Learnable Data Augmentation for One-Shot Unsupervised Domain Adaptation

Julio Ivan Davila Carrazco[1,3]
julio.davila@iit.it
Pietro Morerio[1]
pietro.morerio@iit.it
Alessio Del Bue[1]
alessio.delbue@iit.it
Vittorio Murino[1,2,4]
vittorio.murino@iit.it

[1] Pattern Analysis and Computer Vision (PAVIS)
Italian Institute of Technology
Genoa, Italy

[2] Department of Computer Science and Technology, Bioengineering, Robotics and Systems Engineering
University of Genova
Genoa, Italy

[3] Department of Marine, Electrical, Electronic and Telecommunications Engineering Robotics and Systems Engineering
University of Genova
Genoa, Italy

[4] Department of Computer Science
University of Verona
Verona, Italy

Abstract

This paper presents a classification framework based on learnable data augmentation to tackle the One-Shot Unsupervised Domain Adaptation (OS-UDA) problem. OS-UDA is the most challenging setting in Domain Adaptation, as only one single unlabeled target sample is assumed to be available for model adaptation. Driven by such single sample, our method LearnAug-UDA learns how to augment source data, making it perceptually similar to the target. As a result, a classifier trained on such augmented data will generalize well for the target domain. To achieve this, we designed an encoder-decoder architecture that exploits a perceptual loss and style transfer strategies to augment the source data. Our method achieves state-of-the-art performance on two well-known Domain Adaptation benchmarks, DomainNet and VisDA. The project code is available at https://github.com/IIT-PAVIS/LearnAug-UDA

1 Introduction

Although deep learning empowers us to tackle intricate tasks with exceptional performance, it comes with an inherent trade-off—a voracious appetite for copious amounts of data for effective model training. To overcome this, new areas of research were born, specifically designed to cope with either the scarcity of labeled data, the availability of unlabeled data only, or situations where only few samples can be exploited. In these settings, Domain Adaptation focuses on the task of transferring knowledge from a (usually) richly annotated Source domain to a Target domain, where data is assumed to be scarce or labelling is absent, in any case insufficient to train a model from scratch. When target data is completely unlabeled, we deal with the research area of Unsupervised Domain Adaptation (UDA). In UDA, the scarcity of target samples could be a further issue. Few-shot Unsupervised Domain Adaptation (FS-UDA) focuses on solving the DA problem where only few labeled source samples are available [20, 21], while one-shot Unsupervised Domain Adaptation (OS-UDA) is the case where target data is reduced to *only one unlabeled sample* [3, 12]. Last, an even more extreme setting is the one referred to as Domain Generalization [2, 9, 13], where no data whatsoever is available for the target domain(s), yet one would like the model to generalize as best as it can to unseen domains.

To the best of our knowledge, there are only two methods focusing on solving OS-UDA. Adversarial Style Mining (ASM) [12] leverages style transfer to synthesize target samples to be used for training a classifier. Target-driven One-shot Unsupervised Domain Adaptation (TOS-UDA)[3] applies a two-step strategy to learn a set of fixed transformations guided by a perceptual loss.

In this paper, we propose a new approach to tackle the OS-UDA task. We design an original method to *learn* augmentations for the source samples to resemble the (only) target data by exploiting style transfer strategies. To ensure the quality and fidelity of the learned augmentations, we employ a perceptual loss function. By doing so, we guide the learned augmentations to closely align with the target domain. In summary, in our work:.

- We present LearnAug-UDA a novel approach for tackling One-shot Unsupervised Domain Adaptation setting, which is a challenging UDA task where only one single unlabeled target sample is available.
- The proposed architecture is composed by two encoder-decoder modules to learn augmentations by enforcing a perceptual similarity and conditioning with two types of methods: Mixup[23] and Style disentanglement[7, 24].
- We achieve state-of-the-art results for OS-UDA on DomainNet and VisDA benchmarks.

The rest of the paper is organized as follows. In Section 2, we illustrate the related work, in comparison with our proposed solution. In Section 3, we detail the description of our proposed model and its component modules. In Section 4, we introduce the domain adaptation benchmarks used for testing, and we present the results obtained by our method. In Section 5, we draw the conclusions of the work, while commenting our proposed approach in relation to both the methodological choices and the experimental analysis.

2 Related work

One-Shot Unsupervised Domain Adaptation (OS-UDA): as mentioned above, OS-UDA refers to the scenario in Unsupervised Domain Adaptation where the available samples of the Target domain are reduced to only one unlabeled sample and the literature focusing on this setting is very limited. In [12], They introduced *Adversarial Style Mining* (ASM). ASM leverages style transfer to augment source samples with the style of a single target sample, by introducing a Random Adaptive Instance Normalization (RAIN) module, which in practice performs style transfer. However, RAIN is pretrained for such a task, with extra information from an external dataset (wikiArts). Therefore, while a single target sample is used to drive RAIN at inference time, extra data is actually exploited to learn the style transfer task. ASM's training pipeline is not end-to-end and the additional dataset for pre-training needs to be manually selected based on task and characteristics of source and target dataset. In [3], They introduced *Target-driven One-shot Unsupervised Domain Adaptation* (TOS-UDA) which applies adversarial training and a perceptual loss to guide an augmentation module to learn the parameters of two fixed transformations. TOS-UDA augments source samples to train a classifier, which is expected to generalize well for target data.

An alternative to using domain adaptation methods is to re-purpose methods that does not tackle the domain adaptation problem per se. *TeachAugment* [19] is presented as a method for data augmentation to improve model generalization. TeachAugment augments data samples by performing two types of fixed transformations: a geometrical and color transformations. The parameters of such transformations are then defined by an optimization-based approach and an adversarial training strategy is used to drive the process. This strategy aims to generate augmented samples that are “hard” for the model to learn from. Our approach, although drawing similarities from the aforementioned methods, obtains better results by exploiting the unused potential that the other methods omitted. First of all, we applied an encoder-decoder architecture for our augmentation module. This allows the model to learn better augmentations as it is not bound to a set of fixed transformations. Furthermore, we introduce a modified perceptual loss, able to measure the similarity in both style and content.

Style transfer: Style transfer refers to task of synthesizing an image that resembles the characteristics in style from an image and the content from a different image. In [5], a first approach to use Convolutional Neural Networks (CNNs) for style transfer was presented. This process was later referred as Neural Style Transfer (NST). NST disentangles content and style by using features representations of different layers of a VGG network [18]. In [8], they proposed a perceptual loss for training a feed-forward network for the style transfer task. The perceptual loss measures the perceptual similarity between the synthesized image, content image, and the style image. The first methods solving style transfer were only capable of transfer one specific style [5, 8]. Later, methods [4, 22] raised the expectations by learning different styles from a set of target images. Finally, the introduction of arbitrary style transfer allowed to perform style transfer without being bound to a set of style targets [7, 11, 24]. We can regard our proposed augmentation method as a style transfer module, which has to deal with the difficult task of learning style from a single target image and transferring it to the source samples.

3 Methodology

Our proposed model is constituted by three elements: an augmentation module, a style alignment module and a classifier. Each of these elements focuses on a specific task. The Augmentation Module (AUM) augments source samples to resemble the Target domain. The AUM achieves this by conditioning the image synthesis on the only target sample that is available. The Style Alignment Module (SAM) is responsible of measuring the perceptual similarities (at different levels) between Source samples, the Target sample, and the synthesized augmented samples. Similar to style transfer, AUM and SAM work together to learn the augmentations for the Source data (see Fig. 1, Step 2). In other words, SAM guides the learning process of AUM to obtain augmentations that have a similar style as the Target domain. Our intuition is that the classifier trained on these augmented samples will generalize better for the Target domain (see Fig. 1, Step 1). A two-step training strategy is adopted (see Fig. 1). In the first step, the AUM synthesizes augmented samples for the classifier to train on. In this step, only the classifier is updated via back propagation while the AUM's weights remain frozen. In the second step, AUM is trained by learning augmentations that favour a similarity with the Target domain's style. More detalis are provided in the following.

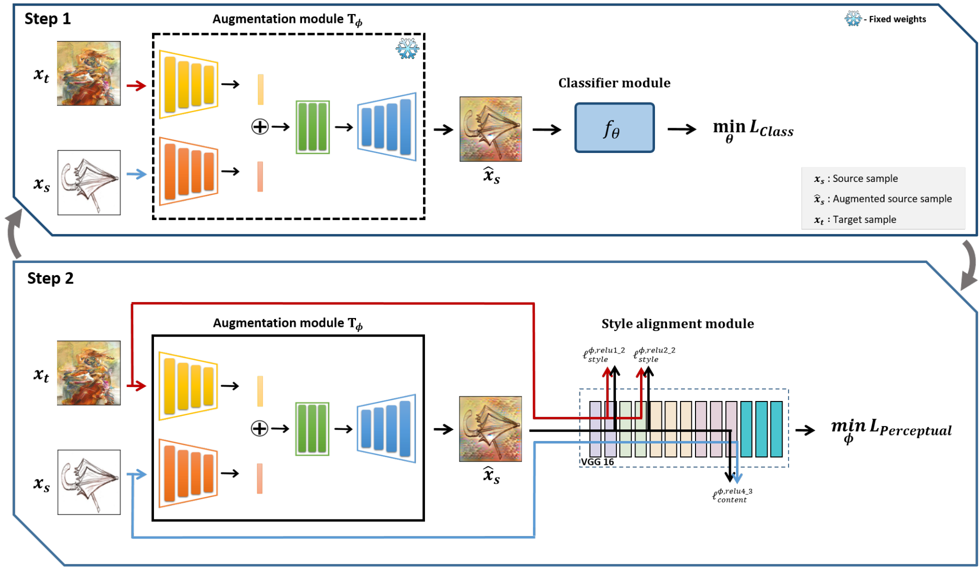

Figure 1: Our proposed approach with its three elements: Augmentation module (AUM), the Style Alignment Module (SAM) and the Classifier Module (CM). The training strategy focuses on alternating between updating the CM and AUM. In step 1, the CM is trained by minimizing a cross-entropy loss. In step 2, AUM learns to synthesize augmentations guided by SAM whom measures the perceptual similarity with the Target sample.

3.1 Augmentation module

The Augmentation Module (AUM) manages the synthesis of augmented samples used for training the Classifier module (CM). We designed two variations of the augmentation module architecture to explore different approaches for achieving style transfer in data augmentation.

The first variation, the Shared Encoder (see Fig. 2a), is a simple encoder-decoder architecture. Both style and content samples are encoded using the same encoder, with conditioning performed at the bottleneck through Mixup between the embeddings [7, 23]. Namely a convex combination of source and target sample is fed as input to the decoder, in order to synthesize the augmented sample. We designed the Share Encoder to exploit a Mixup-inspired data augmentation approach where we wanted the resulting feature maps to contain characteristics of both domains.

The second variation consists of Disentangled Encoders (see Fig. 2b). This design leverages the concept of domain separation [1], the intuition being that different domains exhibit shared characteristics and domain-specific characteristics. Similarly, image style and content can be considered separate aspects. Therefore, our disentangled encoders consist of a Style encoder and a Content encoder. In this case, the intuition is that differences in style are responsible for the performance drop of the source classifier tested on target data. Each encoder specializes in learning different aspects of the image and generates an embedding representing that particular aspect. To synthesize an augmented sample, the concatenated embeddings are passed to the Bottleneck module, to reduce dimensionality, and subsequently a decoder generates the augmented sample. In contrast to the shared encoder, we do not perform Mixup between embeddings with this architecture, as fusion is left to the bottleneck module. To improve the disentanglement between style and content, we introduce a reconstruction loss which allows the encoders to see samples of both domains. The reconstruction loss is defined as:

$$L_{rec} = \|\mathrm{T}_{\phi}(x,x) - x\|_2^2 \tag{1}$$

where $\mathrm{T}_{\phi}(\cdot)$ is the augmentation module, x is an input sample from the source or target data.

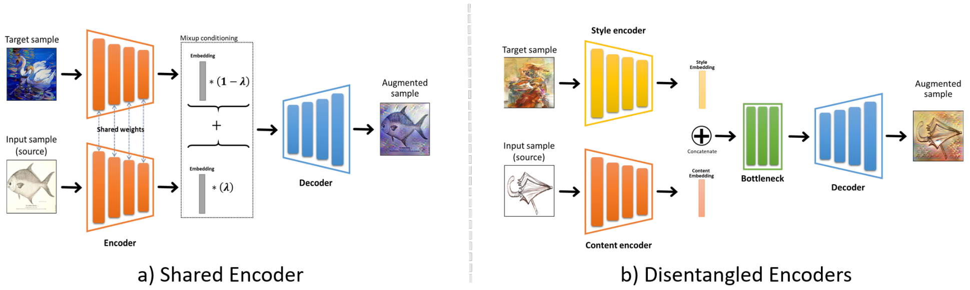

Figure 2: The Augmentation Module (AUM) in its two variations: Shared encoder (a) and Disentangled encoders (b). The Shared encoder applies the same encoding to both the source and target samples, while performing style conditioning through mixup at the bottleneck. The Disentangled encoders treat style and content as separate elements within the image, with each individual encoder learning a distinct feature representation.

3.2 Style alignment module

The Style alignment module (SAM) is in charge of enforcing a perceptual similarity in style and content between a source sample, the only target sample and the synthesized augmented sample. SAM imposes a similarity in style with respect to the target sample. By doing so, AUM learns to synthesize augmented samples for the Target domain. Furthermore, by imposing a similarity in content with respect to the input source sample, i.e. the shared characteristics between source and target, the classifier can use labels from the input source to train for the target domain. SAM consists of a frozen VGG-16 [18] from which feature maps of selected layers are extracted to evaluate the perceptual similarity [8]. The perceptual loss comprises two type of losses: a style and content loss. The style loss evaluates similarity between aspects such as color, texture, and common patterns. Gatys et al. [6] defined the representation of style as the correlation between filter responses of the feature map. These feature correlations are represented as a Gram matrix which is defined as:

$$G_j(x) = \frac{1}{C_j H_j W_j} \sum_{h=1}^{H_j} \sum_{w=1}^{W_j} H_j(x) H_j^T(x) \tag{2}$$

where x is the input image, H_j represents the feature map from the j-layer of the SAM; C_j, H_j, and W_j denote the number of channels, the height, and the width of H_j, respectively. The Gram matrix is calculated for the target and the augmented samples. The style loss of a single layer is calculated as:

$$\ell_i^{style}(\hat{x}_s, x_t) = \|G_i(\hat{x}_s) - G_i(x_t)\|_F^2 \text{ , where } \hat{x}_s = \mathrm{T}_\phi(x_s, x_t) \tag{3}$$

where $\mathrm{T}_\phi(\cdot)$ is the Augmentation module, $\hat{x}_s$ is then the augmented sample, x_s represents a source sample, x_t indicates the target sample, $\|\cdot\|_F^2$ is the Frobenius norm, and i represents the selected layers of SAM. For the content loss which focuses on preserving the semantic content, i.e. the actual objects present in the scene, we define it as:

$$\ell_j^{content}(\hat{x}_s, x_s) = \frac{1}{C_j H_j W_j} \|H_j(\hat{x}_s) - H_j(x_s)\|_2^2 \tag{4}$$

where H_j represents the feature map from the j-layer of the SAM; C_j, H_j, and W_j denote the number of channels, the height, and the width of H_j, respectively. As established by [6, 8], multiple layers could extract the information regarding the style and content. So, it stands to reason to add up the individual loss of different layers to better capture the style and content information. Therefore, these cumulative losses are defined as:

$$L_{style} = \sum_i \ell_i^{style}(\hat{x}_s, x_t), \qquad L_{content} = \sum_j \ell_j^{content}(\hat{x}_s, x_s) \tag{5}$$

where i and j represent the selected layers of SAM for evaluating style and content, respectively. Finally, The perceptual loss is calculated as:

$$L_{perceptual\_G} = L_{style} + L_{content} \tag{6}$$

The perceptual loss, as defined in the previous equations, has limitations when performing style transfer with only one target sample. Naively enforcing a perceptual similarity against the single target sample leads to locally injecting its feature's patterns always in the same position. The synthesized samples tend to exhibit characteristics related to the target sample's content, such as shapes and contours. To address this issue, we propose a new variation of the perceptual loss that incorporates an average pooling operation over the feature maps. By using a spatial average pooling operation, we instead enforce a global perceptual similarity of the feature maps. This operation introduces a smoothing effect on the reduced feature map, leading to a smoother synthesized sample. We apply this new variation to both the style and content losses, defining the new loss as follows:

$$\ell_i(\hat{x}_s, x_c) = \frac{1}{CHW} \|AvP(H_i(\hat{x}_s)) - AvP(H_i(x_c))\|_2^2 \tag{7}$$

where $\hat{x}_s$ represents the augmented sample, x_c represents the sample to compare against with, i.e. the target or input source sample, H_i represents the feature map from the i-layer of the SAM, AvP specifies the average pooling operation, C, H, and W denote the number of channels, the height, and the width of reduced feature map after the average pooling operation, respectively. Finally, the new perceptual loss is redefined as:

$$L_{perceptual\_AvP} = \sum_i \ell_i(\hat{x}_s, x_s) + \sum_j \ell_j(\hat{x}_s, x_t) \tag{8}$$

where x_s represents input source sample, x_t represents the only target sample, i and j represent the selected layers of SAM for evaluating style and content, respectively.

3.3 Classifier module

The Classifier module (CM) is trained on the augmented samples and the labels of their corresponding input source sample. The intuition behind training the classifier with augmented samples is that by doing so it should generalize well for the target domain. The classifier is trained in supervised manner by minimizing a traditional cross entropy loss:

$$L_{CE} = -\sum_{x_s \in X} y_k \log f_\theta(\mathrm{T}_\phi(x_s, x_t)) \tag{9}$$

where X is the source sample set, $y \in \{0,1\}^K$ denotes the one-hot ground-truth vector, K is the number of classes, $\mathrm{T}_\phi(\cdot)$ denotes the AUM with parameters ϕ, and $f_\theta(\cdot)$ represents the classifier with parameters θ.

3.4 Training strategy

A two-step training strategy alternates the learning of the CM and the AUM to enhance the performance and generalization of the model (see Fig. 1). This strategy is applied to stabilize the learning process. This process will allow the individual modules to update their parameters without causing drastic changes in the other modules. The division of task leads to a more effective and specialized learning in each module.

In the first step, the classifier is trained on the augmented samples to learn the underlying patterns and features. The CM is trained by minimizing Eq. 9. During step 1, the weights of AUM are frozen, allowing the backpropagation process to update only the Classifier Module (CM).

In the second step, AUM is trained, guided by SAM, to learn augmentations that resemble the Target domain. AUM can be trained using either variation of the perceptual loss (Eq. 6 or Eq. 8). When the Disentangled encoders are used, the reconstruction loss (eq. 1) can be used together with the perceptual loss to improve AUG's performance. The learning process of AUM can be controlled by conducting several iterations solely focused on step 1.

4 Experiments

4.1 Datasets

To assess the performance of our proposed method, we evaluate it on two well-known benchmarks for domain adaptation: **DomainNet** [15] and **VisDA** [14]. DomainNet consists of four distinct domains: Real, Painting, Clipart, and Sketch. Following the evaluation procedure established in [17], we conduct our assessment across seven domain adaptation tasks: Real to Clipart (**R** → **C**), Real to Painting (**R** → **P**), Real to Sketch (**R** → **S**), Painting to Clipart

Table 1: Classification accuracies of our proposed method for DomainNet on seven DA tasks. For Few-shot, three target samples are used. (SE) refers to the Shared encoder, while (DE) is the Disentangled encoders. (RL) specifies a model trained with the reconstruction loss.

Method	Type	R → C	R → P	R → S	P → C	P → R	C → S	S → P	Average
Source only	-	56.59 ± 0.79	56.79 ± 0.50	46.25 ± 0.86	55.55 ± 0.83	66.20 ± 0.72	52.07 ± 1.01	44.81 ± 1.59	54.04
TeachAugment[19]	-	53.84 ± 0.56	56.70 ± 0.59	46.70 ± 1.34	50.40 ± 1.27	58.64 ± 0.68	50.52 ± 0.09	44.89 ± 0.83	51.67
ASM[12]	One-shot	39.74 ± 0.56	46.39 ± 1.53	31.37 ± 5.51	4.31 ± 0.60	5.87 ± 2.33	37.12 ± 1.12	19.67 ± 2.99	26.35
TOS-UDA [ ]	One-shot	**58.11** ± 0.38	58.57 ± 0.20	**49.87** ± 0.97	54.24 ± 0.62	62.72 ± 0.32	52.88 ± 0.25	47.94 ± 1.12	54.90
Our model (SE)	One-shot	49.89 ± 4.96	57.52 ± 7.05	39.07 ± 16.88	51.55 ± 3.27	58.08 ± 6.41	37.47 ± 19.76	42.09 ± 16.43	47.95
Our model (DE+RL)	One-shot	56.74 ± 0.62	61.02 ± 1.31	47.03 ± 3.41	54.24 ± 2.66	69.06 ± 0.87	53.42 ± 0.76	52.95 ± 1.77	56.35
Our model (SE)	Few-shot (3)	57.06 ± 0.54	61.95 ± 0.57	49.18 ± 0.19	52.52 ± 3.01	66.79 ± 1.91	51.69 ± 1.99	50.08 ± 3.25	55.61
Our model (DE+RL)	Few-shot (3)	57.96 ± 0.81	**62.43** ± 0.88	47.95 ± 1.09	**56.70** ± 0.80	**69.59** ± 0.41	**55.37** ± 0.89	**54.58** ± 0.97	**57.80**

(**P** → **C**), Painting to Real (**P** → **R**), Clipart to Sketch (**C** → **S**), and Sketch to Painting (**S** → **P**). The VisDA dataset, on the other hand, comprises two domains: Synthetic and Real. The data is divided into three splits: training, validation, and testing. The training split represents the synthetic domain, while both the validation and testing splits consist of real domain samples sourced from different image datasets (COCO [10] and the Youtube Bounding Box dataset [16]). For our experiments, we treat the validation split as the training set for the Real domain, as it is from this set that we obtain our target samples.

4.2 Setup

The pipeline training setup is as follows: the classifier CM is a ResNet-101 pretrained on ImageNet. AUM was tested in its two variations: Shared encoder (SE) and Disentangled encoders (DE). SAM, is a VGG-16 [18] pretrained on ImageNet. In the VGG-16, the features maps from the layers $relu1\_2$ and $relu2\_2$ were used for the style loss while the features map from layer $relu4\_3$ was used for the content loss. AUM's layers descriptions will be presented on the supplementary material. All the experiments were run five times. We present mean and std for DomainNet and only the mean for VisDA. VisDA results' standard deviations will be presented on the supplementary.

4.3 Implementation details

To train LearnAug-UDA, we employed two optimizers. Each of them focusing on individually updating the CM and AUG modules. The CM is trained using stochastic gradient descend (SDG) while AUG is trained using the AdamW optimizer. The perceptual loss' weights for layers $relu1\_2$, $relu2\_2$, and $relu4\_3$ were set as 0.25,1.0, and 1.0 respectively. The model was trained for 20 epochs with a learning rate of $1e-4$ for the CM optimizer and $1e-3$ for the AUG optimizer, batch size of 8 samples with input size of $224x224$. When training with SE and Mixup, the parameters for the beta distribution were set as 5.0 for the alpha value and 1.0 for the beta value. Futher information can be found on the supplementary material and the code repository of LearnAug-UDA: https://github.com/IIT-PAVIS/LearnAug-UDA.

4.4 Results

In table 1, the results for DomainNet are presented. we evaluate the accuracy of our method against four different competitors. **Source only**: a classifier trained only on source data using a cross entropy loss. **TeachAugment**[19]: as explained in section 2, this is a method for model generalization that learns fixed augmentations. **ASM**[12]. One of the few methods

Table 2: Classification accuracy of our proposed method on VisDA. For Few-shot, three target samples are used. (SE) refers to the Shared encoder, while (DE) is the Disentangled encoders. (RL) specifies a model trained with the reconstruction loss.

Method	Type	Aeroplane	Bicycle	Bus	Car	Horse	Knife	Motorcycle	Person	Plant	Skateboard	Train	Truck	Mean
Source only	-	68.86	3.24	46.05	**97.61**	30.48	8.08	50.69	5.90	72.14	16.97	62.21	14.84	39.76
TeachAugm [illegible]	-	26.47	0.35	39.49	40.38	1.28	1.21	31.76	0.40	39.36	9.67	55.69	10.45	21.38
ASM[illegible]	One-shot	62.49	25.17	**81.61**	77.23	47.72	11.84	39.51	5.68	**83.93**	30.07	48.77	**31.49**	45.46
TOS-UDA[3]	One-shot	15.05	0.01	13.96	17.31	2.47	20.38	0.53	1.34	11.46	7.15	20.51	5.32	9.63
Ours (DE+RL)	One-shot	59.90	12.77	71.99	91.46	48.44	**23.70**	**59.88**	11.56	76.38	**40.22**	63.19	24.26	48.64
ASM[illegible]	Few-shot (3)	**70.27**	**35.57**	78.62	83.36	43.95	13.88	36.39	4.04	83.14	32.06	37.65	30.31	45.77
TOS-UDA[3]	Few-shot (3)	21.92	1.02	19.66	11.56	7.32	7.60	5.15	2.24	11.67	11.29	17.90	5.74	10.26
Ours (DE+RL)	Few-shot (3)	62.21	10.68	68.38	90.93	**53.88**	22.99	58.91	**12.96**	70.14	39.49	**67.24**	27.95	**48.79**

that tackles OS-UDA setting; it leverages style transfer to generate target samples. **TOS-UDA**[3]: it implements a perceptual loss to guide the learning process to learn fixed transformations. Together with the aforementioned methods, we evaluate two variations of our model: the one using the shared encoder and the other with the disentangled encoders. The results shows that our method the disentangled encoders performs better than any of the other evaluated methods. Instead, our model with shared encoder obtained a lower accuracy than TOS-UDA. This could imply that one-shot is not enough for the shared encoder to transfer the style to the augmented sample. To expand our experiments, we also trained our method in the Few-shot setting (3 Target samples) to evaluate its accuracy if more samples were available. In this setting, our model with shared decoder increase its accuracy which proves that the shared encoder works only if more target samples are available. Our model with the disentangled encoders has an increase of (2.0%) on its performance. This indicates that our method could possibly be used for other DA scenarios where there are more available target samples, without the need to be modified.

In table 2, the results for VisDA are presented. we evaluated our proposed approach against the same methods that were used for DomainNet. To the best of our knowledge, it is the first time that VisDA is used as a benchmark for the OS-UDA setting. We generate the results presented for ASM and TOS-UDA following their respective training strategies. As before, we present the results using differnt number of target samples. For VisDA, we only tested the best performing configuration of our method, i.e. the disentangled encoders and reconstruction loss are used. Based on the results, we conclude that methods which use fixed transformations are not capable of closing the gap between synthetic and real domains. The aforementioned methods obtained lower accuracy than the source only method. Furthermore, our method with its disentangled encoders is able to obtained the highest mean accuracy among all evaluated methods. When using three target samples, our method obtains a high accuracy similar to the one-shot version. This could indicate that unlike the DomainNet experiments, the model would need to be modified for it to reach higher accuracy.

4.5 Ablations

We conducted additional experiments to analyze various configurations of our model. We compared our modified version of the perceptual loss, which incorporates an average pooling (AvgP) operation, with the original approach presented in [8] that utilizes a Gram matrix (eq. 2). We tested two different architectures for AUM: a shared encoder with conditioning mixup (SE) and the disentangled encoders (DE). Additionally, for the second AUM architecture, we also examined its variation with the inclusion of a reconstruction loss (RL). In Table 3, we present the results obtained from testing such ablations. **AUM architectures**. When comparing both architectures without any other variations, i.e. SE and DE, there is not

Table 3: Comparative results for different ablations of our proposed approach are presented in the table. (SE) refers to Shared encoder, while (DE) represents the Disentangled encoders. (AvgP) indicates the use of average pooling by the Style Alignment module, and (RL) specifies a model trained with the reconstruction loss.

Model	R → C	R → P	R → S	P → C	P → R	C → S	S → P	Average
SE	56.53 ± 1.44	62.89 ± 1.03	**50.33** ± 2.27	55.02 ± 1.96	64.32 ± 4.42	53.62 ± 0.73	50.33 ± 2.27	56.73
SE+AvP	54.16 ± 3.25	62.83 ± 0.40	48.64 ± 1.79	48.94 ± 5.24	66.21 ± 0.66	50.04 ± 4.69	52.91 ± 0.69	54.82
DE	56.12 ± 1.97	**63.55** ± 0.61	49.87 ± 2.49	54.38 ± 2.00	66.53 ± 1.70	51.70 ± 1.41	54.34 ± 2.01	56.64
DE+AvP	56.79 ± 0.84	62.50 ± 0.54	48.71 ± 0.66	56.18 ± 1.37	69.11 ± 0.61	54.50 ± 1.70	**55.12** ± 0.61	57.56
DE+AvP+RL	**57.96** ± 0.81	62.43 ± 0.88	47.95 ± 1.09	**56.70** ± 0.80	**69.59** ± 0.41	**55.37** ± 0.89	54.58 ± 0.97	**57.80**

a considerable difference in the average performance between them as the 0.09% difference can be neglected. The same can be said about their evaluations on individual tasks as they differ only about 1.0% in most of the task (6 out of 7). **Average pooling**. The use of average pooling by SAM shows opposite results based on the AUM architecture tested. we believe that the disentangled encoders is able to transfer more style information of the target sample than the shared encoder. therefore after applying average pooling, there is still enough information to measure the perceptual similarity for the style. **Reconstruction Loss**. Applying the reconstruction loss when training the AUM (DE) shows a slight increase in the average accuracy compare to the method which uses only the perceptual loss for training. However, this configuration of the model achieves the best results not only in average accuracy among all configurations but also obtains the best performances in four out of seven tasks.

5 Conclusions

In this paper, we introduced our novel approach called Learnable Data Augmentation for One-Shot Unsupervised Domain Adaptation (LearnAug-UDA). Our method addresses the difficult domain adaptation scenario known as OS-UDA, where the availability of only a single unlabeled target sample poses a challenge. To overcome this limitation, we developed an Augmentation Module that utilizes style transfer techniques to synthesize augmented samples. To guide the learning process of the AUM, we incorporated a Style Alignment Module, which enforces a perceptual similarity between the augmented samples and the target sample. We improved the perceptual loss by introducing an average pooling operation to smooth out the synthesized samples. Through these key components, our approach enables effective adaptation to the target domain. We tested our approach in the image classification task for two well-known Domain Adaptation benchmarks, DomainNet and Visda. We demonstrated the superior performance of our method compared to the selected baselines across a majority of the domain adaptation tasks.

References

[1] Konstantinos Bousmalis, George Trigeorgis, Nathan Silberman, Dilip Krishnan, and Dumitru Erhan. Domain separation networks. *Advances in neural information processing systems*, 29, 2016.

[2] Fabio M. Carlucci, Antonio D'Innocente, Silvia Bucci, Barbara Caputo, and Tatiana Tommasi. Domain generalization by solving jigsaw puzzles. In *Proceedings of the*

IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR), June 2019.

[3] Julio Ivan Davila Carrazco, Suvarna Kishorkumar Kadam, Pietro Morerio, Alessio Del Bue, and Vittorio Murino. Target-driven one-shot unsupervised domain adaptation. In *International Conference on Image Analysis and Processing*, pages 87–99. Springer, 2023.

[4] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. *arXiv preprint arXiv:1610.07629*, 2016.

[5] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. A neural algorithm of artistic style. *arXiv preprint arXiv:1508.06576*, 2015.

[6] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 2414–2423, 2016.

[7] Xun Huang and Serge Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *Proceedings of the IEEE international conference on computer vision*, pages 1501–1510, 2017.

[8] Justin Johnson, Alexandre Alahi, and Li Fei-Fei. Perceptual losses for real-time style transfer and super-resolution. In *Computer Vision–ECCV 2016: 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part II 14*, pages 694–711. Springer, 2016.

[9] Haoliang Li, Sinno Jialin Pan, Shiqi Wang, and Alex C Kot. Domain generalization with adversarial feature learning. In *Proceedings of the IEEE conference on computer vision and pattern recognition*, pages 5400–5409, 2018.

[10] Tsung-Yi Lin, Michael Maire, Serge Belongie, James Hays, Pietro Perona, Deva Ramanan, Piotr Dollár, and C Lawrence Zitnick. Microsoft coco: Common objects in context. In *Computer Vision–ECCV 2014: 13th European Conference, Zurich, Switzerland, September 6-12, 2014, Proceedings, Part V 13*, pages 740–755. Springer, 2014.

[11] Songhua Liu, Tianwei Lin, Dongliang He, Fu Li, Meiling Wang, Xin Li, Zhengxing Sun, Qian Li, and Errui Ding. Adaattn: Revisit attention mechanism in arbitrary neural style transfer. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 6649–6658, October 2021.

[12] Yawei Luo, Ping Liu, Tao Guan, Junqing Yu, and Yi Yang. Adversarial style mining for one-shot unsupervised domain adaptation. *Advances in neural information processing systems*, 33:20612–20623, 2020.

[13] Krikamol Muandet, David Balduzzi, and Bernhard Schölkopf. Domain generalization via invariant feature representation. In *International conference on machine learning*, pages 10–18. PMLR, 2013.

[14] Xingchao Peng, Ben Usman, Neela Kaushik, Judy Hoffman, Dequan Wang, and Kate Saenko. Visda: The visual domain adaptation challenge. *arXiv preprint arXiv:1710.06924*, 2017.

[15] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proceedings of the IEEE/CVF international conference on computer vision*, pages 1406–1415, 2019.

[16] Esteban Real, Jonathon Shlens, Stefano Mazzocchi, Xin Pan, and Vincent Vanhoucke. Youtube-boundingboxes: A large high-precision human-annotated data set for object detection in video. In *proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 5296–5305, 2017.

[17] Kuniaki Saito, Donghyun Kim, Stan Sclaroff, Trevor Darrell, and Kate Saenko. Semi-supervised domain adaptation via minimax entropy. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[18] Karen Simonyan and Andrew Zisserman. Very deep convolutional networks for large-scale image recognition. *arXiv preprint arXiv:1409.1556*, 2014.

[19] Teppei Suzuki. Teachaugment: Data augmentation optimization using teacher knowledge. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 10904–10914, June 2022.

[20] Wanqi Yang, Chengmei Yang, Shengqi Huang, Lei Wang, and Ming Yang. Few-shot unsupervised domain adaptation via meta learning. In *2022 IEEE International Conference on Multimedia and Expo (ICME)*, pages 1–6. IEEE, 2022.

[21] Xiangyu Yue, Zangwei Zheng, Shanghang Zhang, Yang Gao, Trevor Darrell, Kurt Keutzer, and Alberto Sangiovanni Vincentelli. Prototypical cross-domain self-supervised learning for few-shot unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 13834–13844, 2021.

[22] Hang Zhang and Kristin Dana. Multi-style generative network for real-time transfer. In *Proceedings of the European Conference on Computer Vision (ECCV) Workshops*, September 2018.

[23] Hongyi Zhang, Moustapha Cisse, Yann N Dauphin, and David Lopez-Paz. mixup: Beyond empirical risk minimization. *arXiv preprint arXiv:1710.09412*, 2017.

[24] Yexun Zhang, Ya Zhang, and Wenbin Cai. Separating style and content for generalized style transfer. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, June 2018.