Exploiting image translations via ensemble self-supervised learning for Unsupervised Domain Adaptation

Citation for published version (APA):
Piva, F., & Dubbelman, G. (2023). Exploiting image translations via ensemble self-supervised learning for Unsupervised Domain Adaptation. *Computer Vision and Image Understanding*, *234*, 1-10. Article 103745. https://doi.org/10.1016/j.cviu.2023.103745

Document license:
CC BY

DOI:
10.1016/j.cviu.2023.103745

Document status and date:
Published: 01/09/2023

Document Version:
Publisher’s PDF, also known as Version of Record (includes final page, issue and volume numbers)

Computer Vision and Image Understanding 234 (2023) 103745

Contents lists available at ScienceDirect

Computer Vision and Image Understanding

journal homepage: www.elsevier.com/locate/cviu

Exploiting image translations via ensemble self-supervised learning for Unsupervised Domain Adaptation

Fabrizio J. Piva *, Gijs Dubbelman

Eindhoven University of Technology, Department of Electrical Engineering, Groene Loper 12, 5612AZ Eindhoven, The Netherlands

ARTICLE INFO

Communicated by Nikos Paragios

Keywords:
Ensemble learning
Self-supervised learning
Unsupervised domain adaptation
Image translations

ABSTRACT

Unsupervised Domain Adaptation (UDA) aims to improve the generalization capacity of models when they are tested on a real-world target domain by learning a model on a source labeled domain. Recently, a UDA method was proposed that addresses the adaptation problem by combining ensemble learning with self-supervised learning. However, this method uses only the source domain to pretrain the model and employs a limited amount of classifiers to create target pseudo labels. To mitigate these deficiencies, in this work, we explore the usage of image translations in combination with ensemble learning and self-supervised learning. To increase the model's exposure to more variable pretraining data, our method creates multiple diverse image translations, which encourages the learning of domain-invariant features, desired to increase generalization. With these image translations, we are able to learn translation-specific classifiers, which also allows to maximize the amount of ensemble's classifiers resulting in more robust target pseudo labels. In addition, we propose to use the target domain in pretraining stage to mitigate source domain bias in the network. We evaluate our method on the standard UDA benchmarks, i.e., adapting GTA V and Synthia to Cityscapes, and achieve state-of-the-art results on the mIoU metric. Extensive ablation experiments are reported to highlight the advantageous properties of our UDA strategy.

1. Introduction

Despite the impressive breakthroughs achieved in many computer vision tasks, deep learning models still often require training with abundant annotated data. Especially in dense prediction tasks like semantic segmentation, annotating data can be costly and labor-intensive, as it requires providing per-pixel labels for entire images. This makes labeled datasets particularly scarce. One of the consequences of training with limited labeled data is a lack of *generalization*, because models are unable to learn all the possible scenarios that they are likely to encounter in the real-world. This poor generalization prevents models from being deployed on real-world applications. Therefore, to mitigate this issue, many different training strategies has been proposed (Wang et al., 2022; Neven et al., 2021; Bucher et al., 2021), one of which is Unsupervised Domain Adaptation (UDA) (Hoffman et al., 2016). In this work, we propose a novel UDA method that is able to generalize well to unseen data.

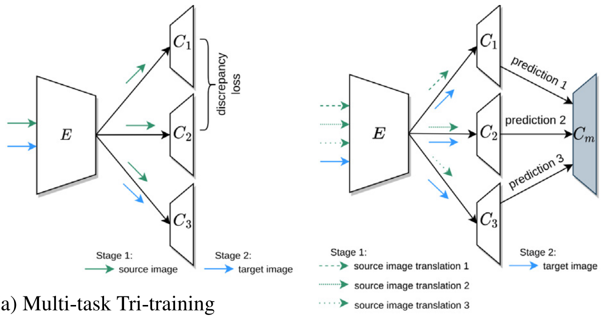
One of the strengths of UDA is that it avoids the burden of manually annotating data by employing synthetic data (Ros et al., 2016; Richter et al., 2016), which represents the so-called *source* domain. In addition, the *target* domain, i.e. the domain on which the model is likely to be tested, is also accessible, but in the form of unlabeled data. However, training a model with data from different domains requires addressing the underlying domain gap. Therefore, the goal in UDA is to train a model with data from a source labeled domain and a target unlabeled domain, so that the resulting model can perform well on the target domain.

Recently, a UDA strategy was proposed (Ruder and Plank, 2018; Zhang et al., 2018), which combines ensemble learning (Dietterich, 2000) with self-supervised learning (SSL) (Hendrycks et al., 2019). This training strategy is performed in two stages using a shared encoder with three classifiers (see Fig. 1(a)). In the first stage, the annotated source set is used to train the encoder and classifiers in a supervised manner, while enforcing disagreement between the first two classifiers with a discrepancy loss (Bousmalis et al., 2016). In the second training stage, the predictions of the first two classifiers are employed to create pseudo labels on the target set. Since these two classifiers were trained to disagree, majority voting is used to determine the winner classes on the target predictions (Hernández-González et al., 2019). Finally, the strategy proposes to train the third classifier using these pseudo labels, to obtain a target-specific classifier. This process can be repeated a certain number of times, known as rounds, until convergence.

Although this training strategy is suitable for UDA, it has the following limitations we are looking to address:

* Corresponding author.
E-mail address: f.j.piva@tue.nl (F.J. Piva).

https://doi.org/10.1016/j.cviu.2023.103745
Received 1 July 2021; Received in revised form 30 May 2023; Accepted 31 May 2023
Available online 8 June 2023



(a) Multi-task Tri-training (Ruder and Plank, 2018; Zhang et al., 2018)

(b) Our approach

Fig. 1. To adapt a source annotated set to an unlabeled target set, our proposed method teaches each classifier to learn the features of an independent source image translation creating discrepancy across all classifiers. This differs from Multi-task Tri-training, as it requires the use of a discrepancy loss between C_1 and C_2 because they are learning from the same features. By learning translation-specific features on each classifier, we remove the need of using discrepancy losses and allow the model to use all classifiers to label the target domain. Unlike Zhang et al. (2018), which uses the predictions of C_1 and C_2 with a rule-based mechanism, we propose a meta-learner C_m that is able to learn, for each specific class, the optimal combination of the predictions of all classifiers to create robust pseudo labels for the target domain.

1. The first training stage relies only on the source domain without adding any variability on these inputs, which introduces domain bias (Tommasi et al., 2017), harming the generalization capacity of the model.
2. The second training stage employs only two classifiers to create pseudo labels, but the performance of ensemble learning increases with the amount of classifiers (Wolpert, 1992).

To overcome these deficiencies, we research an alternative approach for both training stages (see Fig. 1(b)). In the first training stage, our method avoids introducing bias and encourages the model to learn generalizable features. We do this by creating diverse image translations, that each classifier has to learn independently; and by using the target domain information already from the first training stage. This is desired because when a model is deployed in the real-world, the images that the model encounters are likely to contain high variability. At the same time, this removes the need of using a discrepancy loss, because each classifier learns from a specific translation. In addition, during the second training stage, we propose a meta-learning layer that is able to exploit the translation-specific knowledge of each classifier to create pseudo labels. Unlike Ruder and Plank (2018), Zhang et al. (2018) that uses a rule-based mechanism, our meta-learner learns to combine, for each specific class, the expertise of each classifier to label the target domain. By doing so, we are able to maximize the amount of classifiers for the ensemble, which increases the quality of the pseudo labels, and allows all classifiers to learn target specific features.

In summary, the main contributions of our work are:

- Usage of multiple different image translations within an ensemble learning, which benefits the generalization capacity of the model on learning domain invariant representations (Lu et al., 2022).
- Usage of a meta-learner, which exploits the translation-specific knowledge of each classifier to create robust pseudo-labels for the target domain.
- Mitigation of domain bias by using the target domain from the first training stage.

2. Related work

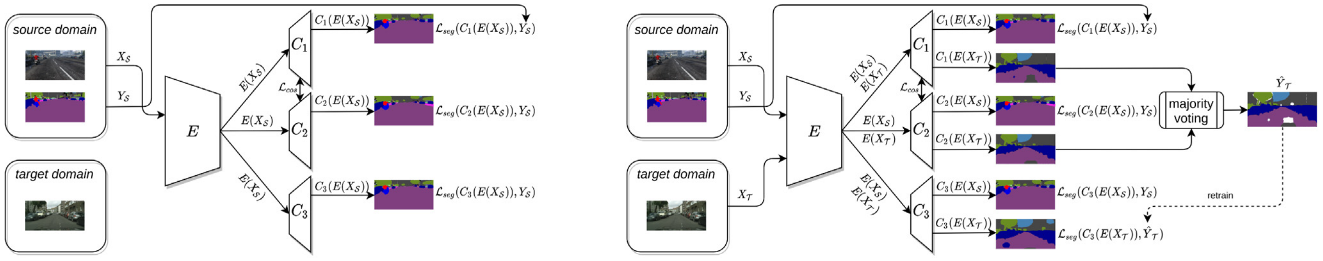
Most of the state-of-the-art methods combine different strategies to achieve competitive results. In this section, we focus on related work that, similar to our approach, use strategies such as image translation, self-supervised learning, and/or ensemble methods.

Image translation methods for UDA have recently been widely used to improve the performance of UDA methods. Since these networks can be trained without the need of labels, the underlying idea of using image translations in UDA is to transform the source domain images to a new set of images with a visual appearance similar to the images of the target domain. This is the case for several UDA methods, that choose to transform the source to the target set, either by using a deep neural network (Hoffman et al., 2018; Li et al., 2019; Wu et al., 2018a) or image processing techniques such as the Fourier transform (Yang and Soatto, 2020). Other works have considered mapping the source and target images to an intermediate space (Murez et al., 2018), where the features are domain agnostic. In addition, Yang et al. (2020a) has shown that mapping the target domain images to the source is also effective, leading to state-of-the-art results. Alternatively, Gong et al. (2019) explores the possibility of generating multiple intermediate representations between the source and the target domain, where each arbitrary representation belongs to a point in a manifold of domains. Regardless of the chosen space to which the source domain is mapped, to the best of our knowledge, no works have considered using multiple representations in parallel to improve UDA models, as explored in this work.

Self-supervised Learning in UDA. Many recent UDA methods leverage self-supervised learning as a way of using the model's predictions to learn from the unlabeled target domain. When using the model's outputs, it is needed to establish criteria to filter out spurious predictions and select reliable label candidates. Many methods that use a single encoder-single decoder architecture propose to use the most confident predictions as selection criteria (Li et al., 2019; Yang and Soatto, 2020; Zou et al., 2018), but they still suffer from the propagation of errors as the model can still consider highly confident but mistaken predictions as correct pseudo labels. Instead, we mitigate this issue by using multiple classifiers that were trained on different inputs, and unlike traditional majority voting (Hernández-González et al., 2019), our meta-learning layer combines the knowledge of each classifier by learning to weight each classifier's decision, for every semantic class.

Ensemble methods for UDA propose to increase the number of predictions for a single input by changing the network architecture. For instance, Co-Training (CT) utilizes two classifiers to create different points of view from the same sample to produce pseudo-labels (Blum and Mitchell, 1998; Hady and Schwenker, 2008), used later for the unlabeled training data. Recent applications of CT in UDA have demonstrated promising results (Luo et al., 2018). Tri-Training (TT) (Zhou and Li, 2005) can be conceived as an extension of CT, where three members participate in the ensemble, each of these consisting of a feature extractor and a classifier. Since TT is computationally expensive, Multitask tri-training (MTri) (Ruder and Plank, 2018) was proposed, where a common feature extractor is shared among three classifiers, computing different outputs from the same features. The idea behind MTri is to make the feature extractor learn those features that are invariant across the source and target domain, whilst forcing a discrepancy between the classifiers through a discrepancy operator.

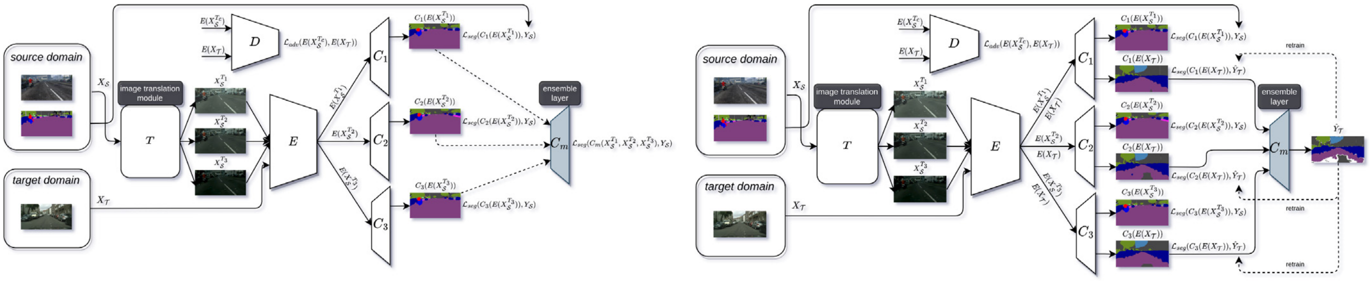
When sharing a feature extractor in MTri, the discrepancy across classifiers becomes a key factor to generate pseudo-labels during SSL (Zhang et al., 2018). While a cosine distance might help to enforce a certain diversity, we hypothesize that feeding constantly the same features to all the classifiers does not optimally allow the encoder to learn domain invariant representations. If we obtain these alternative representations from an image translation model, we can encourage discrepancy by feeding the features of a specific representation to a different classifier, improving simultaneously the generalization capacity of the encoder. These are the principles on which our method is based, combining the benefits of image transformations, ensemble learning, and self-supervised learning.



(a) Stage 1

(b) Stage 2

Fig. 2. *Training strategy of Multi-task Tri-training (MTri)*. To reach a certain level of knowledge during the pretraining phase, MTri uses only the source labeled domain, without introducing any type of variability on these inputs. To learn from the target domain during self-supervised learning, MTri utilizes only the first two classifiers C_1 and C_2 to create pseudo labels, using the third one C_3 to learn a target-specific classifier.


source domain
target domain
X_S
X_T
image translation module
T
$X_S^{T_1}$
$X_S^{T_2}$
$X_S^{T_3}$
$E(X_S^{T_c})$
$E(X_T)$
D
$\mathcal{L}_{adv}(E(X_S^{T_c}), E(X_T))$
E
$E(X_S^{T_1})$
$E(X_S^{T_2})$
$E(X_S^{T_3})$
C_1
C_2
C_3
$C_1(E(X_S^{T_1}))$
$C_2(E(X_S^{T_2}))$
$C_3(E(X_S^{T_3}))$
$C_1(E(X_T))$
$C_2(E(X_T))$
$C_3(E(X_T))$
$\mathcal{L}_{seg}(C_1(E(X_S^{T_1})), Y_S)$
$\mathcal{L}_{seg}(C_2(E(X_S^{T_2})), Y_S)$
$\mathcal{L}_{seg}(C_3(E(X_S^{T_3})), Y_S)$
$\mathcal{L}_{seg}(C_1(E(X_T)), \hat{Y}_T)$
$\mathcal{L}_{seg}(C_2(E(X_T)), \hat{Y}_T)$
$\mathcal{L}_{seg}(C_3(E(X_T)), \hat{Y}_T)$
ensemble layer
C_m
$\mathcal{L}_{seg}(C_m(X_S^{T_1}, X_S^{T_2}, X_S^{T_3}), Y_S)$
$\hat{Y}_T$
retrain

(a) Stage 1

(b) Stage 2

Fig. 3. *Proposed training strategy*. Our method combines different translations generated by an image translation module T to learn translation-specific classifiers. This removes the need of using a discrepancy loss, as each classifier learns independently from a specific translation, and allows the model to use all classifiers to label the target domain. By teaching a meta-learner C_m to find the optimal combination of all classifiers' outputs, we exploit all members of the ensemble and eliminate the usage of rule-based mechanisms. To facilitate the comparison with MTri, we illustrate Eq. (4) in Fig. 3(a), and more details in the selection of $X_S^{T_c}$ can be found in Section 5.2.

3. Preliminary

Goal of UDA. Given the source dataset $\mathcal{S}$ consisting of a set of images $X_\mathcal{S}$ with their corresponding semantic labels $Y_\mathcal{S}$ (e.g. synthetic data generated by computer graphic simulations) and the unlabeled target dataset $\mathcal{T}$ consisting only of the images $X_\mathcal{T}$ without labels, the goal of UDA is to design and train a neural network for semantic segmentation and to make it perform as close as possible to a model hypothetically trained on $X_\mathcal{T}$ with ground truth labels $Y_\mathcal{T}$.

3.1. Multi-task Tri-training (MTri) for UDA

In MTri, the model is trained in two stages: (1) pretraining, and (2) self-supervised learning. The goal of the first training stage is to ensure a certain level of knowledge in the network, while enforcing disagreement between the classifiers C_1 and C_2 (see Fig. 2(a)). This knowledge and disagreement is necessary for the self-supervised learning process, as in the second training stage C_1 and C_2 will play the role of teachers, labeling $X_\mathcal{T}$ using their predictions to learn a target-specific classifier C_3 (see Fig. 2(b)).

3.1.1. First training stage

To pretrain the model, batches of source images $X_\mathcal{S}$ are fed into the encoder and each classifier learns from the extracted features in a supervised way using standard cross entropy loss $\mathcal{L}_{seg}$ (Goodfellow et al., 2016). The disagreement between C_1 and C_2 is implemented using a cosine distance loss $\mathcal{L}_{cos}$ between the weights of these classifiers (Bousmalis et al., 2016). Overall, the loss for the first training stage is as follow:

$$\mathcal{L}_{stage1} = \mathcal{L}_{seg}(C_k(E(X_\mathcal{S})), Y_\mathcal{S}) + \mathcal{L}_{cos}(w_{C_1}, w_{C_2}) \tag{1}$$

where $k \in \{1, 2, 3\}$ represents the classifier index, E is the encoder depicted in Fig. 2, and w_{C_1}, w_{C_2} are the weights of the classifiers C_1 and C_2 respectively. This loss is optimized on the source data during a certain amount of iterations, yielding three subnetworks E, C_1, C_2 and C_3 that will be used in the following stage.

3.1.2. Second training stage

To perform self-supervised learning, the knowledge acquired in the network during the previous training stage is used to label the target images $X_\mathcal{T}$. This is done by using the predictions of the teacher networks C_1 and C_2 on $X_\mathcal{T}$, and then merging them using traditional majority voting (Hernández-González et al., 2019) in combination with confidence thresholding (Zou et al., 2018). This filters out spurious predictions resulting in the label set $\hat{Y}_\mathcal{T}$ for $X_\mathcal{T}$. As this process can be repeated a certain amount of rounds $i = \{1, 2, \ldots\}$, a typical self-supervision round i consists of the following substeps:

Substep 1: Creating the pseudo labels $\hat{Y}_\mathcal{T}^{(i-1)}$.

$$\hat{Y}_\mathcal{T}^{(i-1)} = \text{thresholding}(\text{majority\_voting}(C_1(X_\mathcal{T}), C_2(X_\mathcal{T}))) \tag{2}$$

Substep 2: Retraining to obtain a target-specific classifier C_3.

$$\begin{aligned}\mathcal{L}_{stage2} = {} & \mathcal{L}_{seg}(C_k(E(X_\mathcal{S})), Y_\mathcal{S}) + \mathcal{L}_{cos}(w_{C_1}, w_{C_2}) + \\ & \mathcal{L}_{seg}(C_3(E(X_\mathcal{T})), \hat{Y}_\mathcal{T}^{(i-1)})\end{aligned} \tag{3}$$

The overall training strategy is illustrated in Fig. 2.

4. Method

Our method brings forward a general design where different architecture components are proposed to address the weaknesses of Multi-task Tri-training. We present the overall arrangement of these components in Fig. 3. We describe the goal of each component and how they are trained in the following subsections.

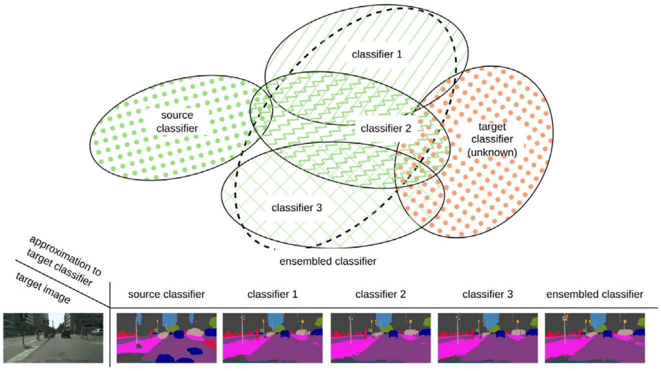


Fig. 4. *Conceptual illustration of our ensemble approach in the solution space*. In UDA, the goal is to approximate the target classifier, which remains unknown due to the unavailability of target labels. Meanwhile, the source classifier, trained solely on the source domain without any adaptation techniques, may fail to fully encompass the solution space of the target domain, as it lacks awareness of the underlying domain gap between the two domains. To tackle this challenge, our work generates multiple image translations from the source domain to train multiple classifiers, each focusing on a specific area of the solution space influenced by its corresponding image translation. Through the application of ensemble learning, we combine the diverse knowledge captured by each individual classifier, resulting in a unified classifier that integrates the collective solutions.

4.1. Image translation module

In contrast to MTri method using the source domain without introducing any visual variability on its images (see Eqs. (1) and (2)), we instead employ different image translations for each classifier in the ensemble. The benefit of using image translations on the source domain is threefold:

1. When these representations are fed to the encoder, the model is encouraged to learn feature invariance to the translations applied on the input images.
2. When the encoder output features of a specific translation are fed to a dedicated classifier, we enforce learning translation-specific classifiers that enables using all classifiers to create pseudo labels for the target domain.
3. When translation-specific classifiers are learnt, we implicitly ensure discrepancy across all classifiers without using an explicit discrepancy loss as in Eq. (1).

Our approach draws inspiration from recent works that have leveraged translation-specific knowledge to enhance the robustness of models in the face of domain shifts and, consequently, improve the quality of pseudo labels (Li et al., 2019; Yang and Soatto, 2020). In these studies, regardless of the specific translation method employed, it has been demonstrated that incorporating image translations into the training process helps the network become more resilient to potential domain shifts that may arise when the model is evaluated on previously unseen data, such as the target domain. By utilizing image translations during training, the network is exposed to various visual variations and transformations that simulate the differences between the source and target domains. This exposure enables the model to capture and learn representations that are invariant or robust to these domain shifts, thus increasing its ability to generalize well on unseen data.

Our method expands on this idea by proposing the use of multiple image translations with multiple classifiers. By using multiple image translations, we can generate abundant and diverse representations of the same source image, which can be used to learn translation-specific classifiers (see Fig. 4). When these classifiers are ensembled into a single classifier, the resulting predictor is more likely to capture the underlying structure of the target domain, as it can incorporate the specific domain shifts learned by each classifier. Therefore, in this work we hypothesize that the ensemble layer becomes more robust, which leads to higher quality pseudo-labels.

Nowadays, image-to-image translation research already provides many trained methods that we can directly use to obtain multiple image translations (Gatys et al., 2016; Huang and Belongie, 2017; Murez et al., 2018; Zhu et al., 2017). To facilitate a direct comparison with Multi-task Tri-training, we select an image translation method for T that provides three transformations, to obtain $X_S^{T_1}$, $X_S^{T_2}$ and $X_S^{T_3}$ from X_S (see Fig. 3). Implementation details on the used image translation network can be found in Section 5.2.

4.2. Ensemble layer

As mentioned, learning different translation-specific classifiers is desirable because it allows us to use all classifiers to generate high quality pseudo labels. To maximize the usage of all classifiers, we propose an ensemble layer C_m (see Fig. 3), which combines the predictions of all classifiers to determine the optimal pseudo labels for the target images. The proposed learnable layer is a form of weighted ensemble, which should be preferred over using a rule-based voting mechanism as shown in Eq. (2).

4.3. First training stage

Given the source annotated set of images X_S with label maps Y_S, and the alternative sets of image translations $X_S^{T_1}$, $X_S^{T_2}$ and $X_S^{T_3}$, the semantic segmentation outputs from each classifier C_k, are computed from the features $E(X_S^{T_k})$ of each transformation and used to train the encoder as well as the classifiers during the first training stage:

$$\mathcal{L}'_{stage1,k} = \mathcal{L}_{seg}(C_k(E(X_S^{T_k})), Y_S) \tag{4}$$

$\forall k \in \{1,2,3\}$. $\mathcal{L}_{seg}$ represents the standard cross entropy loss (Goodfellow et al., 2016) and $\mathcal{L}'_{stage1,k}$ is optimized for each classifier C_k independently. Because each classifier learns the features of a different image translation, there is a one-to-one correspondence between a classifier C_k and its transformation X^{T_k}, as depicted in Fig. 3. As observed in Eq. (4) and comparing with Eq. (1), making use of multiple different image translations removes the need of using $\mathcal{L}_{cos}$.

In addition, we add to Eq. (4) an adversarial loss (Li et al., 2019) and an entropy minimization loss (Yang and Soatto, 2020):

$$\begin{aligned}\mathcal{L}_{stage1,k} =& \mathcal{L}'_{stage1,k} + \lambda_{adv}\mathcal{L}_{adv}(E(X_S^{T_c}), E(X_\mathcal{T})) \\ &+ \lambda_{ent}\mathcal{L}_{ent}(C_k(E(X_\mathcal{T})))\end{aligned} \tag{5}$$

$\forall k \in \{1,2,3\}$. The adversarial loss, computed using the discriminator D depicted in Fig. 3, helps to align the encoder's features between $X_S^{T_c}$ and $X_\mathcal{T}$, which promotes learning domain invariant features (Tsai et al., 2018). Furthermore, the entropy minimization loss attempts to minimize the model's self-information on the target domain (Shannon, 1948), which is beneficial to avoid low confidence predictions (Vu et al., 2019).

Algorithm 1: Training process of our method

```
Input  : $(X_S, Y_S)$, $(X_\mathcal{T}, Y_\mathcal{T} = \emptyset)$
Output: $E$, $C_1$, $C_2$, $C_3$ and $C_m$
obtain $X_S^{T_1}$, $X_S^{T_2}$ and $X_S^{T_3}$ from $T$                // stage 1
train $E$, $C_k$ with Eq. (5) $\forall k \in \{1,2,3\}$
train $C_m$ with Eq. (6) using $(X_S, Y_S)$
generate $\hat{Y}_\mathcal{T}^{(0)}$ from $C_m$ using $X_\mathcal{T}$            // stage 2
for $i \leftarrow 1$ to number of rounds do
    train $E$, $C_1$, $C_2$, $C_3$ with Eq. (8) $\forall k \in \{1,2,3\}$
    retrain $C_m$ with Eq. (9) using $(X_\mathcal{T}, \hat{Y}_\mathcal{T}^{(i-1)})$
    generate $\hat{Y}_\mathcal{T}^{(i)}$ from $C_m$ using $X_\mathcal{T}$
end
```

In contrast to Eq. (1) used by MTri, Eq. (5) encourages the model to use the target domain information already from the first training stage, which helps to mitigate the domain gap between the source and target domain.

To finish the first training stage, the meta-learner C_m is trained to ensemble the outputs of the classifiers C_1, C_2 and C_3 with respect to the transformations of the source dataset. This is done by freezing the weights of $\{E, C_1, C_2, C_3\}$, to obtain the weight vectors w_1, w_2 and w_3 by minimizing the cross-entropy loss:

$$\underset{\mathrm{w}_1,\mathrm{w}_2,\mathrm{w}_3}{\arg\min}\, \mathcal{L}_{seg}(C_m(X_S^{T_1}, X_S^{T_2}, X_S^{T_3}), Y_S), \tag{6}$$

where the output of the meta-learner is computed for every pixel (h, w) as follows:

$$\begin{aligned} &C_m(X_S^{T_1}, X_S^{T_2}, X_S^{T_3})^{(h,w)} = \mathrm{w}_1 \odot C_1(E(X_S^{T_1}))^{(h,w)} \\ &+\mathrm{w}_2 \odot C_2(E(X_S^{T_2}))^{(h,w)} + \mathrm{w}_3 \odot C_3(E(X_S^{T_3}))^{(h,w)}. \end{aligned} \tag{7}$$

To ensure convergence of the weight vectors w_k, we restrict their range to be between 0 and 1. The dimension of the weight vectors is the same as the total number of classes, and $\odot$ denotes element-wise multiplication. Since the output of C_m for a given pixel and class depends only on the three classifier outputs for that specific pixel and class, we refer to it as a sparse version of the standard Multinomial Logistic Regression (Bishop, 2006).

4.4. Second training stage

The second stage consists mainly in self-supervised learning, a process in which the meta-learner C_m generates pseudo-labels $\hat{Y}_\mathcal{T}$ for the target images $X_\mathcal{T}$ to retrain the semantic segmentation network. Since C_m is already trained, we feed the features of the target images $E(X_\mathcal{T})$ into all the classifiers in Eq. (7). As a result, we obtain the ensembled probability maps, on which we apply global confidence thresholding (Li et al., 2019; Yang and Soatto, 2020) to obtain the initial pseudo-labels $\hat{Y}_\mathcal{T}^{(0)}$. Hereafter, we start the self-supervised learning rounds $i = \{1, 2, \dots\}$, each one consisting of three steps. First, the semantic segmentation network is retrained through the loss:

$$\mathcal{L}_{stage2,k} = \mathcal{L}_{stage1,k} + \mathcal{L}_{seg}(C_k(E(X_\mathcal{T})), \hat{Y}_\mathcal{T}^{(i-1)}) \tag{8}$$

$\forall k \in \{1, 2, 3\}$. Second, the meta-learner C_m is retrained on the predictions of the three updated classifiers on the target images along with the pseudo-labels $\hat{Y}_\mathcal{T}^{(i-1)}$:

$$\underset{\mathrm{w}_1,\mathrm{w}_2,\mathrm{w}_3}{\arg\min}\, \mathcal{L}_{seg}(C_m(X_\mathcal{T}, X_\mathcal{T}, X_\mathcal{T}), \hat{Y}_\mathcal{T}^{(i-1)}) \tag{9}$$

And finally, with the updated weight vectors w_k, the meta-learner is able to generate new pseudo-labels for the target images $\hat{Y}_\mathcal{T}^{(i)}$ to be used for the next rounds. The number of SSL rounds will be dictated by C_m, specifically until the performance gap between C_m and the three classifiers C_k is no longer significant. The entire training procedure is summarized in Algorithm 1.

5. Experiments

To show the benefits of our proposed approach, we evaluate the method on the challenging, synthetic-to-real UDA benchmarks for semantic segmentation by performing the following experiments:

1. Comparison with Multi-task Tri-training. Since our approach improves upon the baseline described in Section 3 by adding image translations, the goal of this experiment is to show how our method, that combines image translations with ensemble learning performs in comparison to Multi-task Tri-training (Zhang et al., 2018), which does not use image translations but uses ensemble learning. We do this by implementing the corresponding losses of these methods from the first training stage. In addition, to show the advantages of using multiple classifiers, we compare our method against a vanilla single encoder–decoder architecture (SED) (Li et al., 2019; Yang and Soatto, 2020; Yang et al., 2020a; Zou et al., 2018), that is trained on Eq. (5) with $k = 1$. Finally, we show the distribution of the weights for the meta-learner over all the classes and analyze the influence of using the target domain information via entropy minimization.

2. Comparison with state-of-the-art methods. As it is a standard practice in UDA research, this experiment compares our proposed UDA approach with current state-of-the-art methods, putting our model into context with approaches that rely on a combination of image translation, feature alignment, model regularization and self-supervised learning.

3. Generalization test to unseen data. We conduct an additional experiment to assess the generalization capacity of UDA methods to unseen data, which is often overlooked in traditional UDA benchmarks. The protocol for this experiment consists of first adapting the source to the target dataset, and then evaluating the resulting model on a dataset that was not seen during training. We propose WildDash (Zendel et al., 2018) to be the unseen dataset, as it contains many real-world images collected from challenging driving scenarios. To compare with other state-of-the-art methods, we select those whose code is publicly available and provide an evaluation script, and proceed to (1) reproduce their result on the proposed synthetic-to-real benchmark and (2) evaluate the model on WildDash. In this experiment, we also included an analysis of the architecture arrangement to evaluate the efficiency of the methods and reported the results for both the target dataset and the unseen dataset.

4. Flexibility assessment of ensemble learning. To evaluate the flexibility of our ensemble strategy and study the role of each classifier, we propose a perturbation analysis. Specifically, we randomly assign one member of the ensemble to predict noise, thereby exposing the network to a failure scenario. The goal of this experiment is to study the adaptation capacity of the meta-learner C_m and how the remaining classifiers, including each individual classifier, can adjust to the perturbed inputs. We limit our analysis to the first training stage of the framework and evaluate the individual performance of each classifier, as well as that of the meta-learner C_m.

For all experiments, we use GTA V (Richter et al., 2016) and Cityscapes (Cordts et al., 2016) to represent the source and target domains respectively, measuring the performance using the mIoU (Everingham et al., 2014) on a unseen portion of the target domain. As it is a standard practice when comparing state-of-the-art UDA methods, we also adapt SYNTHIA (Ros et al., 2016) to Cityscapes (Cordts et al., 2016) in the second experiment.

Details regarding these experiments are provided in the following subsections, and in Section 6 we present the results.

5.1. Datasets

Target dataset. Cityscapes (Cordts et al., 2016) is a large-scale and real urban scene semantic segmentation dataset that provides 5000 finely annotated images split into three sets: train (2975), validation (500) and test (1525). These sets are pixel-wise labeled, with a resolution of 1024 × 2048 pixels. The number of classes is 34 but only 19 are officially considered in the evaluation protocol.

Source datasets. GTA V (Richter et al., 2016) is a synthetic dataset that contains 24966 labeled frames taken from a realistic open-world computer game called Grand Theft Auto V (GTA V). The resolution of the images is 1052 × 1914 pixels and most of the frames are vehicle-egocentric. All the classes are compatible with the 19 official classes of Cityscapes. SYNTHIA (Ros et al., 2016) is a synthetic dataset consisting of driving scenes rendered from a virtual city. We use the SYNTHIA-RAND-CITYSCAPES subset as source set, which contains 9400 1280 × 760 images for training and 16 common classes with Cityscapes, and we evaluate the resulting model on these 16 classes.

Table 1
Adaptation from GTA V→Cityscapes, analyzing different architectures as well as the impact of entropy minimization for the first training stage. We show IoU for each class and total mean IoU. Apart from indicating the best IoU in bold, we make an intra comparison between C_1, C_2 and C_3 against C_m when using entropy minimization (underlined with red), and when not using the entropy loss (underlined with blue). Note that although entropy minimization helps to close the gap between SED and our method, there is still a remarkable difference specially considering the performance of C_m.

GTA V → Cityscapes																				
Experiment	road	side.	buil.	wall	fence	pole	light	sign	veget.	terr.	sky	person	rider	car	truck	bus	train	motor	bike	**mIoU**
SED (w/o ent)	77.72	35.69	78.91	**33.2**	19.95	34.86	25.25	3.28	80.77	34.37	71.85	60.06	18.46	84.6	22.66	21.41	1.09	23.29	21.82	39.43
SED (w/ ent)	84.89	34.30	**82.64**	31.24	18.91	**36.78**	32.18	15.20	82.33	31.89	72.78	63.42	13.43	83.36	24.20	25.15	0.06	30.96	30.08	41.78
MTri (Zhang et al., 2018) (C_1) (w/ ent)	64.88	19.33	61.55	12.76	20.81	30.61	**42.13**	14.69	75.2	12.17	60.8	**64.45**	**29.6**	82.1	25.61	32.41	5.29	32.92	27.09	37.6
MTri (Zhang et al., 2018) (C_2) (w/ ent)	62.1	19.64	59.0	15.18	20.87	30.43	41.99	14.55	75.41	12.2	60.67	64.35	29.47	82.18	25.74	32.46	5.34	33.04	27.04	37.46
MTri (Zhang et al., 2018) (C_3) (w/ ent)	58.11	19.18	55.65	16.78	21.13	30.39	41.91	13.93	75.84	12.14	58.99	64.1	29.0	82.95	25.99	**32.51**	**5.61**	**33.75**	27.46	37.13
Ours (C_1) (w/o ent)	82.9	34.06	74.9	25.74	15.76	33.8	33.6	17.09	84.94	34.37	74.21	60.81	14.65	84.73	23.86	26.31	0.64	22.14	32.0	40.87
Ours (C_2) (w/o ent)	78.24	31.48	71.71	26.37	19.18	36.22	32.49	25.61	85.1	31.41	**84.28**	60.28	18.06	84.79	26.16	29.64	0.28	23.29	32.9	41.97
Ours (C_3) (w/o ent)	81.91	30.15	77.22	26.38	15.0	34.63	31.53	27.42	83.75	35.18	81.37	61.71	17.32	**85.07**	**26.5**	29.86	0.2	21.36	33.43	42.10
Ours (C_m) (w/o ent)	82.78	35.74	75.81	26.83	19.89	34.96	34.47	25.60	**85.23**	**35.35**	79.75	62.02	14.82	84.68	25.30	32.05	0.03	27.48	41.23	43.39
Ours (C_1) (w/ ent)	83.24	33.4	78.04	27.48	18.37	33.26	35.34	22.34	83.87	27.47	82.2	62.7	28.26	80.76	20.49	15.41	0.22	27.12	37.78	41.99
Ours (C_2) (w/ ent)	84.67	33.64	80.30	27.33	19.37	35.95	33.10	27.49	83.84	30.29	81.53	61.98	26.54	81.50	21.48	20.18	0.03	29.05	40.57	43.10
Ours (C_3) (w/ ent)	**87.22**	**36.57**	81.26	28.65	17.82	35.55	32.58	29.11	83.46	30.39	77.06	62.48	28.78	81.49	22.75	22.85	0.06	29.85	35.14	43.32
Ours (C_m) (w/ ent)	85.29	35.57	81.69	29.93	**20.24**	35.53	36.63	**35.94**	83.24	28.1	81.75	63.75	29.18	81.8	23.44	24.58	4.67	31.0	**47.26**	**45.24**

Unseen dataset. WildDash (Zendel et al., 2018) is a real-world dataset containing 4256 finely annotated images in a pixel-wise manner, created with the purpose of testing the robustness of models under different driving scenarios (e.g. rain, road coverage, darkness, overexposure). These images have a resolution of 1920 × 1080 pixels and the labels are fully compatible with Cityscapes.

5.2. Implementation details

Image translation module and adversarial loss. We have chosen CycleGAN (Zhu et al., 2017) as our image translation model T, as it is a commonly used approach that can be trained efficiently. Its architecture provides two image translations: the translation from the source to the target domain X^t_S and the reconstruction back to the source set X^r_S. These two translations are combined with the original source images to obtain three different representation $X^{T_1}_S = X_S$, $X^{T_2}_S = X^r_S$ and $X^{T_3}_S = X^t_S$. Since feature alignments between target-like and target images is frequently done when using CycleGAN's as image translation module (Li et al., 2019; Hoffman et al., 2018), we assign $X^{T_c}_S$ to be $X^{T_3}_S = X^t_S$ in Eq. (5).

Training protocols for MTri and SED. In the first experiment, we have respected the training protocol of MTri as described in Section 3.1.1, by implementing Eq. (1). To allow for a fair comparison with our method, we also added the entropy minimization term to the MTri loss. As for the single encoder–decoder (SED) approach, all our losses were implemented using one encoder and one classifier, while using all three available transformations. In essence, the SED approach is similar to our approach but does not use the ensemble approach with the three classifiers.

Hardware and network architecture. In our experiments, we have implemented our method using Tensorflow (Abadi et al., 2016) and trained our model using a single NVIDIA TITAN RTX with 24 GB memory. Regarding the segmentation network, we have chosen ResNet101 (He et al., 2016) pretrained on ImageNet (Deng et al., 2009) as feature extractor for E. When it comes to the decoders $C_k, k \in \{1, 2, 3\}$, DeepLab-v2 (Chen et al., 2018) framework was used. Throughout the training process, we use SGD (Bottou, 2010) as optimizer with momentum of 0.9, encoder and decoders follow a poly learning rate policy, where the initial learning rate is set to $2.5e^{-4}$. In the second and third experiments where our model is trained end-to-end, we use the same hyperparameters reported in Li et al. (2019), Yang and Soatto (2020) for the adversarial and entropy minimization loss respectively. During the first stage the network is trained for 150 k iterations. Then we perform SSL until convergence is reached on each round. We use a crop size of 512 × 1024 during training, and we evaluate on full resolution 1024 × 2048 images from Cityscapes validation split.

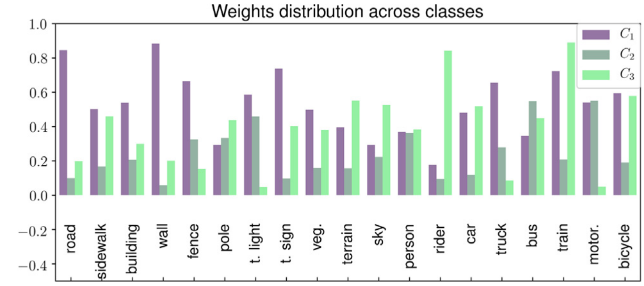


Fig. 5. *Weights after optimizing Eq. (7) for the adaption GTA V→ Cityscapes.* Before starting SSL, C_1 and C_3 are the most dominant predictors on the output space of C_m.

6. Results

6.1. Comparison with multi-task tri-training

The results of this experiment are summarized in Table 1. Our method, which combines image translations with ensemble learning, outperforms Multi-task Tri-training, that uses ensemble learning without any image translations. The superiority of our method is particularly pronounced when taking into account the predictions of the meta-learner C_m. Notably, C_m achieves better performance than the individual classifiers in a significant number of classes (14 with entropy minimization, 11 without entropy minimization).

If our approach were to use only one classifier, as is the case with the baseline SED, the method would still outperform Multi-task Tri-training by around +4.32 mIoU points. Note that using one classifier would essentially disregard the influence of ensemble learning, as an ensemble requires by definition more than one classifier. This means that using image translations alone has a positive effect in our method, which can be further boosted in combination with ensemble learning.

Fig. 5 shows the weights learnt by the meta-learner, that quantify the proportion that each classifier contributes to the final prediction of C_m. Analyzing the classes in Table 1 where our method outperforms all three classifiers (e.g., *fence*, *traffic light*, *rider*, *bus*, *bike*), we observe a pattern in Fig. 5. The meta-learner tends to amplify the contribution of the two best performing classifiers while penalizing the classifier with the lowest mIoU. For classes like *traffic sign* and *truck*, the meta-learner prefers a combination of weak classifiers, resulting in improved predictions that surpass the performance of the strongest member of the ensemble.

These findings highlight the advantages of incorporating image translations into the ensemble learning framework, as our method demonstrates superior performance compared to Multi-task Tri-training.

Table 2
Adapting from GTA V to Cityscapes. S1 and S2 indicate the training stage, while R1 and R2 denote the first and second round of SSL, respectively.

GTA V→ Cityscapes																				
Method	road	side.	buil.	wall	fence	pole	light	sign	veget.	terr.	sky	person	rider	car	truck	bus	train	motor	bike	**mIoU**
DCAN (Wu et al., 2018b)	85.0	30.8	81.3	25.8	21.2	22.2	25.4	26.6	83.4	36.7	76.2	58.9	24.9	80.7	29.5	42.9	2.5	26.9	11.6	41.7
DLOW (Gong et al., 2019)	87.1	33.5	80.5	24.5	13.2	29.8	29.5	26.6	82.6	26.7	81.8	55.9	25.3	78.0	33.5	38.7	0.0	22.9	34.5	42.3
CLAN (Luo et al., 2018)	87.0	27.1	79.6	27.3	23.3	28.3	35.5	24.2	83.6	27.4	74.2	58.6	28.0	76.2	33.1	36.7	6.7	31.9	31.4	43.2
ABStruct (Chang et al., 2019)	91.5	47.5	82.5	31.3	25.6	33.0	33.7	25.8	82.7	28.8	82.7	62.4	30.8	85.2	27.7	34.5	6.4	25.2	24.4	45.4
ADVENT (Vu et al., 2019)	89.4	33.1	81.0	26.6	26.8	27.2	33.5	24.7	83.9	36.7	78.8	58.7	30.5	84.8	38.5	44.5	1.7	31.6	32.4	45.5
BDL (Li et al., 2019)	91.0	44.7	84.2	34.6	27.6	30.2	36.0	36.0	85.0	**43.6**	83.0	58.6	31.6	83.3	35.3	49.7	3.3	28.8	35.6	48.5
FDA-MBT (Yang and Soatto, 2020)	92.5	**53.3**	82.4	26.5	27.6	36.4	40.6	38.9	82.3	39.8	78.0	62.6	**34.4**	84.9	34.1	**53.1**	16.9	27.7	46.4	50.45
PCEDA (Yang et al., 2020b)	91.0	49.2	85.6	**37.2**	**29.7**	33.7	38.1	39.2	85.4	35.4	85.1	61.1	32.8	84.1	**45.6**	46.9	0.0	34.2	44.5	50.5
Ours S1 (C_3)	87.22	36.57	81.26	28.65	17.82	35.55	32.58	29.11	83.46	30.39	77.06	62.48	28.78	81.49	22.75	22.85	0.06	29.85	35.14	43.32
Ours S1 (C_m)	85.29	35.57	81.69	29.93	20.24	35.53	36.63	35.94	83.24	28.1	81.75	63.75	29.18	81.8	23.44	24.58	4.67	31.0	47.26	45.24
Ours S2-R1 (C_3)	90.6	46.94	84.06	31.9	23.88	37.53	34.81	34.37	85.69	36.02	84.32	66.53	29.41	85.46	27.77	32.48	7.15	**36.05**	54.96	48.94
Ours S2-R1 (C_m)	90.81	47.85	85.01	32.08	24.55	37.73	38.15	42.13	85.37	34.32	84.97	66.51	28.0	84.51	27.0	25.14	15.23	35.03	56.02	49.50
Ours S2-R2 (C_2)	92.27	51.59	86.19	35.28	26.84	36.73	35.68	42.4	**86.84**	37.3	**85.49**	66.9	27.6	85.75	32.26	32.85	**20.59**	33.89	**58.1**	51.29
Ours S2-R2 (C_m)	**92.59**	53.05	**86.31**	34.2	27.17	**39.13**	**41.0**	**44.8**	86.1	34.32	84.69	**67.23**	29.77	**85.78**	32.73	29.9	20.12	35.55	57.05	**51.66**

Table 3
Adapting from SYNTHIA to Cityscapes. Total mIoU values with * are reported only on 13 subclasses (excluding *wall*, *fence* and *pole*). Our method achieves the best performance over all the 16 classes.

SYNTHIA→ Cityscapes																	
Method	road	side.	buil.	wall*	fence*	pole*	light	sign	veget.	sky	person	rider	car	bus	motor	bike	**mIoU**
AdaptPatch (Tsai et al., 2019)	82.4	38.0	78.6	8.7	0.6	26.0	3.9	11.1	75.5	84.6	53.5	21.6	71.4	32.6	19.3	31.7	40.0
AdaptSegNet (Tsai et al., 2018)	79.2	37.2	78.8	–	–	–	9.9	10.5	78.2	80.5	53.5	19.6	67.0	29.5	21.6	31.3	45.9*
BDL (Li et al., 2019)	**86.0**	**46.7**	80.3	–	–	–	14.1	11.6	79.2	81.3	54.1	27.9	73.7	**42.2**	25.7	45.3	51.4*
CLAN (Luo et al., 2018)	81.3	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*
FDA-MBT (Yang and Soatto, 2020)	79.3	35.0	73.2	–	–	–	19.9	24.0	61.7	82.6	61.4	**31.1**	83.9	40.8	**38.4**	**51.1**	52.5*
PCEDA (Yang et al., 2020b)	85.9	44.6	**80.8**	**9.0**	**0.8**	32.1	24.8	23.1	79.5	83.1	57.2	29.3	73.5	34.8	32.4	48.2	46.2
Ours (C_m)	81.90	41.88	78.21	3.38	0.02	**44.76**	**24.82**	**27.17**	**86.59**	**85.18**	**68.74**	30.55	**84.65**	24.42	20.12	40.77	**46.45**

Table 4
Number of classes where each classifier outperforms the others on GTA V→ Cityscapes. Although there is a clear dominance of C_3 before starting SSL, this trend tends to wear off as the self supervision process unfolds.

classifier	stage 1	SSL: round 1	SSL: round 2
C_1	5	6	4
C_2	4	7	7
C_3	10	6	8

Table 5
Influence of T on GTA V→ Cityscapes during first round of SSL. Using T during SSL by feeding a transformation to its corresponding classifier provokes a slight performance drop in the mIoU.

classifier	stage 1	SSL without T	SSL with T
C_1	41.99	48.87	47.9
C_2	43.10	48.91	47.8
C_3	43.32	48.94	48.0

6.2. Comparison with state-of-the-art methods

The quantitative results for the adaption GTA V→ Cityscapes can be seen in Table 2. When comparing it with state-of-the-art methods, we can see that our approach outperforms PCEDA (Yang et al., 2020b), FDA-MBT (Yang and Soatto, 2020) and BDL (Li et al., 2019), three recent methods that use a combination of strategies. Qualitative results in Fig. 7 shows satisfactory results on the output space, leading to consistently clean predictions.

As for SYNTHIA→ Cityscapes, the mIoU values are shown in Table 3. We achieved competitive results over all 16 classes with respect to state-of-the-art methods such as PCEDA (Yang et al., 2020b) and AdaptPatch (Tsai et al., 2019), dominating on some difficult classes such as *pole*, *traffic light* and *traffic sign*.

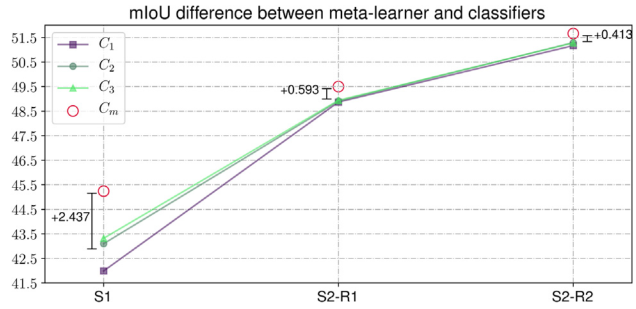


Fig. 6. *Effect of SSL for C_m.* Retraining the meta-learner after each round of SSL makes sure that it keeps outperforming the other classifiers with a decreasing margin.

If we analyze the improvements during SSL, we see that the meta-learner consistently scores better than the individual three classifiers, although its gain diminishes with each round of SSL (see Fig. 6). This can be attributed to the fact that all three classifiers are being optimized with the same images, and thus the model is losing the capability to keep diversity among the predictors. The results from also show that the dominance of the members of the ensemble can alternate since C_3 is the best predictor after the first round (R1) and C_2 takes over after the second one (R2). This suggests that some predictors can learn more than others, even if they share the same input images, showing that all of them are equally important. This can be better appreciated in

Table 6
Generalization test on WildDash after adapting GTA V→ Cityscapes.

Method	# encoders	# classifiers	mIoU Cityscapes	mIoU WildDash
ADVENT (Vu et al., 2019)	1	1	45.5	25.9
BDL (Li et al., 2019)	1	1	48.5	26.57
FDA-MBT (Yang and Soatto, 2020)	3	3	50.45	31.07
Ours	1	3	51.66	**31.2**

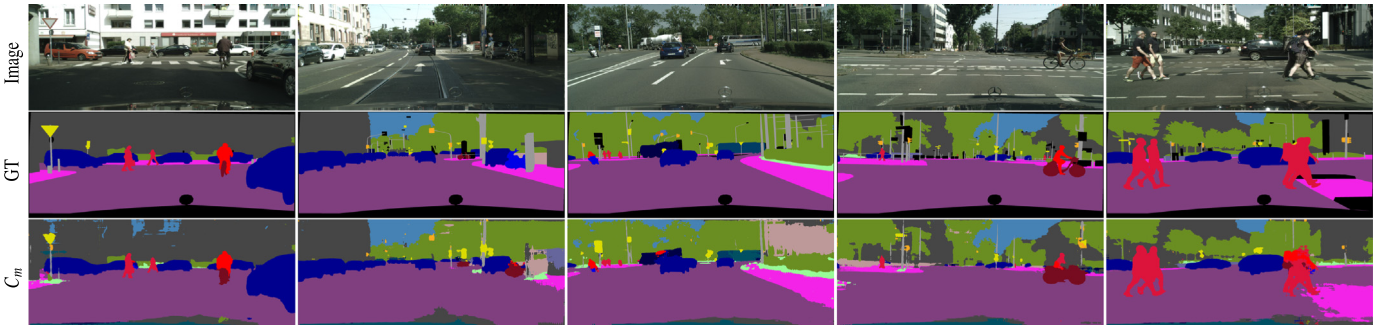


Fig. 7. *Qualitative comparison from GTA V to Cityscapes*. The meta-learner rebalances the predictions from C_1, C_2 and C_3 to achieve a smoother output over all the classes, where less predominant classes such as *t. sign* and *pole* have more presence on the pixel space of C_m.

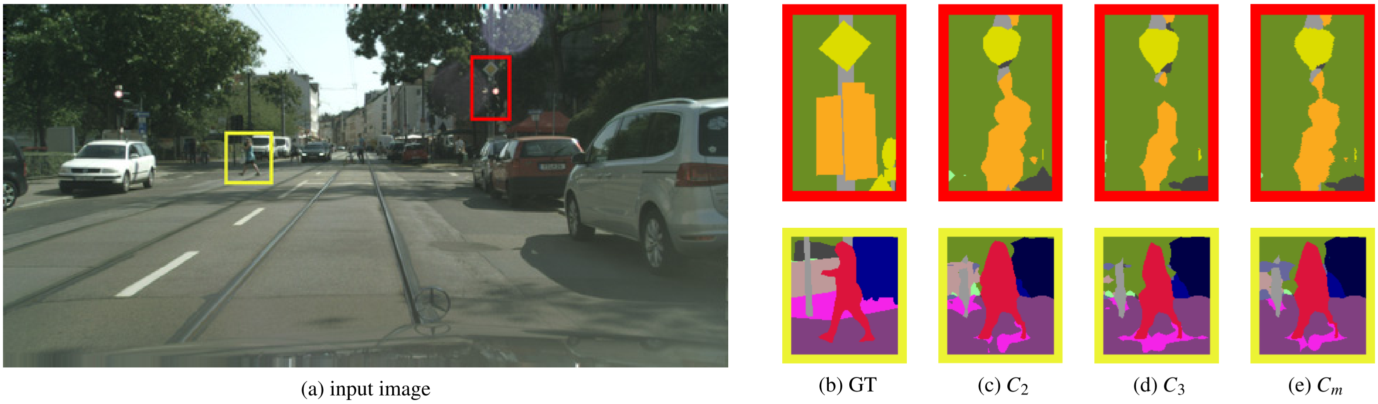

(a) input image
(b) GT
(c) C_2
(d) C_3
(e) C_m

Fig. 8. *Qualitative results during stage 1 when C_1 simulates a corrupt classifier.* The image belongs to the validation set of Cityscapes, and the predictions were made for the adaption GTA V→ Cityscapes. The meta-learner has built its final output by taking considerably into account the prediction from C_2 for the class *traffic light*, but for *traffic sign* it opted for C_3 (see red crop). On the other hand, C_m relies on C_3 for the classes *person* and *pole*, explaining the similarity between Figs. 8(b) and 8(c) when looking at the yellow crop.

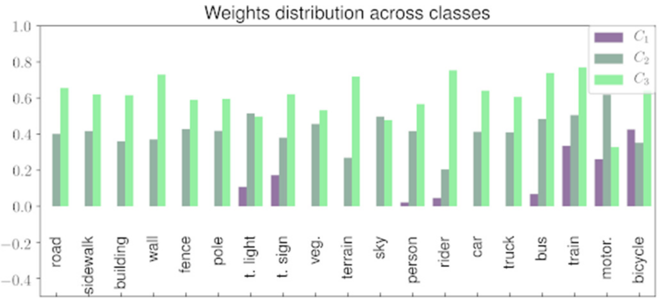


Fig. 9. *Weights' distribution after training the meta-learner in Eq. (7), when C_1 outputs only random noise for the set-up GTA V→ Cityscapes.*.

Table 4 where C_2 stands out after R1 and remains close to C_3 after R2 when analyzing the mIoU per class during SSL.

Using the image translation module T during the second stage by transforming the target set into the closest transformation possible to each classifier, i.e., transforming the target images to the source domain for the first two classifiers while keeping them unaltered for the third one, leads to slightly worse performance (see Table 5). This can be attributed to the fact that, since SSL aims to close the gap for the target distribution, it is needed to keep the inputs as similar as possible to those that the algorithm would receive during inference.

6.3. Generalization test to unseen data

The results of the proposed generalization test in Table 6 shows different UDA methods along with their arrangement for the semantic segmentation network and the corresponding mIoU performance on Cityscapes and WildDash, after adapting GTA V to Cityscapes. ADVENT (Vu et al., 2019) is a UDA approach that does not leverage any image translation strategy, while BDL (Li et al., 2019) and FDA-MBT (Yang and Soatto, 2020) make use of one and three image translations respectively. If we consider the amount of encoders and classifiers, we can notice that using a single encoder–decoder gives a limiting generalization performance, although BDL outperforms ADVENT. This slightly better performance of BDL can be attributed to the usage of one image translation (transforming the source to the target with CycleGAN) to increase the robustness of the model.

FDA-MBT uses three image representations, mapping the source annotated images to the target using three different parameters for the image translation module, and performing UDA by training each encoder–decoder segmentation model with a specific representation.

Table 7
mIoU per classifier for GTA V→ Cityscapes during stage 1 when C_1 plays as a noisy predictor. The reported values are compared with those reported in Section 6.1, achieving a slightly similar performance when having only two functional classifiers.

experiment	mIoU C_1	mIoU C_2	mIoU C_3	mIoU C_m
Normal conditions	41.99	43.10	43.32	45.24
Noisy classifier	0.1	41.99	43.10	45.11

The reported performance of 31.07 is the result of averaging the three trained models, and although it is close to ours, our semantic segmentation network takes up significantly fewer parameters to train (one encoder and three classifiers). This makes our approach attractive as it is a good trade-off between its number of parameters and performance. More importantly, this experiment shows the advantageous effect of using multiple image translations, as done in FDA-MBT and our approach, on the generalizability of the trained models.

6.4. Flexibility assessment of ensemble learning

We conducted an analysis to assess the flexibility of our ensemble approach to learn to disregard useless classifiers. We do this by subjecting one member of the ensemble to predict random noise. Specifically, we selected C_1 as the random noise predictor, while C_2 and C_3 remained trained and intact after the first training stage. We modeled C_1 following a standard normal distribution with mean 0 and standard deviation of 1.

The results of this experiment are shown in Table 7, which demonstrates that the meta-learner C_m was able to bypass C_1 while maintaining a relatively similar mIoU performance compared to the experiment conducted in Section 6.1, where all three classifiers were used. This indicates that C_m has successfully re-adapted to a new scenario where only two productive classifiers are available, reducing the participation of the corrupt classifier to the minimum. Further analysis shows that the individual classifiers also demonstrated strong performance, with C_2 and C_3 achieving mIoU scores of 41.99 and 43.10, respectively. However, C_1 had a significantly lower mIoU score, indicating that it did not contribute meaningfully to the overall ensemble performance.

The effect of this rebalancing is shown in more detail in Fig. 8, depicting the output of all members of the ensemble in two particular crops. The red crop focuses on the classes *traffic light*, *traffic sign* and *pole*; while the yellow one analyzes the segmentation maps of a *pedestrian*. We can observe in Fig. 8(c) from the red crop that the meta-learner tends to share a similar shape in the prediction map than C_2 in Fig. 8(a) when analyzing the class *traffic light*, although the amount of pixels is clearly different. This behavior is expected in accordance to the weights distribution from Fig. 9, since C_2 has a slightly higher coefficient than C_3 for that particular class. The opposite situation takes place when we analyze the yellow crop, since C_m acts more aligned with the third classifier for *pedestrian* and *pole*, differing slightly in shape with the latter.

Our results suggest that our ensemble approach is robust to perturbations in individual classifiers, and that the meta-learner is able to effectively adapt to new scenarios. The individual classifier analysis highlights the importance of selecting high-performing classifiers for the ensemble approach, as poor-performing classifiers can have a negative impact on the overall performance. These findings can inform the development of more robust ensemble approaches in the future.

7. Conclusions

In this work, we proposed an UDA approach for semantic segmentation that effectively combines image translations, self-supervised learning, and ensemble learning in a novel way. We used challenging synthetic-to-real semantic segmentation UDA benchmarks to show that the proposed method improves the accuracy not only on target data, but also on unseen data that was never used during training.

We can conclude from the results that increasing the input variability via different image translations induces the network to learn domain invariant representations in the feature extractor while enforcing translation-specific features on each classifier. In addition, by employing a meta-learning layer, multiple ensemble classifiers can be used to generate high-quality pseudo-labels and thereby improve the self-supervised learning process.

We should note that although we focused on the standard synthetic-to-real UDA benchmarks, it is also possible to extend this work to real-to-real applications where both source and target domains are real-world datasets. While this can represent a more realistic application of UDA, we consider that in this work we made a step in improving the generalization capability of deep learning models.

CRediT authorship contribution statement

Fabrizio J. Piva: Conceptualization, Methodology, Software, Validation, Formal analysis, Investigation, Data curation, Visualization, Project administration, Writing – original draft, Writing – review & editing. **Gijs Dubbelman:** Resources, Writing – original draft, Writing – review & editing, Supervision, Funding acquisition.

Declaration of competing 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.

Data availability

We used publicly available data, that can be accessed in our References section

Acknowledgments

This work was funded by the Nederlandse Organisatie voor Wetenschappelijk Onderzoek (NWO), within the Efficient Deep Learning program, funding ID P16-25 project 3 (VAS).

References

Abadi, M., Agarwal, A., Barham, P., Brevdo, E., Chen, Z., Citro, C., Corrado, G.S., Davis, A., Dean, J., Devin, M., Ghemawat, S., Goodfellow, I.J., Harp, A., Irving, G., Isard, M., Jia, Y., Józefowicz, R., Kaiser, L., Kudlur, M., Levenberg, J., Mané, D., Monga, R., Moore, S., Murray, D.G., Olah, C., Schuster, M., Shlens, J., Steiner, B., Sutskever, I., Talwar, K., Tucker, P.A., Vanhoucke, V., Vasudevan, V., Viégas, F.B., Vinyals, O., Warden, P., Wattenberg, M., Wicke, M., Yu, Y., Zheng, X., 2016. TensorFlow: A system for large-scale machine learning. In: USENIX. OSDI, pp. 265–283.

Bishop, C.M., 2006. Pattern Recognition and Machine Learning. Springer, pp. 209–210.

Blum, A., Mitchell, T., 1998. Combining labeled and unlabeled data with co-training. In: COLT. pp. 92–100.

Bottou, L., 2010. Large-scale machine learning with stochastic gradient descent. In: COMPSTAT. pp. 177–186.

Bousmalis, K., Trigeorgis, G., Silberman, N., Krishnan, D., Erhan, D., 2016. Domain separation networks. In: NIPS. pp. 343–351.

Bucher, M., Vu, T., Cord, M., Pérez, P., 2021. Handling new target classes in semantic segmentation with domain adaptation. Comput. Vis. Image Underst. 212, 103258.

Chang, W., Wang, H., Peng, W., Chiu, W., 2019. All about structure: Adapting structural information across domains for boosting semantic segmentation. In: CVPR. pp. 1900–1909.

Chen, L., Papandreou, G., Kokkinos, I., Murphy, K., Yuille, A.L., 2018. DeepLab: Semantic image segmentation with deep convolutional nets, atrous convolution, and fully connected CRFs. TPAMI 834–848.

Cordts, M., Omran, M., Ramos, S., Rehfeld, T., Enzweiler, M., Benenson, R., Franke, U., Roth, S., Schiele, B., 2016. The cityscapes dataset for semantic urban scene understanding. In: CVPR. pp. 3213–3223.

Deng, J., Dong, W., Socher, R., Li, L., Li, K., Li, F., 2009. ImageNet: A large-scale hierarchical image database. In: CVPR. pp. 248–255.

Dietterich, T.G., 2000. Ensemble methods in machine learning. In: Kittler, J., Roli, F. (Eds.), Multiple Classifier Systems, First International Workshop, MCS 2000, Cagliari, Italy, June 21-23, 2000, Proceedings. In: Lecture Notes in Computer Science, vol. 1857, Springer, pp. 1–15.

Everingham, M., Eslami, S.M.A., Gool, L.V., Williams, C.K.I., Winn, J.M., Zisserman, A., 2014. The pascal visual object classes challenge: A retrospective. IJCV 98–136.

Gatys, L.A., Ecker, A.S., Bethge, M., 2016. Image style transfer using convolutional neural networks. CVPR 2414–2423.

Gong, R., Li, W., Chen, Y., Van Gool, L., 2019. DLOW: Domain flow for adaptation and generalization. CVPR 2477–2486.

Goodfellow, I.J., Bengio, Y., Courville, A., 2016. Deep Learning. MIT Press, Cambridge, MA, USA.

Hady, M.F.A., Schwenker, F., 2008. Co-training by committee: A new semi-supervised learning framework. In: ICDM. pp. 563–572.

He, K., Zhang, X., Ren, S., Sun, J., 2016. Deep residual learning for image recognition. In: CVPR. pp. 770–778.

Hendrycks, D., Mazeika, M., Kadavath, S., Song, D., 2019. Using self-supervised learning can improve model robustness and uncertainty. In: Wallach, H.M., Larochelle, H., Beygelzimer, A., d'Alché-Buc, F., Fox, E.B., Garnett, R. (Eds.), Advances in Neural Information Processing Systems 32: Annual Conference on Neural Information Processing Systems 2019, NeurIPS 2019, December 8-14, 2019, Vancouver, BC, Canada. pp. 15637–15648.

Hernández-González, J., Inza, I., Lozano, J.A., 2019. A note on the behavior of majority voting in multi-class domains with biased annotators. IEEE Trans. Knowl. Data Eng. 195–200.

Hoffman, J., Tzeng, E., Park, T., Zhu, J., Isola, P., Saenko, K., Efros, A.A., Darrell, T., 2018. CyCADA: Cycle-consistent adversarial domain adaptation. In: ICML. pp. 1994–2003.

Hoffman, J., Wang, D., Yu, F., Darrell, T., 2016. FCNs in the wild: Pixel-level adversarial and constraint-based adaptation. CoRR, arXiv:1612.02649.

Huang, X., Belongie, S., 2017. Arbitrary style transfer in real-time with adaptive instance normalization. In: ICCV. pp. 1510–1519.

Li, Y., Yuan, L., Vasconcelos, N., 2019. Bidirectional learning for domain adaptation of semantic segmentation. CVPR 6929–6938.

Lu, W., Wang, J., Li, H., Chen, Y., Xie, X., 2022. Domain-invariant feature exploration for domain generalization. CoRR, arXiv:2207.12020.

Luo, Y., Zheng, L., Guan, T., Yu, J., Yang, Y., 2018. Taking a closer look at domain shift: Category-level adversaries for semantics consistent domain adaptation. In: CVPR. pp. 2507–2516.

Murez, Z., Kolouri, S., Kriegman, D., Ramamoorthi, R., Kim, K., 2018. Image to image translation for domain adaptation. In: CVPR. pp. 4500–4509.

Neven, R., Neven, D., Brabandere, B.D., Proesmans, M., Goedemé, T., 2021. Weakly-supervised semantic segmentation by learning label uncertainty. In: IEEE/CVF International Conference on Computer Vision Workshops, ICCVW 2021, Montreal, BC, Canada, October 11-17, 2021. IEEE, pp. 1678–1686.

Richter, S.R., Vineet, V., Roth, S., Koltun, V., 2016. Playing for data: Ground truth from computer games. In: ECCV. pp. 102–118.

Ros, G., Sellart, L., Materzynska, J., Vázquez, D., López, A.M., 2016. The SYNTHIA dataset: A large collection of synthetic images for semantic segmentation of urban scenes. In: CVPR. pp. 3234–3243.

Ruder, S., Plank, B., 2018. Strong baselines for neural semi-supervised learning under domain shift. In: ACL. pp. 1044–1054.

Shannon, C.E., 1948. A mathematical theory of communication. Bell Syst. Tech. J. 379–423.

Tommasi, T., Patricia, N., Caputo, B., Tuytelaars, T., 2017. A deeper look at dataset bias. In: Csurka, G. (Ed.), Domain Adaptation in Computer Vision Applications. In: Advances in Computer Vision and Pattern Recognition, pp. 37–55.

Tsai, Y., Hung, W., Schulter, S., Sohn, K., Yang, M., Chandraker, M., 2018. Learning to adapt structured output space for semantic segmentation. In: CVPR. pp. 7472–7481.

Tsai, Y., Sohn, K., Schulter, S., Chandraker, M., 2019. Domain adaptation for structured output via discriminative patch representations. In: ICCV. pp. 1456–1465.

Vu, T., Jain, H., Bucher, M., Cord, M., Pérez, P., 2019. ADVENT: Adversarial entropy minimization for domain adaptation in semantic segmentation. In: CVPR. pp. 2517–2526.

Wang, Y., Wang, H., Shen, Y., Fei, J., Li, W., Jin, G., Wu, L., Zhao, R., Le, X., 2022. Semi-supervised semantic segmentation using unreliable pseudo-labels. In: IEEE/CVF Conference on Computer Vision and Pattern Recognition, CVPR 2022, New Orleans, la, USA, June 18-24, 2022. IEEE, pp. 4238–4247.

Wolpert, D., 1992. Stacked generalization. Neural Netw. 5, 241–259.

Wu, Z., Han, X., Lin, Y., Uzunbas, M.G., Goldstein, T., Lim, S., Davis, L.S., 2018a. DCAN: dual channel-wise alignment networks for unsupervised scene adaptation. In: ECCV. pp. 135–153.

Wu, H., Sun, Z., Yuan, W., 2018b. Direction-aware neural style transfer. In: ACM Multimedia. pp. 1163–1171.

Yang, J., An, W., Wang, S., Zhu, X., Yan, C., Huang, J., 2020a. Label-driven reconstruction for domain adaptation in semantic segmentation. In: ECCV. pp. 480–498.

Yang, Y., Lao, D., Sundaramoorthi, G., Soatto, S., 2020b. Phase consistent ecological domain adaptation. In: CVPR. pp. 9008–9017.

Yang, Y., Soatto, S., 2020. FDA: Fourier domain adaptation for semantic segmentation. In: CVPR. pp. 4084–4094.

Zendel, O., Honauer, K., Murschitz, M., Steininger, D., Domínguez, G.F., 2018. WildDash - creating hazard-aware benchmarks. In: ECCV. pp. 407–421.

Zhang, J., Chen, L., Kuo, C.J., 2018. A fully convolutional tri-branch network (FCTN) for domain adaptation. In: ICASSP. pp. 3001–3005.

Zhou, Z., Li, M., 2005. Tri-training: exploiting unlabeled data using three classifiers. ITKDE 1529–1541.

Zhu, J., Park, T., Isola, P., Efros, A.A., 2017. Unpaired image-to-image translation using cycle-consistent adversarial networks. In: ICCV. pp. 2242–2251.

Zou, Y., Yu, Z., Kumar, B.V.K.V., Wang, J., 2018. Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In: ECCV. pp. 297–313.