becoming worse than the source baseline. Therefore, it is reasonable to assume that it is the pseudo-labelling component, and not the mixing, that cause class conflation, similar to the conclusion of existing work [17]. It is possible that incorporating existing techniques for pseudo-labelling would solve this issue. Though as previously stated, our proposed method of mixing images across domains offers a simple correction purely by changing the nature of the augmentation technique.

Distribution Alignment. In DACS, the issue of classes being conflated is handled by pasting classes from the source domain images onto the target images, which largely solves the problem. Another way to solve the same problem would be to impose a prior class distribution on the pseudo-labels. This has been done previously for the same task in the context of entropy regularization [15], a related technique also based on entropy minimization. It is therefore insightful to see if a similar solution works for correcting pseudo-labels in Naive ClassMix. To this end we use Distribution Alignment, as used in [48], meaning that a distribution of classes is forced upon the predictions. This is a different way of injecting entropy into the pseudo-labels, meaning it also makes it more likely that the network learns to correctly segment the target domain and avoid class conflation. We perform experiments where the ground-truth distribution \(\bar{p}\) of the target dataset is used to guide the training. This is done for each sample by transforming the output prediction \(\tilde{q}\) for each pixel is into \(\tilde{q} = \text{Normalize}(q \times p / \bar{p})\), where \(\bar{p}\) is a running average of all predictions made by the network on the target data. The setup is otherwise identical to when using Naive ClassMix. The results from this are shown in Table 5 in our case, this is clearly not a legitimate way of doing this, as the ground-truth class distribution would not be known for an unlabelled dataset in a realistic setting. However, it is interesting to see that this approach also solves the issue of conflating classes, as all classes are represented in the results. This further strengthens our hypothesis that artificial injection of entropy in training can help the network avoid class conflation. An interesting direction of future research would be to use an estimated class distribution in a similar way.
Table 3: Results for GTA5 to Cityscapes, from experiments when using only pseudo-labelling and when using Distribution alignment on top of Naive ClassMix. For comparison the results presented in Table 1 are also included.

| Method             | Road | SW  | Build | Wall  | Fence | Pole | TL  | TS  | Veg  | Terrain | Sky | Person | Rider | Car  | Truck | Bus  | Train | MC   | Bike | Label |
|--------------------|------|-----|-------|-------|-------|------|-----|-----|------|---------|-----|--------|-------|------|-------|------|-------|------|------|-------|
| Source             | 63.31| 15.65| 59.39 | 8.56  | 15.17 | 18.31| 26.94 | 15.00 | 80.46 | 15.25   | 72.97| 51.04  | 17.67 | 59.68 | 28.19 | 33.07 | 3.53  | 23.21| 16.73 |
| Pseudo-labelling   | 85.14| 0.03 | 75.84 | 0.35  | 0.03  | 0.23 | 3.19 | 1.30 | 78.05 | 36.53   | 95.78| 4.89   | 1.01  | 79.17 | 4.24  | 1.30  | 0.00  | 0.26 | 0.02  |
| Naive ClassMix     | 84.78| 0.00 | 82.81 | 0.34  | 0.05  | 0.23 | 3.19 | 1.30 | 78.05 | 36.53   | 95.78| 4.89   | 1.01  | 79.17 | 4.24  | 1.30  | 0.00  | 0.26 | 0.02  |
| Distribution alignment | 85.05| 28.88| 86.79 | 16.92 | 36.89 | 30.38| 49.73 | 53.91 | 85.61 | 32.20   | 92.78| 66.61  | 23.53 | 84.00 | 34.81 | 27.70 | 0.20  | 16.65| 60.08 |
| DACS               | 89.90| 39.66| 87.87 | 30.71 | 39.52 | 38.52| 46.43 | 52.79 | 87.98 | 45.96   | 88.76| 67.20  | 35.78 | 84.45 | 45.73 | 50.19 | 0.00  | 27.25| 33.96 |

5 Conclusion

We proposed DACS, Domain Adaptation via Cross-domain mixed Sampling, a novel algorithm for unsupervised domain adaptation in semantic segmentation, based on an adaptation of the ClassMix [9] data augmentation method. We show how the naive application of ClassMix to UDA results in systematic problems in the predictions, and detail the changes performed in order to correct these issues. Furthermore, we perform an evaluation of DACS in two popular domain adaptation benchmarks, GTA→Cityscapes and SYNTHIA→Cityscapes, and show that it outperforms other existing methods and pushes the state of the art for both tasks.

Acknowledgements

Part of this work was carried out during a master thesis at Volvo Cars, who also provided the necessary hardware to facilitate our experiments. We would like to thank Oskar Karnblad, Saudin Botonjic, Srikar Muppirisetty, Adam Fejne, together with the remaining members of the machine learning team for their support.

References

[1] Jonathan Long, Evan Shelhamer, and Trevor Darrell. Fully convolutional networks for semantic segmentation. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 3431–3440, 2015.

[2] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Semantic image segmentation with deep convolutional nets and fully connected crfs. In Yoshua Bengio and Yann LeCun, editors, *3rd International Conference on Learning Representations, ICLR 2015, San Diego, CA, USA, May 7-9, 2015, Conference Track Proceedings*, 2015.

[3] Hengshuang Zhao, Jianping Shi, Xiaojian Qi, Xiaogang Wang, and Jiaya Jia. Pyramid scene parsing network. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2881–2890, 2017.

[4] Wei-Chih Hung, Yi-Hsuan Tsai, Yan-Ting Liou, Yen-Yu Lin, and Ming-Hsuan Yang. Adversarial learning for semi-supervised semantic segmentation. In *British Machine Vision Conference 2018, BMVC 2018, Newcastle, UK, September 3-6, 2018*, page 65. BMVA Press, 2018.

[5] Sudhanshu Mittal, Maxim Tatarchenko, and Thomas Brox. Semi-supervised semantic segmentation with high-and low-level consistency. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 2019.

[6] Geoffrey French, Timo Aila, Samuli Laine, Michal Mackiewicz, and Graham D. Finlayson. Semi-supervised semantic segmentation needs strong, high-dimensional perturbations. *CoRR*, abs/1906.01916, 2019.

[7] Zhengyang Feng, Qianyu Zhou, Guangliang Cheng, Xin Tan, Jianping Shi, and Lizhuang Ma. Semi-supervised semantic segmentation via dynamic self-training and class-balanced curriculum, 2020.

[8] Liang-Chieh Chen, Raphael Gontijo Lopes, Bowen Cheng, Maxwell D. Collins, Ekin D. Cubuk, Barret Zoph, Hartwig Adam, and Jonathon Shlens. Leveraging semi-supervised learning in video sequences for urban scene segmentation, 2020.

[9] Viktor Olsson, Wilhelm Tranheden, Juliano Pinto, and Lennart Svensson. Classmix: Segmentation-based data augmentation for semi-supervised learning, 2020.

[10] Yi-Hsuan Tsai, Wei-Chih Hung, Samuel Schulter, Kihyuk Sohn, Ming-Hsuan Yang, and Mannmohan Chandraker. Learning to adapt structured output space for semantic segmentation. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 7472–7481. IEEE Computer Society, 2018.

[11] Yawei Luo, Ping Liu, Tao Guan, Junqing Yu, and Yi Yang. Significance-aware information bottleneck for domain adaptive semantic segmentation. In *2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019*, pages 6777–6786. IEEE, 2019.
[12] Yawei Luo, Liang Zheng, Tao Guan, Junqing Yu, and Yi Yang. Taking a closer look at domain shift: Category-level adversaries for semantics consistent domain adaptation. In IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019, pages 2507–2516. Computer Vision Foundation / IEEE, 2019.

[13] Jihan Yang, Ruijia Xu, Ruiyu Li, Xiaojuan Qi, Xiaoyong Shen, Guanbin Li, and Liang Lin. An adversarial perturbation oriented domain adaptation approach for semantic segmentation. In The Thirty-Fourth AAAI Conference on Artificial Intelligence, AAAI 2020, The Thirty-Second Innovative Applications of Artificial Intelligence Conference, IAAI 2020, The Tenth AAAI Symposium on Educational Advances in Artificial Intelligence, EAAI 2020, New York, NY, USA, February 7-12, 2020, pages 12613–12620. AAAI Press, 2020.

[14] Yi-Hsuan Tsai, Kihyuk Sohn, Samuel Schulter, and Manmohan Chandraker. Domain adaptation for structured output via discriminative patch representations. In 2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019, pages 1456–1465. IEEE, 2019.

[15] Tuan-Hung Vu, Himalaya Jain, Maxime Bucher, Matthieu Cord, and Patrick Pérez. ADVENT: adversarial entropy minimization for domain adaptation in semantic segmentation. In IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019, pages 2517–2526. Computer Vision Foundation / IEEE, 2019.

[16] Yiheng Zhang, Zhaofan Qiu, Ting Yao, Dong Liu, and Tao Mei. Fully convolutional adaptation networks for semantic segmentation. In 2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018, pages 6810–6818. IEEE Computer Society, 2018.

[17] Yang Zou, Zhiding Yu, B. V. K. Vijaya Kumar, and Jinsong Wang. Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In Vittorio Ferrari, Martial Hebert, Cristian Sminchisescu, and Yair Weiss, editors, Computer Vision - ECCV 2018 - 15th European Conference, Munich, Germany, September 8-14, 2018, Proceedings, Part III, volume 11207 of Lecture Notes in Computer Science, pages 297–313. Springer, 2018.

[18] Yang Zou, Zhiding Yu, Xiaofeng Liu, B. V. K. Vijaya Kumar, and Jinsong Wang. Confidence regularized self-training. In 2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019, pages 5981–5990. IEEE, 2019.

[19] Jinyu Yang, Weizhi An, Chaochao Yan, Peilin Zhao, and Junzhou Huang. Context-aware domain adaptation in semantic segmentation, 2020.

[20] Zhedong Zheng and Yi Yang. Unsupervised scene adaptation with memory regularization in vivo. CoRR, abs/1912.11164, 2019.

[21] Zhezhong Zheng and Yi Yang. Rectifying pseudo label learning via uncertainty estimation for domain adaptive semantic segmentation, 2020.

[22] Dong-Hyun Lee. Pseudo-label : The simple and efficient semi-supervised learning method for deep neural networks. ICML 2013 Workshop : Challenges in Representation Learning (WREPL), 07 2013.

[23] Xiaoxiao Li, Ziwei Liu, Ping Luo, Chen Change Loy, and Xiaoou Tang. Not all pixels are equal: Difficulty-aware semantic segmentation via deep layer cascade. In 2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017, pages 6459–6468. IEEE Computer Society, 2017.

[24] Yi Zhu, Karan Sapra, Fitsum A. Reda, Kevin J. Shih, Shawn D. Newsam, Andrew Tao, and Bryan Catanzaro. Improving semantic segmentation via video propagation and label relaxation. In IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019, pages 8856–8865. Computer Vision Foundation / IEEE, 2019.

[25] Yun-Chun Chen, Yen-Yu Lin, Ming-Hsuan Yang, and Jia-Bin Huang. Crdoco: Pixel-level domain transfer with cross-domain consistency. In IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2019, Long Beach, CA, USA, June 16-20, 2019, pages 1791–1800. Computer Vision Foundation / IEEE, 2019.

[26] Qianyu Zhou, Zhengyang Feng, Guangliang Cheng, Xin Tan, Jianping Shi, and Lizhuang Ma. Uncertainty-aware consistency regularization for cross-domain semantic segmentation, 2020.

[27] Marco Toldo, Andrea Maracani, Umberto Michieli, and Pietro Zanuttigh. Unsupervised domain adaptation in semantic segmentation: a review, 2020.

[28] Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. In 2017 IEEE Conference on Computer Vision
DACS: Domain Adaptation via Cross-Domain Mixed Sampling

[30] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A. Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In Jennifer G. Dy and Andreas Krause, editors, Proceedings of the 35th International Conference on Machine Learning, ICML 2018, Stockholmsmässan, Stockholm, Sweden, July 10-15, 2018, volume 80 of Proceedings of Machine Learning Research, pages 1994–2003. PMLR, 2018.

[31] Zuxuan Wu, Xin Wang, Joseph Gonzalez, Tom Goldstein, and Larry Davis. ACE: adapting to changing environments for semantic segmentation. In 2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019, pages 2121–2130. IEEE, 2019.

[32] Xiangyu Yue, Yang Zhang, Sicheng Zhao, Alberto L. Sangiovanni-Vincentelli, Kurt Keutzer, and Boqing Gong. Domain randomization and pyramid consistency: Simulation-to-real generalization without accessing target domain data. In 2019 IEEE/CVF International Conference on Computer Vision, ICCV 2019, Seoul, Korea (South), October 27 - November 2, 2019, pages 2100–2110. IEEE, 2019.

[33] Shanshan Wang, Lei Zhang, Wangmeng Zuo, and Bob Zhang. Class-specific reconstruction transfer learning for visual recognition across domains. IEEE Transactions on Image Processing, PP:1–1, 11 2019.

[34] Shanshan Wang, Lei Zhang, Wangmeng Zuo, and Bob Zhang. Class-specific reconstruction transfer learning for visual recognition across domains. IEEE Transactions on Image Processing, PP:1–1, 11 2019.

[35] Xiaofu Wu, Suofei hang, Quan Zhou, Zhen Yang, Chunming Zhao, and Longin Jan Latecki. Entropy minimization vs. diversity maximization for domain adaptation, 2020.

[36] David Berthelot, Nicholas Carlini, Ian J. Goodfellow, Nicolas Papernot, Avital Oliver, and Colin Raffel. Mixmatch: A holistic approach to semi-supervised learning. In H Anna M. Wallach, Hugo Larochelle, Alina Beygelzimer, Florence d’Alché-Buc, Emily B. Fox, and Roman Garnett, editors, Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, 8-14 December 2019, Vancouver, BC, Canada, pages 5050–5060, 2019.

[37] Qizhe Xie, Zihang Dai, Eduard Hovy, Minh-Thang Luong, and Quoc V. Le. Unsupervised data augmentation for consistency training, 2019.

[38] Kihyuk Sohn, David Berthelot, Chun-Liang Li, Zizhao Zhang, Nicholas Carlini, Ekin D. Cubuk, Alex Kurakin, Han Zhang, and Colin Raffel. Fixmatch: Simplifying semi-supervised learning with consistency and confidence. ArXiv, 2020.

[39] Jongmok Kim, Jooyoung Jang, and Hyunwoo Park. Structured consistency loss for semi-supervised semantic segmentation. ArXiv, abs/2001.04647, 2020.

[40] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In Isabelle Guyon, Ulrike von Luxburg, Samy Bengio, Hanna M. Wallach, Rob Fergus, S. V. N. Vishwanathan, and Roman Garnett, editors, Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, NeurIPS 2017, 4-9 December 2017, Long Beach, CA, USA, pages 1195–1204, 2017.

[41] Liang-Chieh Chen, George Papandreou, Iasonas Kokkinos, Kevin Murphy, and Alan L. Yuille. Deeplab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected crfs. IEEE Trans. Pattern Anal. Mach. Intell., 40(4):834–848, 2018.

[42] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In 2016 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2016, Las Vegas, NV, USA, June 27-30, 2016, pages 770–778. IEEE Computer Society, 2016.

[43] J. Deng, W. Dong, R. Socher, L.-J. Li, K. Li, and L. Fei-Fei. ImageNet: A Large-Scale Hierarchical Image Database. In CVPR09, 2009.

[44] Tsung-Yi Lin, Michael Maire, Serge J. Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C. Lawrence Zitnick. Microsoft COCO: common objects in context. In David J. Fleet, Tomás Pajdla, Bernt Schiele, and Tinne Tuytelaars, editors, Computer Vision - ECCV 2014 - 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V, volume 8693 of Lecture Notes in Computer Science, pages 740–755. Springer, 2014.

[45] Marius Cordts, Mohamed Omran, Sebastian Ramos, Timo Rehfeld, Markus Enzweiler, Rodrigo Benenson, Uwe Franke, Stefan Roth, and Bernt Schiele. The cityscapes dataset for semantic urban scene understanding. In 2016
[46] Stephan R. Richter, Vibhav Vineet, Stefan Roth, and Vladlen Koltun. Playing for data: Ground truth from computer games. In Bastian Leibe, Jiri Matas, Nicu Sebe, and Max Welling, editors, European Conference on Computer Vision (ECCV), volume 9906 of LNCS, pages 102–118. Springer International Publishing, 2016.

[47] German Ros, Laura Sellart, Joanna Materzynska, David Vazquez, and Antonio Lopez. The SYNTHIA Dataset: A large collection of synthetic images for semantic segmentation of urban scenes. 2016.

[48] David Berthelot, Nicholas Carlini, Ekin D. Cubuk, Alex Kurakin, Kihyuk Sohn, Han Zhang, and Colin Raffel. Remixmatch: Semi-supervised learning with distribution alignment and augmentation anchoring. ArXiv, abs/1911.09785, 2019.