Learning to Adapt Neural Networks Across Visual Domains

UNIVERSITY OF TRENTO - Italy

Department of Information Engineering and Computer Science

Subhankar Roy

IECS Doctoral School,
Department of Information Engineering and Computer Science

The University of Trento

Advisors: Prof. Dr. Elisa Ricci and Prof. Dr. Nicu Sebe
A thesis submitted for the degree of

Doctor of Philosophy (Ph.D.)

September 29, 2022

Abstract

In the field of machine learning (ML) a very commonly encountered problem is the lack of generalizability of learnt classification functions when subjected to new samples that are not representative of the training distribution. The discrepancy between the training (*a.k.a.* source) and test (*a.k.a.* target) distributions are caused by several latent factors such as change in appearance, illumination, viewpoints and so on, which is also popularly known as *domain-shift*. In order to make a classifier cope with such domain-shifts, a sub-field in machine learning called *domain adaptation* (DA) has emerged that jointly uses the annotated data from the source domain together with the unlabelled data from the target domain of interest. For a classifier to be adapted to an unlabelled target data set is of tremendous practical significance because it has no associated labelling cost and allows for more accurate predictions in the environment of interest. A majority of the DA methods which address the single source and single target domain scenario are not easily extendable to many practical DA scenarios. As there has been as increasing focus to make ML models deployable, it calls for devising improved methods that can handle inherently complex practical DA scenarios in the real world.

In this work we build towards this goal of addressing more practical DA settings and help realize novel methods for more real world applications: *(i)* We begin our work with analyzing and addressing the single source and single target setting by proposing whitening-based embedded normalization layers to align the marginal feature distributions between two domains. To better utilize the unlabelled target data we propose an unsupervised regularization loss that encourages both confident and consistent predictions. *(ii)* Next, we build on top of the proposed normalization layers and use them in a generative framework to address multi-source DA by posing it as an image translation problem. This proposed framework TriGAN allows a single generator to be learned by using all the source domain data into a single network, leading to better generation of target-like source data. *(iii)* We address multi-target DA by learning a single classifier for all of the target domains. Our proposed framework exploits feature aggregation with a graph convolutional network to align feature representations of similar samples across domains. Moreover, to counteract the noisy pseudo-labels we propose to use a co-teaching strategy with a dual classifier head. To enable smoother adaptation, we propose a domain curriculum learning ,when the domain labels are available, that adapts to one target domain at a time, with increasing domain gap. *(iv)* Finally, we address the challenging source-free DA where the only source of supervision is a source-trained model. We propose to use Laplace

Approximation to build a probabilistic source model that can quantify the uncertainty in the source model predictions on the target data. The uncertainty is then used as importance weights during the target adaptation process, down-weighting target data that do not lie in the source manifold.

Contents

1

Introduction

1.1 Motivation

Building intelligent systems that have the ability to solve tasks by recognizing patterns and making predictions has been the longstanding goal of the machine learning (ML) community. Although the pinnacle of general intelligence has not been achieved so far, the ML community has made a steadfast progress in many challenging tasks, with the learning algorithms sometimes attaining super-human performance [56]. This has led to the increased deployment of ML algorithms in self-driving cars [50], computer-aided diagnostics [84], e-commerce websites [146], social media content moderation [3] and so on.

In essence the goal of a typical learning algorithm (or system) is to predict a desired output given some input. This is achieved through a process called *training*. In the supervised case the training process would require a set of inputs and their corresponding outputs. For instance, in the simplest case of image classification the input will be an image and the output consists of one of the several categorical labels. Then the goal of the training process is to approximate a function (commonly modelled with neural networks) that maps from the input to the output space, by minimizing a cost function on a given finite data set. To evaluate the goodness of fit, the learned function is often tested on unseen data points under the independent and identically distributed (or *i.i.d.*) assumption [55]. If the *i.i.d.* assumption is not violated, or in other words if the test points are similar to the training points, the model will make accurate predictions in future.

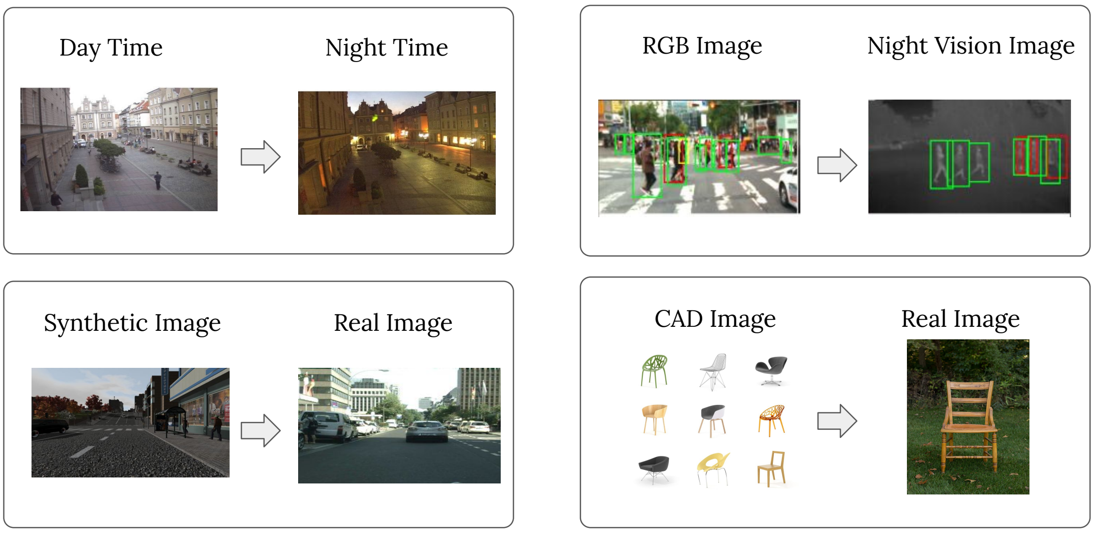


Figure 1.1: Illustrations of various kinds of domain-shift in the real world. In each of the four boxes, the image on the left depicts an example of training distribution images and the image on the right depicts the testing distribution images. For instance, the domain shift between CAD images and images taken in the real world is quite high. Due to difference in the data distributions the neural networks will exhibit a generalization gap when deployed out-of-the-box.

However, in the real world the test data is quite often not very representative of the training data. As a consequence the learned function will fail to *generalize* well on such distribution. The shift in the distributions between the training and test data is caused by a phenomenon called *domain-shift* [157] (see Fig. 1.1). For example, training data collected in a particular urban setting for an autonomous driving application might be biased with respect to other urban or rural settings due to different layout of roads, compounded with different weather conditions. A learning system trained on that particular urban setting will be unreliable and error prone when deployed in a different road or weather setting. While a naive solution would be collecting data from every possible city and weather configuration and then train specific systems where the system would be deployed, it would be unfeasible due to expensive and laborious annotated data collection process. Instead, the researchers have tried to answer if the information from the annotated urban training data set could be used to adapt the learning system to work well in rural settings?

To answer the above question a broad field of study called *domain adaptation* (DA) [4] has emerged that attempts to bridge the domain gap between the training and test distributions. In the literature the labelled data is referred to belong to the *source* domain and the unlabelled data to the *target* domain, where the learning system will potentially be deployed. In most of the DA works it is assumed that plentiful of labelled data is available from the source domain(s) and little or no labelled data from the target domain of interest. The main goal in DA is to mitigate the domain gap between the source and target domains such that the classifier trained on the source data set generalizes well to the target data set. To this end plethora of unsupervised domain adaptation (UDA) methods have been proposed that include optimizing statistical moments [13, 14, 15, 98, 124, 152, 161], adversarial training [38, 96, 159], generative modelling [61, 94, 136], to name a few.

In the recent times many real world DA settings have been proposed that depart from the traditional UDA setting, which involves a single labelled source domain and a single unlabelled target domain. In more detail, as the real world is more complex, the assumption of a single source and target domain no longer holds true. In some cases, the labelled source data set may become unavailable due to privacy or storage concerns and the traditional UDA approaches fail to operate under such conditions. Thus, to address such non-conventional DA problems new research avenues have opened up that comprise of multi-source DA [121], multi-target DA [20], source-free DA [90] and so on. This hints at the fact that to build real-world ready and robust learning systems more challenging and realistic DA scenarios need to be addressed and rightfully deserves thorough investigation.

1.2 Outline

A very natural starting point for investigating DA research topic is by studying and analyzing the closed-set single source and single target UDA (STDA), as it is the simplest UDA setting. In Chapter 2 "Domain Whitening Transform" we address STDA by means of marginal feature alignment between domains with the help of embedded domain adaptive blocks that are based on the Whitening transform. In Chapter 3 titled "TriGAN for Multi-source Domain Adaptation" we re-purpose the adaptation components of the previous chapter to address multi-source domain adaptation (MSDA) through a generative framework. In Chapter 4 "Curriculum Graph Co-teaching for Multi-target Domain Adaptation" we address a more challenging UDA setting when a single model need to learned for multiple unlabelled target domains. To this end we propose to align the feature representations across domains with a graph neural network (GNN). Finally, in Chapter 5 we address a significantly harder UDA problem when a pre-trained source model

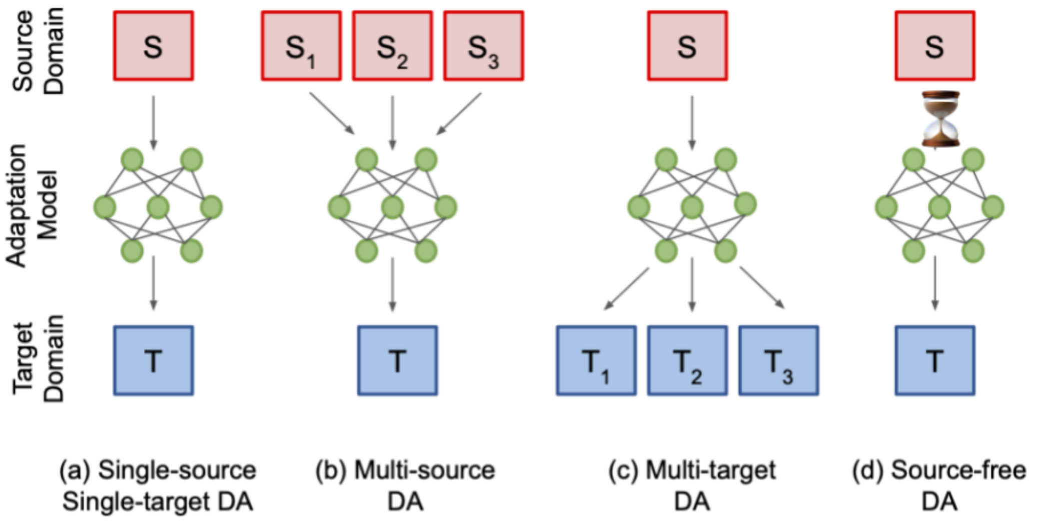


Figure 1.2: An overview of the different DA scenarios that have been addressed in this dissertation. (a) in single-source single-target DA the goal is to adapt from a single labelled source data set to a single unlabelled target data set; (b) in multi-source DA the task is to leverage the knowledge from several source domains in order to adapt to an unlabelled target domain; (c) in multi-target DA the goal is to adapt a single model than can work well in various target domains; and (d) represents source-free DA where the only source of supervision is a pre-trained model and the source data set is discarded during target adaptation.

need to be adapted on a target data set with the source data set becoming absent. To address source-free DA we highlight the importance of quantifying uncertainty and how to incorporate such estimates into target adaption process. All the DA scenarios that have been addressed in this chapter are summarized in Fig. 1.2. Since, the related literature is non-homogeneous and differs from one DA setting to the other, we have described the related works in the respective chapters.

1.2.1 Domain Whitening Transform

In Chapter 2 we address the STDA task where the goal is to adapt a model on a desired target domain of interest by leveraging a related labelled source data set along with an unlabelled target data set. In this chapter, to bridge the domain-gap we propose to align the marginal feature distributions between the source and target domains through our proposed Domain

Whitening Transform (DWT) layers that are embedded inside the neural network. In details, our DWT layers align the first and the second order moments of the features and can be seen as a generalization of the correlation alignment and batch normalization-based alignment layers, which are commonly used in several UDA approaches. Secondly, we propose an unsupervised loss on the target data called Min-Entropy Consensus (MEC) loss that unifies entropy minimization and consistency regularization losses. Specifically, the proposed MEC loss simultaneously encourages coherent predictions between two perturbed versions of the same target sample and exploits these predictions as pseudo-labels for training. Through extensive experiments on several UDA benchmarks we show that our proposed components consistently improve performance with respect to the existing state-of-the-art methods.

1.2.2 TriGAN for Multi-source Domain Adaptation

In the next Chapter 3 we address a slightly more challenging UDA setting called MSDA where labelled source data comes from multiple source domains and the goal is to adapt a model to work well on a single target domain. In this work we adopt a generative image translation approach where the goal is to generate target-like source images such that they can be leveraged to train a target-specific classifier. In details, we build our generative adversarial network (TriGAN) inspired by the observation that the appearance of a given image depends on three factors: the domain, the style (characterized in terms of low-level features variations) and the content. To this end, we use the previously proposed DWT layers (and their variations) to project the image features onto a space where only the dependence from the content is kept, and then re-project this invariant representation onto the pixel space using the target domain and style. With our proposed approach image translation between any pair of source and target domain can be achieved with a single generator network, which greatly simplifies training, especially when the number of source domains are large. We conduct thorough experiments on the MSDA benchmarks and show that our end results are both quantitatively and qualitatively superior.

1.2.3 Curriculum Graph Co-teaching for Multi-target Domain Adaptation

In Chapter 4 we address a more real-world and practical problem of multi-target domain adaptation (MTDA) where the goal is to adapt a single model towards multiple unlabelled target domains using only a single labelled source data set. Since multiple domain shifts need to be addressed we propose to learn an unified feature space through a graph convolutional network (GCN) that aggregates features from similar samples across the domains. As our GCN depends on pseudo-labels to connect similar samples in the graph, which can by noisy, we develop a co-teaching strategy with a dual classifier head that is assisted by curriculum learning to obtain more reliable pseudo-labels. Additionally, when the domain labels are available, we propose Domain-aware Curriculum Learning (DCL), a sequential adaptation strategy that first adapts on the easier target domains, followed by the harder ones. Through our extensive experiments on several MTDA benchmarks we show the beneficial impact of each of our proposed components, attaining state-of-the-art results in this task.

1.2.4 Uncertainty-aware Source-free Domain Adaptation

Finally, in Chapter 5 we turn our attention to very emerging and challenging DA problem called source-free domain adaptation (SFDA) where the task is to adapt a classifier to an unlabelled target data set by only using a pre-trained source model. SFDA is challenging because the absence of the source data and the domain shift makes the predictions of the source model on the target data becomes unreliable. Therefore, in this chapter we propose quantifying the uncertainty in the source model predictions and utilizing it to guide the target adaptation. We construct a probabilistic source model by incorporating priors on the network parameters inducing a distribution over the model predictions. Uncertainties are estimated by employing a Laplace approximation and incorporated to identify target data points that do not lie in the source manifold and to down-weight them when maximizing the mutual information on the target data. We show the advantages of uncertainty-guided SFDA over traditional SFDA in the closed-set and open-set settings and provide empirical evidence that our approach is more robust to strong domain shifts.

1.3 Contributions

Working in the context of visual domain adaptation using deep learning techniques we have made the following contributions:

- *Domain Whitening Transform* layers to align the marginal feature distributions between the source and target domains, which are embedded inside a neural network. DWT layers generalizes the Batch Normalization based domain alignment layers.

- *Min Entropy Consensus Loss*, an unsupervised regularization loss, on the unlabelled target domain images that enforces consistent predictions between two perturbed versions of a target image and at the same time encourages peaked prediction on one of the semantic classes.
- *TriGAN* an generative framework that performs image-to-image translation between multiple source domains and one target domain in order to generate synthetic target-like source images.
- *Instance Whitening Transform*, *conditional Domain Whitening Transform* and *Adaptive Instance Whitening Transform*, which are based on the DWT, are proposed to help realize the TriGAN generator.
- *Curriculum Graph Co-teaching*, a co-teaching and graph neural network based feature aggregation framework that aligns similar samples from different domains in order to obtain a unified feature space in MTDA.
- *Domain Curriculum Learning* that follows an easy to hard target domain selection strategy in MTDA where the feature alignment process begins with the easiest target domain and gradually progresses to the hardest one.
- *Bayesian framework* U-SFAN that constructs a probabilistic source model to quantify the uncertainty in the source model predictions on the target data and utilizes it to guide the adaptation. We show U-SFAN is more robust under strong domain shifts.
- *Laplace Approximation* is employed to estimate the uncertainties, which we show to entail well for the SFDA setting as it decouples source training from target adaptation.

1.4 Publications

The following is the list of the papers published during the doctoral study period in the reverse chronological order. Note that works marked with * are not included in this thesis:

- Yangsong Zhang, Subhankar Roy, Hongtao Lu, Elisa Ricci, Stéphane Lathuilière. “Co-operative Self-Training for Multi-Target Adaptive Semantic Segmentation”. In WACV, 2023.*

- Subhankar Roy, Mingxuan Liu, Zhun Zhong, Nicu Sebe, Elisa Ricci. “Class-incremental Novel Class Discovery”. In ECCV, 2022.*

- Subhankar Roy, Martin Trapp, Andrea Pilzer, Juho Kannala, Nicu Sebe, Elisa Ricci, Arno Solin. “Uncertainty-guided Source-free Domain Adaptation”. In ECCV, 2022. (Chapter 5 is mainly based on the content of this publication)

- Subhankar Roy, Evgeny Krivosheev, Zhun Zhong, Nicu Sebe, Elisa Ricci. “Curriculum Graph Co-Teaching for Multi-Target Domain Adaptation”. In CVPR, 2021. (Chapter 4 is mainly based on the content of this publication).

- Zhun Zhong, Enrico Fini, Subhankar Roy, Zhiming Luo, Elisa Ricci, Nicu Sebe. “Neighborhood Contrastive Learning for Novel Class Discovery”. In CVPR, 2021.*

- Subhankar Roy, Aliaksandr Siarohin, Enver Sangineto, Samuel Rota Bulo, Nicu Sebe, Elisa Ricci. “TriGAN: Image-to-Image Translation for Multi-Source Domain Adaptation”. In MVAP, 2021. (Chapter 3 is mainly based on the content of this publication).

- Aliaksandr Siarohin, Subhankar Roy, Stéphane Lathuilière, Sergey Tulyakov, Elisa Ricci, Nicu Sebe. “Motion-supervised Co-Part Segmentation”. In ICPR, 2021.*

- Subhankar Roy, Willi Menapace, Sebastiaan Oei, Ben Luijten, Enrico Fini, Cristiano Saltori, Iris Huijben, Nishith Chennakeshava, Federico Mento, Alessandro Sentelli, Emanuele Peschiera, Riccardo Trevisan, Giovanni Maschietto, Elena Torri, Riccardo Inchingolo, Andrea Smargiassi, Gino Soldati, Paolo Rota, Andrea Passerini, Ruud JG Van Sloun, Elisa Ricci, Libertario Demi. “Deep learning for classification and localization of COVID-19 markers in point-of-care lung ultrasound”. In TMI, 2020.*

- Subhankar Roy, Aliaksandr Siarohin, Nicu Sebe. “Unsupervised Domain Adaptation Using Full-Feature Whitening and Colouring”. In ICIAP, 2019.*

- Cristiano Saltori, Subhankar Roy, Nicu Sebe, Giovanni Iacca. “Regularized Evolutionary Algorithm for Dynamic Neural Topology Search”. In ICIAP, 2019.*

- Subhankar Roy, Aliaksandr Siarohin, Enver Sangineto, Samuel Rota Bulo, Nicu Sebe, Elisa Ricci. “Unsupervised Domain Adaptation using Feature-Whitening and Consensus Loss”. In CVPR, 2019. (Chapter 2 is mainly based on the content of this publication).

1.5 Works Under Review

The following is the list of the works during the doctoral study period that are under peer review. Note that works marked with * are not included in this thesis:

- Giacomo Zara, Victor Turrisi da Costa, Subhankar Roy, Paolo Rota, Paolo Rota and Elisa Ricci. “Simplifying Open-Set Video Domain Adaptation with Contrastive Learning”. Under review in WACV 2023 (2nd round)

2

Domain Whitening Transform

A classifier trained on a dataset seldom works on other datasets obtained under different conditions due to domain shift. This problem is commonly addressed by domain adaptation methods. In this chapter we introduce a novel deep learning framework which unifies different paradigms in unsupervised domain adaptation. Specifically, we propose domain alignment layers which implement feature whitening for the purpose of matching source and target feature distributions. Additionally, we leverage the unlabeled target data by proposing the Min-Entropy Consensus loss, which regularizes training while avoiding the adoption of many user-defined hyper-parameters. We report results on publicly available datasets, considering both digit classification and object recognition tasks. We show that, in most of our experiments, our approach improves upon previous methods, setting new state-of-the-art performances.[1]

2.1 Introduction

Deep learning methods have been successfully applied to different visual recognition tasks, demonstrating an excellent generalization ability. However, analogously to other statistical machine learning techniques, deep neural networks also suffer from the problem of *domain shift* [157], which is observed when predictors trained on a dataset do not perform well when applied to novel domains.

Since collecting annotated training data from every possible domain is expensive and sometimes even impossible, over the years several Domain Adaptation (DA) methods [25, 119]

[1] The content of this chapter is based on the CVPR 2019 paper [131]

have been proposed. DA approaches leverage labeled data in a source domain in order to learn an accurate prediction model for a target domain. Specifically, in the special case of Unsupervised Domain Adaptation (UDA), no annotated target data are available at training time. Note that, even if target-sample labels are not available, unlabeled data can and usually are exploited at training time.

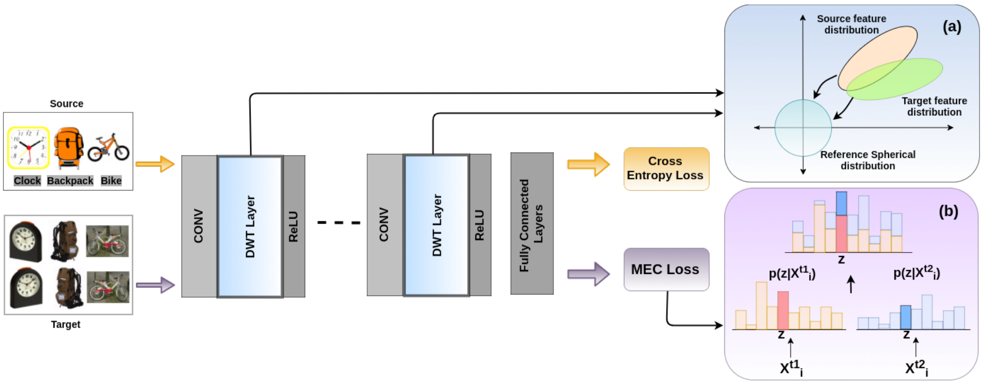


Figure 2.1: Overview of the proposed deep architecture embedding our DWT layers and trained with the proposed MEC loss. (a) Due to domain shift the source and the target data have different marginal feature distributions. Our DWT estimates these distributions using dedicated sample batches and then "whitens" them projecting them into a common, spherical distribution. (b) The proposed MEC loss univocally selects a pseudo-label z that maximizes the agreement between two perturbed versions $\mathbf{x}_i^{t_1}$ and $\mathbf{x}_i^{t_2}$ of the same target sample.

Most UDA methods attempt to reduce the domain shift by directly aligning the source and target marginal distributions. Notably, approaches based on the *Correlation Alignment* paradigm model domain data distributions in terms of their second-order statistics. Specifically, they match distributions by minimizing a loss function which corresponds to the difference between the source and the target covariance matrices obtained using the network's last-layer activations [111, 151, 152]. Another recent and successful UDA paradigm exploits *domain-specific alignment layers*, derived from Batch Normalization (BN) [66], which are directly embedded within the deep network [13, 88, 108]. Other prominent research directions in UDA correspond to those methods which also exploit the target data posterior distribution. For instance, the *entropy minimization* paradigm adopted in [13, 51, 140], enforces the network's prediction probability distribution on each target sample to be peaked with respect to some (unknown) class, thus penalizing high-entropy target predictions. On the other hand, the *consistency-enforcing* paradigm [31, 143, 154] is based on specific loss functions which penalize inconsistent predictions over perturbed copies of the same target samples.

In this paper we propose to unify the above paradigms by introducing two main novelties. First, we align the source and the target data distributions using covariance matrices similarly to [111, 151, 152]. However, instead of using a loss function computed on the last-layer activations, we use domain-specific alignment layers which compute domain-specific covariance matrices of intermediate features. These layers "whiten" the source and the target features and project them into a common spherical distribution (see Fig. 2.1 (a), blue box). We call this alignment strategy *Domain-specific Whitening Transform* (DWT). Notably, our approach generalizes previous BN-based DA methods [13, 88, 107] which do not consider inter-feature correlations and rely only on feature standardization.

The second novelty we introduce is a novel loss function, the Min-Entropy Consensus (MEC) loss, which merges both the entropy [13, 51, 140] and the consistency [31] loss function. The motivation behind our proposal is to avoid the tuning of the many hyper-parameters which are typically required when considering several loss terms and, specifically, the confidence-threshold hyper-parameters [31]. Indeed, due to the mismatch between the source and the target domain, and because of the unlabeled target-data assumption, hyper-parameters are hard to be tuned in UDA [111]. The proposed MEC loss simultaneously encourages coherent predictions between two perturbed versions of the same target sample and exploits these predictions as pseudo-labels for training. (Fig. 2.1 (b), purple box).

We plug our proposed DWT and the MEC loss into different network architectures and we empirically show a significant boost in performance. In particular, we achieve state-of-the-art results in different UDA benchmarks: MNIST [82], USPS [32], SVHN [115], CIFAR-10, STL10 [23] and Office-Home [163].

2.2 Related Work

Unsupervised Domain Adaptation. Several previous works have addressed the problem of DA, considering both shallow models and deep architectures. In this section we focus on only deep learning methods for UDA, as these are the closest to our proposal.

UDA methods mostly differ in the strategy used to reduce the discrepancy between the source and the target feature distributions and can be grouped in different categories. The first category includes methods modeling the domain distributions in terms of their first and second order statistics. For instance, some works aim at reducing the domain shift by minimizing the Maximum Mean Discrepancy [98, 99, 163] and describe distributions in terms of their first order statistics. Other works consider also second-order statistics using the *correlation alignment* paradigm (Sec. 2.1) [111, 152]. Instead of introducing additional loss functions, more recent works deal with the domain-shift problem by directly embedding into a deep network *domain alignment layers* which exploit BN [13, 88, 106, 108].

A second category of methods include approaches which learn domain-invariant deep representations. For instance, in [36] a gradient reversal layer learns discriminative domain-agnostic representations. Similarly, in [158] a domain-confusion loss is introduced, encouraging the network to learn features robust to the domain shift. Haeusser *et al.* [52] present Associative Domain Adaptation, an approach which also learns domain-invariant embeddings.

A third category includes methods based on Generative Adversarial Networks (GANs) [9, 136, 144, 147, 153]. The main idea behind these approaches is to directly transform images from the target domain to the source domain. While GAN-based methods are especially successful in adaptation from synthetic to real images and in case of non-complex datasets, they have limited capabilities for complex images.

Entropy minimization, first introduced in [47], is a common strategy in semi-supervised learning [184]. In a nutshell, it consists in exploiting the high-confidence predictions of unlabeled samples as pseudo-labels. Due to its effectiveness, several popular UDA methods [13, 99, 136, 140] have adopted the entropy-loss for training deep networks.

Another popular paradigm in UDA, which we refer to as the *consistency-enforcing* paradigm, is realized by perturbing the target samples and then imposing some consistency between the predictions of two perturbed versions of the same target input. Consistency is imposed by defining appropriate loss functions, as shown in [31, 140, 143]. The consistency loss paradigm is effective but it becomes uninformative if the network produces uniform probability distributions for corresponding target samples. Thus, previous methods also integrate a Confidence Thresholding (CT) technique [31], in order to discard unreliable predictions. Unfortunately, CT introduces additional user-defined and dataset-specific hyper-parameters which are difficult to tune in an UDA scenario [111]. Differently, as demonstrated in our experiments, our MEC loss eliminates the need of CT and the corresponding hyper-parameters. We refer the readers to the comprehensive domain adaptation survey [25] for further readings.

Feature Decorrelation. Recently, Huang *et al.* [63] and Siarohin *et al.* [150] proposed to replace BN with feature whitening in a discriminative and generative setting, respectively. However, none of these works consider a DA problem. We show in this paper that feature whitening can be used to align the source and the target marginal distributions using layer-specific covariance matrices without the need of a dedicated loss function as in previous correlation alignment methods.

2.3 Methods

In this section we present the proposed UDA approach. Specifically, after introducing some preliminaries, we describe our Domain-Specific Whitening Transform and, finally, the proposed Min-Entropy Consensus loss.

2.3.1 Preliminaries

Let $\mathcal{S} = \{(I_j^s, y_j^s)\}_{j=1}^{n_s}$ be the labeled source dataset, where I_j^s is an image and $y_j^s \in \mathcal{Y} = \{1, 2 \ldots, C\}$ its associated label, and $\mathcal{T} = \{I_i^t\}_{i=1}^{n_t}$ be the unlabeled target dataset. The goal of UDA is to learn a predictor for the target domain by using samples from both $\mathcal{S}$ and $\mathcal{T}$. Learning a predictor for the target domain is not trivial because of the issues discussed in Sec. 2.1.

A common technique to reduce domain shift is to use BN-based layers inside a network, such as to project the source and target feature distributions to a reference distribution through feature standarization. As mentioned in Sec. 2.1, in this work we propose to replace feature standardization with whitening, where the whitening operation is domain-specific. Before introducing the proposed whitening-based distribution alignment, we recap below BN. Let $B = \{\mathbf{x}_1, ..., \mathbf{x}_m\}$ be a mini-batch of m input samples to a given network layer, where each element $\mathbf{x}_i \in B$ is a d-dimensional feature vector, *i.e.* $\mathbf{x}_i \in \mathbb{R}^d$. Given B, in BN each $\mathbf{x}_i \in B$ is transformed as follows:

$$\mathit{BN}(x_{i,k}) = \gamma_k \frac{x_{i,k} - \mu_{B,k}}{\sqrt{\sigma_{B,k}^2 + \epsilon}} + \beta_k, \tag{2.1}$$

where k ($1 \leq k \leq d$) indicates the k-th dimension of the data, $\mu_{B,k}$ and $\sigma_{B,k}$ are, respectively, the mean and the standard deviation computed with respect to the k-th dimension of the samples in B and ϵ is a constant used to prevent numerical instability. Finally, γ_k and β_k are scaling and shifting learnable parameters.

In the next section we present our DWT, while in Sec. 2.3.3 we present the proposed MEC loss. It is worth noting that each proposed component can be plugged independently in a network without having to rely on each other.

2.3.2 Domain-specific Whitening Transform

As stated above, BN is based on a per-dimension *standardization* of each sample $\mathbf{x}_i \in B$. Hence, once normalized, the batch samples may still have correlated feature values. Since our goal is to use feature normalization in order to alleviate the domain-shift problem (see below), we argue that plain standardization is not enough to align the source and the target marginal distributions. For this reason we propose to use Batch Whitening (BW) instead of BN, which is defined as:

$$\mathit{BW}(x_{i,k}; \Omega) = \gamma_k \hat{x}_{i,k} + \beta_k, \tag{2.2}$$

$$\hat{\mathbf{x}}_i = W_B(\mathbf{x}_i - \boldsymbol{\mu}_B). \tag{2.3}$$

In Eq. (2.3), the vector $\boldsymbol{\mu}_B$ is the mean of the elements in B (being $\mu_{B,k}$ its k-th component) while the matrix W_B is such that: $W_B^\top W_B = \Sigma_B^{-1}$, where Σ_B is the covariance matrix computed using B. $\Omega = (\boldsymbol{\mu}_B, \Sigma_B)$ are the batch-dependent first and second-order statistics. Eq. (2.3) performs the *whitening* of $\mathbf{x}_i$ and the resulting set of vectors $\hat{B} = \{\hat{\mathbf{x}}_1, ..., \hat{\mathbf{x}}_m\}$ lie in a spherical distribution (i.e., with a covariance matrix equal to the identity matrix).

Our network takes as input two different batches of data, randomly extracted from $\mathcal{S}$ and $\mathcal{T}$, respectively. Specifically, given any arbitrary layer l in the network, let $B^s = \{\mathbf{x}_1^s, ..., \mathbf{x}_m^s\}$ and $B^t = \{\mathbf{x}_1^t, ..., \mathbf{x}_m^t\}$ denote the batch of intermediate output activations, from layer l, for the source and target domain, respectively. Using Eq. (2.2)-(2.3) we can now define our Domain-specific Whitening Transform (DWT). Let x^s and x^t denote the inputs to the DWT layer from the source and the target domain, respectively. Our DWT is defined as follows (we drop the sample index i and dimension index k for the sake of clarity):

$$\mathit{DWT}(x^s; \Omega^s) = BW(x^s, \Omega^s), \tag{2.4}$$

$$\mathit{DWT}(x^t; \Omega^t) = BW(x^t, \Omega^t). \tag{2.5}$$

We estimate separate statistics ($\Omega^s = (\boldsymbol{\mu}_B^s, \Sigma_B^s)$ and $\Omega^t = (\boldsymbol{\mu}_B^t, \Sigma_B^t)$) for B^s and B^t and use them for whitening the corresponding activations, projecting the two batches into a common spherical distribution (Fig. 2.1 (a)).

W_B^s and W_B^t are computed following the approach described in [150], which is based on the Cholesky decomposition [28]. The latter is faster [150] than the ZCA-based whitening [71] adopted in [63]. In the Supplementary Material we provide more details on how W_B^s and W_B^t are computed. Differently from [150] we replace the "coloring" step after whitening with simple scale and shift operations, thereby preventing the introduction of extra parameters in the network. Also, differently from [150] we use *feature grouping* [63] (Sec. 2.3.2.1) in order to make the batch-statistics estimate more robust when m is small and d is large. During training, the DWT layers accumulate the statistics for the target domain using a moving average of the batch statistics (Ω_{avg}^t).

In summary, the proposed DWT layers replace the correlation alignment of the last-layer feature activations with the intermediate-layer feature whitening, performed at different levels of abstraction. In Sec. 2.3.2.1 we show that BN-based domain alignment layers [13, 88] can be seen as a special case of DWT layers.

2.3.2.1 Implementation Details

Given a typical block (Conv layer → BN → ReLU) of a CNN, we replace the BN layer with our proposed DWT layer (see in Fig. 2.1), obtaining: (Conv layer → DWT → ReLU). Ideally, in order to project the source and target feature distributions to a reference one, the DWT layers should perform full-feature whitening using a $d \times d$ whitening matrix, where d is the number of features. However, the computed covariance matrix Σ_B can be ill-conditioned if d is large and m is small. For this reason, unlike [150] and similar to [63] we use *feature grouping*, where the features are grouped into subsets of size g. This results in better-conditioned covariance matrices but into partially whitened features. In this way we reach a compromise between full-feature whitening and numerical stability. Interestingly, when $g = 1$, the whitening matrices reduce to diagonal matrices, thus realizing feature standardization as in [13, 88].

2.3.3 Min-Entropy Consensus Loss

The impossibility of using the cross-entropy loss on the unlabeled target samples is commonly circumvented using some common unsupervised loss, such as the entropy [13, 140] or the consistency loss [31, 143]. While minimizing the entropy loss ensures that the predictor maximally separates the target data, minimization of the consistency loss forces the predictor to deliver consistent predictions for target samples coming from identical (yet unknown) category. Therefore, given the importance of exploiting better the unlabeled target data and the limitations of the above two losses (see Sec. 2.1), we propose a novel Min-Entropy Consensus (MEC) loss within the framework of UDA. We explain below how MEC loss merges both the entropy and the consistency loss into a single unified function.

Similar to the consistency loss, the proposed MEC loss requires input data perturbations. Unless otherwise explicitly specified, we apply common data-perturbation techniques on both $\mathcal{S}$ and $\mathcal{T}$ using affine transformations and Gaussian blurring operations. When we use the MEC loss, the network is fed with three batches instead of two. Specifically, apart from B^s, we use two different target batches (B_1^t and B_2^t), which contain duplicate pairs of images differing only with respect to the adopted image perturbation.

Conceptually, we can think of this pipeline as three different networks with three separate domain-specific statistics Ω^s, Ω_1^t and Ω_2^t but with shared network weights. However, since both B_1^t and B_2^t are drawn from the same distribution, we estimate a single Ω^t using both the target batches ($B_1^t \bigcup B_2^t$). As an additional advantage, this makes it possible to use $2m$ samples for computing Σ_B^t.

Let $B^s = \{\mathbf{x}_1^s, ..., \mathbf{x}_m^s\}$, $B_1^t = \{\mathbf{x}_1^{t_1}, ..., \mathbf{x}_m^{t_1}\}$ and $B_2^t = \{\mathbf{x}_1^{t_2}, ..., \mathbf{x}_m^{t_2}\}$ be three batches of the last-layer activations. Since the source samples are labeled, the cross-entropy loss (L^s) can be used in case of B^s:

$$L^s(B^s) = -\frac{1}{m}\sum_{i=1}^{m} \log p(y_i^s|\mathbf{x}_i^s), \tag{2.6}$$

where $p(y_i^s|\mathbf{x}_i^s)$ is the (soft-max-based) probability prediction assigned by the network to a sample $\mathbf{x}_i^s \in B^s$ with respect to its ground-truth label y_i^s. However, ground-truth labels are not available for target samples. For this reason, we propose the following MEC loss (L^t):

$$L^t(B_1^t, B_2^t) = \frac{1}{m}\sum_{i=1}^{m} \ell^t(\mathbf{x}_i^{t_1}, \mathbf{x}_i^{t_2}), \tag{2.7}$$

$$\ell^t(\mathbf{x}_i^{t_1}, \mathbf{x}_i^{t_2}) = -\frac{1}{2} \max_{y \in \mathcal{Y}} \Big(\log p(y|\mathbf{x}_i^{t_1}) + \log p(y|\mathbf{x}_i^{t_2}) \Big). \tag{2.8}$$

In Eq. (2.8), $\mathbf{x}_i^{t_1} \in B_1^t$ and $\mathbf{x}_i^{t_2} \in B_2^t$ are activations of two corresponding perturbed target samples.

The intuitive idea behind our proposal is that, similarly to consistency-based losses [31, 143], since $\mathbf{x}_i^{t_1}$ and $\mathbf{x}_i^{t_2}$ correspond to the same image, the network should provide similar predictions. However, unlike the aforementioned methods which compute the L2-norm or the binary cross-entropy between these predictions, the proposed MEC loss finds the class z such that $z = \text{argmin}_{y \in \mathcal{Y}} \Big(\log p(y|\mathbf{x}_i^{t_1}) + \log p(y|\mathbf{x}_i^{t_2}) \Big)$. z is the class in which the posteriors corresponding to $\mathbf{x}_i^{t_1}$ and $\mathbf{x}_i^{t_2}$ maximally agree. We then use z as the pseudo-label, which can be selected without ad-hoc confidence thresholds. In other words, instead of using high-confidence thresholds to discard unreliable target samples [31], we use all the samples but we backpropagate the error with respect to only z.

The dynamics of MEC loss is the following. First, similarly to the consistency losses, it forces the network to provide coherent predictions. Second, differently from consistency losses, which are prone to attain a near zero value with uniform posterior distributions, it enforces peaked predictions. See the Supplementary Material for a more formal relation between the MEC loss and both entropy and consistency loss.

The final loss L is a weighted sum of L^s and L^t: $L(B^s, B_1^t, B_2^t) = L^s(B^s) + \lambda L^t(B_1^t, B_2^t)$.

2.3.4 Discussion

The proposed DWT generalizes the BN-based DA approaches by decorrelating the batch features. Besides the analogy with the correlation-alignment methods mentioned in Sec. 2.1, in which covariance matrices are used to estimate and align the source and the target distributions, a second reason for which we believe that full-whitening is important is due to the relation between feature normalization and the smoothness of the loss [63, 74, 83, 137, 148]. For instance, previous works [83, 137] showed that better conditioning of the input-feature covariance matrix leads to better conditioning of the Hessian of the loss function, making the gradient descent weight updates closer to Newton updates. However, BN only performs standardization, which barely improves the conditioning of the covariance matrix when the features are correlated

[63]. Conversely, feature whitening completely decorrelates the batch samples, thus potentially improving the smoothness of the landscape of the loss function.

The importance of a smoothed loss function is even higher when entropy-like losses on unlabeled data are used. For instance, Shu *et al.* [148] showed that minimizing the entropy forces the classifier to be confident on the unlabeled target data, thus potentially driving the classifier's decision boundaries away from the target data. However, without a locally-Lipschitz constraint on the loss function (*i.e.* with a non smoothed loss landscape), the decision boundaries can be placed close to the training samples even when the entropy is minimized [148]. Since our MEC loss is related with both the entropy and the consistency loss, we employ DWT also to improve the smoothness of our loss function in order to alleviate overfitting phenomena related to the use of unlabeled data.

2.4 Experiments

In this section we provide details about our implementation and training protocols and we report our experimental evaluation. We conduct experiments on both small and large-scale datasets and we compare our method with state-of-the-art approaches. We also present an ablation study to analyze the impact of each of our contributions on the classification accuracy.

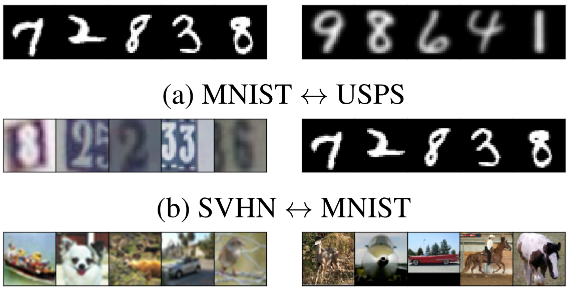
2.4.1 Datasets

We conduct experiments on the following datasets:

MNIST ↔ **USPS.** The **MNIST** dataset [82] contains grayscale images (28 × 28 pixels) depicting handwritten digits ranging from 0 to 9. The **USPS** [32] dataset is similar to MNIST, but images have smaller resolution (16 × 16 pixels). See Fig. 2.2(a) for sample images.

MNIST ↔ **SVHN.** Street View House Number (SVHN) [115] images are 32 × 32 pixels RGB images. Similarly to the MNIST dataset digits range from 0 to 9. However, in SVHN images have variable colour intensities and depict non-centered digits. Thus, there is a significant domain shift with respect to MNIST (Fig. 2.2(b))

CIFAR-10 ↔ **STL**: CIFAR-10 is a 10 class dataset of RGB images depicting generic objects and with resolution 32 × 32 pixels. STL [23] is similar to the CIFAR-10, except



(a) MNIST ↔ USPS

(b) SVHN ↔ MNIST

(c) CIFAR-10 ↔ STL

Figure 2.2: Small image datasets used in our experiments.

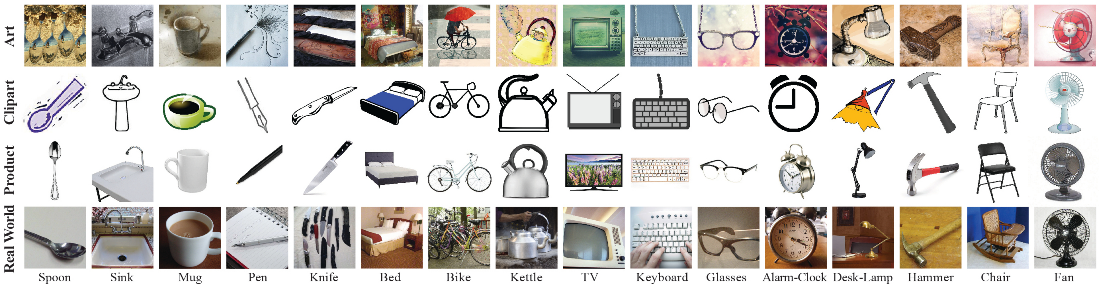


Figure 2.3: Sample images from the Office-Home dataset.

it has fewer labelled training images per class and has images of resolution 96 × 96 pixels. The non-overlapping classes - “frog” and “monkey” are removed from CIFAR-10 and STL, respectively. Samples are shown in Fig. 2.2.(c).

Office-Home: The Office-Home [163] dataset comprises 4 distinct domains, each corresponding to 65 different categories (Fig. 2.3). There are 15,500 images in the dataset, thus this represents large-scale benchmark for testing domain adaptation methods. The domains are: `Art`(**Ar**), `Clipart` (**Cl**), `Product` (**Pr**) and `Real World` (**Rw**).

2.4.2 Experimental Setup

To fairly compare our method with other UDA approaches, in the digits experiments we adopt the same base networks proposed in [38]. For the CIFAR-10↔STL experiments we use the network described in [31]. We train the networks using the Adam optimizer [72] with a mini-batch of cardinality $m = 64$ samples, an initial learning rate of 0.001 and weight decay of 5×10^{-4}. The networks are trained for a total of 120 epochs with learning rate being decreased by a factor of 10 after 50 and 90 epochs. We use the SVHN → MNIST setting to fix the value of the hyperparameter λ to 0.1 and to set group size (g) equal to 4. These hyperparameters values are used for all the datasets.

In the Office-Home dataset experiments we use a ResNet-50 [57] following [97]. In our experiments we modify ResNet-50 by replacing the first BN layer and the BN layers in the first residual block (with 64 features) with DWT layers. The network is initialized with weights taken from a pre-trained model trained on the ILSVRC-2012 dataset. We discard the final fully-connected layer and we replace it with a randomly initialized fully-connected layer with 65 output logits. During training, each domain-specific batch is limited to m = 20 samples (due to GPU memory constraints). The SGD optimizer is used with an initial learning rate of 10^{-2} for the randomly initialized final layer and 10^{-3} for the rest of the trainable parameters of the network. The network is trained for a total of 60 epochs where one "epoch" is the pass through the entire data set having the lower number of training samples. The learning rates are then decayed by a factor of 10 after 54 epochs. Differently from the small-scale datasets experiments, where target samples have predefined train and test splits, in the Office-Home experiments, all the target samples (without labels) are used during training and evaluation.

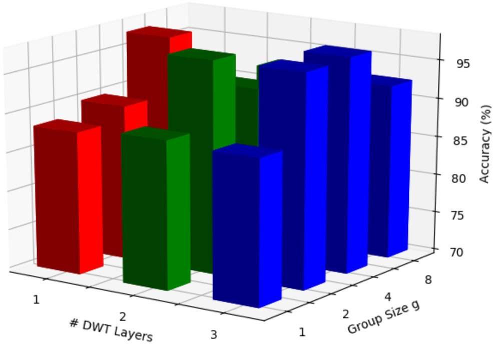


Figure 2.4: SVHN → MNIST experiment: accuracy at varying number of DWT layers and group size. Different colors are used to improve readability.

To demonstrate the effect our contributions, we consider three different variants for the proposed method. In the first variant (denoted as **DWT** in Sec. 2.3.2), we only consider DWT layers *without* the proposed MEC loss. In practice, in the considered network architectures we replace the BN layers which follows the convolutional layers with DWT layers. Supervised cross-entropy loss is used for the labeled source samples and the entropy-loss as in [13] is used for the unlabeled target samples. No data-augmentation is used here. In the second variant, denoted as **DWT-MEC**, we also exploit the proposed MEC loss (this corresponds to our full method). In this case we need perturbations of the input data, which are obtained by following some basic data-perturbation schemes like image translation by a factor of [0.05, 0.05], Gaussian blur ($\sigma = 0.1$) and random affine transformation as proposed in [31]. In the third variant (**DWT-MEC (MT)**) we plug our proposed DWT layers and the MEC loss in the Mean-Teacher (MT) training paradigm [154].

Method	Source Target	MNIST USPS	USPS MNIST	SVHN MNIST
SE (w/ CT) [31]		99.29	99.26	97.88
SE (w/o CT) [31]		98.71	97.63	26.80
DWT-MEC (MT)		**99.30**	**99.15**	**99.14**

Table 2.1: Accuracy (%) on the digits datasets. Comparison between the consistency loss in SE method [31] (with and without CT) and our threshold-free MEC loss.

2.4.3 Results

In this section we present an extensive experimental analysis of our approach, showing both the results of an ablation study and a comparison with state-of-the-art methods.

2.4.3.1 Ablation Study

We first conduct a thorough analysis of our method assessing, in isolation, the impact of our two main contributions: (i) aligning source and target distributions by embedded DWT layers; and (ii) leveraging target data through our threshold-free MEC loss.

First, we consider the SVHN→MNIST setting and we show the benefit of feature whitening over BN. We vary the number of whitening layers from 1 to 3 and simultaneously change the group size (g) from 1 to 8 (see Sec. 2.3.2.1). With group size equal to 1, DWT layers reduces to DA layers as proposed in [13, 88]. Our results are shown in Fig. 2.4 and from the figure it is clear that when $g = 1$ the accuracy stays consistently below 90%. This behaviour can be ascribed to the sub-optimal alignment of source and target data distributions achieved with previous BN-based DA layers. When the group size increases, the feature decorrelation performed by the DWT layers comes into play and results into a significant improvement in terms of performance. The accuracy increases monotonically as the group size grows until the value of $g = 4$, then it start to decrease. This final drop is probably due to ill-conditioned covariance matrices. Indeed, a covariance matrix with size 8 × 8 is perhaps poorly estimated due to the lack of samples in a batch (Sec. 2.3.2.1). Importantly, Fig. 2.4 also shows that increasing the number of DWT layers has a positive impact on the accuracy. This is in contrast with [63], where feature decorrelation is used only in the first layer of the network.

	Source	MNIST	USPS	SVHN	MNIST
Methods	Target	USPS	MNIST	MNIST	SVHN
Source Only		78.9	57.1±1.7	60.1±1.1	20.23±1.8
w/o augmentation					
CORAL [151]		81.7	-	63.1	-
MMD [158]		81.1	-	71.1	-
DANN [38]		85.1	73.0±2.0	73.9	35.7
DSN [11]		91.3	-	82.7	-
CoGAN [94]		91.2	89.1±0.8	-	-
ADDA [160]		89.4±0.2	90.1±0.8	76.0±1.8	-
DRCN [41]		91.8±0.1	73.7±0.1	82.0±0.2	40.1±0.1
ATT [140]		-	-	86.20	52.8
ADA [51]		-	-	97.6	-
AutoDIAL [13]		97.96	97.51	89.12	10.78
SBADA-GAN [136]		97.6	95.0	76.1	**61.1**
GAM [62]		95.7±0.5	98.0±0.5	74.6±1.1	-
MECA [111]		-	-	95.2	-
DWT		**99.09**±0.09	**98.79**±0.05	**97.75**±0.10	28.92 ±1.9
Target Only		96.5	99.2	99.5	96.7
w/ augmentation					
SE [a] [31]		88.14±0.34	92.35±8.61	93.33±5.88	33.87±4.02
SE [b] [31]		98.23±0.13	**99.54**±0.04	**99.26**±0.05	**37.49**±2.44
SE [†] [b] [31]		99.29±0.16	99.26±0.04	97.88±0.03	24.09±0.33
DWT-MEC[b]		99.01±0.06	99.02±0.05	97.80±0.07	30.20±0.92
DWT-MEC (MT)[b]		**99.30**±0.19	99.15±0.05	99.14±0.02	31.58±2.34

Table 2.2: Accuracy (%) on the digits datasets: comparison with state of the art. [a] indicates minimal usage of data augmentation and [b] considers augmented source and target data. [†] indicates our implementation of SE [31].

In Tab. 2.1 we evaluate the effectiveness of the proposed MEC loss and we compare our approach with the consistency based loss adopted by French *et al.* [31]. We use Self-Ensembling (SE) [31] with and without confidence thresholding (CT) on the network predictions of the teacher network. To fairly compare our approach with SE we also consider a mean-teacher (MT) scheme in our framework. MT follows the training scheme of SE where the weights of the teacher network is the exponential moving average of those of the student. During training, the augmented versions of the target sample are passed through both the student and teacher network.

We observe that SE have excellent performance when the CT is set to a very high value (0.936 as in [31]) but it performance drops when CT is set equal to 0, especially in the SVHN→MNIST setting. This shows that the consistency loss in [31] may be harmful when the network is not confident on the target samples. Conversey, the proposed MEC loss leads to results which are on par to SE in the MNIST↔USPS settings and to higher accuracy in the SVHN→MNIST setting. This clearly demonstrates that our proposed loss avoids the need of introducing the CT hyper-parameter and, at the same time, yields to better performance. It is important to remark that, in the case of UDA, tuning hyper-parameters is hard as target samples are unlabeled and cross-validation on source data is unreliable because of the domain shift problem [111].

	Source	Ar	Ar	Ar	Cl	Cl	Cl	Pr	Pr	Pr	Rw	Rw	Rw	
Method	Target	Cl	Pr	Rw	Ar	Pr	Rw	Ar	Cl	Rw	Ar	Cl	Pr	Avg
ResNet-50 [57]		34.9	50.0	58.0	37.4	41.9	46.2	38.5	31.2	60.4	53.9	41.2	59.9	46.1
DAN [98]		43.6	57.0	67.9	45.8	56.5	60.4	44.0	43.6	67.7	63.1	51.5	74.3	56.3
DANN [38]		45.6	59.3	70.1	47.0	58.5	60.9	46.1	43.7	68.5	63.2	51.8	76.8	57.6
JAN [99]		45.9	61.2	68.9	50.4	59.7	61.0	45.8	43.4	70.3	63.9	52.4	76.8	58.3
SE [31]		48.8	61.8	72.8	54.1	63.2	65.1	50.6	49.2	72.3	66.1	55.9	78.7	61.5
CDAN-RM [97]		49.2	64.8	72.9	53.8	63.9	62.9	49.8	48.8	71.5	65.8	56.4	79.2	61.6
CDAN-M [97]		**50.6**	65.9	73.4	55.7	62.7	64.2	51.8	**49.1**	74.5	68.2	**56.9**	80.7	62.8
DWT-MEC		50.3	**72.1**	**77.0**	**59.6**	**69.3**	**70.2**	**58.3**	48.1	**77.3**	**69.3**	53.6	**82.0**	**65.6**

Table 2.3: Accuracy(%) on Office-Home dataset with Resnet-50 as base network and comparison with the state-of-the-art methods.

2.4.3.2 Comparison with State-of-the-Art Methods

In this section we present our results and compare with previous UDA methods. Tab. 2.2 reports the results obtained on the digits datasets. We compare with several baselines: Correlation Alignment (CORAL) [151], Simultaneous Deep Transfer (MMD) [158], Domain-Adversarial Training of Neural Networks (DANN) [38], Domain separation networks [11], Coupled generative adversarial net-works (CoGAN) [94], Adversarial discriminative domain adaptation (ADDA) [160], Deep reconstruction-classification networks (DRCN), [41], Asymmetric tri-training [140], Associative domain adaptation (ADA) [51], AutoDIAL [13], SBADA-GAN [136], Domain transfer through deep activation matching (GAM) [62], Minimal-entropy correlation alignment (MECA) [111] and SE [31]. Note that the Virtual Adversarial Domain Adaptation (VADA) [148] use a different network, thus cannot be compared with the other

	Source	CIFAR-10	STL
	Target	STL	CIFAR-10
Source Only		60.35	51.88
w/o augmentation			
DANN [38]		66.12	56.91
DRCN [41]		66.37	58.65
AutoDIAL [13]		79.10	70.15
DWT		**79.75**±0.25	**71.18**±0.56
Target Only		67.75	88.86
w/ augmentation			
SE [a] [31]		77.53±0.11	71.65±0.67
SE [b] [31]		80.09±0.31	69.86±1.97
DWT-MEC[b]		80.39±0.31	**72.52**±0.94
DWT-MEC (MT)[b]		**81.83**±0.14	71.31±0.22

Table 2.4: Accuracy (%) on the CIFAR-10↔STL: comparison with state of the art. [a] indicates minimal data augmentation and [b] considers augmented source and target data.

methods (including ours) which are based on a different capacity network. For this reason, [148] is not reported in Tab. 2.2. Results associated with each method are taken from the corresponding papers. We re-implemented SE as the numbers reported in the original paper [31] refer to different network architectures.

Tab. 2.2 is split in two sections, separating those methods that exploit data augmentation from those which use only the original training data. Compared with no-data augmentation methods, our DWT performs better than previous UDA methods in the three settings. Our method is less effective in the MNIST→SVHN due to the strong domain shift between the two domains. In this setting, GAN-based methods [136] are more effective. Looking at methods which consider data augmentation, we compare our approach with SE [31]. To be consistent with other methods, we plug the architectures described in [36] in SE. Comparing the proposed approach with our re-implementation of SE (SE[†b]) we observe that DWT-MEC (MT) is almost on par with SE in the MNIST↔USPS setting and better than SE in the SVHN→MNIST. For the sake of completeness, we also report the performance of SE taken from the original paper [31], considering SE with minimal augmentation (only gaussian blur) and SE with full augmentation.

With the rapid progress of deep DA methods, the results in the digits datasets have saturated. This makes it difficult to gauge the merit of the proposed contributions. Therefore, we also consider the CIFAR10 $\leftrightarrow$ STL setting. Our results are reported in Tab. 2.4. Similarly to the experiments in Tab. 2.2, we separate those methods exploiting data augmentation from those not using target-sample perturbations. Tab. 2.4 shows that our method (DWT), outperforms all previous baselines which also do not consider augmentation. Furthermore, by exploiting data perturbation and the proposed MEC loss our approach (with and without Mean-Teacher) reaches higher accuracy than SE.[1]

Finally, we also perform experiments on the large-scale Office-Home dataset and we compare with the baselines methods as reported by Long *et al.* [97]. The results reported in Tab. 2.3 show that our approach outperforms all the other methods. On average, the proposed approach improves over Conditional Domain Adversarial Networks (CDAN) by 2.8% and it is also more accurate than SE.

2.5 Conclusions

In this chapter we addressed UDA by proposing domain-specific feature whitening with DWT layers and the MEC loss. On the one hand, whitening of intermediate features enables the alignment of the source and the target distributions at intermediate feature levels and increases the smoothness of the loss landscape. On the other hand, our MEC loss better exploits the target data. Both these components can be easily integrated in any standard CNN. Our experiments on standard benchmarks show state-of-the-art performance on digits categorization and object recognition tasks. As future work, we plan to extend our method to handle multiple source and target domains.

[1] In this case the accuracy values reported for SE are taken directly from the original paper as the underlying network architecture is the same.

3

TriGAN for Multi-source Domain Adaptation

Most domain adaptation methods consider the problem of transferring knowledge to the target domain from a single source dataset. However, in practical applications, we typically have access to multiple sources. In this chapter we propose the first approach for Multi-Source Domain Adaptation (MSDA) based on Generative Adversarial Networks. Our method is inspired by the observation that the appearance of a given image depends on three factors: the *domain*, the *style* (characterized in terms of low-level features variations) and the *content*. For this reason, we propose to project the source image features onto a space where only the dependence from the content is kept, and then re-project this invariant representation onto the pixel space using the target domain and style. In this way, new labeled images can be generated which are used to train a final target classifier. We test our approach using common MSDA benchmarks, showing that it outperforms state-of-the-art methods. [1]

3.1 Introduction

A well known problem in computer vision is the need to adapt a classifier trained on a given *source* domain in order to work on a different, *target* domain. Since the two domains typically

[1] The content of this chapter is based on the MVAP 2021 paper [132]

have different marginal feature distributions, the adaptation process needs to reduce the corresponding *domain shift* [157]. In many practical scenarios, the target data are not annotated and Unsupervised Domain Adaptation (UDA) methods are required.

While most previous adaptation approaches consider a single source domain, in real world applications we may have access to multiple datasets. In this case, Multi-Source Domain Adaptation (MSDA) methods [109, 121, 169, 175] may be adopted, in which more than one source dataset is considered in order to make the adaptation process more robust. However, despite more data can be used, MSDA is challenging as multiple domain-shift problems need to be simultaneously and coherently solved.

In this chapter we deal with (unsupervised) MSDA using a data-augmentation approach based on a Generative Adversarial Network (GAN) [45]. Specifically, we generate artificial target samples by "translating" images from all the source domains into target-like images. Then the synthetically generated images are used for training the target classifier. While this strategy has been recently adopted in the single-source UDA scenario [61, 94, 113, 136, 145], we are the first to show how it can be effectively used in a MSDA setting. In more detail, our goal is to build and train a "universal" translator which can transform an image from an input domain to a target domain. The translator network is "universal" because it is not specific for a given source dataset but can transform images from multiple source domains into the target domain, given a domain label as input (see Fig. 3.1). The proposed translator is based on an *encoder*, which extracts domain-invariant intermediate features, and a *decoder*, which projects these features onto the domain-specific target distribution.

To make this image translation effective, we assume that the appearance of an image depends on three factors: the *content*, the *domain* and the *style*. The *domain* models properties that are shared by the elements of a dataset but which may not be shared by other datasets. On the other hand, the *style* factor represents properties which are shared among different *local* parts of *a single image* and describes low-level features which concern a specific image (e.g., the color or the texture). The *content* is the semantics that we want to keep unchanged during the translation process: typically, it is the foreground object shape which corresponds to the image label associated with each source data sample. Our encoder obtains the intermediate representations in a two-step process: we first generate style-invariant representations and then we compute the domain-invariant representations. Symmetrically, the decoder transforms the intermediate representations, first projecting these features onto a domain-specific distribution and then onto a style-specific distribution. In order to modify the underlying distribution of a set of features, inspired by [131], in the encoder we use *whitening* layers which progressively align the style-and-domain feature distributions. Then, in the decoder, we project the intermediate invariant representation onto a new domain-and-style specific distribution with *Whitening and Coloring* (WC) [150] batch transformations, according to the target data.

A "universal" translator similar in spirit to our proposed generator is StarGAN [22]. The goal of StarGAN is pure image translation and it is not used for discriminative tasks (e.g., UDA tasks). The main advantage of a universal translator with respect to train N specific source-to-target translators (being N the number of source domains) is that the former can jointly use all the source datasets, thus alleviating the risk of overfitting [22]. However, differently from our proposed generator, in StarGAN the domain label is represented by a one-hot vector concatenated with the input image. As shown in [150] this procedure is less effective than using domain-label conditioned batch transforms. We empirically show that, when we use StarGAN in our MSDA scenario, the synthesized images are much less effective for training the target classifier, which confirms that our batch-based transformations of the image distribution are more effective for our translation task.

Contributions. Our main contributions can be summarized as follows. (i) We propose the first generative MSDA method. We call our approach TriGAN because it is based on representing the image appearance using three different factors: the style, the domain and the content. (ii) The proposed image translation process is based on style and domain specific statistics which are first removed from and then added to the source images by means of modified WC layers. Specifically, we use the following feature transformations (associated with a corresponding layer type): Instance Whitening Transform (IWT), Domain Whitening Transform (DWT) [131], conditional Domain Whitening Transform ($cDWT$) and Adaptive Instance Whitening Transform ($AdaIWT$). IWT and $AdaIWT$ are novel layers introduced in this chapter. (iii) We test our method on two MSDA datasets, Digits-Five [169] and Office-Caltech10 [44], outperforming state-of-the-art methods.

3.2 Related Work

In this section we review the previous approaches on UDA, considering both single source and multi-source methods. Since the proposed generator is also related to deep models used for image-to-image translation, we also analyse related work on this topic.

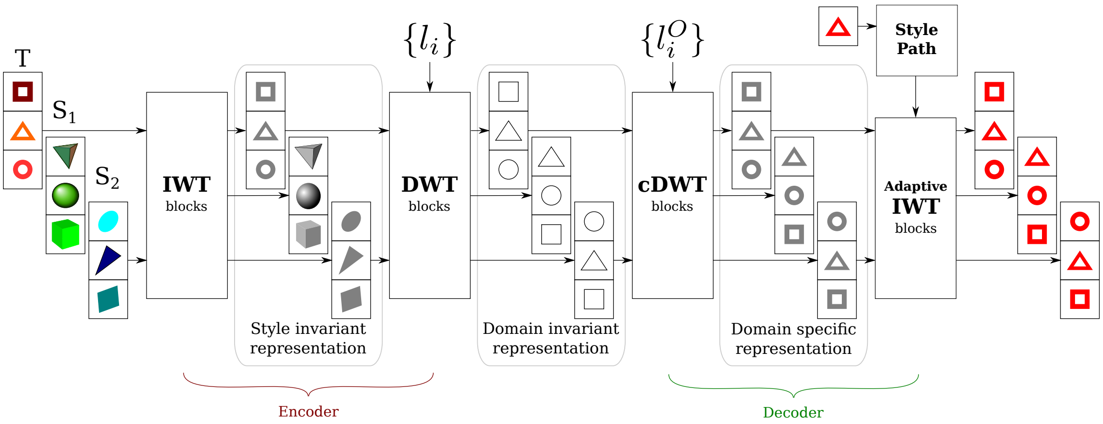


Figure 3.1: An overview of the TriGAN generator. We schematically show 3 domains $\{T, S_1, S_2\}$ - objects with *holes*, *3D objects* and *skewed* objects, respectively. The content is represented by the object's shape - square, circle or triangle. The style is represented by the color: each image input to $\mathcal{G}$ has a different color and each domain has its own set of styles. First, the encoder $\mathcal{E}$ creates a style-invariant representation using IWT blocks. DWT blocks are then used to obtain a domain-invariant representation. Symmetrically, the decoder $\mathcal{D}$ brings back domain-specific information with cDWT blocks (for simplicity we show only a single output domain, T). Finally, we apply a reference style. The reference style is extracted using the style path and it is applied using the Adaptive IWT blocks. In this figure, l_i and l_i^o denote, respectively, the input and the output domain labels.

Single-source UDA. Single-source UDA approaches assume a single labeled source domain and can be broadly classified under three main categories, depending on the strategy adopted to cope with the domain-shift problem. The first category uses first and second order statistics to model the source and the target feature distributions. For instance, [98, 99, 161, 163] minimize the Maximum Mean Discrepancy, *i.e.* the distance between the mean of feature distributions between the two domains. On the other hand, [111, 124, 152] achieve domain invariance by aligning the second-order statistics through correlation alignment. Differently, [13, 88, 108] reduce the domain shift by domain alignment layers derived from batch normalization (BN) [66]. This idea has been recently extended in [131], where grouped-feature whitening (DWT) is used instead of feature standardization as in BN. In our proposed encoder we also use the DWT layers, which we adapt to work in a generative network. In addition, we also propose other style and domain dependent batch-based normalizations (i.e., IWT, $cDWT$ and $AdaIWT$).

The second category of methods computes domain-agnostic representations by means of an adversarial learning based approach. For instance, discriminative domain-invariant representations are constructed through a gradient reversal layer in [36]. Similarly, Tzeng et al. [159] use a domain confusion loss and a domain discriminator to align the source and the target domain.

The third category of methods uses adversarial learning in a generative framework (i.e., GANs [45]) to create artificial source and/or target images and perform domain adaptation. Notable approaches are SBADA-GAN [136], CyCADA [61], CoGAN [94], I2I Adapt [113] and Generate To Adapt (GTA) [145]. While these generative methods have been shown to be very successful in UDA, none of them deals with a multi-source setting. Note that trivially extending these approaches to an MSDA scenario involves training N different generators, being N the number of source domains. In contrast, in our universal translator, only a subset of parameters grow linearly with the number of domains (Sec. 3.3.2.3), while the others are shared over all the domains. Moreover, since we train our generator using $(N+1)^2$ translation directions, we can largely increase the number of training sample-domain pairs effectively used (Sec. 3.3.3).

Multi-source UDA. In [175], multiple-source knowledge transfer is obtained by borrowing knowledge from the target *k* nearest-neighbour sources. Similarly, a distribution weighted combining rule is proposed in [109] to construct a target hypothesis as a weighted combination of source hypotheses. Recently, Deep Cocktail Network (DCTN) [169] uses the distribution-weighted combining rule in an adversarial setting. A Moment Matching Network (M^{3}SDA) is introduced in [121] to reduce the discrepancy between the multiple-source and the target domains. Zhao et al. [180] investigate multi-source domain adaptation for segmentation tasks, while Rakshit et al. [127] adversarially train an ensemble of source domain classifiers in order to align the source domains to each other. Adversarial training is used also in [181], where the authors propose to use the Wasserstein distance between the source samples and the target distribution in order to select those samples which are the closest to the target domain.

Differently from these methods which operate in a discriminative setting, we propose the first generative approach for an MSDA scenario, where the target dataset is populated with artificial "translations" of the source images.

Image-to-image Translation. Image-to-image translation approaches, i.e. those methods which transform an image from one domain to another, possibly keeping its semantics, are the basis of our method. In [67] a U-Net network translates images under the assumption that paired images in the two domains are available at training time. In contrast, CycleGAN [183] can learn to translate images using unpaired training samples. Note that, by design, these methods work with two domains. ComboGAN [2] partially alleviates this issue by using N generators for translations among N domains. Our work is also related to StarGAN [22] which handles unpaired image translation amongst *N* domains (*N* ≥ 2) through a single generator. However, StarGAN achieves image translation without explicitly forcing the image representations to be domain invariant, and this may lead to a significant reduction of the network representation power as the number of domains increases. On the other hand, our goal is to obtain an explicit, intermediate image representation which is style-and-domain independent. We use *IWT* and *DWT* to achieve this. We also show that this invariant representation can simplify the re-projection process onto a desired style and target domain. This is achieved through $AdaIWT$ and $cDWT$ which results into very realistic translations amongst domains. Very recently, a whitening and colouring based image-to-image translation method was proposed in [21], where the whitening operation is *weight-based*: the transformation is embedded into the network weights. Specifically, whitening is approximated by enforcing the convariance matrix, computed using the intermediate features, to be equal to the identity matrix. Conversely, our whitening transformation is *data dependent* (i.e., it depends on the specific batch statistics, Sec. 3.3.2.1) and uses the Cholesky decomposition [27] to compute the whitening matrices of the input samples in a closed form, thereby eliminating the need of additional ad-hoc losses. Finally, most related to this chapter, the work in [87] uses a whitening and coloring transform for the task of universal style transfer. However, different from Li *et al.* [87], we conduct domain-specific whitening transform, which yields improved results over when no domain assumptions are made.

Asides from traditional neural network based image translation models, alternatively, Yang *et al.* [174] proposed a light-weight image translation technique that relies on fourier transform to transfer style between domains. Since this is a concurrent work with this current chapter, it is outside the scope of comparison.

3.3 Style-and-Domain based Image Translation

In this section we describe the proposed approach for MSDA. We first provide an overview of our method and we introduce the notation adopted throughout the paper (Sec. 3.3.1). Then we describe the TriGAN architecture (Sec. 3.3.2) and our training procedure (Sec.3.3.3).

3.3.1 Notation and Overview

In the MSDA scenario we have access to N labeled source datasets $\{S_j\}_{j=1}^N$, where $S_j = \{(\mathbf{x}_k, y_k)\}_{k=1}^{n_j}$, and a target unlabeled dataset $T = \{\mathbf{x}_k\}_{k=1}^{n_t}$. All the datasets (target included) share the same semantic categories, and each of them is associated to a domain $\mathbf{D}_1^s, ..., \mathbf{D}_N^s, \mathbf{D}^t$, respectively. Our final goal is to build a classifier for the target domain $\mathbf{D}_t$ exploiting the data in $\{S_j\}_{j=1}^N \cup T$.

Our method is based on two separate training stages. We initially train a generator $\mathcal{G}$ which learns how to change the appearance of a real input image in order to adhere to a desired domain and style. $\mathcal{G}$ learns $(N+1)^2$ mappings between every possible pair of image domains, in this way exploiting much more supervisory information with respect to a plain strategy in which N different source-to-target generators are separately trained [22] (Sec. 3.3.3). Once $\mathcal{G}$ is trained, in the second stage we use it to generate target data having the same content of the source data, thus creating a new, *labeled*, target dataset, which is finally used to train a target classifier $\mathcal{C}$. However, in training $\mathcal{G}$ (first stage), we do not use class labels and T is treated in the same way as the other datasets.

As mentioned in Sec. 3.1, $\mathcal{G}$ is composed of an encoder $\mathcal{E}$ and a decoder $\mathcal{D}$ (Fig. 3.1). The role of $\mathcal{E}$ is to "whiten", *i.e.*, to remove, both domain-specific and style-specific aspects of the input image features in order to obtain domain and style invariant representations. Symmetrically, $\mathcal{D}$ "colors" the domain-and-style invariant features generated by $\mathcal{E}$, by progressively projecting these intermediate representations onto a domain-and-style specific space.

In the first training stage, $\mathcal{G}$ takes as input a batch of images $B = \{\mathbf{x}_1, ..., \mathbf{x}_m\}$ with corresponding *domain* labels $L = \{l_1, ..., l_m\}$, where $\mathbf{x}_i$ belongs to the domain $\mathbf{D}_{l_i}$ and $l_i \in \{1, ..., N+1\}$. Moreover, $\mathcal{G}$ takes as input a batch of output domain labels $L^O = \{l_1^O, ..., l_m^O\}$, and a batch of reference style images $B^O = \{\mathbf{x}_1^O, ..., \mathbf{x}_m^O\}$, such that $\mathbf{x}_i^O$ has domain label l_i^O. For a given $\mathbf{x}_i \in B$, the task of $\mathcal{G}$ is to transform $\mathbf{x}_i$ into $\hat{\mathbf{x}}_i$ such that: (1) $\mathbf{x}_i$ and $\hat{\mathbf{x}}_i$ share the same content but (2) $\hat{\mathbf{x}}_i$ belongs to domain $\mathbf{D}_{l_i^O}$ and has the same style of $\mathbf{x}_i^O$.

3.3.2 TriGAN Architecture

The TriGAN architecture is composed of a generator network $\mathcal{G}$ and a discriminator network $\mathcal{D}_{\mathcal{P}}$. As above mentioned, $\mathcal{G}$ comprises an encoder $\mathcal{E}$ and decoder $\mathcal{D}$, which we describe in

(Sec. 3.3.2.2-3.3.2.3). The discriminator $\mathcal{D}_{\mathcal{P}}$ is based on the Projection Discriminator [110]). Before describing the details of $\mathcal{G}$, we briefly review the WC transform [150]) (Sec. 3.3.2.1) which is used as the basic operation in our proposed batch-based feature transformations.

3.3.2.1 Preliminaries: Whitening $\&$ Coloring Transform

Let $F(\mathbf{x}) \in \mathbb{R}^{h \times w \times d}$ be the tensor representing the activation values of the convolutional feature maps in a given layer corresponding to the input image $\mathbf{x}$, with d channels and $h \times w$ spatial locations. We treat each spatial location as a d-dimensional vector, thus, each image $\mathbf{x}_i$ contains a set of vectors $X_i = \{\mathbf{v}_1, ..., \mathbf{v}_{h \times w}\}$. With a slight abuse of the notation, we use $B = \bigcup_{i=1}^{m} X_i = \{\mathbf{v}_1, ..., \mathbf{v}_{h \times w \times m}\}$, which includes all the spatial locations in all the images in a batch. The WC transform is a multivariate extension of the per-dimension normalization and shift-scaling transform (BN) proposed in [66] and widely adopted in both generative and discriminative networks. WC can be described by:

$$WC(\mathbf{v}_j; B, \boldsymbol{\beta}, \boldsymbol{\Gamma}) = Coloring(\bar{\mathbf{v}}_j; \boldsymbol{\beta}, \boldsymbol{\Gamma}) = \boldsymbol{\Gamma}\bar{\mathbf{v}}_j + \boldsymbol{\beta} \tag{3.1}$$

where:

$$\bar{\mathbf{v}}_j = Whitening(\mathbf{v}_j; B) = \boldsymbol{W}_B(\mathbf{v}_j - \boldsymbol{\mu}_B). \tag{3.2}$$

In Eq. 3.2, $\boldsymbol{\mu}_B$ is the centroid of the elements in B, while $\boldsymbol{W}_B$ is such that: $\boldsymbol{W}_B^\top \boldsymbol{W}_B = \boldsymbol{\Sigma}_B^{-1}$, where $\boldsymbol{\Sigma}_B$ is the covariance matrix computed using B. The result of applying Eq. 3.2 to the elements of B, is a set of *whitened* features $\bar{B} = \{\bar{\mathbf{v}}_1, ..., \bar{\mathbf{v}}_{h \times w \times m}\}$, which lie in a spherical distribution (*i.e.*, with a covariance matrix equal to the identity matrix). On the other hand, Eq. 3.1 performs a *coloring* transform, *i.e.* projects the elements in $\bar{B}$ onto a learned multivariate Gaussian distribution. While $\boldsymbol{\mu}_B$ and $\boldsymbol{W}_B$ are computed using the elements in B (they are data-dependent), Eq. 3.1 depends on the d dimensional learned parameter vector $\boldsymbol{\beta}$ and on the $d \times d$ dimensional learned parameter matrix $\boldsymbol{\Gamma}$. Eq. 3.1 is a linear operation and can be simply implemented using a convolutional layer with kernel size $1 \times 1 \times d$. We refer to [150] for more details on how WC can be efficiently implemented.

In this chapter we use the WC transform in our encoder $\mathcal{E}$ and decoder $\mathcal{D}$, in order to first obtain a style-and-domain invariant representation for each $\mathbf{x}_i \in B$, and then transform this representation accordingly to the desired output domain $\mathbf{D}_{l_i^O}$ and style image sample $\mathbf{x}_i^O$. The next sub-sections show the details of the proposed architecture.

3.3.2.2 Encoder

The encoder $\mathcal{E}$ is composed of a sequence of standard $Convolution_{k \times k}$ - $Normalization$ - $ReLU$ - *Average Pooling* blocks and some $ResBlocks$ (more details in the Supplementary Material), in which we replace the common BN layers [66] with our proposed normalization modules, which are detailed below.

Obtaining Style Invariant Representations. In the first two blocks of $\mathcal{E}$ we whiten the first and second-order statistics of the low-level features of each $X_i \subseteq B$, which are mainly responsible for the *style* of an image [39]. To do so, we propose the *Instance Whitening Transform* (IWT), where the term *instance* is inspired by Instance Normalization (IN) [162] and highlights that the proposed transform is applied to a set of features extracted from a single image $\mathbf{x}_i$. Specifically, for each $\mathbf{v}_j \in X_i$, $IWT(\mathbf{v}_j)$ is defined as:

$$IWT(\mathbf{v}_j) = WC(\mathbf{v}_j; X_i, \boldsymbol{\beta}, \boldsymbol{\Gamma}). \tag{3.3}$$

Note that in Eq. 3.3 we use X_i as the batch, where X_i contains only feautures of a specific image $\mathbf{x}_i$ (Sec. 3.3.2.1). Moreover, each $\mathbf{v}_j \in X_i$ is extracted from the first two convolutional layers of $\mathcal{E}$, thus $\mathbf{v}_j$ has a small receptive field. This implies that whitening is performed using an *image-specific* feature centroid $\boldsymbol{\mu}_{X_i}$ and covariance matrix $\boldsymbol{\Sigma}_{X_i}$, which represent the first and second-order statistics of the low-level features of $\mathbf{x}_i$. On the other hand, coloring is based on the parameters $\boldsymbol{\beta}$ and $\boldsymbol{\Gamma}$, which *do not depend* on $\mathbf{x}_i$ or l_i. The coloring operation is the analogous of the shift-scaling per-dimension transform computed in BN just after feature standardization [66] and is necessary to avoid decreasing the network representation capacity [150].

Obtaining Domain Invariant Representations. In the subsequent blocks of $\mathcal{E}$ we whiten the first and second-order statistics which are *domain specific*. For this operation we adopt the *Domain Whitening Transform* (DWT) proposed in [131]. Specifically, for each $X_i \subseteq B$, let l_i be its domain label (see Sec. 3.3.1) and let $B_{l_i} \subseteq B$ be the subset of features which have been extracted from *all* those images in B which share *the same domain label* l_i. Then, for each $\mathbf{v}_j \in B_{l_i}$:

$$DWT(\mathbf{v}_j) = WC(\mathbf{v}_j; B_{l_i}, \boldsymbol{\beta}, \boldsymbol{\Gamma}). \tag{3.4}$$

Similarly to Eq. 3.3, Eq. 3.4 performs whitening using a subset of the current feature batch. Specifically, all the features in B are partitioned depending on the domain label of the image they have been extracted from, so obtaining $B_1, B_2, ...$, etc, where all the features in B_l belong to images of the domain $\mathbf{D}_l$. Then, B_l is used to compute domain-dependent first and second order statistics ($\boldsymbol{\mu}_{B_l}, \boldsymbol{\Sigma}_{B_l}$). These statistics are used to project each $\mathbf{v}_j \in B_l$ onto a domain-invariant spherical distribution. A similar idea was recently proposed in [131] in a discriminative network for single-source UDA. However, differently from [131], we also use coloring by re-projecting the whitened features onto a new space governed by a learned multivariate distribution. This is done using the (layer-specific) parameters $\boldsymbol{\beta}$ and $\boldsymbol{\Gamma}$ which do not depend on l_i.

3.3.2.3 Decoder

Our decoder $\mathcal{D}$ is functionally and structurally symmetric with respect to $\mathcal{E}$: it takes as input the domain and style invariant features computed by $\mathcal{E}$ and projects these features onto the desired domain $\mathbf{D}_{l_i^O}$ with the style extracted from the reference image $\mathbf{x}_i^O$.

Similarly to $\mathcal{E}$, $\mathcal{D}$ is a sequence of $ResBlocks$ and a few $Upsampling$ - $Normalization$ - $ReLU$ - $Convolution_{k \times k}$ blocks (more details in the Supplementary Material). Similarly to Sec. 3.3.2.2, in the $Normalization$ layers we replace BN with our proposed feature normalization approaches, which are detailed below.

Projecting Features onto a Domain-specific Distribution. Apart from the last two blocks of $\mathcal{D}$ (see below), all the other blocks are dedicated to project the current set of features onto a domain-specific subspace. This subspace is learned from data using domain-specific coloring parameters $(\boldsymbol{\beta}_l, \boldsymbol{\Gamma}_l)$, where l is the label of the corresponding domain. To this purpose we introduce the *conditional Domain Whitening Transform* (*cDWT*), where the term "conditional" specifies that the coloring step is conditioned on the domain label l. In more detail: Similarly to Eq. 3.4, we first partition B into $B_1, B_2, ...$, etc. However, the membership of $\mathbf{v}_j \in B$ to B_l is decided taking into account the *desired output* domain label l_i^O for each image rather than its original domain as in case of Eq. 3.4. Specifically, if $\mathbf{v}_j \in X_i$ and the output domain label of X_i is l_i^O, then $\mathbf{v}_j$ is included in $B_{l_i^O}$. Once B has been partitioned, we define $cDWT$ as follows:

$$cDWT(\mathbf{v}_j) = WC(\mathbf{v}_j; B_{l_i^O}, \boldsymbol{\beta}_{l_i^O}, \boldsymbol{\Gamma}_{l_i^O}). \tag{3.5}$$

Note that, after whitening, and differently from Eq. 3.4, coloring in Eq. 3.5 is performed using *domain-specific* parameters $(\boldsymbol{\beta}_{l_i^O}, \boldsymbol{\Gamma}_{l_i^O})$.

Applying a Specific Style. In order to apply a specific style to $\mathbf{x}_i$, we first extract the output style from the reference image $\mathbf{x}_i^O$ associated with $\mathbf{x}_i$ (Sec. 3.3.1). This is done using the *Style Path* (see Fig. 3.1), which consists of two $Convolution_{k \times k}$ - IWT - $ReLU$ - *Average Pooling* blocks (which share the parameters with the first two layers of the encoder) and a MultiLayer Perceptron (MLP) $\mathcal{F}$. Following [39] we represent a style using the first and the second order statistics $\boldsymbol{\mu}_{X_i^O}, \boldsymbol{W}_{X_i^O}^{-1}$, which are extracted using the IWT blocks (Sec. 3.3.2.2). Then we use $\mathcal{F}$ to adapt these statistics to the domain-specific representation obtained as the output of the previous step. In fact, in principle, for each $\mathbf{v}_j \in X_i^O$, the $Whitening()$ operation inside the IWT transform could be "inverted" using:

$$Coloring(\mathbf{v}_j; \boldsymbol{\mu}_{X_i^O}, \boldsymbol{W}_{X_i^O}^{-1}). \tag{3.6}$$

Indeed, the coloring operation (Eq. 3.1) is the inverse of whitening (Eq. 3.2). However, the elements of X_i now lie in a feature space which is different from the output space of Eq. 3.3, thus the transformation defined by *Style Path* needs to be adapted. For this reason, we use a MLP ($\mathcal{F}$) which implements this adaptation:

$$[\boldsymbol{\beta}_i \| \boldsymbol{\Gamma}_i] = \mathcal{F}([\boldsymbol{\mu}_{X_i^O} \| \boldsymbol{W}_{X_i^O}^{-1}]). \tag{3.7}$$

Note that, in Eq. 3.7, $[\boldsymbol{\mu}_{X_i^O} \| \boldsymbol{W}_{X_i^O}^{-1}]$ is the (concatenated) input and $[\boldsymbol{\beta}_i \| \boldsymbol{\Gamma}_i]$ is the MLP output, one input-output pair per image $\mathbf{x}_i^O$.

Once $(\boldsymbol{\beta}_i, \boldsymbol{\Gamma}_i)$ have been generated, we use them as the coloring parameters of our *Adaptive IWT* ($AdaIWT$):

$$AdaIWT(\mathbf{v}_j) = WC(\mathbf{v}_j; X_i^O, \boldsymbol{\beta}_i, \boldsymbol{\Gamma}_i). \tag{3.8}$$

Eq. 3.8 imposes style-specific first and second order statistics to the features of the last blocks of $\mathcal{D}$ in order to mimic the style of $\mathbf{x}_i^O$.

3.3.3 Network Training

GAN Training. For the sake of clarity, in the rest of the paper we use a simplified notation for $\mathcal{G}$, in which $\mathcal{G}$ takes as input only one image instead of a batch. Specifically, let $\hat{\mathbf{x}}_i = \mathcal{G}(\mathbf{x}_i, l_i, l_i^O, \mathbf{x}_i^O)$ be the generated image, starting from $\mathbf{x}_i$ ($\mathbf{x}_i \in \mathbf{D}_{l_i}$) and with desired output domain l_i^O and style image $\mathbf{x}_i^O$. $\mathcal{G}$ is trained using the combination of three different losses, with the goal of changing the style and the domain of $\mathbf{x}_i$ while preserving its content.

First, we use an *adversarial loss* based on the Projection Discriminator [110] ($\mathcal{D}_{\mathcal{P}}$), which is conditioned on the input labels (i.e., domain labels, in our case) and uses a hinge loss:

$$\mathcal{L}_{cGAN}(\mathcal{G}) = -\mathcal{D}_{\mathcal{P}}(\hat{\mathbf{x}}_i, l_i^O) \tag{3.9}$$

$$\begin{aligned} \mathcal{L}_{cGAN}(\mathcal{D}_{\mathcal{P}}) = \max(0, 1 + \mathcal{D}_{\mathcal{P}}(\hat{\mathbf{x}}_i, l_i^O)) \\ + \max(0, 1 - \mathcal{D}_{\mathcal{P}}(\mathbf{x}_i, l_i)) \end{aligned} \tag{3.10}$$

The second loss is the *Identity loss* proposed in [183]), which in our framework is implemented as follows:

$$\mathcal{L}_{ID}(\mathcal{G}) = ||\mathcal{G}(\mathbf{x}_i, l_i, l_i, \mathbf{x}_i) - \mathbf{x}_i||_1. \tag{3.11}$$

In Eq. 3.11, $\mathcal{G}$ computes an identity transformation, being the input and the output domain and style the same. After that, a pixel-to-pixel $\mathcal{L}_1$ norm is computed.

Finally, we propose to use a third loss which is based on the rationale that the generation process should be *equivariant* with respect to a set of simple transformations which preserve the main content of the images (e.g., the foreground object shape). Specifically, we use the set of the affine transformations $\{h(\mathbf{x}; \boldsymbol{\theta})\}$ of image $\mathbf{x}$ which are defined by the parameter $\boldsymbol{\theta}$ ($\boldsymbol{\theta}$ is a 2D transformation matrix). The affine transformation is implemented by a differentiable bilinear kernel as in [68]. The *Equivariance loss* is:

$$\mathcal{L}_{Eq}(\mathcal{G}) = ||\mathcal{G}(h(\mathbf{x}_i; \boldsymbol{\theta}_i), l_i, l_i^O, \mathbf{x}_i^O) - h(\hat{\mathbf{x}}_i; \boldsymbol{\theta}_i)||_1. \tag{3.12}$$

In Eq. 3.12, for a given image $\mathbf{x}_i$, we randomly choose a geometric parameter $\boldsymbol{\theta}_i$ and we apply $h(\cdot; \boldsymbol{\theta}_i)$ to $\hat{\mathbf{x}}_i = \mathcal{G}(\mathbf{x}_i, l_i, l_i^O, \mathbf{x}_i^O)$. Then, using the same $\boldsymbol{\theta}_i$, we apply $h(\cdot; \boldsymbol{\theta}_i)$ to $\mathbf{x}_i$ and we get $\mathbf{x}_i' = h(\mathbf{x}_i; \boldsymbol{\theta}_i)$, which is input to $\mathcal{G}$ in order to generate a second image. The two generated images are finally compared using the $\mathcal{L}_1$ norm. This is a form of self-supervision, in which equivariance to geometric transformations is used to extract semantics. Very recently a similar loss has been proposed in [65], where equivariance to affine transformations is used for image co-segmentation.

The complete loss for $\mathcal{G}$ is:

$$\mathcal{L}(\mathcal{G}) = \mathcal{L}_{cGAN}(\mathcal{G}) + \lambda(\mathcal{L}_{Eq}(\mathcal{G}) + \mathcal{L}_{ID}(\mathcal{G})). \tag{3.13}$$

Note that Eq. 3.9, 3.10 and 3.12 depend on the pair $(\mathbf{x}_i, l_i^O)$: This means that the supervisory information we effectively use, grows with $O((N+1)^2)$, which is quadratic with respect to a plain strategy in which N different source-to-target generators are trained (Sec. 3.2).

Classifier Training. Once $\mathcal{G}$ is trained, we use it to artificially create a labeled training dataset (T^L) for the target domain. Specifically, for each S_j and each $(\mathbf{x}_i, y_i) \in S_j$, we randomly pick $\mathbf{x}_t \in T$, which is used as the reference style image, and we generate: $\hat{\mathbf{x}}_i = \mathcal{G}(\mathbf{x}_i, l_i, N+1, \mathbf{x}_t)$, where $N+1$ is fixed and indicates the target domain ($\mathbf{D}_t$) label (see Sec. 3.3.1). $(\hat{\mathbf{x}}_i, y_i)$ is added to T^L and the process is iterated. T^L is generated on the fly during the training of $\mathcal{C}$, and, every time that a given $(\mathbf{x}_i, y_i) \in S_j$ is selected, we randomly select a different reference style image $\mathbf{x}_t \in T$.

Finally, we train a classfier $\mathcal{C}$ on T^L using the cross-entropy loss:

$$\mathcal{L}_{Cls}(\mathcal{C}) = -\frac{1}{|T^L|} \sum_{(\hat{\mathbf{x}}_i, y_i) \in T^L} \log p(y_i | \hat{\mathbf{x}}_i). \tag{3.14}$$

Some previous works such as CyCADA [60] and SBADA-GAN [136] propose including the image classification loss in the unpaired translation stage, so that image content is not lost during the translation. While it is indeed possible to couple the $\mathcal{L}_{Cls}(\mathcal{C})$ loss in the image translation stage, making the whole process end-to-end, but we observe that our TriGAN generations do not suffer from such issue. It can be attributed due to the geometric constraints induced by the $\mathcal{L}_{Eq}$ loss and due to the well established intuition that image translation methods can solely translate the low-level textures without modifying the semantic content.

3.4 Experiments

In this section we describe the experimental setup and then we evaluate our approach using common MSDA datasets. We also present an ablation study in which we separately analyse the impact of each TriGAN component. In the Supplementary Material we show additional experiments in a single-source UDA scenario.

3.4.1 Datasets

In our experiments we consider two common domain adaptation benchmarks, namely the Digits-Five benchmark [169] and the Office-Caltech dataset [44].

Digits-Five [169] is composed of five different digit-recognition datasets: USPS [32], MNIST [82], MNIST-M [36], SVHN [115] and the Synthetic numbers dataset [38] (SYNDIGITS). SVHN [115] contains Google Street View images of real-world house numbers. Synthetic numbers [38] includes 500K computer-generated digits with different sources of variations (*i.e.* position, orientation, color, blur). USPS [32] is a dataset of digits scanned from U.S. envelopes, MNIST [82] is a popular benchmark for digit recognition and MNIST-M [36] is its colored counterpart. We adopt the experimental protocol described in [169]: in each domain the train/test split is composed of a subset of 25000 images for training and 9000 images for testing. For USPS, the entire dataset is used.

Office-Caltech [44] is a domain-adaptation benchmark, obtained selecting the subset of those 10 categories which are shared between Office31 and Caltech256 [49]. It contains 2533 images, about half of which belonging to Caltech256. There are four different domains: Amazon (A), DSLR (D), Webcam (W) and Caltech256 (C).

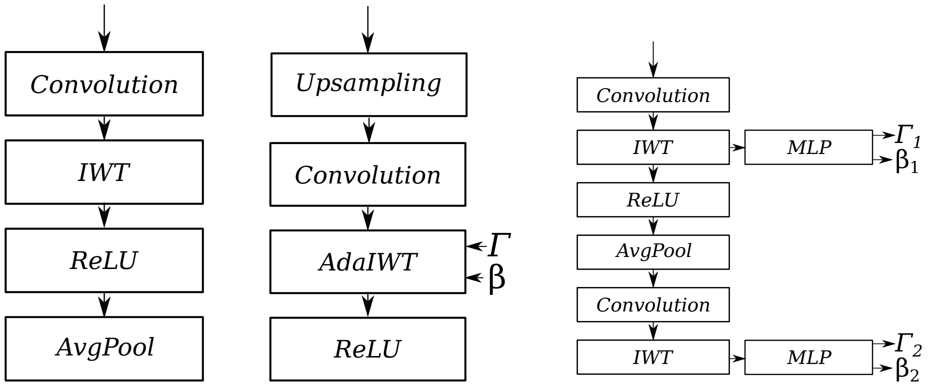
3.4.2 Implementation details

In this section we provide the architectural details of the TriGAN generator $\mathcal{G}$ and the discriminator $\mathcal{D}_{\mathcal{P}}$.

Instance Whitening Transform (IWT) blocks. As shown in Fig 3.2 (a), each **IWT** block is a sequence composed of: $Convolution_{k\times k} - IWT - ReLU - AvgPool_{m\times m}$, where k and m denote the kernel sizes. There are two **IWT** blocks in $\mathcal{E}$. In the first **IWT** block, we use $k = 5$ and $m = 2$, while in the second we use $k = 3$ and $m = 2$.

Adaptive Instance Whitening (AdaIWT) blocks. The **AdaIWT** blocks are analogous to the **IWT** blocks, except from the *IWT* layers which are replaced with *AdaIWT* layers. Specifically, the **AdaIWT** block is a sequence: $Upsampling_{m\times m} - Convolution_{k\times k} - AdaIWT - ReLU$, where $m = 2$ and $k = 3$. *AdaIWT* also takes as input the coloring parameters ($\mathbf{\Gamma}$, $\boldsymbol{\beta}$) (see Sec. 3.2.3 of the main paper and Fig. 3.2 (b)). Two **AdaIWT** blocks are consecutively used in $\mathcal{D}$. The last **AdaIWT** block is followed by a $Convolution_{5\times 5}$ layer.

Style Path. The **Style Path** is composed of: *Convolution* $_{5\times 5} - (IWT - MLP) - ReLU - AvgPool_{2\times 2} - Convolution_{3\times 3} - (IWT - MLP)$ (Fig. 3.2 (c)). The output of the **Style**



(a) IWT block **(b) AdaIWT** block **(c) Style Path**

Figure 3.2: A schematic representation of (a) the **IWT** block; (b) the **AdaIWT** block; and (c) the **Style Path**.

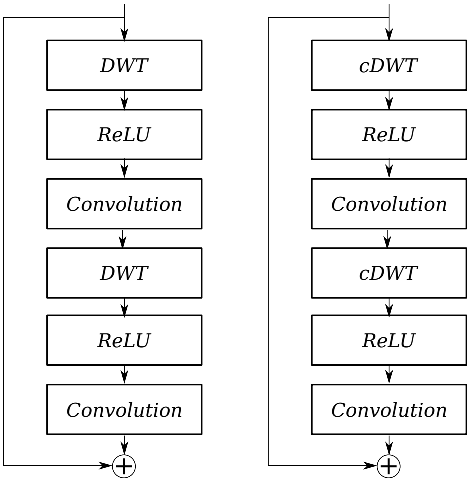
Path is $(\boldsymbol{\beta}_1 \| \boldsymbol{\Gamma}_1)$ and $(\boldsymbol{\beta}_2 \| \boldsymbol{\Gamma}_2)$, which are input to the second and the first **AdaIWT** blocks, respectively (see Fig. 3.2 (b)). The MLP is composed of five fully-connected layers with 256, 128, 128, 256 neurons, with the last fully-connected layer having a number of neurons equal to the cardinality of the coloring parameters $(\boldsymbol{\beta} \| \boldsymbol{\Gamma})$.

Domain Whitening Transform (**DWT**) **blocks**. The schematic representation of a **DWT** block is shown in Fig. 3.3 (a). For the **DWT** blocks we adopt a residual-like structure [57]: $DWT - ReLU - Convolution_{3\times3} - DWT - ReLU - Convolution_{3\times3}$. We also add identity shortcuts in the **DWT** residual blocks to aid the training process.

Conditional Domain Whitening Transform (**cDWT**) **blocks**. The proposed **cDWT** blocks are schematically shown in Fig. 3.3 (b). Similarly to a **DWT** block, a **cDWT** block contains the following layers: $cDWT - ReLU - Convolution_{3\times3} - cDWT - ReLU - Convolution_{3\times3}$. Identity shortcuts are also used in the **cDWT** residual blocks.

All the above blocks are assembled to construct $\mathcal{G}$, as shown in Fig. 3.4. Specifically, $\mathcal{G}$ contains two **IWT** blocks, one **DWT** block, one **cDWT** block and two **AdaIWT** blocks. It also contains the **Style Path** and 2 $Convolution_{5\times5}$ (one before the first **IWT** block and another after the last **AdaIWT** block), which is omitted in Fig. 3.4 for the sake of clarity. $\{\boldsymbol{\Gamma}_1, \boldsymbol{\beta}_1, \boldsymbol{\Gamma}_2, \boldsymbol{\beta}_2\}$ are computed using the **Style Path**.

For the discriminator $\mathcal{D}_{\mathcal{P}}$, we use a Projection Discriminator architecture [110]. In $\mathcal{D}_{\mathcal{P}}$ we use projection shortcuts instead of identity shortcuts. In Fig 3.5 we schematically show a



(a) DWT block **(b) cDWT** block

Figure 3.3: A schematic representation of (a) the **DWT** block; and (b) the **cDWT** block.

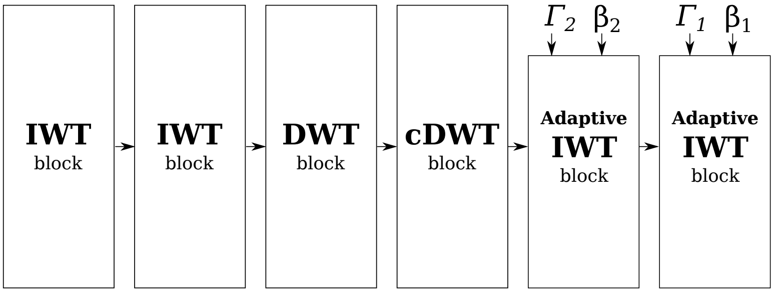


Figure 3.4: A schematic representation of the Generator $\mathcal{G}$.

discriminator block. $\mathcal{D}_{\mathcal{P}}$ is composed of 2 such blocks. We use spectral normalization [110] in $\mathcal{D}_{\mathcal{P}}$.

3.4.3 Experimental Setup

We train TriGAN for 100 epochs using the Adam optimizer [72] with the learning rate set to 1e-4 for $\mathcal{G}$ and 4e-4 for $\mathcal{D}_{\mathcal{P}}$ as in [59]. The loss weighing factor λ in Eq. 3.13 is set to 10 as in [183]. Since $\mathcal{L}_{Eq}$ can be seen as a replacement of the cycle loss in Zhu *et al.* [183], and both being L1-loss we also use the same λ value for $\mathcal{L}_{Eq}$, as used in the cycle loss.

In the Digits-Five experiments we use a mini-batch of size 256. Due to the difference in

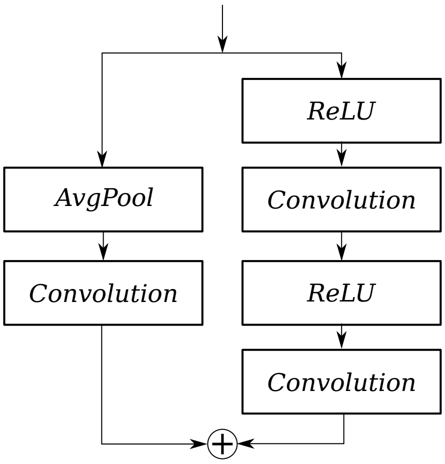


Figure 3.5: A schematic representation of the Discriminator $\mathcal{D}_{\mathcal{P}}$.

image resolution and image channels, the images of all the domains are converted to 32 × 32 RGB. For a fair comparison, for the final target classifier $\mathcal{C}$ we use exactly the same network architecture used in [38, 121].

In the Office-Caltech10 experiments we downsample the images to 164 × 164 to accommodate more samples in a mini-batch. We use a mini-batch of size 24 for training with 1 GPU. For the back-bone target classifier $\mathcal{C}$ we use the ResNet101 [57] architecture used in [121]. The weights are initialized with a network pre-trained on the ILSVRC-2012 dataset [135]. In our experiments we remove the output layer and we replace it with a randomly initialized fully-connected layer with 10 logits, one for each class of the Office-Caltech10 dataset. $\mathcal{C}$ is trained with Adam with an initial learning rate of 1e-5 for the randomly initialized last layer and 1e-6 for all other layers. In the Office-Caltech10 experiments, we also include $\{S_j\}_{j=1}^N$ in T^L when training $\mathcal{C}$.

3.4.4 Results

In this section we quantitatively analyse TriGAN (Sec. 3.4.4.1, 3.4.4.2 and 3.4.4.3) and we show some qualitative image-translation results (Sec. 3.4.4.4).

Protocol	Models	**mt, up, sv, sy → mm**	**mm, up, sv, sy → mt**	**mt, mm, sv, sy → up**	**mt, up, mm, sy → sv**	**mt, up, sv, mm → sy**	**Avg**
Source Combine	Source Only	63.70±0.83	92.30±0.91	90.71±0.54	71.51±0.75	83.44±0.79	80.33±0.76
	DAN[98]	67.87±0.75	97.50±0.62	93.49±0.85	67.80±0.84	86.93±0.93	82.72±0.79
	DANN[36]	70.81±0.94	97.90±0.83	93.47±0.79	68.50±0.85	87.37±0.68	83.61±0.82
Multi- Source	Source Only	63.37±0.74	90.50±0.83	88.71±0.89	63.54±0.93	82.44±0.65	77.71±0.81
	DAN[98]	63.78±0.71	96.31±0.54	94.24±0.87	62.45±0.72	85.43±0.77	80.44±0.72
	CORAL[151]	62.53±0.69	97.21±0.83	93.45±0.82	64.40±0.72	82.77±0.69	80.07±0.75
	DANN[36]	71.30±0.56	97.60±0.75	92.33±0.85	63.48±0.79	85.34±0.84	82.01±0.76
	ADDA[160]	71.57±0.52	97.89±0.84	92.83±0.74	75.48±0.48	86.45±0.62	84.84±0.64
	DCTN[169]	70.53±1.24	96.23±0.82	92.81±0.27	77.61±0.41	86.77±0.78	84.79±0.72
	M^{3}SDA[121]	72.82±1.13	**98.43**±0.68	**96.14**±0.81	81.32±0.86	89.58±0.56	87.65±0.75
	StarGAN [22]	44.71±1.39	96.26±0.62	55.32±3.71	58.93±1.95	63.36±2.41	63.71±2.01
	TriGAN (Ours)	**83.20**±0.78	97.20±0.45	94.08±0.92	**85.66**±0.79	**90.30**±0.57	**90.08**±0.70

Table 3.1: Classification accuracy (%) on **Digits-Five**. *MNIST-M, MNIST, USPS, SVHN* and *Synthetic Digits* are abbreviated as **mm**, **mt**, **up**, **sv** and **sy**, respectively. Each setting is denoted as a set of source domains (before the arrow) and a target domain (after the arrow). Following [121], we indicate with "Source Combine" the protocol in which all the source datasets are combined in one, hence performing a standard single-source domain adaptation task. Best values are in bold and second best values are underlined.

3.4.4.1 Comparison with State-of-the-Art Methods

Tab. 3.1 and Tab. 3.2 show the results on the Digits-Five and the Office-Caltech10 datset, respectively. In Tab. 3.1 and Tab. 3.2 the methods have been grouped under source-combine and multi-source categories. As the name suggest, source-combine do not assume the knowledge of source domain labels and combines all the source samples into a single domain. Whereas, in multi-source the knowledge about source domain labels are exploited. In Tab. 3.1 all the baselines have been taken from Peng *et al.* [121] and we report the numbers of our method averaged over three runs.

Tab. 3.1 shows that TriGAN achieves an average accuracy of 90.08% which is higher than all other methods. M^{3}SDA is better than TriGAN in the **mm**, **up**, **sv**, **sy** → **mt** and in the **mt**, **mm**, **sv**, **sy** → **up** settings, where TriGAN is the second best. In all the other settings, TriGAN outperforms all the other approaches. As an example, in the **mt**, **up**, **sv**, **sy** → **mm** setting, TriGAN is better than the second best method, M^{3}SDA, by a significant margin of 10.38%. In the same table we also show the results obtained when we replace TriGAN with StarGAN [22], which is another "universal" image translator (Sec. 3.2). Specifically, we use StarGAN to generate synthetic target images and then we train the target classifier using the same protocol described in Sec. 3.3.3. The corresponding results in Table 3.1 show that StarGAN, despite to be known to work well for aligned face translation, drastically fails when used in this UDA scenario.

Finally, we also use Office-Caltech10, which is considered to be difficult for generative-based UDA methods because of the high-resolution images. Although the dataset is quite saturated, TriGAN achieves a classification accuracy of 97.0%, outperforming all the other methods and beating the previous state-of-the-art approach (M^{3}SDA) by a margin of 0.6% on average (see Tab. 3.2).

Protocol	Models	All/W → W	All/D → D	All/C → C	All/A → A	Avg
Source Combine	Source only	99.0	98.3	87.8	86.1	92.8
	DAN [98]	99.3	98.2	89.7	94.8	95.5
Multi-Source	Source only	99.1	98.2	85.4	88.7	92.9
	DAN [98]	99.5	99.1	89.2	91.6	94.8
	DCTN [169]	99.4	99.0	90.2	92.7	95.3
	M^{3}SDA [121]	99.5	99.2	92.2	94.5	96.4
	StarGAN [22]	99.6	**100.0**	89.3	93.3	95.5
	TriGAN (Ours)	**99.7**	**100.0**	**93.0**	**95.2**	**97.0**

Table 3.2: Classification accuracy (%) on **Office-Caltech10**. The target domains are indicated as follows: Amazon (A), DSLR (D), Webcam (W) and Caltech256 (C). In each setting, the source domains are all the remaining datasets except the target dataset.

3.4.4.2 Ablation Study

In this section we analyse the different components of our method and study in isolation their impact on the final accuracy. Specifically, we use the Digits-Five dataset and the following models: i) Model **A**, which is our full model containing the following components: *IWT*, *DWT*, *cDWT*, *AdaIWT* and $\mathcal{L}_{Eq}$. ii) Model **B**, which is similar to Model **A** except we replace $\mathcal{L}_{Eq}$ with the cycle-consistency loss $\mathcal{L}_{Cycle}$ of CycleGAN [183]. iii) Model **C**, where we replace *IWT*, *DWT*, *cDWT* and *AdaIWT* of Model **A** with *IN* [162], *BN* [66], conditional Batch Normalization (*cBN*) [29] and Adaptive Instance Normalization (*AdaIN*) [64]. This comparison highlights the difference between feature whitening and feature standardization. iv) Model **D**, which ignores the style factor. Specifically, in Model **D**, the blocks related to the style factor, i.e., the *IWT* and the *AdaIWT* blocks, are replaced by *DWT* and *cDWT* blocks, respectively. v) Model **E**, in which the style path differs from Model **A** in the way the style is applied to the domain-specific representation. Specifically, we remove the MLP $\mathcal{F}(.)$ and we directly apply $(\boldsymbol{\mu}_{X_i^O}, \boldsymbol{W}^{-1}_{X_i^O})$. vi) Finally, Model **F** represents no-domain assumption (e.g. the DWT and cDWT blocks are replaced with standard WC blocks).

Model	Description	Avg. Accuracy (%) (Difference)
A	TriGAN (full method)	**90.08**
B	Replace Equivariance Loss with Cycle Loss	88.38 (-1.70)
C	Replace Whitening with Feature Standardisation	89.39 (-0.68)
D	No Style Assumption	88.32 (-1.76)
E	Applying style directly instead of style path	88.36 (-1.71)
F	No Domain Assumption	89.10 (-0.98)
StarGAN (Baseline)	No Style Assumption, Domain Labels concatenated with the input image	63.71 (-26.37)

Table 3.3: An analysis of the main TriGAN components using Digits-Five.

The results, reported in Tab. 3.3, show that all the components of the proposed generator play a role in the accuracy reached by the full model (**A**). Specifically, $\mathcal{L}_{Cycle}$ (Model **B**) is detrimental for the accuracy because $\mathcal{G}$ may focus on semantically meaningless details when reconstructing-back the image. Conversely, the affine transformations used in case of $\mathcal{L}_{Eq}$, force $\mathcal{G}$ to focus on the shape (i.e., the content) of the images. Also Model **C** is outperformed by model **A**, demonstrating the importance of feature whitening with respect to feature standardisation, corroborating the findings of [131] in a pure-discriminative scenario. Moreover, the no-style assumption in Model **D** hurts the classification accuracy by a margin of 1.76% when compared with Model **A**. We believe this is due to the fact that, when instance-specific style information is missing in the image translation process, then the diversity of the translations decreases, consequently reducing the final accuracy. In fact, we remind that, when T^L is created (Sec. 3.3.3), for the same $(\mathbf{x}_i, y_i) \in S_j$, we randomly select multiple, different reference style images $\mathbf{x}_t \in T$: this diversity cannot be obtained if only domain-specific latent factors are modeled. Finally, Model **E** and Model **F** show the importance of the proposed style path and the domain factor, respectively.

Note that the ablation analysis in Tab. 3.3 is done by removing a single component from the full model **A**, and the marginal differences with respect to Model **A** show that all the components are important. On the other hand, simultaneously removing all the components makes our model become similar to StarGAN, which we also report in Tab. 3.3 as a baseline comparison. In fact, in StarGAN, there is no style information and the domain labels are concatenated with the input image (Sec. 3.2). Conversely, our main contribution is the generation of intermediate invariant features and their re-projection onto the target domain and style distribution. This is obtained using a combination of blocks which extend WC [150] in order to first remove and then re-introduce style and domain specific statistics. The difference between the use of StarGAN and the use of Model **A** to populate of T^L (-26.37%) empirically shows that the proposed image translation approach is effective in a MSDA scenario.

3.4.4.3 Multi domain image-to-image translation

Our proposed generator can be used for a pure generative (non-UDA), multi-domain image-to-image translation task. We conduct experiments on the Alps Seasons dataset [2] which consists of images of Alps mountains with 4 different domains (corresponding to 4 seasons). Fig. 3.6 shows some images generated using our generator. For this experiment, we compare our generator with StarGAN [22] using the FID [59] metrics. FID measures the realism of the

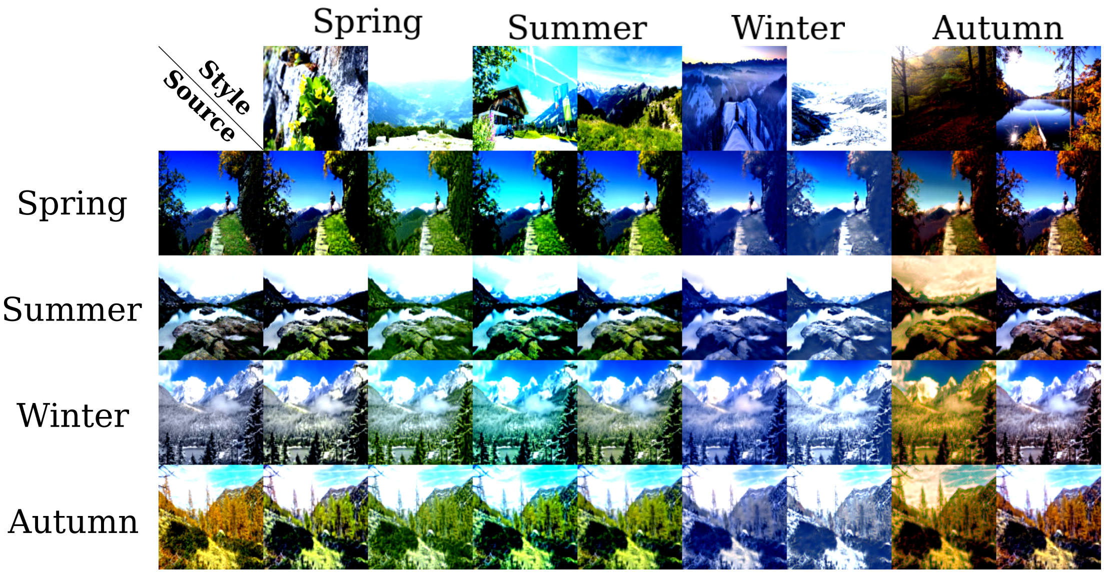


Figure 3.6: Image translation examples generated by TriGAN across different domains (i.e., seasons) using the Alps Seasons dataset. We show two generated images for each domain combination. The leftmost column shows the *source* images, one from each domain, and the topmost row shows the *style* of the target domain, two reference style images for each target domain.

generated images (the lower the better). The FID scores are computed considering all the real samples in the target domain and generating an equivalent number of synthetic images in the target domain. Tab. 3.4 shows that the TriGAN FID scores are significantly lower than the StarGAN scores. This further highlights that decoupling the style and the domain and using WC-based layers to progressively "whiten" and "color" the image statistics, yields to a more realistic cross-domain image translation than using domain labels as input as in the case of StarGAN.

	All/Wint. →Winter	All/Sum. →Summer	All/Spr. →Spring	All/Aut. →Autumn
StarGAN [22]	148.45	180.36	175.40	145.24
TriGAN (Ours)	41.03	38.59	40.75	32.71

Table 3.4: Alps Seasons dataset, FID scores: Comparing TriGAN with StarGAN [22].

3.4.4.4 Qualitative Image Translation Results

In Figs. 3.7 and 3.8, we show some translations examples obtained using our generator $\mathcal{G}$ and different datasets, which show how both the domain and the style is used in transforming an image sample. For instance, the fourth row of Fig. 3.7 shows an SVHN digit image which contains other digits in the background (which is a common characteristic of the SVHN dataset). When $\mathcal{G}$ translates this image to MNIST or MNIST-M, the background digit disappears, accordingly to the common uniform background of the target datasets. When the reference style image is, e.g., the MNIST-M "three" (fifth column), $\mathcal{G}$ correctly applies the instance-specific style (i.e., a blue foreground digit with a red background). A similar behaviour can be observed in Fig. 3.8.

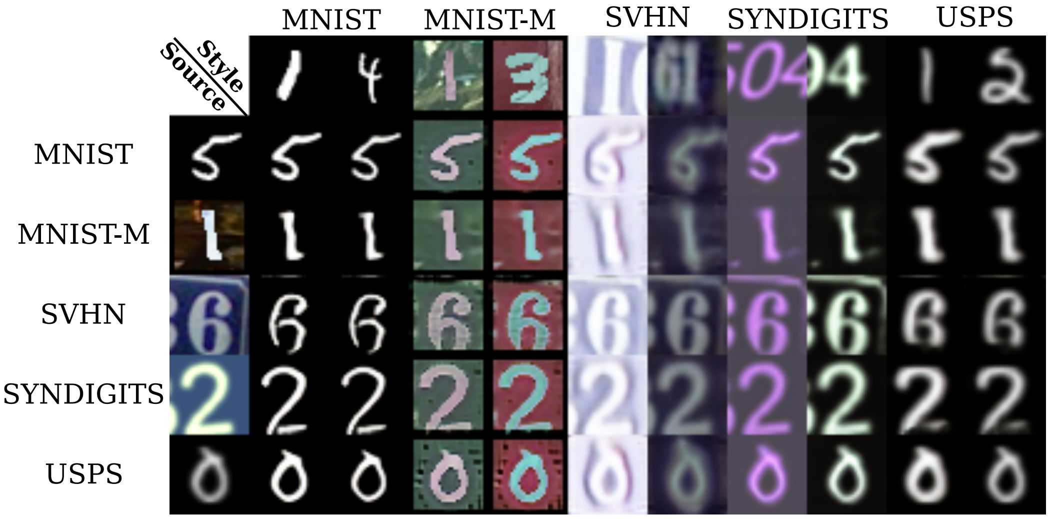


Figure 3.7: Image translation examples obtained using our generator with the Digits-Five dataset. The leftmost column shows the *source* images, one from each domain, and the topmost row shows the *style* image from the target domain, two reference images for each target domain.

3.5 Experiments for single-source UDA

Since TriGAN has can handle N-source domain translations, we also conduct experiments for a Single-Source UDA scenario where $N = 1$ and the source domain is grayscale MNIST. Below we describe the adopted UDA settings with the Digits-Five dataset and the corresponding results.

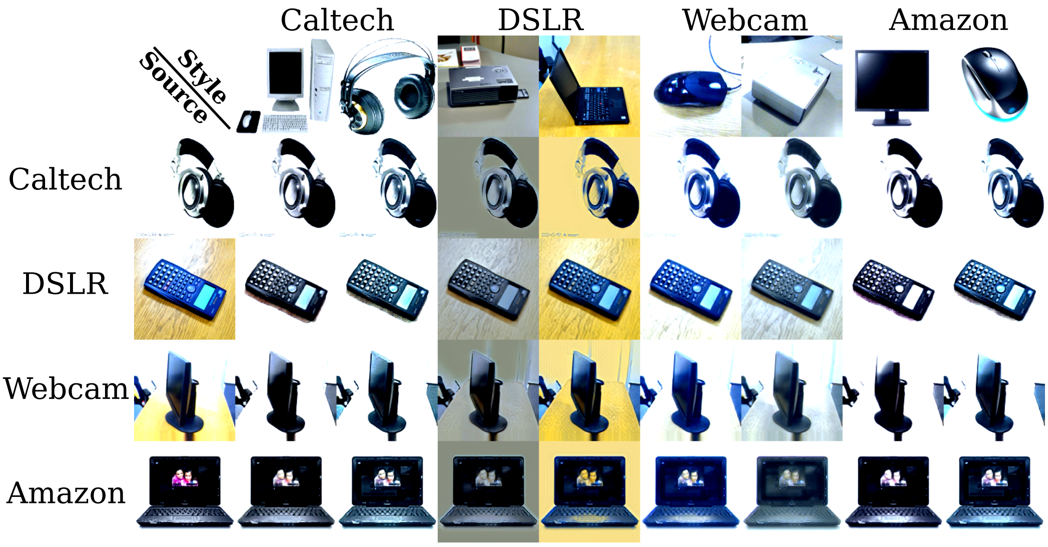


Figure 3.8: Image translation examples obtained using our generator with the Office-Caltech10 dataset. The leftmost column shows the *source* images, one from each domain, and the topmost row shows the *style* image from the target domain, two reference images for each target domain.

3.5.1 Datasets

MNIST → USPS. The MNIST dataset contains grayscale images of handwritten digits from 0 to 9. The image resolution in MNIST is 28 × 28. USPS contains similar grayscale handwritten digits, except from the resolution which is 16 × 16. We up-sample the images of both domains to 32 × 32 during training. For training TriGAN, 50000 MNIST and 7438 USPS samples are used. For evaluation, we use 1860 test samples from USPS.

MNIST → MNIST-M. MNIST-M is a coloured version of the grayscale MNIST digits. MNIST-M has RGB images with resolution 28 × 28. For the TriGAN training, all the 50000 training samples from both MNIST and MNIST-M are used, and the dedicated 10000 MNIST-M test samples are used for evaluation. Training images are up-sampled to 32 × 32.

MNIST → SVHN. SVHN is the short form of Street View House Number and contains real-world images of digits, ranging from 0 to 9. The samples in SVHN are RGB images, with a resolution of 32 × 32. SVHN has non-centered digits with varying colour intensities. One challenging characteristic of the SVHN images is the presence of other digits, partially shown

Methods	Source	MNIST	MNIST	MNIST
	Target	USPS	MNIST-M	SVHN
Source Only		78.9	63.6	26.0
DANN [38]		85.1	77.4	35.7
CoGAN [94]		91.2	62.0	-
ADDA [159]		89.4	-	-
PixelDA [10]		95.9	98.2	-
UNIT [93]		95.9	-	-
SBADA-GAN [136]		97.6	**99.4**	61.1
GenToAdapt [145]		92.5	-	36.4
CyCADA [61]		94.8	-	-
I2I Adapt [113]		92.1	-	-
TriGAN (Ours)		**98.0**	95.7	**66.3**

Table 3.5: Classification Accuracy (%) of generation-based methods on the *single-source* UDA scenario for digit recognition. The best value is in bold and the second best is underlined.

in the background. For the TriGAN training, 60000 MNIST and 73257 SVHN samples are used. During the evaluation, all the 26032 SVHN test samples are utilized.

3.5.2 Comparison with generation-based state-of-the-art methods

In this section we compare our proposed TriGAN with generation-based state-of-the-art UDA methods, either based on GANs or based reconstruction approaches. Tab. 3.5 reports the performance of our TriGAN alongside the results obtained from the following baselines: Domain Adversarial Neural Network [38] (**DANN**), Coupled generative adversarial networks [94] (**CoGAN**), Adversarial discriminative domain adaptation [159] (**ADDA**), Pixel-level domain adaptation [10] (**PixelDA**), Unsupervised image-to-image translation networks [93] (**UNIT**), Symmetric bi-directional adaptive gan [136] (**SBADA-GAN**), Generate to adapt [145] (**GenToAdapt**), Cycle-consistent adversarial domain adaptation [61] (**CyCADA**) and Image to image translation for domain adaptation [113] (**I2I Adapt**). Tab. 3.5 shows that TriGAN outperforms all the other generative methods in two out of the three adaptation settings. In the MNIST → MNIST-M setting, TriGAN is the third best. It is interesting to note that TriGAN achieves significantly better results in the MNIST → SVHN setting, which is considered as a hard setting, where TriGAN is 5.2% better than the second best method SBADA-GAN.

3.6 Conclusions

In this chapter we proposed TriGAN, an MSDA framework which is based on data-generation from multiple source domains using a single generator. The underlying principle of our approach to to obtain intermediate, domain and style invariant representations in order to simplify the generation process. Specifically, our generator progressively removes style and domain specific statistics from the source images and then re-projects the intermediate features onto the desired target domain and style. We obtained state-of-the-art results on two MSDA datasets, showing the potentiality of our approach.

4

Curriculum Graph Co-teaching for Multi-target Domain Adaptation

In this chapter we address multi-target domain adaptation (MTDA), where given one labeled source dataset and multiple unlabeled target datasets that differ in data distributions, the task is to learn a robust predictor for all the target domains. We identify two key aspects that can help to alleviate multiple domain-shifts in the MTDA: feature aggregation and curriculum learning. To this end, we propose Curriculum Graph Co-Teaching (CGCT) that uses a dual classifier head, with one of them being a graph convolutional network (GCN) which aggregates features from similar samples across the domains. To prevent the classifiers from over-fitting on its own noisy pseudo-labels we develop a co-teaching strategy with the dual classifier head that is assisted by curriculum learning to obtain more reliable pseudo-labels. Furthermore, when the domain labels are available, we propose Domain-aware Curriculum Learning (DCL), a sequential adaptation strategy that first adapts on the easier target domains, followed by the harder ones. We experimentally demonstrate the effectiveness of our proposed frameworks on several benchmarks and advance the state-of-the-art in the MTDA by large margins (e.g. +5.6% on the DomainNet w.r.t to the competitor method)[1].

[1] The content of this chapter is based on the CVPR 2021 paper [130].

4.1 Introduction

Deep learning models suffer from the well known drawback of failing to generalize well when deployed in the real world. The gap in performance arises due to the difference in the distributions of the training (a.k.a source) and the test (a.k.a target) data, which is popularly referred to as *domain-shift* [157]. Since, collecting labeled data for every new operating environment is prohibitive, a rich line of research, called Unsupervised Domain Adaptation (UDA), has evolved to tackle the task of leveraging the source data to learn a robust predictor on a desired target domain.

In the literature, UDA methods have predominantly been designed to adapt from a single source domain to a single target domain (STDA). Such methods include optimizing statistical moments [13, 14, 15, 98, 124, 131, 152, 161], adversarial training [38, 96, 159], generative modelling [61, 94, 136], to name a few. However, given the proliferation in unlabeled data acquisition, the need to adapt to just a single target domain has lost traction in the real world scenarios. As the number of target domains grows, the number of models that need to be trained also scales linearly. For this reason, the research focus has very recently been steered to address a more practical scenario of adapting simultaneously to multiple target domains from a single source domain. This adaptation setting is formally termed as Multi-target Domain Adaptation (MTDA). The goal of the MTDA is to learn more compact representations with a single predictor that can perform well in all the target domains. Straightforward application of the STDA methods for the MTDA may be sub-optimal due to the presence of multiple domain-shifts, thereby leading to negative transfer [20, 178]. Thus, the desideratum to align multiple data distributions makes the MTDA considerably more challenging.

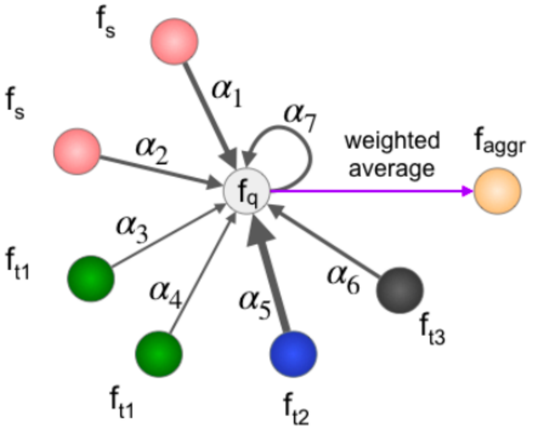


Figure 4.1: Schematic diagram of aggregating features from similar samples across domains using a graph convolutional network. Each color represents a domain

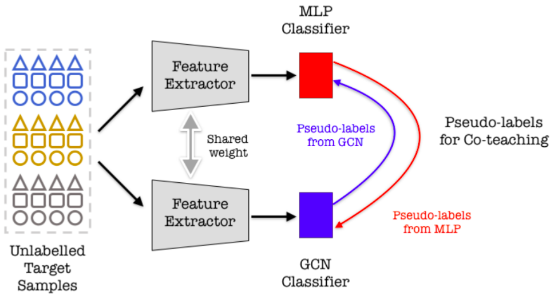


Figure 4.2: Schematic diagram of co-teaching with dual-head classifier

In this chapter we build our framework for the MTDA pivoted around two key concepts: *feature aggregation* and *curriculum learning*. Firstly, we argue that given the intrinsic nature of the task, learning robust features in a unified space is a prerequisite for attaining minimum risk across multiple target domains. For this purpose we propose to represent the source and the target samples as a graph and then leverage Graph Convolutional Networks [73] (GCN) to aggregate semantic information from similar samples in a *neighbourhood* across different domains (see Fig. 4.1). For the GCN to be operative, partial relationships among the samples (nodes) in the graph must at least be known apriori in the form of class labels. However, this information is absent for the target samples. To this end, we design a *co-teaching* framework where we train two classifiers: a MLP classifier and a GCN classifier that provide target pseudo-labels to each other (see Fig. 4.2). On the one hand, the MLP classifier is utilized to make the GCN learn the pairwise similarity between two nodes in the graph. While, on the other hand, the GCN classifier, due to its feature aggregation property, provides better pseudo-labels to assist the training of the MLP classifier. Given that co-teaching works on the assumption that different networks capture different aspects of learning [8], it is beneficial for suppressing noisy pseudo-labels. his feature aggregation and/or co-teaching aspects are largely missing in existing MTDA methods [20, 42, 123, 173] (see Tab. 4.1).

Method	Domain labels	Feature aggregation	Curriculum learning	Co-teaching
AMEAN [20]	✗	✗	✗	✗
DADA [123]	✗	✗	✗	✗
MTDA-ITA [42]	✓	✗	✗	✗
HGAN [173]	✓	✓	✗	✗
CGCT (**Ours**)	✗	✓	✓	✓
D-CGCT (**Ours**)	✓	✓	✓	✓

Table 4.1: Comparison with recent the state-of-the-art MTDA methods in terms of the operating regimes.

Secondly, we make a crucial observation, very peculiar to the MTDA setting, *i.e.*, during training as the network tries to adapt to multiple domain-shifts of varying degree, pseudo-labels obtained on-the-fly from the network for the target samples are very noisy. Self-training the network with unreliable pseudo-labeled target data further deteriorates the performance. To further combat the impact of noisy pseudo-labels, we propose to obtain pseudo-labels in an episodic fashion, and advocate the use of *curriculum learning* in the context of MTDA. In particular, when the domain labels of the target are latent, each episode or *curriculum step* consists of a fixed number of training iterations. Fairly consistent and reliable pseudo-labels are obtained from the GCN classifier at the end of each curriculum step. We call this proposed framework as **C**urriculum **G**raph **C**o-**T**eaching (CGCT) (see Fig. 4.4 (a)).

Furthermore, when the domain labels of the target are available, we propose an Easy-To-Hard Domain Selection (EHDS) strategy where the feature alignment process begins with the target domain that is closest to the source and then gradually progresses towards the hardest one (see Fig. 4.3). This makes adaptation to multiple targets smoother. In this case, each curriculum step involves adaptation with a single new target domain. The CGCT when combined with this proposed **D**omain-aware **C**urriculum **L**earning (DCL) (see Fig. 4.4 (b)) is referred to as D-CGCT. The Tab. 4.1 highlights the operating regimes of our frameworks versus the state-of-the-art MTDA methods.

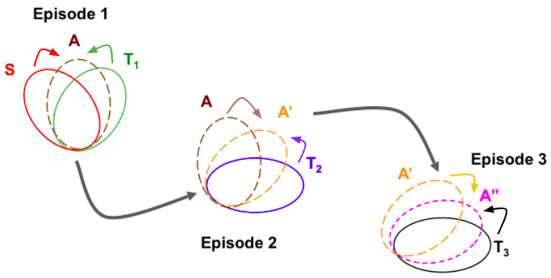


Figure 4.3: Schematic diagram of domain curriculum learning (DCL) when the domain labels are known. Each solid ellipse represents a domain and dotted ones represent the adapted domain

To summarize, the contributions of this work are threefold:

- We propose Curriculum Graph Co-Teaching (CGCT) for MTDA that exploits the co-teaching strategy with the dual classifier head, together with the curriculum learning, to learn more robust representations across multiple target domains.
- To better utilize the domain labels, we propose a Domain-aware Curriculum Learning (DCL) strategy to make the feature alignment process smoother.
- In the MTDA setting, we outperform the state-of-the-art for several UDA benchmarks by significant margins (including +5.6% on the large scale DomainNet [122]).

4.2 Related Works

Single-source single-target DA (STDA) refers to the task of adapting a classifier from a single labeled source dataset to a single unlabeled target dataset. In the UDA literature, a plethora of STDA methods have been proposed, which can be broadly classified into three major categories based upon the adaptation strategy. The first category uses first (Maximum Mean Discrepancy [98, 99, 161, 163]) or second order (correlation alignment [13, 15, 88, 108, 111, 124, 131, 133, 152]) statistics of the source and target features to align the marginal feature distributions. The second category of STDA methods [12, 17, 38, 96, 159] adopts adversarial training strategy to align the marginal feature distributions of the two domains. Essentially, these methods use a gradient reversal layer [38] to make the feature extractor network agnostic to domain specific information. The final category of STDA methods [61, 94, 136, 145] resort to pixel-level adaptation by generating synthetic *target-like* source images or *source-like* target images with the help of generative adversarial network (GAN) [45]. However, practical applications go beyond the single-source and single-target setting and often involve multiple source [132, 169, 171] or target domains.

Multi-target DA aims to transfer knowledge from a single labeled source dataset to multiple unlabeled target datasets. While the research in STDA is quite mature, most STDA methods can not be trivially extended to a multi-target setting. So far only a handful of methods [20, 42, 69, 95, 123, 173] for MTDA can be found in the literature. AMEAN [20] performs clustering on the blended target domain samples to obtain *sub-targets* and then learns domain-invariant features from the source and the obtained sub-targets using a STDA method [148]. The approaches introduced in [42, 69, 123] are derived from STDA and do not exploit any peculiarity of the MTDA setting. Conversely, our CGCT and D-CGCT are tailor-made for the multi-target setting as we propose to use feature aggregation of similar samples across multiple domains.

Curriculum for DA involves adopting an adaptive strategy that evolves over time to better address the adaptation across domains. Shu *et. al.* [149] propose a strategy based on curriculum learning that exploits the loss of the network as weights to identify and eliminate unreliable source samples. An Easy-to-Hard Transfer Strategy (EHTS) is proposed in PFAN [16] that progressively selects the pseudo-labeled target samples which have higher cosine similarity to the per-category source prototypes. Similarly, our CGCT is inspired by the EHTS strategy except we progressively recruit the pseudo-labeled targets [6] from the robust GCN classification head to better train the MLP classifier, which in turn regularizes the GCN head (see Sec.4.3.2). For the multi-source DA setting, CMSS [171] trains a separate network to weigh the most relevant samples across several source domains for adapting to a single target domain. However, differently from CMSS, our proposed DCL utilizes the domain information to adapt over time from the easiest to the hardest target domain in the MTDA setting (see Sec. 4.3.3).

Graph Neural Networks (GNN) are neural network models applied on graph-structured data that can capture the relationships between the objects (nodes) in a graph via message passing through the edges [46, 167]. Relevant to our work are GNN-derived Graph Convolutional Networks (GCN) [73] that have recently been applied for addressing DA [101, 102, 173]. For instance, Luo *et. al.* [101] propose PGL for open-set DA to capture the relationship between the overlapping classes in the source and the target. Notably, Yang *et. al.* [173] introduce heterogeneous Graph Attention Network (HGAN) for MTDA to learn the relationship of similar samples among multiple domains and then utilize the graph-based pseudo-labeled target samples to align their centroids with that of the source. Unlike [101, 173], we incorporate the idea of co-teaching [53] in a GCN framework for combating noisy pseudo-labels.

4.3 Methods

In this section we present our proposed Curriculum Graph Co-Teaching (CGCT) and thereafter Domain Curriculum Learning (DCL) for the task of MTDA. We also discuss some preliminaries that are used to address the task.

Problem Definition. In the MTDA scenario, we are provided with a single source dataset $\mathcal{S} = \{(\mathbf{x}_{s,i}, y_{s,i})\}_{i=1}^{n_s}$, containing n_s labeled samples, and N unlabeled target datasets $\mathcal{T} = \{\mathcal{T}_j\}_{j=1}^{N}$, where $\mathcal{T}_j = \{\mathbf{x}_{t_j,k}\}_{k=1}^{n_j}$ with each containing n_j unlabeled samples. As in any DA scenario, the fundamental assumption is that the underlying data distributions of the source and the targets are different from each other. It is also assumed that the label space of the source and targets are the same. Under these assumptions, the goal of the MTDA is to learn a single predictor for all the target domains by using the data in $\mathcal{S} \cup \{\mathcal{T}_j\}_{j=1}^{N}$.

4.3.1 Preliminaries

Baseline for Multi-target Domain Adaptation. Domain Adversarial Network (DANN) [38], originally designed for STDA, aligns the feature distributions of the source and the target domains by using an adversarial training. DANN comprises of three networks: the feature extractor, the classifier and the domain discriminator. The classifier is responsible for classifying the features obtained from the feature extractor into n_c classes. On the one hand, the domain discriminator distinguishes the source from the target features. While on the other hand, the feature extractor is trained to fool the discriminator and simultaneously learn good features for semantic classification.

Formally, let $F_\theta : \mathbb{R}^{3\text{x}w\text{x}h} \to \mathbb{R}^d$ be the feature extractor network, parameterized by θ, that outputs a feature $\mathbf{f} = F(\mathbf{x})$ for a given sample $\mathbf{x}$. The classifier network, parameterized by ϕ, is denoted by $G_\phi : \mathbb{R}^d \to \mathbb{R}^{n_c}$, which takes as input a feature $\mathbf{f}$ and outputs class logits $\mathbf{g} = G(\mathbf{f})$. The discriminator network $D_\psi : \mathbb{R}^d \to \mathbb{R}^1$, parameterized by ψ, takes in the same feature $\mathbf{f}$ and outputs a single logit. By treating all the target domains as one combined target domain, the overall training objective of DANN for MTDA is given by:

$$\max_{\psi} \min_{\theta,\phi} \ell_{ce} - \lambda_{adv}\, \ell_{adv}, \tag{4.1}$$

$$\begin{aligned} \text{where } \ell_{ce} &= -\,\mathbb{E}_{(\mathbf{x}_{s,i},y_{s,i})\sim\mathcal{S}}\ \widetilde{y}_{s,i}\log G(F(\mathbf{x}_{s,i})),\\ \text{and } \ell_{adv} &= -\,\mathbb{E}_{\mathbf{x}_{s,i}\sim\mathcal{S}}\ \log D(F(\mathbf{x}_{s,i}))\\ &\quad -\,\mathbb{E}_{x_{t,j}\sim\mathcal{T}}\ \log\,[1 - D(F(\mathbf{x}_{t,j}))]. \end{aligned}$$

$\widetilde{y}_{s,i}$ is the one-hot label for a source label $y_{s,i}$. The first term, ℓ_{ce}, in Eq. 4.1 is the cross-entropy loss computed on the source domain samples and minimized w.r.t. θ, ϕ. The second term, ℓ_{adv}, in Eq. 4.1 is the adversarial loss that is maximized w.r.t ψ but minimized w.r.t θ. λ_{adv} is the weighing factor for ℓ_{adv}. To capture the multi-modal nature of the distributions, CDAN [96] is proposed where D can be additionally conditioned on the classifier predictions $\mathbf{g}$. In CDAN [96], the D takes as input $\mathbf{h} = (\mathbf{f}, \mathbf{g})$, the joint variable of $\mathbf{f}$ and $\mathbf{g}$, instead of just $\mathbf{f}$. In this chapter we use CDAN for aligning the feature distributions.

Graph Convolutional Network. For the GCN [73] classifier we construct an undirected and fully-connected graph $\Gamma = (\mathcal{V}, \mathcal{E}, \mathcal{A})$ from all samples in mini-batch. In details, given a mini-batch of images, we represent each image $\mathbf{x}_i$ as a node $\mathbf{v}_i \in \mathcal{V}$ in the Γ. $e_{i,j} \in \mathcal{E}$ indicates an edge between nodes $\mathbf{v}_i$ and $\mathbf{v}_j$, and $a_{i,j}$ is the semantic similarity score for nodes $(\mathbf{v}_i, \mathbf{v}_j)$ forming an affinity matrix $\mathcal{A}$.

Following [101], we compute the semantic similarity scores $\hat{a}_{i,j}^{(l)}$ at the l-th layer for all pairs $(\mathbf{v}_i, \mathbf{v}_j) \in \mathcal{E}$:

$$\hat{a}_{i,j}^{(l)} = f_{edge}^{(l)}(\mathbf{v}_i^{(l-1)}, \mathbf{v}_j^{(l-1)}), \tag{4.2}$$

where $f_{edge}^{(l)}$ is a non-linear similarity function parameterized by φ, and $\mathbf{v}_i^{(l-1)}$ is features at l-1 GCN layer of a sample $\mathbf{v}_i$. The initial node features $\mathbf{v}_i$ are instantiated with $\mathbf{f}_i$, the embedding obtained from F. Then, we add self-connections for nodes in the graph and normalize the obtained similarity scores as:

$$\mathcal{A}^{(l)} = M^{-\frac{1}{2}}(\hat{\mathcal{A}}^{(l)} + I)M^{-\frac{1}{2}}, \tag{4.3}$$

where M is the degree matrix, I is the identity matrix, and $\hat{\mathcal{A}}$ is the un-normalized affinity matrix.

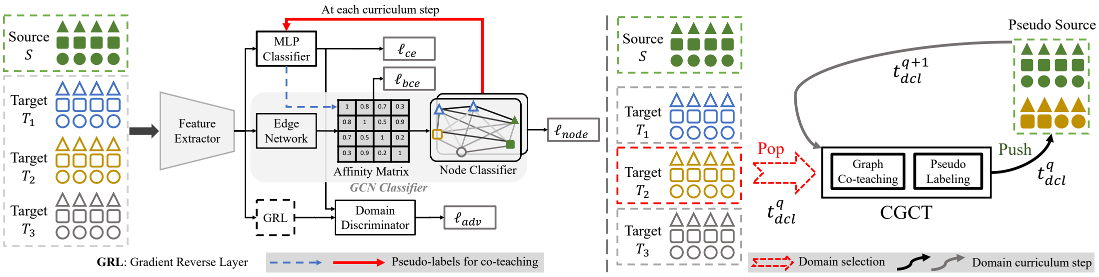
Finally, given the affinity matrix $\mathcal{A}^{(l)}$, we update the node features with the following propagation rule:

$$\mathbf{v}_i^{(l)} = f_{node}^{(l)}\Big([\mathbf{v}_i^{(l-1)}, \sum_{j \in \mathcal{B}} a_{i,j}^{(l)} \cdot \mathbf{v}_j^{(l-1)}]\Big), \tag{4.4}$$

where $f_{node}^{(l)}$ is a non-linear function parameterized by φ', $\mathcal{B}$ is a set of samples in the mini-batch, and $[\cdot, \cdot]$ is the feature concatenation function. The final $f_{node}^{(L)}$ layer is the output layer with n_c outputs. We slightly abuse the notations and drop the superscript l in our subsequent formulations for the sake of clarity.

4.3.2 Curriculum Graph Co-Teaching

In this chapter we introduce the Curriculum Graph Co-Teaching (CGCT) that employs feature aggregation with a GCN and uses curriculum learning for pseudo-labeling. In details, as shown in Fig. 4.4(a), it is composed of: a feature extractor F, a domain discriminator D, a MLP classifier G_{mlp} and a GCN classifier G_{gcn}. The G_{mlp} is a fully-connected output layer with n_c output logits. The G_{gcn} consists of an edge network f_{edge} and a node classifier f_{node}. The f_{node} aggregates the features of the samples in $\mathcal{B}$ by considering the learnt pairwise similarity in the affinity matrix $\hat{\mathcal{A}}$ produced by the f_{edge}. The G_{gcn} also outputs n_c logits. Since, the G_{mlp} and the G_{gcn} capture different aspects of learning, they are exploited to provide feedback



(a) Curriculum Graph Co-Teaching

(b) Domain-aware Curriculum Learning

Figure 4.4: The pipeline of the proposed framework: a) CGCT: Curriculum Graph Co-Teaching and b) DCL: Domain-aware curriculum learning. (a) In the CGCT, the MLP Classifier provides pseudo-labels (PL) (- - → arrow) for the target samples to guide the Edge Network to learn the Affinity Matrix, whereas the Node Classifier of the GCN provides PL (bold → arrow) to the MLP Classifier at the end of each curriculum step, realizing the *co-teaching*. (b) In the DCL, the target domains are selected for adaptation, one at a time per domain curriculum step t^q_{dcl}, with the "easier" domains selected first and then the "harder" ones. After PL are obtained, the pseudo-labeled target dataset is added to the Pseudo Source dataset, which is then used in the next adaptation step.

to each other in a co-teaching fashion. The CGCT is trained for Q *curriculum steps* where a curriculum step, t^q_{cur}, is an episode in which the network is trained for K training iterations. Each curriculum step t^q_{cur} is further decomposed into two stages: i) Adaptation stage and ii) Pseudo-labeling stage. Each stage in a t^q_{cur} is described below. Note that, as in [20], we assume that the domains labels of the target are latent and not observed during training.

Adaptation stage. In this stage we mainly perform the feature alignment using CDAN [96]. In details, initially at step t^0_{cur} we start with a source set $\hat{\mathcal{S}}^0 = \{\mathcal{S}\}$ and a target set $\mathcal{T}$. We sample mini-batches $\mathcal{B}^0 = \{\mathcal{B}^0_s, \mathcal{B}^0_t\} = \{\mathcal{B}^0_{s,i}, \mathcal{B}^0_{t,i}\}^B_{i=1}$ with size B such that $\mathcal{B}^0_{s,i} \sim \hat{\mathcal{S}}$ and $\mathcal{B}^0_{t,i} \sim \mathcal{T}$. Each mini-batch of images is first fed to the feature extractor F to obtain $\mathcal{F}^0 = \{\mathbf{f}^0_{s,i}, \mathbf{f}^0_{t,i}\}^B_{i=1}$ which are then simultaneously fed to both the G_{mlp} and G_{gcn}. When fed to the G_{mlp} it outputs the logits $\hat{\mathcal{G}}^0 = \{\hat{\mathbf{g}}^0_{s,i}, \hat{\mathbf{g}}^0_{t,i}\}^B_{i=1}$. On the other hand, $\mathcal{F}^0$ are input to the f_{edge} to estimate the pairwise similarity of the samples in $\mathcal{B}^0$. Specifically, the f_{edge} outputs an affinity matrix $\hat{A}$ following Eq. 4.2, where the entries $\hat{a}_{i,j}$ in $\hat{A}$ denote the strength of similarity between samples i and j in $\mathcal{B}^0$. Intuitively, higher the value of $\hat{a}_{i,j}$, higher is the likelihood of samples i and j belonging to the same semantic category. Finally, following Eq. 4.4, the f_{node} aggregates the features in $\mathcal{F}^0$ based on the estimated $\hat{A}$ such that for each node the most similar samples in the neighbourhood contribute more to its final representation. Subsequently, the f_{node} outputs its logits as $\bar{\mathcal{G}}^0 = \{\bar{\mathbf{g}}^0_{s,i}, \bar{\mathbf{g}}^0_{t,i}\}^B_{i=1}$. The elements in $\hat{\mathcal{G}}^0$ and $\bar{\mathcal{G}}^0$ are then passed through a softmax function to obtain the probabilities for each sample as $p(\hat{y} = c|\hat{\mathbf{g}}; c \in n_c)$ and $p(\bar{y} = c|\bar{\mathbf{g}}; c \in n_c)$, where $\hat{y}$ and $\bar{y}$ are the predictions, respectively.

To guide the f_{edge} to learn the pairwise similarity between the samples in $\mathcal{B}^0$ we propose the concept of co-teaching where the G_{mlp} provides feedback to the f_{edge}. Since, G_{mlp} makes instance-level independent predictions on the samples in $\mathcal{B}^0$, it is not susceptible to the accumulation of potential noise from the dissimilar neighbours. To this end, for a $\mathcal{B}^0$ we construct a "*target*" affinity matrix $\hat{\mathcal{A}}^{tar}$ and enforce the predictions of f_{edge} to be as close as possible to the $\hat{\mathcal{A}}^{tar}$. Each entry $\hat{a}^{tar}_{i,j}$ in the $\hat{\mathcal{A}}_{tar}$ is given by:

$$\hat{a}^{tar}_{i,j} = \begin{cases} 1, & \text{if } y_i = y_j = c \\ 0, & \text{otherwise} \end{cases}, \tag{4.5}$$

where c is the class label. While the class labels of $\mathcal{B}^0_s$ are provided as ground truth, we do not have access to the labels of $\mathcal{B}^0_t$. Therefore, a target domain sample $\mathbf{x}_{t,j} \in \mathcal{B}^0_t$ is assigned a definitive pseudo-label $\hat{y}_{t,j} = c'$ where $c' = \text{argmax}_{c \in n_c} p(\hat{y}_{t,j} = c|\hat{\mathbf{g}}_{t,j})$ if the maximum likelihood $\max_{c \in n_c} p(\hat{y}_{t,j} = c|\hat{\mathbf{g}}_{t,j})$ is greater than a threshold τ. The entries $\hat{a}^{tar}_{i,j}$ involving $\mathbf{x}_{t,j} \in \mathcal{B}^0_t$ not passing the τ are not optimized during training. We train the f_{edge} using a binary cross-entropy loss as:

$$\ell^{edge}_{bce} = \hat{a}^{tar}_{i,j} \log p(\hat{a}_{i,j}) + (1 - \hat{a}^{tar}_{i,j}) \log (1 - p(\hat{a}_{i,j})). \tag{4.6}$$

Finally, for training the G_{mlp} and the f_{node} in the G_{gcn} we compute the standard cross-entropy loss with the samples in $\mathcal{B}^0_s$ as:

$$\ell^{mlp}_{ce} = -\frac{1}{|\mathcal{B}^0_s|} \sum_{i=1}^{|\mathcal{B}^0_s|} \tilde{y}_i \log \ p(\hat{y}_{s,i}|\hat{\mathbf{g}}^0_{s,i}), \tag{4.7}$$

$$\ell^{node}_{ce} = -\frac{1}{|\mathcal{B}^0_s|} \sum_{i=1}^{|\mathcal{B}^0_s|} \tilde{y}_i \log \ p(\bar{y}_{s,i}|\bar{\mathbf{g}}^0_{s,i}). \tag{4.8}$$

We feed the features $\{\hat{\mathbf{h}}^0_{s,i}, \hat{\mathbf{h}}^0_{t,i}\}^B_{i=1} = \{(\mathbf{f}^0_{s,i}, \hat{\mathbf{g}}^0_{s,i}), (\mathbf{f}^0_{t,i}, \hat{\mathbf{g}}^0_{t,i})\}^B_{i=1}$, corresponding to $\mathcal{B}^0$, to the domain discriminator D and compute the conditional adversarial loss following Eq. 4.1. Thus, the final objective function for the CGCT can be written as:

$$\begin{aligned} \max_{\psi} \min_{\theta,\phi,\varphi,\varphi'} \; & \ell_{ce}^{mlp} + \lambda_{edge}\ell_{bce}^{edge} \\ & + \lambda_{node}\ell_{ce}^{node} - \lambda_{adv}\ell_{adv}, \end{aligned} \tag{4.9}$$

where λ_{edge}, λ_{node} and λ_{adv} are the weighing factors.

Pseudo-labelling stage. Upon completion of the adaptation stage in a curriculum step t_{cur}^{q} we put the network in inference mode and obtain pseudo-labels $\forall \mathbf{x}_{t,j} \in \mathcal{T}$. The G_{gcn} is employed for this task because, owing to its aggregating characteristics, it learns more robust features [173] than the G_{mlp}. This is the *curriculum* aspect of our proposed co-teaching training strategy in CGCT where the obtained pseudo-labeled target samples are then used to train the G_{mlp}, besides the f_{node}.

At any step t_{cur}^{q}, the criterion for pseudo-label selection is formally written as:

$$\forall \mathbf{x}_{t,j} \in \mathcal{T}, w_j = \begin{cases} 1, & \text{if } \max_{c \in n_c} p(\bar{y}_{t,j} = c | \bar{\mathbf{g}}_{t,j}) > \tau \\ 0, & \text{otherwise} \end{cases}, \tag{4.10}$$

where $w_j = 1$ signifies that $\mathbf{x}_{t,j}$ is selected with a pseudo-label $\bar{y}_{t,j} = c'$ where $c' = \text{argmax}_{c \in n_c} p(\bar{y}_{t,j} = c | \bar{\mathbf{g}}_{t,j})$, whereas $w_j = 0$ denotes no pseudo-label is assigned. After the pseudo-labeling stage in a t_{cur}^{q} we obtain a pseudo-labeled target set $\mathcal{D}_t^q = \{(\mathbf{x}_{t,j}, \bar{y}_{t,j})\}_{j=1}^{\bar{n}_t}$ where $\bar{n}_t$ is the number of recruited pseudo-labeled target samples. Post pseudo-labeling we update and prepare the source set for the succeeding step t_{cur}^{q+1} as:

$$\hat{\mathcal{S}}^{q+1} = \mathcal{S} \cup \mathcal{D}_t^q. \tag{4.11}$$

The update rule in Eq. 4.11 allows us to compute the supervised losses ℓ_{ce}^{node} and ℓ_{ce}^{mlp} from Eq. 4.9 for $\mathbf{x}_{t,j} \sim \mathcal{D}_t$. Note that we do not alter the domain labels in $\mathcal{D}_t^q$ and hence, the formulation for ℓ_{adv} remains unchanged.

At the culmination of Q curriculum steps, $\hat{\mathcal{S}}^{Q}$ is obtained using Eq. 4.11 and the network is fine-tuned with only the supervised losses in Eq. 4.9 for K' training iterations.

4.3.3 Domain-aware Curriculum Learning

Now we consider the case when the domain labels of the target are available, *i.e.* $\mathcal{T} = \{\mathcal{T}_j\}_{j=1}^{N}$, N being the number of target domains. In principle, when the domain labels are available, one can either train N domain discriminators or a $(N+1)$ way single domain discriminator. Apart from over-parameterization, it also suffers from limited gradients coming from the discriminator(s) due to single point estimates [78]. Thus, we propose **D**omain-aware **C**urriculum **L**earning (DCL) as an alternate learning paradigm to better utilize the target domain labels in the MTDA setting.

To this end we design the DCL that is based on our proposed Easy-to-Hard Domain Selection (EHDS) strategy. Our proposal for the DCL stems from the observation that different target domains exhibit different domain shifts from the source domain, where some domain shifts are larger than the others. Evidently, the network will find it easier to adapt to the closest target domain while performing sub-optimally on the domain with the largest domain shift. When adaptation is performed with N domains at tandem then the large domain shifts of harder domains will interfere with the feature alignment on the easier target domains, thereby compromising the overall performance. To overcome this problem, in the EHDS strategy, as the name suggests, the network performs feature adaptation one domain at a time, starting from the easiest target domain and gradually moving towards the hardest. The "*closeness*" of a target domain from the source is measured by the uncertainty in the target predictions with a source-trained model. Lesser the uncertainty in predictions, closer the target from the source domain. Therefore, measuring the entropy on a target domain can serve as a good proxy for domain selection, and is defined as:

$$H(\mathcal{T}_j) = - \mathop{\mathbb{E}}_{\mathbf{x}_{t_j,k} \sim \mathcal{T}_j} \sum_{c=1}^{|n_c|} p(\hat{y}_{t_j,k,c} | \mathbf{x}_{t_j,k}) \log p(\hat{y}_{t_j,k,c} | \mathbf{x}_{t_j,k}). \tag{4.12}$$

Due to this step-by-step adaptation through domain traversal, the intermediate target domains help in reducing large domain shifts by making the farthest domain shift considerably closer than that at the start. Differently from the CGCT, in the DCL, each curriculum step, defined as t_{dcl}^q, consist in learning over one target domain, with a total of N steps. Since, the simulation of single-source and single-target adaptation inside the MTDA setup yields better domain-invariant features, at the end of each t_{dcl}^q we also consider extracting pseudo-labels for the target samples from the classifier and add them to the source set (see Fig. 4.4(b)) for computing the supervised losses. This further reduces the domain gaps for the forthcoming harder domains. The t_{dcl}^q is split into three stages and are described below:

Domain selection stage. Given a source-trained model $F_{\theta^*}(G_{\phi^*})$, where θ^* and ϕ^* are the trained parameters of F and G, and initial source and target sets $\hat{\mathcal{S}}^0 = \{\mathcal{S}\}$ and $\hat{\mathcal{T}}^0 = \{\mathcal{T}_j\}_{j=1}^N$, the closest target domain is selected as:

$$\mathbb{D}^0 = \underset{j}{\text{argmin}}\{H_j(\mathcal{T}_j) \mid \forall \mathcal{T}_j \in \hat{\mathcal{T}}^0\}, \tag{4.13}$$

where $\mathbb{D}^0$ is the target domain selected at step t_{dcl}^0 and is used for performing adaptation in the subsequent stage.

Adaptation stage. This stage is similar to the one in t_{cur}^q, described in Sec. 4.3.2, except the feature adaptation at any step t_{dcl}^q is performed using $\hat{\mathcal{S}}^q \cup \mathcal{T}_{\mathbb{D}^q}$, rather than the entire target set $\mathcal{T}$. The model is trained using the losses described in Eq. 4.9.

Pseudo-labeling stage. The criterion for pseudo-label selection still remains the same, as described in Eq. 4.10, with the exception of target samples being drawn only from the current target domain $\mathbb{D}^q$, yielding a pseudo-labeled target set $\mathcal{D}_t^{\mathbb{D}^q}$. Consequently, the source and target set update changes as following:

$$\hat{\mathcal{S}}^{q+1} = \hat{\mathcal{S}}^q \cup \mathcal{D}_t^{\mathbb{D}^q}, \tag{4.14}$$

$$\hat{\mathcal{T}}^{q+1} = \hat{\mathcal{T}}^q \setminus \mathcal{T}_{\mathbb{D}^q}. \tag{4.15}$$

These three stages are repeated until all N domains have been exhausted. Then similarly, as in CGCT, the final model is fine-tuned with $\hat{\mathcal{S}}^Q$. When CGCT is trained using the DCL strategy we refer to the model as D-CGCT. We would like to point that the DCL can also be realized with a single classifier model (see Sec. 4.4).

4.3.4 Discussion

Here we highlight the keys differences between the CGCT and PGL [101] as well as the dual classifier-based methods [53, 141]. The PGL [101] exploits the graph learning framework in an episodic fashion to obtain pseudo-labels for the unlabeled target samples, which are then used to bootstrap the model by training on the pseudo-labeled target data. While our proposed method is similar in spirit to the episodic training in [101], we do not solely rely on the GCN to obtain the pseudo-labels. We conjecture that due to the fully-connected nature of the graph and lack of target labels, the GCN will be prone to accumulate features of dissimilar neighbours, thereby, resulting in the erroneous label propagation. To address this peculiarity, we propose to resort to the co-teaching paradigm, where the G_{mlp} is exploited to train the f_{edge} network. As the two classifiers will capture different aspects of training [53], it will prevent the f_{edge} to be trained with the same erroneous pseudo-labels as the f_{node}. We validate this conjecture empirically, where a network with a single GCN classifier with pseudo-labels performs sub-optimally compared to CGCT (see Tab. 5 row 7 of the main paper). Finally, the dual classifier-based methods maintain two classifiers to identify and filter either harder target samples [141] or noisy samples [53]. Contrarily, we maintain G_{mlp} and G_{gcn} to provide feedback to each other by exploiting the key observation that each classifier learns different patterns during training. Furthermore, given the intrinsic design of the G_{gcn}, we also do away with an extra adhoc loss of keeping the weights of two networks different.

4.4 Experiments

4.4.1 Dataset and Experimental Details

Datasets. We conduct experiments on five standard UDA benchmarks: Digits-five [169], Office-31 [138], PACS [85], Office-Home [163] and the very large scale DomainNet [122] (**0.6 million** images). The statistics of the datasets are summarized in Tab. 4.2.

Digits-five [169] is composed of five domains that are drawn from the: i) grayscale handwritten digits MNIST [82] (**mt**); ii) a coloured version of **mt**, called as MNIST-M [38] (**mm**); iii) USPS [32] (**up**), which is a lower resolution, 16×16, of the handwritten digits **mt**; iv) a real-world dataset of digits called SVHN [115] (**sv**); and v) a synthetically generated dataset *Synthetic Digits* [38] (**sy**). Following the protocol of [20], we sub-sample 25,000 and 9,000 samples from the training and test sets of **mt**, **mm**, **sv** and **sy** and use as train and test sets, respectively. For the **up** domain we use all the 7,348 training and 1,860 and test samples, for our experiments. All the images are re-scaled to a 28×28 resolution.

Office31 [138] is a standard visual DA dataset comprised of three domains: Amazon, DSLR and Webcam. The dataset consists of 31 distinct object categories with a total of 4,652 samples.

Office-Home [163] is a relatively newer DA benchmark that is larger than Office31 and is composed of four different visual domains: Art, Clipart, Product and Real. It consists of 65 object categories and has 15,500 images in total.

PACS [85] is another visual DA benchmark that also consists of four domains: Photo (P), Art Painting (A), Cartoon (C) and Sketch (S). This dataset is captured from 7 object categories and has 9,991 images in total.

DomainNet [122] is the most challenging and very large scale DA benchmark, which has six different domains: Clipart (C), Infograph (I), Painting (P), Quickdraw (Q), Real (R) and Sketch (S). It has around **0.6 million** images, including both train and test images, and has 345 different object categories. We use the official training and testing splits, as mentioned in [123], for our experiments.

Dataset	#domains	#classes	#images
Digits-five	5	10	$\sim$ 145K
PACS	4	7	9,991
Office-31	3	31	4,652
Office-Home	4	65	15,500
DomainNet	6	345	$\sim$ 0.6M

Table 4.2: Dataset details for multi-target domain adaptation.

Evaluation protocol. We use the classification accuracy to evaluate the performance. The classification accuracy is computed for every possible combination of one source domain and the rest of the target domains. The performance for a given direction, *i.e.*, *source→rest*, is given by averaging the accuracy on all the target domains, where *source* signifies the source domain and *rest* indicates all the unlabeled domains except the *source*. Importantly, in all our experiments we always report the final classification accuracy obtained with the G_{mlp} because the G_{gcn} always requires a mini-batch at inference, an assumption which is easily violated when deployed in the real world.

Implementation details. To be fairly comparable with the state-of-the-art methods, we adopted comparable backbone feature extractors in the corresponding experiments and datasets. For Digits-five, we have used a small convolutional network as the backbone feature extractor, which is adapted from [20] and includes two *conv* layers and two *fc* layers. We trained the model using a Stochastic Gradient Descent (SGD) optimizer with an initial learning rate of 1e-3. For the rest of the datasets, we have adoptd ResNet [57] based feature extractors. Specifically, for the ablation studies on Office-Home, we have used ResNet-18 as the backbone network. For the state-of-the-art comparisons on Office31, PACS and Office-Home we have used ResNet-50. For the DomainNet, we have utilized ResNet-101 as used by the competitor methods. Similarly to the Digits-five, SGD optimizer is used with an initial learning rate of 1e-3 and is decayed exponentially. Each curriculum step consists of $K = 10,000$ training iterations for all the datasets, except the DomainNet, where $K = 50,000$ due to large size of the dataset. The final fine-tuning step is trained with $K' = 15,000$ iterations for all datasets.

For the GCN architecture, we have implemented f_{node} network with 2 conv layers followed by a Batch Normalization (BN) layer and ReLU activation, except the final layer. The first layer takes as input image features concatenated with the context of the mini-batch, *i.e.*, the aggregated features of other images in a mini-batch (based on the affinity matrix estimated by the f_{edge}). The second conv layer outputs the logits that are equal to the number of classes n_c. We have used 1x1 convolution kernels in the f_{node}. Similarly, we have implemented the f_{edge} network with 3 conv layers and 1x1 kernels, where the first two layers are followed by the BN layers and ReLU activations, except the last. The third conv layer has a single channel as output, thus, representing the similarity scores between samples in a mini-batch.

Hyperparameter selection. In our final model we used only a single set of hyperparameters, which are $\lambda_{edge} = 1$, $\lambda_{node} = 0.3$, $\lambda_{adv} = 1$ and $\tau = 0.7$. Following the standard protocol in [148], we used a held-out validation set of 1000 samples for the MNIST $\rightarrow$ *rest* direction to tune these hyper-parameters.

4.4.2 Ablations

In this section we discuss the design choices of our proposed contributions and report the results of a thorough ablation study. Our ablation analysis highlights the importance of the *graph co-teaching* and the *curriculum learning*. We run the ablation experiments on Office-Home with ResNet-18 [57] as backbone network and on Digits-five with a network adopted from AMEAN [20]. We adopt the CDAN as a baseline for adaptation in Tab. 4.3 and Tab. 4.4.

Graph co-teaching. The goal of this particular ablation study is to analyse why our proposed graph co-teaching is beneficial and the manner in which it should be realised in an adaptation framework. To this end, as shown in the Tab. 4.3, we design some baselines that can be distinguished in the manner in which the G_{mlp} and the G_{gcn} provide pseudo-labels to the

		Pseudo-labels from			
Model	Co-teaching	G_{mlp}	f_{edge}	f_{node}	**Avg**(%)
M1	✗	self	G_{mlp}	G_{mlp}	57.4
M2	✗	G_{gcn}	G_{gcn}	G_{gcn}	59.6
M3	✓	self	G_{mlp}, G_{gcn}	G_{mlp}	58.2
D-CGCT (Ours)	✓	G_{gcn}	G_{mlp}	G_{gcn}	**60.8**

Table 4.3: Ablation study of different co-teaching strategies on Office-Home. We reported the classification accuracy averaged across all the *source* → *rest* directions.

each other (columns 3 to 5) and then compare it to our D-CGCT. In more details, the baseline models can be described as: i) M1: a baseline where the G_{mlp} provides pseudo-labels to itself, f_{edge} and f_{node} after each curriculum step t^q_{dcl}; ii) M2: a baseline similar to M1, except that the G_{gcn} provides the pseudo-labels; iii) M3: another baseline which is similar to M1 but with an exception that the G_{gcn} also provides pseudo-labels to f_{edge} for the current target domain in an ongoing t^q_{dcl} step.

Unsurprisingly, M1 performs the worst of all the baselines because the pseudo-labels computed by the G_{mlp} are less accurate due to G_{mlp} not taking into account the *feature aggregation* from multiple domains. Contrarily, the baseline M2 performs better than the M1 due to the fact that M2 uses G_{gcn} for pseudo-labeling, which are more accurate. This highlights the importance of feature aggregation in the MTDA setting. One other thing that separates D-CGCT from both M1 and M2 is the co-teaching, which is absent in the latter baselines. Since, the D-CGCT enables co-teaching, with the G_{mlp} and the G_{gcn} providing pseudo-labels to each other, it does not overfit on the same “incorrect" pseudo-label, thereby achieving more robust predictions. Contrarily, M3 uses co-teaching and yet it fails to achieve comparable performance. We speculate that, since the f_{edge} is also trained with the pseudo-labels obtained from the G_{gcn} for the current target domain in a t^q_{dcl} step, it becomes susceptible to noise. Thus, in summary, the graph co-teaching is the most effective when the G_{gcn} is exploited to provide pseudo-labels only after each curriculum step.

Curriculum learning. We also study the effect of domain-aware curriculum learning in isolation from co-teaching. For that purpose, as shown in the Tab. 4.4, we start with the baseline model CDAN by treating all the target domains as one single domain. When the domain labels of the target are available, the baseline improves by 1.33%, indicating that the domain labels can

	Office-Home				
Model	Art	Clipart	Product	Real	**Avg**(%)
Source train	51.45	43.93	42.41	54.50	48.07
Baseline	50.70	50.78	47.95	57.63	51.77
Base.†	52.08	53.21	48.62	58.49	53.10
Base.†+PL	54.61	56.13	50.25	**61.04**	55.51
Base.† + DCL	**55.94**	**56.66**	**52.85**	60.18	**56.41**
Base.†+GCN‡	50.19	49.09	46.52	60.76	51.64
Base.†+GCN‡ + PL	54.52	57.60	53.20	65.49	57.70
CGCT	60.81	60.00	54.13	62.62	59.39
D-CGCT	**61.42**	**60.73**	**57.27**	**63.8**	**60.81**

Table 4.4: Ablation results of different baselines using ResNet-18 as backbone on Office-Home. **Baseline**: CDAN [96] model that combines all the target domains into a single target domain. "†" indicates the baseline models that use the domain labels of the target. **GCN‡**: the baseline model with the GCN as the single classification head. **PL**: using pseudo-labels.

indeed improve the performance of an adaptation model. To show the benefit of the DCL without co-teaching, we train the **Base**† + **DCL**, and it yields an average accuracy that is higher than the Base.† + PL counterpart. The advantage of using DCL is further amplified when coupled with the CGCT, where the D-CGCT outperforms all other baselines, including the CGCT. Due to the gradual adaptation, the D-CGCT also leads to the better cluster formation than the CGCT, as shown by the *t*-SNE visualization in the Fig. 4.5.

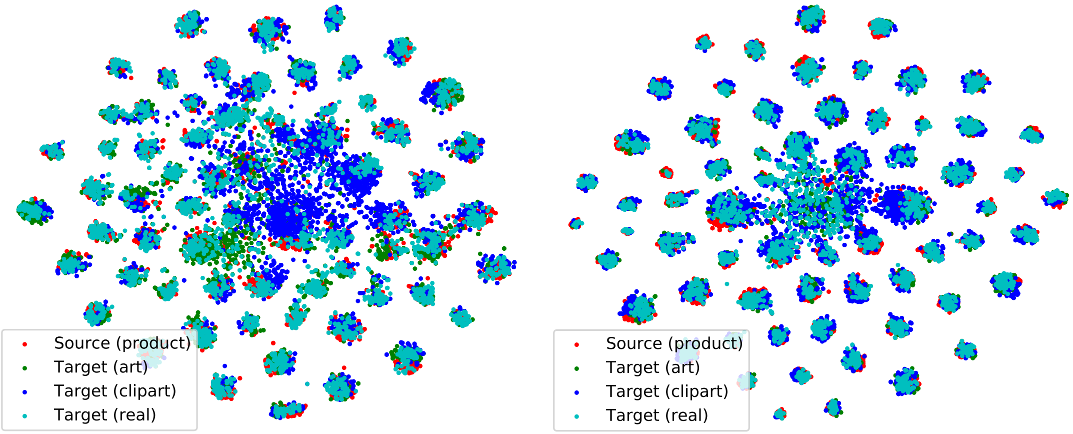
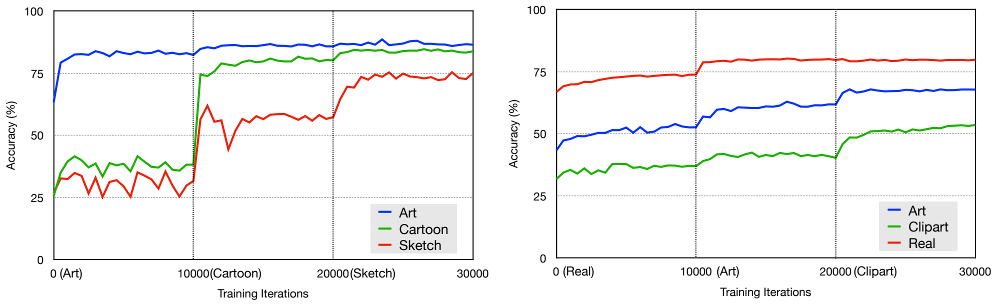


Figure 4.5: *t*-SNE plots of the feature embeddings with Product → *rest* in Office-Home. Left: CGCT. Right: D-CGCT.

To demonstrate that the order of target domains selection in the DCL indeed makes a difference, we consider a reverse-domain curriculum learning where the hardest domain is selected first, followed by the less hard ones. To this end, we train two models: i) Baseline†+DCL; and ii) Baseline†+Rev-DCL and compare their performances in the Fig. 4.7. In both the datasets we observe the same phenomenon that the reverse-curriculum being detrimental. This once again re-establishes the importance of the proposed DCL in the MTDA setting.

Additionally, to explain why the step-by-step adaptation in the proposed DCL better addresses the alleviation of the larger domain-shifts in the MTDA setting, we plot the classification accuracy with the D-CGCT in Fig. 4.6. As can be observed from the Fig. 4.6 (a), for Photo → *rest* setting in the PACS, when the adaptation first begins with the Art as target, the performance of the model on the *unseen* Cartoon domain simultaneously improves in the first 10k iterations (or the 1$^{\text{st}}$ curriculum step), despite the network not seeing any sample from the Cartoon domain. This phenomenon is even vividly noticeable in the second curriculum step, where the performance on the unseen Sketch largely increases when the Cartoon is selected for adaptation. This in other words means that the domain-shift between the source (Photo) and the farthest target (Sketch) has already been considerably reduced by the time the Sketch enters the adaptation stage (from 20k iterations on wards). Thus, we empirically demonstrate the prime reason behind the DCL achieving superior performance over other state-of-the-art MTDA methods. Similar observations can also be noticed for the Office-Home. We depict the Product → *rest* setting in the Fig. 4.6 (b).



(a) Photo → *rest* in the PACS (a) Product → *rest* in the Office-Home

Figure 4.6: The classification accuracy line plots with the D-CGCT using ResNet-50 as the backbone. At each indicated training iteration in the x-axis, a new target domain (shown in brackets) is selected for adaptation.

Setting	Model	Digits-five					Avg (%)
		mt → mm,sv ,sy,up	mm → mt,sv, sy,up	sv → mm,mt, sy,up	sy → mm,sv, mt,up	up → mm,sv, sy,mt	
Target Combined	Source only	26.9	56.0	67.2	73.8	36.9	52.2
	ADDA [159]	43.7	55.9	40.4	66.1	34.8	48.2
	DAN [98]	31.3	53.1	48.7	63.3	27.0	44.7
	GTA [145]	44.6	54.5	60.3	74.5	41.3	55.0
	RevGrad [38]	52.4	64.0	65.3	66.6	44.3	58.5
	AMEAN [20]	**56.2**	65.2	67.3	71.3	47.5	61.5
	CDAN [96]	53.0	76.3	65.6	81.5	**56.2**	66.5
	CGCT	54.3	**85.5**	**83.8**	**87.8**	52.4	**72.8**
Multi-Target	CDAN [96]	53.7	76.2	64.4	80.3	46.2	64.2
	CDAN + DCL	62.0	87.8	87.8	92.3	**63.2**	78.6
	D-CGCT	**65.7**	**89.0**	**88.9**	**93.2**	62.9	**79.9**

Table 4.5: Comparison with the state-of-the-art methods on the Digits-five. "Target Combined" indicates methods are performed on one source to one combined target domain. "Multi-Target" indicates methods are performed on one source to multi-target setting. Our proposed models are highlighted in bold.

4.4.3 Comparison with State-of-The-Art

We compare our proposed method and its variants with several state-of-the-art methods that are designed exclusively for the MTDA as well as the STDA methods that can be extended and used in the MTDA setting.

In Tab. 4.6 we report the numbers for Office-31 and Office-Home for single-target, target-combined and multi-target setting. The single-target setting denotes training single-source to single-target models, the target-combined means treating all the target domains as one aggregated target, while the multi-target setting comprise of training a single model for single-source to multiple-targets. As can be observed, in all the settings our proposed CGCT and D-CGCT outperform all the state-of-the-art methods. Specifically, for the Office-31, our CGCT without using domain labels is already 2.4% better than the HGAN [173], which is a MTDA method exploiting domain labels for feature aggregation with a single GCN classifier besides pseudo-labeling. This highlights the importance of having a co-teaching strategy with two classifiers and curriculum learning for counteracting the impact of noisy pseudo-labels in the GCN framework. We also observed that incorporating domain information following the proposed DCL strategy improves the performance in the Office-Home, with the D-CGCT achieving 5.5% improvement

		Office-31				Office-Home				
Setting	Model	Amazon	DSLR	Webcam	**Avg**(%)	Art	Clipart	Product	Real	**Avg**(%)
w/o Target	Source train	68.6	70.0	66.5	68.4	47.6	42.6	44.2	51.3	46.4
Single-Target	DAN [98]	79.5	80.3	81.2	80.4	56.1	54.2	51.7	63.0	56.3
	RevGrad [38]	80.8	82.5	83.2	82.2	58.3	55.4	52.8	63.9	57.6
	JAN [99]	85.0	83.0	85.6	84.3	58.7	57.0	53.1	64.3	58.3
	CDAN [96]	**91.4**	84.1	84.0	86.6	64.2	62.9	59.9	68.1	63.8
	CGCT (ours)	89.6	**85.5**	**87.6**	**87.6**	**67.9**	**68.7**	**62.3**	**70.7**	**67.4**
Target-Combined	DAN [98]	78.0	64.4	66.7	69.7	55.6	56.6	48.5	56.7	54.4
	RevGrad [38]	78.2	72.2	69.8	73.4	58.4	58.1	52.9	62.1	57.9
	JAN [99]	84.2	74.4	72.0	76.9	58.3	60.5	52.2	57.5	57.1
	CDAN [96]	93.6	80.5	81.3	85.1	59.5	61.0	54.7	62.9	59.5
	AMEAN [20]	90.1	77.0	73.4	80.2	64.3	65.5	59.5	66.7	64.0
	CGCT (ours)	**93.9**	**85.1**	**85.6**	**88.2**	**67.4**	**68.1**	**61.6**	**68.7**	**66.5**
Multi-Target	MT-MTDA [116]	87.9	83.7	84.0	85.2	64.6	66.4	59.2	67.1	64.3
	HGAN [173]	88.0	84.4	84.9	85.8	-	-	-	-	-
	CDAN+DCL (ours)	92.6	82.5	84.7	86.6	63.0	66.3	60.0	67.0	64.1
	D-CGCT (ours)	**93.4**	**86.0**	**87.1**	**88.8**	**70.5**	**71.6**	**66.0**	**71.2**	**69.8**

Table 4.6: Comparison with state-of-the-art methods on Office-31 and Office-Home. All methods use the ResNet-50 as the backbone. Single-Target indicates methods are performed on one source to one target setting. Target-Combined indicates methods are performed on one source to aggregated targets setting, while the Multi-Target indicates methods are performed on one source to multi-target setting.

		PACS						
Setting	Model	A → S	A → C	A → P	P → S	P → C	P → A	**Avg** (%)
Target Combined	MSTN [179]	70.4	71.2	96.2	**55.9**	**49.1**	70.8	68.9
	ADDA [159]	65.3	68.0	96.0	48.8	47.1	67.3	65.4
	CDAN [96]	56.8	61.1	95.9	55.7	53.8	49.4	62.1
	CGCT	**70.5**	**75.4**	**98.3**	44.6	44.3	**81.7**	**69.1**
Multi-Target	CDAN [96]	75.9	81.9	95.4	51.3	61.7	65.0	71.9
	HGAN [173]	72.1	78.3	97.7	70.8	62.8	78.8	76.8
	CDAN + DCL	68.7	89.0	98.8	61.2	**82.9**	**89.8**	81.7
	D-CGCT	**84.6**	**90.2**	**99.4**	**76.5**	82.4	88.6	**87.0**

Table 4.7: Comparison with the state-of-the-art methods on the PACS. All methods use the ResNet-50 as the backbone. "Target Combined" indicates methods are performed on one source to one combined target domain. "Multi-Target" indicates methods are performed on one source to multi-target setting. Our proposed models are highlighted in bold.

	DomainNet						
Model	Cli.	Inf.	Pai.	Qui.	Rea.	Ske.	**Avg**(%)
Source train	25.6	16.8	25.8	9.2	20.6	22.3	20.1
SE [31]	21.3	8.5	14.5	13.8	16.0	19.7	15.6
MCD [141]	25.1	19.1	27.0	10.4	20.2	22.5	20.7
DADA [123]	26.1	20.0	26.5	12.9	20.7	22.8	21.5
CDAN [96]	31.6	27.1	31.8	12.5	33.2	35.8	28.7
MCC [69]	33.6	30.0	32.4	13.5	28.0	35.3	28.8
CDAN + DCL	35.1	31.4	37.0	**20.5**	35.4	**41.0**	33.4
CGCT	36.1	**33.3**	35.0	10.0	39.6	39.7	32.3
D-CGCT	**37.0**	32.2	**37.3**	19.3	**39.8**	40.8	**34.4**

Table 4.8: Comparison with the state-of-the-art methods on DomainNet. All methods use the ResNet-101 as the backbone. The classification accuracy are reported for each *source→rest* direction, with each *source* domain being indicated in the columns. All the reported numbers are evaluated on the multi-target setting.

over MT-MTDA [116], a MTDA method that also utilizes domain labels. Finally, as can be seen from the Tab. 4.8, the D-CGCT advances the state-of-the-art results for the very challenging DomainNet dataset by a non-trivial margin of 5.6%. This further verifies the effectiveness of our proposed methods for addressing the MTDA.

In the Tab. 4.5, we report the state-of-the-art comparison on the Digits-five. For a fair comparison, we compare with the baselines reported in [20]. In both the target combined and multi-target settings, our proposed methods outperform all other baselines. For the PACS, reported in the Tab. 4.7, we notice that domain labels is very vital for mitigating multiple domain-shifts. For example, CDAN in the multi-target setting performs 9.8% better than its target combined counterpart. Similar trend can also be observed between our CGCT and D-CGCT, with the D-CGCT outperforming the former by a large margin.

Overcoming negative transfer. Careful inspection of the Tab. 4.6 tells us that the single-target DA methods always outperform the same STDA method when applied in the multi-target setting. For e.g., CDAN is 4.3% better in the single-target than in the multi-target setting. The drop in performance for the multi-target setting clearly hints at the fact that *negative transfer* [20, 123] is quite prevalent in the MTDA, despite having access to more data. Contrarily, our proposed CGCT when applied to both the settings fares equally well for the Office-Home and outperforms the single-target counterpart by 0.6% for the Office-31. This once again

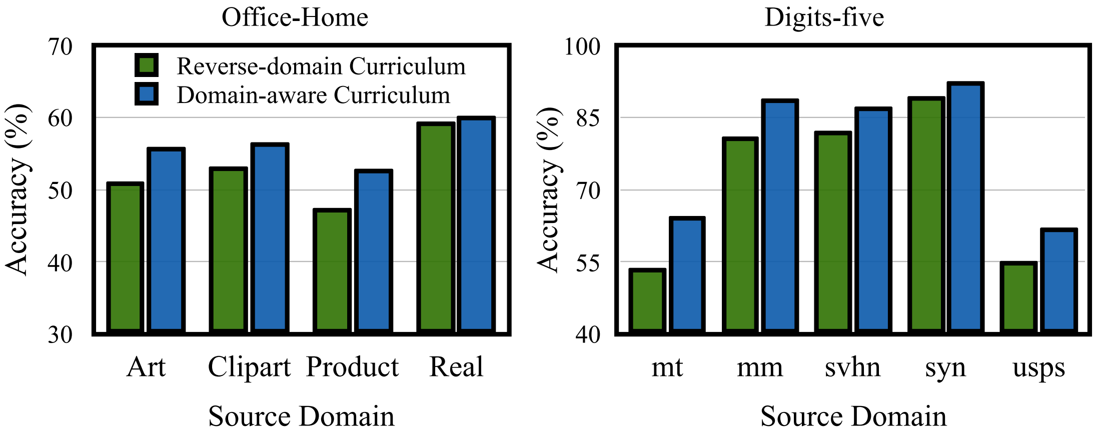
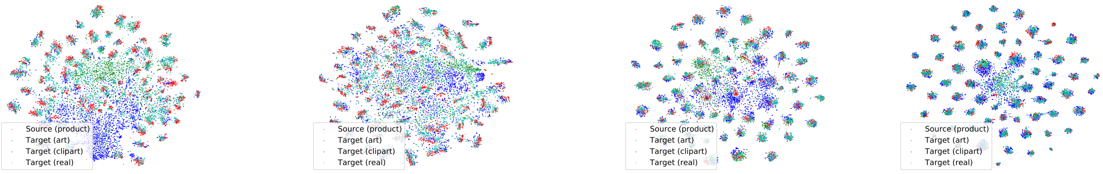


Figure 4.7: Comparison of the DCL with the *reverse*-domain curriculum model on Office-Home and Digits-Five. In the reverse-domain curriculum model the order of selection of target domains is exactly opposite to that of the DCL model.

shows that the design choices made in our CGCT and D-CGCT lead to learning more robust domain-invariant features and provide resilience against negative transfer.

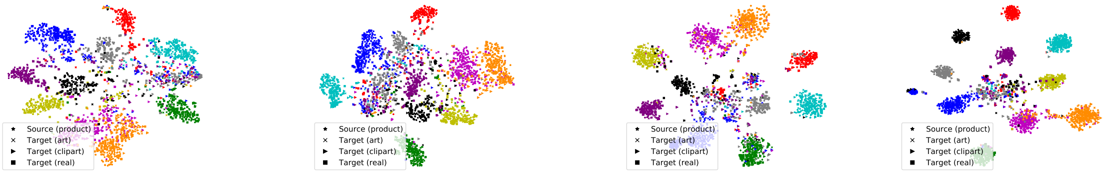


(a) CDAN [96] (b) CDAN (**w/** domain labels) (c) CGCT (d) D-CGCT

Figure 4.8: *t*-SNE plots of the feature embeddings for the Product → *rest* of the Office-Home. All the models use ResNet-50 as backbone. Each colour indicates a different domain. Digital zoom is recommended.

4.4.4 Visualization

In this section we visualize the features learned by our models and compare them with the baseline methods. The Fig. 4.8 depicts the *t*-SNE plots of the feature embeddings computed by feature extractor network (ResNet-50) for the direction Product → *rest* of the Office-Home. The plots in the Fig. 4.8 (c) and (d) demonstrate that the proposed CGCT and D-CGCT result in well clustered and discriminative features compared to CDAN baselines (see Fig. 4.8 (a) and



(a) CDAN [96] (b) CDAN (**w**/ domain labels) (c) CGCT (d) D-CGCT

Figure 4.9: *t*-SNE plots of the feature embeddings for the Product → *rest* of the Office-Home depicting only 10 randomly sampled classes. All the methods use ResNet-50 as backbone. Each colour indicates a different class while each shape represents a different domain. Digital zoom is recommended.

(b)). To better visualize the decision boundaries in the latent feature space, we select 10 classes, randomly from the Office-Home, and depict the *t*-SNE plots of the feature embeddings in the Fig. 4.9. It is can be seen that our models learn features that can be easily separated by a linear classifier, much easier than the CDAN models. In particular, the CDAN when using domain labels (see Fig. 4.9 (b)) produces more overlapping classes than our D-CGCT (see Fig. 4.9 (d)). Thus, when the domain labels are leveraged with our DCL strategy, the model produces features that are more discriminative, thereby leading to an improved performance in the MTDA.

4.5 Conclusions

In this chapter to address multi-target domain adaptation (MTDA), we proposed Curriculum Graph Co-Teaching (CGCT) that uses a graph convolutional network to perform robust feature aggregation across multiple domains, which is then trained with a co-teaching and curriculum learning strategy. To better exploit domain labels of the target we presented a Domain-aware curriculum (DCL) learning strategy that adapts easier target domains first and harder later, enabling a smoother feature alignment. Through extensive experiments we demonstrate that our proposed contributions handsomely outperform the state-of-the-art in the MTDA.

5

Uncertainty-aware Source-free Domain Adaptation

Source-free domain adaptation (SFDA) aims to adapt a classifier to an unlabelled target data set by only using a pre-trained source model. However, the absence of the source data and the domain shift makes the predictions on the target data unreliable. In this chapter we propose quantifying the uncertainty in the source model predictions and utilizing it to guide the target adaptation. For this, we construct a probabilistic source model by incorporating priors on the network parameters inducing a distribution over the model predictions. Uncertainties are estimated by employing a Laplace approximation and incorporated to identify target data points that do not lie in the source manifold and to down-weight them when maximizing the mutual information on the target data. Unlike recent works, our probabilistic treatment is computationally lightweight, decouples source training and target adaptation, and requires no specialized source training or changes of the model architecture. We show the advantages of uncertainty-guided SFDA over traditional SFDA in the closed-set and open-set settings and provide empirical evidence that our approach is more robust to strong domain shifts even without tuning.[1].

[1] The content of this chapter is based on the ECCV 2022 paper [134].

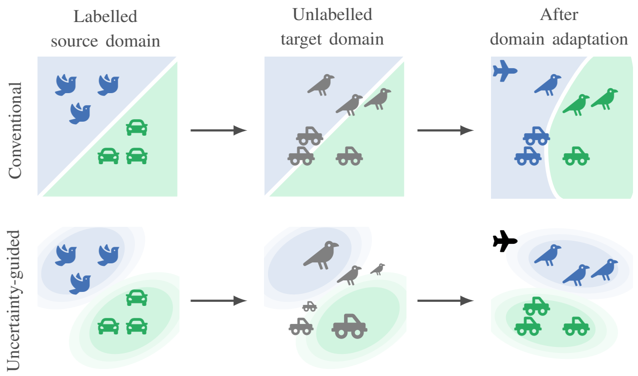


Figure 5.1: Illustrative sketch of source-free domain adaptation (SFDA) on a labelled source domain (🐦, 🚗) and an *unlabelled* target domain (🐦, 🚗) potentially containing additional classes (✈). The **top-row** shows conventional methods which ignore model uncertainties; the **bottom-row** shows our method which incorporates uncertainties about the predictive model, enabling uncertainty-guided SFDA that is more robust to distribution shifts

5.1 Introduction

Deep neural networks have proven to be very successful in a myriad of computer vision tasks such as categorization, detection, and retrieval. However, much of the success has come at the price of excessive human effort put into the manual data-labelling process. Since collecting annotated data can be prohibitive and impossible at times, domain adaptation (DA, see [24] for an overview) methods have gained increasing attention. They enable training on unlabelled target data by conjointly leveraging a previously labelled yet related source data set while mitigating *domain-shift* [157] between the two. Such methods predominantly comprise of minimizing statistical moments between distributions [98, 131, 152, 161], using adversarial objectives to maximize domain confusion [38, 159], or reconstructing data with generative methods [60].

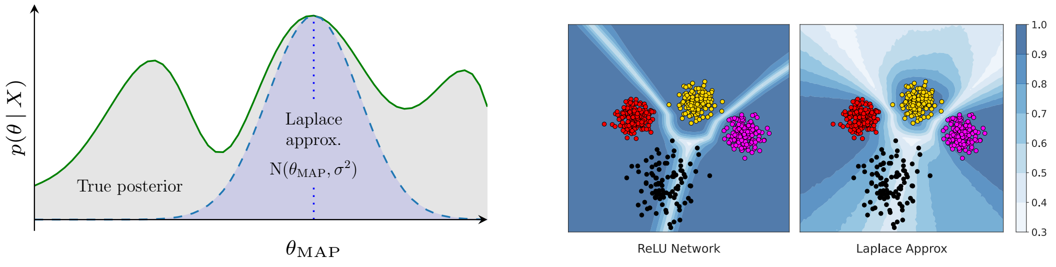
Albeit successful, the preceding methods mandate access to the source data set during the target adaptation phase as they require an estimate of the source distribution for the alignment. With the emergence of regulations on data privacy and bottleneck in data transmission for large data sets, access to the source data can not always be guaranteed. Thus, paving the way to a relatively new and more realistic DA setting, called *source-free* DA (SFDA, [24]), where the task is to adapt to the target data set when the only source of supervision is a source-trained model. SFDA facilitates maintaining data anonymity in privacy-sensitive applications (*e.g.*, surveillance or medical applications) and at the same time reduces data transmission and storage overhead. Towards this goal, recently, several SFDA methods have been proposed that utilize the hypotheses learned from the source data [77, 90, 155]. Notably, SHOT [90] – an information maximization (IM) [43] based SFDA method – has demonstrated to work reasonably well on DA benchmarks, sometimes outperforming traditional DA methods. While promising, these conventional SFDA techniques do not account for the uncertainty in the predictions of the source model on the target data. As a by-product, solely maximizing mutual information [43] on the target data can lead to erroneous decision surfaces (see Fig. 5.1 top).

This work argues that quantification of the uncertainty in predictions is essential in SFDA. Depending on the inductive biases of the model, the source model may predict incorrect target pseudo-labels with high confidence, *e.g.* due to the extrapolation property in ReLU networks [58] (see Fig. 5.2b left). In the literature, uncertainty-guided methods have been proposed in the context of traditional UDA and SFDA settings, employing Monte Carlo (MC) dropout to estimate the uncertainties in the model predictions [128, 182]. However, MC dropout requires specialized training and specialized model architecture, suffers from manual hyperparameter tuning [35], and is known to provide a poor approximation even for simple (*e.g.* linear) models [30, 117, 118].

In this work, we propose to construct a probabilistic source model by incorporating priors on the network parameters, inducing a distribution over the model predictions, on the last layer of the source model. This enables us to perform an efficient local approximation to the posterior using a *Laplace approximation* (LA, [104, 156]), see Fig. 5.2a. This principled Bayesian treatment leads to more robust predictions, especially when the target data set contains out-of-distribution (OOD) classes (see Fig. 5.1 bottom) or in case of strong domain shifts. Once the uncertainty in predictions is estimated, we selectively guide the target model to maximize the mutual information [43] in the target predictions. This alleviates the alignment of the target features with the wrong source hypothesis, resulting in a domain adaptation scheme that is robust to mild and strong domain shifts without tuning. We call our proposed method Uncertainty-guided Source-Free AdaptatioN (U-SFAN). Our approach requires no specialized source training or specialized architecture, opposed to exiting works (*e.g.* [81, 182]), introduces little computational overhead, and decouples source training and target adaptation.

We summarize our contributions as follows. *(i)* We emphasize the need to quantify uncertainty in the predictions for SFDA and propose to account for uncertainties by placing priors on the parameters of the source model. Our approach is computationally efficient by employing a last-layer Laplace approximation and greatly decouples the training of the source and target. *(ii)* We demonstrate that our proposed CGCT successfully guides the target adaptation without specialized loss functions or a specialised architecture. *(iii)* We empirical show the advantage of our method over SHOT [90] in the closed-set and the open-set setting for several benchmarks tasks and provide evidence for the improved robustness against mild and strong domain shifts.

5.2 Related Work



(a) Laplace approximation

(b) Out-of-distribution detection

Figure 5.2: (a) The Laplace approximation is mode-seeking and adapts to the local curvature around the mode θ_{MAP}. It does not necessarily capture the (intractable) full posterior, but gives a proxy for it, is principled, and efficient to evaluate. (b) Example of predictive uncertainty (un ▨ certain) captured by a ReLU network vs. a Laplace approximation that assigns higher uncertainty to inputs (∴) of an unseen class

Closed-set Domain Adaptation, often abbreviated as UDA, refers to the family of DA methods that aim to learn a classifier for an unlabelled target data set while simultaneously using the labelled source data set, which differ in their underlying data distributions. In the literature [164] mainly three categories of UDA methods can be found. First, discrepancy-based UDA methods aim to diminish the domain-shift between the two domains with maximum mean discrepancy (MMD, [98, 100, 161]), or with correlation alignment [111, 131, 152]. The second category of UDA methods exploits the adversarial objective [45] to promote domain confusion between the two data distributions by using domain discriminator [38, 96, 159]. Finally, the third category comprises reconstruction-based UDA methods [11, 41, 60] that casts data reconstruction as an auxiliary objective in order to ensure invariance in the feature space. However, these methods can only work in the presence of the source data set during the adaptation stage, which might be limited in practice due to data privacy or storage concerns.

Open-set Domain Adaptation (OSDA), originally proposed in [120], refers to the DA setting where both the domains have some shared and private classes, with explicit knowledge about the shared classes. However, such a setting was deemed impractical, and later Saito *et al.* [142] proposed the open-set setting where the source labels are a subset of the target labels. Thereon, several OSDA methods have been proposed which use image-to-image translations [177], progressive filtering [92], ensemble of multiple classifiers [33] and one-vs-all classifiers [139] to detect OOD samples. Similar to the UDA, the OSDA methods also require support from the source data to detect target private classes, which make them unsuitable for source-free DA.

Source-free Domain Adaptation (SFDA) aims to adapt a model to the unlabelled target domain when only the source model is available and the source data set is absent during target adaptation. Existing SFDA methods use pseudo-label refinement [1, 18, 90], latent source feature generation using variational inference [176], or disparity among an ensemble of classifiers [81]. Certain SFDA methods resort to *ad hoc* source training protocols to enable the source model to be adapted on the target data. For instance, [81] requires an ensemble of classifiers to be trained during source training so that the disparity among them could be utilized for target adaptation. Similarly, USFDA [77] requires artificially generated negative samples in the source training stage for the model to detect OOD samples. Such coupled source and target training procedures make these SFDA methods less viable for practical applications. On the other hand, our proposed CGCT does not require specialized source training except a computationally lightweight approximate inference, which can be done with a single pass of the source data during the source training phase. Moreover, unlike [1, 81], our CGCT works well on both closed-set and open-set SFDA without any *ad hoc* modifications.

Uncertainty Quantification in the form of Bayesian deep learning (*e.g.*, [70, 114]) is concerned with formalizing prior knowledge and representing uncertainty in model predictions, especially under domain-shift or out-of-distribution samples. Even though the Bayesian methodology gives an explicit way of formalizing uncertainty, computation is often intractable. Thus, approximate inference methods such as Monte Carlo (MC) dropout [34], deep ensembles [80, 166], other stochastic methods (*e.g.*, [105]), variational methods [7], or the Laplace approximation [129] are typically employed in practice. Prior work on DA with semantic segmentation

[182] and UDA [54, 79, 81, 128, 165] applied MC dropout or deep ensembles, respectively, for uncertainty quantification if DA. However, none of those above approaches can be considered practical for the more challenging source-free DA scenario as MC dropout, ensembles, and other stochastic methods do not lend themselves well to the source-free case. In particular, they either require retraining several models on the source, changing the model architecture or requiring a tailored learning procedure on the source data. Thus we take a Laplace approach which allows re-using the source model by linearizing around a point-estimate (see Fig. 5.2), which is *post hoc*, yet grounded in classical statistics [40].

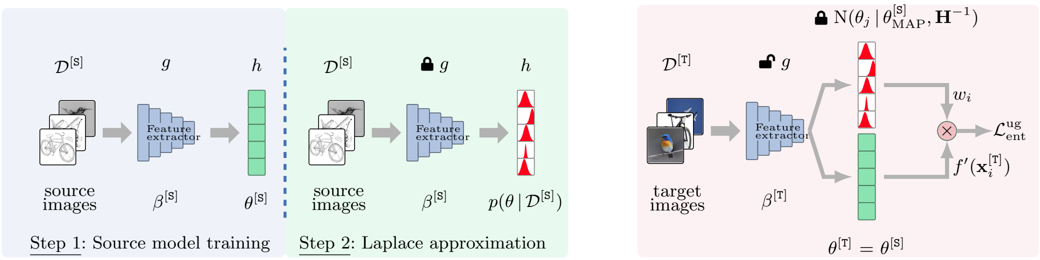
5.3 Methods

Problem Definition and Notation We are given a labelled source data set, having $n^{[\mathtt{S}]}$ instances, $\mathcal{D}^{[\mathtt{S}]} = \{(\mathbf{x}_i^{[\mathtt{S}]}, \mathbf{y}_i^{[\mathtt{S}]})\}_{i=1}^{n^{[\mathtt{S}]}}$, where $\mathbf{x}^{[\mathtt{S}]} \in \mathcal{X}^{[\mathtt{S}]}$ are D-dimensional inputs and $\mathbf{y}^{[\mathtt{S}]} \in \mathcal{Y}^{[\mathtt{S}]}$ where we assume K-dimensional one-hot encoded class labels, *i.e.*, $\mathcal{Y}^{[\mathtt{S}]} = \mathbb{B}^K$. Moreover, we have $n^{[\mathtt{T}]}$ unlabelled target observations $\mathcal{D}^{[\mathtt{T}]} = \{\mathbf{x}_j^{[\mathtt{T}]}\}_{j=1}^{n^{[\mathtt{T}]}}$, where $\mathbf{x}^{[\mathtt{T}]} \in \mathcal{X}^{[\mathtt{T}]}$ are D-dimensional unlabelled inputs. As in any DA scenario, the assumption made is that the marginal distributions of the source and the target are different, but the semantic concept represented through class labels does not change. Formally, we assume that $p(\mathbf{y}^{[\mathtt{S}]} \mid \mathbf{x}^{[\mathtt{S}]}) \approx p(\mathbf{y}^{[\mathtt{S}]} \mid \mathbf{x}^{[\mathtt{T}]})$ and $p(\mathbf{x}^{[\mathtt{S}]}) \neq p(\mathbf{x}^{[\mathtt{T}]})$.

In the SFDA scenario we further assume that the source data set is only available while learning the source function $f \colon \mathcal{X}^{[\mathtt{S}]} \to \mathcal{Y}^{[\mathtt{S}]}$ and becomes unavailable while adapting on the unlabelled data. The goal of SFDA is to adapt the source function f to the target domain solely by using the data in $\mathcal{D}^{[\mathtt{T}]}$. The resulting target function, denoted as $f' \colon \mathcal{X}^{[\mathtt{T}]} \to \mathcal{Y}^{[\mathtt{T}]}$, can then be used to infer the class assignment for $\mathbf{x}^{[\mathtt{T}]} \in \mathcal{X}^{[\mathtt{T}]}$. In this chapter we have considered two settings of the SFDA: i) *vanilla closed-set* SFDA where the label space of the source $\mathtt{S}$ and the target $\mathtt{T}$ is the same, $\mathrm{L}^{[\mathtt{S}]} = \mathrm{L}^{[\mathtt{T}]}$; and ii) *open-set* SFDA where the label space of the $\mathtt{S}$ is a subset of the $\mathtt{T}$, *i.e.*, $\mathrm{L}^{[\mathtt{S}]} \subset \mathrm{L}^{[\mathtt{T}]}$, and $\mathrm{L}^{[\mathtt{T}]} \setminus \mathrm{L}^{[\mathtt{S}]}$ are denoted as target private or OOD classes.

We model the source and target functions f with a neural network that is composed of two sub-networks: feature extractor g and hypothesis function h, such that $f = h \circ g$. The feature extractor g and the hypothesis function h are parameterized by parameters β and θ, respectively. During target adaptation, the model is initialized with parameters learned on $\mathcal{D}^{[\mathtt{S}]}$

and subsequently the feature extractor parameters are updated using backpropagation, *i.e.*, the hypothesis function is kept frozen.



(a) Source Training

(b) Target Adaptation

Figure 5.3: The pipeline for U-SFAN: (a) Initial source model training (1) and the additional step (2) of composing a Laplace approximation for assessing the posterior over model parameters, $p(\theta \mid \mathcal{D}^{[S]})$. (b) At target adaptation, we keep the posterior over the parameters fixed (🔒) and train g under a uncertainty-aware composite loss that weights samples according to predictive uncertainty

Overall Idea Our proposed method for SFDA operates in two stages. We begin the first stage (see Fig. 5.3a) by training a source model on the data set $\mathcal{D}^{[S]}$, which gives us the maximum-a-posteriori probability (MAP) estimate of the source network parameters ($\{\beta^{[S]}_{\mathrm{MAP}}, \theta^{[S]}_{\mathrm{MAP}}\}$). The second stage (see Fig. 5.3b) comprises of maximization of mutual information [43] in the predictions for the target inputs $\mathcal{D}^{[T]}$. However, due to the overconfidence of ReLU networks [58], maximizing mutual information for all inputs equally, including those that are far away from the source data, could be detrimental. To overcome this pathology, we derive a per-sample weight using the model's uncertainty and use it to modulate the mutual information objective in SHOT. To estimate the uncertainty in the predictions on the target data, we perform approximate posterior inference over the parameters of the hypothesis function, *i.e.*, $p(\theta^{[S]} \mid \mathcal{D}^{[S]})$. Inspired by recent works on approximate inference in Bayesian neural networks[75, 103, 156], we propose to estimate the posterior predictive distribution $p(\mathbf{y} \mid \mathbf{x}, \mathcal{D})$ using a Laplace approximation, introducing little computational overhead and without the need for specialized source training. We briefly describe the preliminaries to our approach in the following section.

5.3.1 Preliminaries

Liang *et al.* [90] proposed SHOT (Source HypOthesis Transfer) for the task of SFDA, where the goal is to find a parameterisation $\beta^{[T]}$ of the feature extractor g such that the distribution of latent features $\mathbf{z}^{[\mathrm{T}]} = g_{\beta^{[\mathrm{T}]}}(\mathbf{x}^{[\mathrm{T}]})$ matches the distribution of the latent source features. This enables that the target data can be accurately classified by the hypothesis function parameterized by $\theta^{[\mathrm{S}]}$. To this end, the authors address the SFDA task in two stages where the first and second stage comprise of source model training and maximizing the mutual information [43] between the latent representations and the classifier output, respectively.

The source model $f\colon \mathcal{X}^{[\mathrm{S}]} \to \mathcal{Y}^{[\mathrm{T}]}$ for a K-way classification task is learned using a label-smoothed cross-entropy objective [112], *i.e.*,

$$\mathcal{L}_{\text{src}} = -\,\mathbb{E}_{p(\mathbf{x}^{[\mathrm{S}]},\mathbf{y}^{[\mathrm{S}]})} \textstyle\sum_{k=1}^{K} \tilde{y}_k^{[\mathrm{S}]} \log \phi_k(f(\mathbf{x}^{[\mathrm{S}]})), \tag{5.1}$$

where $\phi_k(\mathbf{a}) = {}^{\exp(a_k)}/_{\sum_j \exp(a_j)}$ denotes the likelihood for the k^{th} component of the model output and $\tilde{y}_{i,k}^{[\mathrm{S}]} = y_{i,k}^{[\mathrm{S}]}(1-\alpha) + {}^{\alpha}/_{K}$ is the class label for the i^{th} label smoothed datum.

After the source training, the $\mathcal{D}^{[\mathrm{S}]}$ is discarded and the target adaptation is conducted on $\mathcal{D}^{[\mathrm{T}]}$ only. To adapt on the target domain, the target function f' is initialized based on the learned source function f and learned with the information maximization (IM) loss [43]. The IM loss ensures that the function mapping will produce one-hot predictions while at the same time enforcing diverse assignments, *i.e.*,

$$\mathcal{L}_{\text{ent}} {=} {-}\mathbb{E}_{p(\mathbf{x}^{[\mathrm{T}]})} \textstyle\sum_{k=1}^{K} \phi_k(f'(\mathbf{x}^{[\mathrm{T}]})) \log \phi_k(f'(\mathbf{x}^{[\mathrm{T}]})), \tag{5.2}$$

$$\mathcal{L}_{\text{div}} {=} \mathrm{D}_{\mathrm{KL}}(\hat{\mathbf{p}} \,\|\, K^{-1}\mathbf{1}_K) - \log K, \tag{5.3}$$

where $\mathbf{1}_K$ is a vector of all ones, and $\hat{p}_k = \mathbb{E}_{p(\mathbf{x}^{[\mathrm{T}]})}[\phi_k(f'(\mathbf{x}^{[\mathrm{T}]}))]$ is the expected network output for the k^{th} class. Intuitively, $\mathcal{L}_{\text{ent}}$ is in charge of making the network output one-hot, while $\mathcal{L}_{\text{div}}$ is responsible for equally partitioning the network prediction into K classes. In practice $\mathcal{L}_{\text{div}}$ operates on a mini-batch level. In this chapter we start from SHOT-IM to adapt to the target domain.

5.3.2 Uncertainty-guided Source-free DA

Distributional shift between source and target data sets causes the network outputs to differ, even for a same underlying semantic concept [24]. In a standard UDA scenario, where the source data is available during target adaptation, it is still possible to align the marginal distributions by using a quantifiable discrepancy metric. The task becomes more challenging in the SFDA scenario because it is not possible to align the target feature distribution to a reference (or source) distribution. Moreover, standard ReLU networks are known to yield overconfident predictions for data points which lie far away from the training (source) data [58]. In other words, the MAP estimates of a neural network has no notion of uncertainty over the learned weights. Thus, blindly trusting the source model predictions for $\mathbf{x}^{[\mathrm{T}]} \in \mathcal{D}^{[\mathrm{T}]}$ while performing information maximization [90] or entropy minimization [48] can potentially lead to misalignment of clusters between the source and target.

In this chapter we propose to incorporate the uncertainty of the neural network's weights into the predictions. This mandates a Bayesian treatment of the networks parameters (θ), which gives a posterior distribution over the model parameters by conditioning onto observed data ($\mathcal{D}$), *i.e.*, $p(\theta \mid \mathcal{D}) = \frac{p(\theta)\, p(\mathcal{D} \mid \theta)}{p(\mathcal{D})} \propto p(\theta)\, p(\mathcal{D} \mid \theta)$. The prediction of the network h_θ for an observation $\mathbf{x}$ is given by predictive posterior distribution, *i.e.*,

$$p(y_k \mid \mathbf{x}, \mathcal{D}) = \int_\theta \phi_k(h_\theta(\mathbf{x}))\, p(\theta \mid \mathcal{D})\, \mathrm{d}\theta. \tag{5.4}$$

Note that the posterior $p(\theta \mid \mathcal{D})$ in Eq. 5.4 does not have an analytical solution in general and need to be approximated. For this, we employ a local approximation to the posterior using a Laplace approximation (LA, [156]). The LA locally approximates the true posterior using a multivariate Gaussian distribution centred at a local maximum and with covariance matrix given by the inverse of the Hessian $\mathbf{H}$ of the negative log-posterior, *i.e.*, $p(\theta \mid \mathcal{D}) \approx \mathrm{N}(\theta \mid \theta_{\mathrm{MAP}}, ^{-1}\mathbf{H})$ with $\mathbf{H} := -\nabla^2_\theta \log p(\theta \mid \mathcal{D}) \mid_{\theta_{\mathrm{MAP}}}$. Note that the LA is a principled and simple, yet effective, approach to approximate posterior inference stemming from a second-order Taylor expansion of the true posterior around θ_{MAP}. Next we will discuss LA in the context of SFDA.

Bayesian Source Model Generation In the source training stage (see Fig. 5.3a), by optimizing Eq. 5.1, we obtain a MAP estimate of the weights for our source model, comprising β_{MAP} and θ_{MAP} for g and h, respectively. Since f is often modelled by a very deep neural network (*e.g.*, ResNet-50), computing the Hessian can be computationally infeasible owing to the large number of parameters. So we make another simplification by applying a Bayesian treatment only to hypothesis function h, known as the *last-layer* Laplace approximation [75]. This gives us a probabilistic source hypothesis with posterior distribution $p(\theta \mid \mathcal{D}^{[\mathrm{S}]})$ for the parameters. The feature extractor g remains deterministic. Formally, let $\mathbf{z} = g_{\beta^{[\mathrm{S}]}}(\mathbf{x})$ be the latent feature representation from the feature extractor. Following Eq. 5.4, the predictive posterior distribution is given as:

$$p(y_k \mid \mathbf{z}, \mathcal{D}^{[\mathrm{S}]}) \approx \int_\theta \phi_k(h_\theta(\mathbf{z}))\, \mathrm{N}(\theta \mid \theta_{\mathrm{MAP}}, ^{-1}\mathbf{H})\, \mathrm{d}\theta. \tag{5.5}$$

While the *last-layer* LA greatly simplifies the computational overhead for large networks, the Hessian can still be difficult to compute in the case the number of classes is large. To simplify computations, we assume that $\mathbf{H}$ can be Kronecker-factored $\mathbf{H} := \mathbf{V} \otimes \mathbf{U}$ and the resulting approximation is referred to as Kronecker-factored Laplace approximation (KFLA, [129]). Such probabilistic treatment allows us to quantify uncertainty in the predictions for data points from the target with little computational overhead. Also, the LA can be readily computed using a single forward pass of the source data through the network. Next, we describe how to use the uncertainty estimates during target adaptation.

Uncertainty-guided Information Maximization Upon completion of the source model generation stage, we exploit the probabilistic source hypothesis to guide the information maximization in the target adaptation stage. SHOT puts equal confidence on all the target predictions and do not make any distinction for the target feature that lies outside of the source manifold. We emphasize that in case of strong domain-shift naively maximizing the IM loss could lead to cluster misalignment. For that reason, we propose to weigh the entropy minimization objective (Eq. 5.2) with a weight which is proportional to the certainty in the target predictions (see Fig. 5.3b). To get the per-sample weight for a $\mathbf{x}^{[\mathtt{T}]}$ we need to compute the predictive posterior distribution, as outlined in Eq. 5.5. However, exactly solving the integration is intractable in many cases and we, therefore, resort to Monte Carlo (MC) integration. Let $\mathbf{z}^{[\mathtt{T}]} = g_{\beta^{[\mathtt{T}]}}(\mathbf{x}^{[\mathtt{T}]})$, the approximate predictive posterior distributions is:

$$p(y_k \,|\, \mathbf{z}^{[\mathtt{T}]}, \mathcal{D}^{[\mathtt{S}]}) \approx \frac{1}{M} \sum_{j=1}^{M} \phi_k\left(h_{\theta_j}(\mathbf{z}^{[\mathtt{T}]})\right), \tag{5.6}$$

where $\theta_j \sim \mathrm{N}(\theta_j \,|\, \theta_{\mathrm{MAP}}, ^{-1}\mathbf{H})$ and M denotes the number of MC steps. To encourage low entropy predictions we additionally scale the outputs of the hypothesis by $1/\tau$, where $0 < \tau \leq 1$. The final weight of each observation $\mathbf{x}_i^{[\mathtt{T}]}$ is then computed as $w_i = \exp(-H)$ where H denotes the entropy of the predictive mean. The *uncertainty-guided entropy loss* is then given as:

$$\mathcal{L}_{\mathrm{ent}}^{\mathrm{ug}} = -\mathbb{E}_{p(\mathbf{x}^{[\mathtt{T}]})} \sum_{k=1}^{K} w\, \sigma_k(f'(\mathbf{x}^{[\mathtt{T}]})) \log \sigma_k(f'(\mathbf{x}^{[\mathtt{T}]})). \tag{5.7}$$

The final training objective is then given as: $\mathcal{L}_{\text{U-SFAN}} = (1-\gamma)\mathcal{L}_{\mathrm{ent}}^{\mathrm{ug}} + \gamma\, \mathcal{L}_{\mathrm{div}}$.

How does this differ from conventional uncertainty estimation? The importance and advantages of adopting a Laplace approximation (LA) over Monte Carlo (MC) dropout to estimate uncertainty in SFDA can be summarized as follows: *(i)* LA does not require specialized network architecture (*e.g.*, dropout layers), loss function, or re-training (as in MC dropout) to estimate predictive uncertainties. This greatly decouples the source training from target adaptation, which is essential to be applicable in SFDA; *(ii)* To have well-calibrated uncertainties, MC dropout requires a grid search over the dropout probabilities [35], a prohibitive operation in deep neural networks, especially as the future target data is not available at source training. LA is a more principled approach that does not require a grid search, making it better suited for SFDA. *(iii)* LA is computationally lightweight since it requires just a single forward pass of the source data through the network after the source training to estimate the posterior over the parameters of the sub-network. *(iv)* LA does not impact the training time during target adaptation because, unlike MC dropout, only a single forward pass is needed to quantify the predictive uncertainties. Because LA employs a Gaussian approximation to the posterior, MC integration is cheap and efficient to compute. *(v)* As used in our work, LA estimates the full posterior over the weights and biases, while MC dropout can only account for the uncertainties over the weights [34] and is known to be a poor approximation to the posterior [30, 117, 118]. *(vi)* LA preserves the decision boundary induced by the MAP estimate, which is not the case for MC dropout [75]. In summary, our contribution goes beyond the uncertainty re-weighting scheme [89, 91] commonly used in UDA, while carrying many advantages over existing works.

5.4 Experiments

We conduct experiments on four standard DA benchmarks: OFFICE31 [138], OFFICE-HOME [163], VISDA-C [125], and the large-scale DOMAINNET [122] (**0.6 million** images). For the experiments in the open-set DA setting we follow the split of [90] for shared and target-private classes.

Evaluation protocol We report the classification accuracy for every possible pair of $source \mapsto target$ directions, except for the VISDA-C where we are only concerned with the transfer from *synthetic* ↦ *real* domain. For the open-set experiments, following the evaluation protocol in [90], we report the OS accuracy which includes the per-class accuracy of the known and the unknown class and is computed as $\text{OS} = \frac{1}{K+1} \sum_{k=1}^{K+1} \text{acc}_k$, where $k = \{1, 2, \ldots, K\}$ denote the shared classes and $(K + 1)^{\text{th}}$ is the target-private or OOD classes. This metric is preferred over the known class accuracy, $\text{OS}^* = \frac{1}{K} \sum_{k=1}^{K} \text{acc}_k$, as it does not take into account the OOD classes.

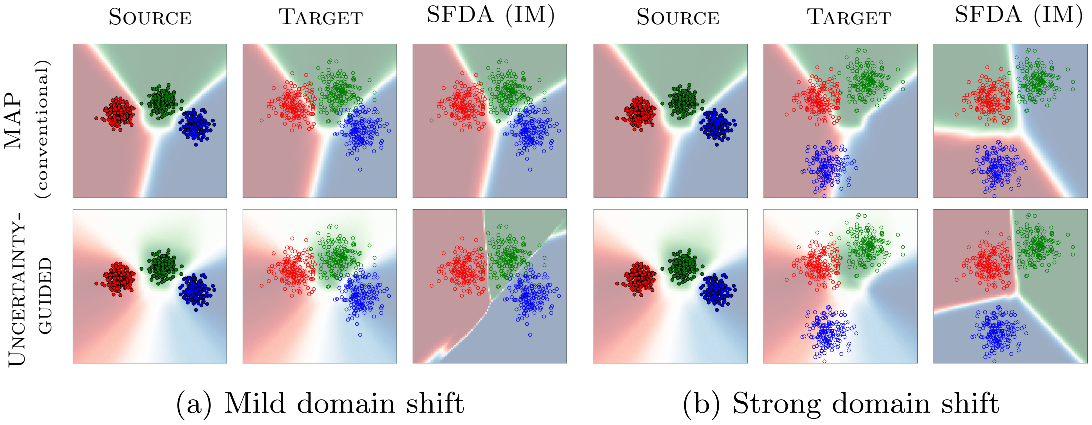
Implementation details We adopted the network architectures used in the SFDA literature, which are ResNet-50 or ResNet-101 [57]. Following [90], we added a bottleneck layer containing 256 neurons which is then followed by a batch normalization layer. The network finally ends with a weight normalized linear classifier that is kept frozen during the target adaptation. For computing the KFLA we use the PyTorch package of Dangel *et al.* [26]. Upon acceptance, the code for U-SFAN will be made available on GitHub.

Hyperparameter selection We re-use the hyperparameters from the baseline of [90], *e.g.*, the standard optimization technique for training such as SGD with an initial learning rate of 10^{-2} and 10^{-3} for ResNet-50 and ResNet-101, respectively. The learning rate is decayed by power decay [37]. We used the a batch size of 64 and we set $\alpha = 0.1$ and $\gamma = 0.5$. Exclusive to our method, we set the prior precision in LA equal to the weight decay, *i.e.* $5 \cdot 10^{-4}$, and set the temperature $\tau = 0.4$ for all our experiments.

5.4.1 Ablation Studies

In this section we experimentally and visually show the issues with minimizing the IM under different varieties of domain-shift. We also show how these can be overcome by incorporating uncertainty during target adaptation.

Limitations of IM loss. As discussed in Sec. 5.3.2, conventional SFDA methods that rely on optimizing the IM loss on the unlabelled target data, *e.g.* SHOT, are prone to misalignment of the target data with the source hypothesis under strong domain shift. To visually demonstrate this phenomenon, we design an experiment of a 3-way classification task on toy data (see Fig. 5.4). We choose a 2D toy data for this demonstration since it is not feasible to plot higher dimensional data without dimensionality reduction techniques. Given a set of source data points, belonging to three classes, we simulate two kinds of domain-shift: *mild* shift (Fig. 5.4a) and *strong* shift (Fig. 5.4b). In the case of mild shift, the target data points stay very close to the

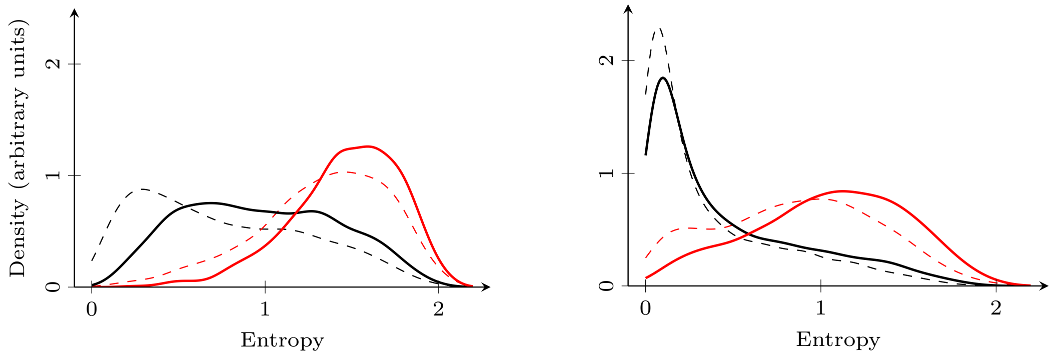


(a) Mild domain shift (b) Strong domain shift

Figure 5.4: Comparison of conventional IM (MAP) with our uncertainty-guided IM on target data under mild and strong domain-shift. The solid • vs. hollow ∘ circles represent the source and the target data, respectively. Each class is colour coded and the decision boundaries are shaded with the corresponding colours. Under strong domain-shift, IM, when used with a MAP estimate, finds a **completely flipped** decision boundary. U-SFAN finds the decision boundary by down-weighting the *far away* target data

source manifold, and the conventional approach (only using the MAP estimate) can classify a majority of target data points without the need of adaptation. Whereas, in the case of strong shift, the target data points for the blue class, in particular, shift drastically away from the source points. The source model based on the MAP estimate missclassifies most of the target data points with high confidence. On the other hand, our uncertainty-guided source model remains certain only for those target points which lie within the source support and assigns low certainty otherwise (proportional to the strength of colours depicting the decision surface in Fig. 5.4), robustifying the adaptation on the target data in case of domain shift.

Given such a set-up, we optimize the IM loss (*i.e.* SHOT-IM) for both the conventional and the uncertainty-guided source models. In the case of mild shift, both can reliably partition the target data points under the right decision surfaces (see Fig. 5.4 (right)). This is intuitive because the decision boundary of the target model already passes through the low-density regions. Hence, the optimization of the IM loss leads to correct target classification with both methods. However, when the domain shift is more substantial, the conventional approach results in *completely flipped* decision boundaries. This happens because most blue target points fall under the red decision surface, and thus, the IM loss assigns them to class ‘red’. On the contrary, our uncertainty-guided approach down-weights the blue points, and *safely* optimizes the IM loss as the model is uncertain about the class assignment for those points Fig. (5.4b (right)). This protects from major changes in the decision boundaries and allows the optimization to find the correct decision boundaries for the target data. Therefore, highlighting the importance of having a notion of uncertainty in the model predictions during adaptation. We will see Subsequently, we show that this intuition also holds well for real-world data sets where our U-SFAN offers more robustness when the data set becomes challenging.



(a) Before adaption

(b) After adaption

Figure 5.5: Entropy density plots for CIFAR9 → STL9 in the closed-set SFDA setting using the MAP estimate (- - - correct, - - - incorrect) or our approach (—— correct,—— incorrect). Our uncertainty-guided SFDA approach places less mass on low-entropy incorrect samples before and after adaptation

Importance of probabilistic outputs. To gain further insights, we visualize the entropy density plots of the source model predictions before and after adaptation with conventional (MAP estimate) and uncertainty-guided models on an image data set (CIFAR [76] as source data set and STL [23] as target), see Fig. 5.5. As seen in Fig. 5.5a, the MAP estimate has lower entropy predictions for both the correct and incorrect predictions, when compared to our uncertainty-guided model. Reduced over-confidence for our approach is expected before the adaptation phase, however, it is non-trivial that this behavior also bears in the post-adaptation phase. The reduced over-confident allows our U-SFAN to down-weight the incorrect predictions during target adaptation, resulting in improved target accuracy over SHOT-IM (77.04% for U-SFAN vs 75.69% for SHOT-IM). This effect can be noticed in Fig. 5.5b where U-SFAN has overall higher entropy incorrect predictions, which is desirable in SFDA.

To further understand the contribution of our uncertainty-guided re-weighting, we run an ablation where the approximate posterior distribution of our method (Eq. 5.7) is replaced by a

Method	Source-Only	SHOT-IM [90]	SHOT-IM + Ent. weighting	U-SFAN (ours)
Avg. Acc.	60.3	70.5	71.2	**71.8**

Table 5.1: Comparison of model performance using entropy weighting during target adaptation on the Office-Home data set. The weights computed using the LA is more beneficial than the weights computed with a MAP network

Method	A→C	A→P	A→R	C→A	C→P	C→R	P→A	P→C	P→R	R→A	R→C	R→P	Avg.
ResNet-50	34.9	50.0	58.0	37.4	41.9	46.2	38.5	31.2	60.4	53.9	41.2	59.9	46.1
DANN [38]	45.6	59.3	70.1	47.0	58.5	60.9	46.1	43.7	68.5	63.2	51.8	76.8	57.6
DWT [131]	50.3	72.1	77.0	59.6	69.3	70.2	58.3	48.1	77.3	69.3	53.6	82.0	65.6
CDAN [96]	50.7	70.6	76.0	57.6	70.0	70.0	57.4	50.9	77.3	70.9	56.7	81.6	65.8
SAFN [170]	52.0	71.7	76.3	64.2	69.9	71.9	63.7	51.4	77.1	70.9	57.1	81.5	67.3
SHOT-IM [90]	55.4	76.6	80.4	66.9	74.3	75.4	65.6	54.8	80.7	73.7	58.4	83.4	70.5
LSC [172]	57.9	78.6	81.0	66.7	77.2	77.2	65.6	56.0	82.2	72.0	57.8	83.4	71.3
U-SFAN (Ours)	**58.5**	78.6	81.1	66.6	75.2	77.9	66.3	**57.9**	80.6	73.6	**61.4**	84.1	71.8
A^{2}Net[168]	58.4	**79.0**	**82.4**	67.5	**79.3**	78.9	**68.0**	56.2	**82.9**	**74.1**	60.5	**85.0**	**72.8**
SHOT [90]	57.1	78.1	81.5	**68.0**	78.2	78.1	67.4	54.9	82.2	73.3	58.8	84.3	71.8
U-SFAN+ (Ours)	57.8	77.8	81.6	67.9	77.3	**79.2**	67.2	54.7	81.2	73.3	60.3	83.9	71.9

Table 5.2: Comparison of the classification accuracy on the Office-Home for the closed-set setting using ResNet-50. High overall performance signifies *milder* distributional shift between domains. The improvement of U-SFAN upon SHOT is moderate, but competitive with respect to A^{2}Net[168], which requires a complex training objective

weight computed from a point estimate from a MAP source model. This model is denoted as SHOT-IM + ENT. WEIGHTING in Tab. 5.1. We observe that such weighting scheme indeed improves the performance over SHOT-IM. However, it still lacks behind our proposed U-SFAN which uses weights computed from the uncertainty-guided model. This clearly shows that the improvement in performance with U-SFAN is not simply caused by the re-weighting but also due to better identification of target samples that are not well explained under the source model.

5.4.2 State-of-the-art Comparison

Closed-set experiments. We compare our U-SFAN with UDA and SFDA methods on multiple data sets for the closed-set setting. As can be seen from Tab. 5.2 and Tab. 5.4 we improve the performance over majority of the baselines. Especially, we consistently improve over SHOT-IM with our method. We also combine the nearest centroid pseudo-labelling, used in SHOT [90], with U-SFAN (indicated as U-SFAN+ in Tab. 5.2 and Tab. 5.3), and we find that it further helps improving the performance. Notably, the recently proposed A^{2}Net [168] (which just addresses

METHOD	ACC.
ResNet-101	52.4
CDAN+BSP [19]	75.9
SAFN [170]	76.1
SHOT-IM† [90]	80.3
U-SFAN (Ours)	81.2
3C-GAN [86]	81.6
A^{2}Net[168]	**84.3**
SHOT† [90]	82.4
U-SFAN+ (Ours)	82.7

Table 5.3: Comparison of the classification accuracy on the VISDA-C for the closed-set DA, pertaining to the *Synthetic* → *Real* direction, using ResNet-101. † indicates the numbers of [90] that are obtained using the official code from the authors. Note that several SFDA methods perform equally well for VISDA-C, hinting at saturating performance

METHOD	A→D	A→W	D→A	D→W	W→A	W→D	AVG.
ResNet-50	68.9	68.4	62.5	96.7	60.7	99.3	76.1
DANN [38]	79.7	82.0	68.2	96.9	67.4	99.1	82.2
DAN [98]	78.6	80.5	63.6	97.1	62.8	99.6	80.4
SAFN [170]	90.7	90.1	73.0	98.6	70.2	99.8	87.1
CDAN [96]	92.9	94.1	71.0	98.6	69.3	100.	87.7
SHOT-IM [90]	90.6	91.2	72.5	98.3	71.4	**99.9**	87.3
U-SFAN (Ours)	91.8	92.3	**75.8**	97.7	**74.4**	99.8	88.6
SHOT [90]	94.0	90.1	74.7	**98.4**	74.3	**99.9**	88.6
U-SFAN+ (Ours)	**94.2**	**92.8**	74.6	98.0	**74.4**	99.0	**88.8**

Table 5.4: Comparison of the classification accuracy on the OFFICE31 for the closed-set SFDA using ResNet-50. Results on the small-scale OFFICE31 are known to be saturated. The visual appearance between the domains do not vary much, thus making the domain shift *milder*

closed-set SFDA) outperforms our U-SFAN in a couple of data sets, but uses a combination of several loss functions. Interplay of multiple losses can be hard to tune in practice. On the other hand, our method is simpler, more versatile and works for both the SFDA settings. Given the performance of the SFDA baseline methods in OFFICE-HOME and VISDA-C are relatively high and closer to each other, the domain shift can be considered milder with respect to more challenging data set like DOMAIN-NET.

SOURCE	SHOT-IM [90]	U-SFAN
CLIPART	25.04	**30.88**
INFOGRAPH	21.58	**26.44**
PAINTING	23.89	**29.91**
QUICKDRAW	**10.76**	10.44
REAL	21.74	**29.32**
SKETCH	28.87	**29.99**
AVG.	21.98	**26.13**

Table 5.5: Comparison of the average accuracy on the DOMAINNET for the closed-set SFDA using ResNet-50. The SOURCE column indicates the domain where the source model has been trained. The data set being challenging (exhibiting *strong* domain-shift), the improvement with our U-SFAN over [90] is substantial

METHOD	A→C	A→P	A→R	C→A	C→P	C→R	P→A	P→C	P→R	R→A	R→C	R→P	AVG.
ResNet-50	53.4	52.7	51.9	69.3	61.8	74.1	61.4	64.0	70.0	78.7	71.0	74.9	65.3
ATI-λ[120]	55.2	52.6	53.5	69.1	63.5	74.1	61.7	64.5	70.7	79.2	72.9	75.8	66.1
OpenMax [5]	56.5	52.9	53.7	69.1	64.8	74.5	64.1	64.0	71.2	80.3	73.0	76.9	66.7
STA [92]	58.1	53.1	54.4	71.6	69.3	81.9	63.4	65.2	74.9	85.0	75.8	80.8	69.5
SHOT-IM [90]	62.5	77.8	83.9	60.9	73.4	79.4	64.7	58.7	83.1	69.1	62.0	82.1	71.5
SHOT [90]	**64.5**	**80.4**	**84.7**	63.1	**75.4**	**81.2**	65.3	59.3	**83.3**	69.6	**64.6**	**82.3**	72.8
U-SFAN (Ours)	62.9	77.9	84.0	**67.9**	74.6	79.6	**68.8**	**61.3**	**83.3**	**76.0**	63.9	**82.3**	**73.5**

Table 5.6: Comparison of the OS classification accuracy on the OFFICE-HOME for the open-set setting using ResNet-50. U-SFAN improves over SHOT without the need for nearest-centroid pseudo-labelling in the case of open-set SFDA

When we compare U-SFAN with SHOT-IM on the challenging SFDA benchmark DOMAIN-NET the advantage of our U-SFAN over SHOT-IM becomes imminent, see Tab. 5.5, which is in line with the ablation study in Sec. 5.4.1. Different from the previous data sets, the difficulty in mitigating domain-shift for DOMAIN-NET is evident from the low overall performance of both SHOT-IM and U-SFAN. This data set can be considered as an instance where strong domain-shift may manifest in the real world. The substantial improvement in the performance of U-SFAN over SHOT-IM for DOMAIN-NET demonstrates that incorporating the uncertainty in the model's predictions plays a crucial role in SFDA. The conventional approach may overfit to noisy model predictions for challenging data sets, leading to poor performance. Whereas, U-SFAN can capture the uncertainty in predictions and down-weight the impact of noisy predictions by attending less to such samples during target adaptation.

Open-set experiments. Following the work SHOT [90] we also conduct experiments on OFFICE-HOME dataset in the open-set setting. We report the OS classification accuracy metric in the Tab. 5.6, which includes the unknown class (see the beginning of Sec. 5.4). While some previous works on open-set adaptation adopt the H-score as a metric to evaluate open-set methods, it has been discussed in OVANet [139] that the H-score metric can put more weight on the open-set classes when the number of such samples are much lower compared to the closed-set classes. For this reason we report the OS score which weighs equally all the classes in the target domain. It can observed that our U-SFAN outperforms SHOT by a clear margin, showing the effectiveness in finding out-of-distribution samples.

5.5 Conclusions

In this chapter, we demonstrated the need for uncertainty quantification in SFDA and proposed U-SFAN that leveraging it during target adaptation. Our uncertainty-guided SFDA approach employs a Laplace approximation to the posterior, does not require specialized source training, and allows for efficient computation of predictive uncertainties. Our experiments showed that down-weighting *distant* target data points in our novel uncertainty-weighted IM loss alleviates the misalignment of target data with the source hypothesis. We ran experiments on closed and open-set SFDA settings and show that U-SFAN consistently improves upon the existing methods. Moreover, U-SFAN has shown to be robust under mild distribution shifts and shows promising results even under severe distribution shifts without additional tuning.

While we mainly focused on the popular IM-based SFDA methods, our proposed uncertainty-guided adaptation is also applicable to other SFDA frameworks, *e.g.*, neighbourhood clustering [172] or extensions to the multi-source SFDA problem. Moreover, the principles we build upon are general, interpretable, and have strong backing in classical statistics. We believe that uncertainty-guided SFDA will become a backbone tool for future methods in DA that generalize over different problem domains, are less sensitive to the training setup, and will provide good results without extensive *ad hoc* tuning to each problem.

6

Final Remarks

In this doctoral thesis we presented and discussed several methods for domain adaptation under different settings, mainly aimed at image classification, using deep learning techniques. Domain adaptation allows to bridge the domain gap between training and testing distributions and hence holds the key in making deep learning models generalizable and deployable in real world applications.

We started our analysis with the simplest case of closed-set unsupervised domain adaptation by proposing *Domain Whitening Transform* layers to align the marginal feature distributions between the source and target domains. To better leverage the unlabelled target data we also presented the *Min Entropy Consensus* loss to encourage the network to minimize entropy and make consistent predictions on two perturbed versions of a single target image. We showed that our proposed contributions can subsume several well known paradigms of approaches commonly followed in UDA.

Next we presented a generative approach for tackling multi-source domain adaptation by proposing the *TriGAN* framework that synthesizes target-like source images for training a target classifier. The main idea behind this framework is to use a universal generator that projects the image features onto a space where only the dependence from the content is kept, and then re-project this invariant representation onto the pixel space using the target domain and style. Having a universal generator for all the source and target domain pairs circumvents training multiple generators and also allows to use all of the source domains to train the single network.

We also addressed a more realistic DA setting of multi-target domain adaptation where we proposed the *Curriculum Graph Co-teaching* framework to learn an unified feature space through a graph convolutional network, which is trained in a co-teaching fashion to curb noisy pseudo-labels. When the domain labels of the target domains are available we proposed the *Domain Curriculum Learning* strategy that first adapts on the easier target domains and then the harder ones. We empirically showed that the order of adaptation is important to obtain better pseudo-labels and prevents negative transfer while learning with multiple target domains.

Finally, we addressed a even more challenging UDA setting called source-free domain adaptation where only a pre-trained source model is available while adapting on a desired target domain. Due to domain shift and lack of source data the source model predictions on target data can be unreliable. Hence, we constructed a probabilistic framework *U-SFAN* that is equipped with a notion of uncertainty. This uncertainty is estimated using Laplace Approximation that is then used to re-weight the target data while optimizing the information maximization loss. Through extensive experiments we showed that uncertainty-aware models are more robust to stronger domain shift.

6.1 Future Research Directions

This final section includes a short overview of the main challenges for domain adaptation and depicts future research directions that could come out from this work. The DA sub-field has overseen a plethora of works from the academic world and have shown to work well in several benchmarks. But when it comes to real world the simplified training assumptions, which are commonly followed in the academic setting, do not hold anymore. For instance, the target data in the real world comes in continuous streams and can not always be stored due to memory limitations. Under such cases, the traditional DA learning techniques may fail or may not be as effective as compared to offline methods. This calls for methods that can learn from unsupervised streams of target data. Another noticeable peculiarity in a vast majority of DA methods is the assumption of the knowledge about the network architecture. In many cases pre-trained models can be proprietary and can come as black box models. It would then require rethinking the adaptation strategy. Also, with the privacy guidelines becoming more rigorous, collecting target data from various end users may hit a brick wall for the practitioners. Thus adaptation needs to be performed in a decentralized fashion, which our proposed CGCT is unable to do. Finally, the DA methods have not exploited the large corpus of paired image-text data sets to build stronger source models to be used in adaptation. Such foundation models such as CLIP [126] have shown remarkable performance in various downstream tasks and offers promise to the DA community. Using stronger pre-trained models trained with captions can help reduce the domain gap which is otherwise not possible with traditional learning techniques.

Bibliography

[1] Waqar Ahmed, Pietro Morerio, and Vittorio Murino. Adaptive pseudo-label refinement by negative ensemble learning for source-free unsupervised domain adaptation. *arXiv preprint arXiv:2103.15973*, 2021. 83

[2] Asha Anoosheh, Eirikur Agustsson, Radu Timofte, and Luc Van Gool. Combogan: Unrestrained scalability for image domain translation. In *CVPR*, 2018. 34, 49

[3] Pinkesh Badjatiya, Shashank Gupta, Manish Gupta, and Vasudeva Varma. Deep learning for hate speech detection in tweets. In *Proceedings of the 26th international conference on World Wide Web companion*, pages 759–760, 2017. 1

[4] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine learning*, 79(1):151–175, 2010. 2

[5] Abhijit Bendale and Terrance E Boult. Towards open set deep networks. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition (CVPR)*, pages 1563–1572, 2016. 95

[6] Yoshua Bengio, Jérôme Louradour, Ronan Collobert, and Jason Weston. Curriculum learning. In *Proc. ICML*, 2009. 59

[7] David M Blei, Alp Kucukelbir, and Jon D McAuliffe. Variational inference: A review for statisticians. *Journal of the American statistical Association*, 112(518):859–877, 2017. 83

[8] Avrim Blum and Tom Mitchell. Combining labeled and unlabeled data with co-training. In *Proc. CLT*, 1998. 57

[9] Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. Unsupervised pixel-level domain adaptation with gans. In *CVPR*, 2017. 14

[10] Konstantinos Bousmalis, Nathan Silberman, David Dohan, Dumitru Erhan, and Dilip Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *CVPR*, 2017. 53

[11] Konstantinos Bousmalis, George Trigeorgis, Nathan Silberman, Dilip Krishnan, and Dumitru Erhan. Domain separation networks. In *NIPS*, 2016. 24, 25, 82

[12] Zhangjie Cao, Mingsheng Long, Jianmin Wang, and Michael I Jordan. Partial transfer learning with selective adversarial networks. In *Proc. CVPR*, 2018. 59

[13] Fabio Maria Carlucci, Lorenzo Porzi, Barbara Caputo, Elisa Ricci, and Samuel Rota Bulò. Autodial: Automatic domain alignment layers. In *Proc. ICCV*, 2017. 3, 12, 13, 14, 17, 18, 22, 23,

24, 25, 26, 32, 56, 59

[14] Fabio Maria Carlucci, Lorenzo Porzi, Barbara Caputo, Elisa Ricci, and Samuel Rota Bulo. Just dial: Domain alignment layers for unsupervised domain adaptation. In *Proc. ICIAP*, 2017. 3, 56

[15] Woong-Gi Chang, Tackgeun You, Seonguk Seo, Suha Kwak, and Bohyung Han. Domain-specific batch normalization for unsupervised domain adaptation. In *Proc. CVPR*, 2019. 3, 56, 59

[16] Chaoqi Chen, Weiping Xie, Wenbing Huang, Yu Rong, Xinghao Ding, Yue Huang, Tingyang Xu, and Junzhou Huang. Progressive feature alignment for unsupervised domain adaptation. In *Proc. CVPR*, 2019. 59

[17] Qingchao Chen, Yang Liu, Zhaowen Wang, Ian Wassell, and Kevin Chetty. Re-weighted adversarial adaptation network for unsupervised domain adaptation. In *Proc. CVPR*, 2018. 59

[18] Weijie Chen, Luojun Lin, Shicai Yang, Di Xie, Shiliang Pu, Yueting Zhuang, and Wenqi Ren. Self-supervised noisy label learning for source-free unsupervised domain adaptation. *arXiv preprint arXiv:2102.11614*, 2021. 83

[19] Xinyang Chen, Sinan Wang, Mingsheng Long, and Jianmin Wang. Transferability vs. discriminability: Batch spectral penalization for adversarial domain adaptation. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1081–1090, 2019. 94

[20] Ziliang Chen, Jingyu Zhuang, Xiaodan Liang, and Liang Lin. Blending-target domain adaptation by adversarial meta-adaptation networks. In *Proc. CVPR*, 2019. 3, 56, 57, 59, 63, 68, 69, 70, 74, 75, 76

[21] Wonwoong Cho, Sungha Choi, David Keetae Park, Inkyu Shin, and Jaegul Choo. Image-to-image translation via group-wise deep whitening-and-coloring transformation. In *CVPR*, 2019. 34

[22] Yunjey Choi, Minje Choi, Munyoung Kim, Jung-Woo Ha, Sunghun Kim, and Jaegul Choo. StarGAN: Unified generative adversarial networks for multi-domain image-to-image translation. In *CVPR*, 2018. 31, 34, 35, 46, 47, 49, 50

[23] Adam Coates, Andrew Ng, and Honglak Lee. An analysis of single-layer networks in unsupervised feature learning. In *Proceedings of the fourteenth international conference on artificial intelligence and statistics*, 2011. 13, 20, 92

[24] Gabriela Csurka. A comprehensive survey on domain adaptation for visual applications. *Domain Adaptation in Computer Vision Applications*, pages 1–35, 2017. 80, 86

[25] Gabriela Csurka, editor. *Domain Adaptation in Computer Vision Applications*. Advances in Computer Vision and Pattern Recognition. Springer, 2017. 11, 14

[26] Felix Dangel, Frederik Kunstner, and Philipp Hennig. Backpack: Packing more into backprop. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2020. 90

[27] Dariusz Dereniowski and Marek Kubale. Cholesky factorization of matrices in parallel and ranking of graphs. In *International Conference on Parallel Processing and Applied Mathematics*, 2003. 34

[28] Dariusz Dereniowski and Kubale Marek. Cholesky factorization of matrices in parallel and ranking of graphs. In *5th Int. Conference on Parallel Processing and Applied Mathematics*, 2004. 17

[29] Vincent Dumoulin, Jonathon Shlens, and Manjunath Kudlur. A learned representation for artistic style. In *ICLR*, 2016. 48

[30] Andrew Foong, David Burt, Yingzhen Li, and Richard Turner. On the expressiveness of approximate inference in bayesian neural networks. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 15897–15908, 2020. 81, 89

[31] Geoff French, Michal Mackiewicz, and Mark Fisher. Self-ensembling for visual domain adaptation. *ICLR*, 2018. 12, 13, 14, 18, 19, 21, 23, 24, 25, 26, 76

[32] Jerome Friedman, Trevor Hastie, and Robert Tibshirani. *The elements of statistical learning*, volume 1. 2001. 13, 20, 42, 68

[33] Bo Fu, Zhangjie Cao, Mingsheng Long, and Jianmin Wang. Learning to detect open classes for universal domain adaptation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 567–583, 2020. 83

[34] Yarin Gal and Zoubin Ghahramani. Dropout as a Bayesian approximation: Representing model uncertainty in deep learning. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1050–1059, 2016. 83, 89

[35] Yarin Gal, Jiri Hron, and Alex Kendall. Concrete dropout. In *Advances in Neural Information Processing Systems (NIPS)*, pages 3581–3590, 2017. 81, 89

[36] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. *ICML*, 2015. 14, 26, 33, 42, 46

[37] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1180–1189, 2015. 90

[38] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *JMLR*, 2016. 3, 21, 24, 25, 26, 42, 45, 53, 56, 59, 60, 68, 74, 75, 80, 82, 93, 94

[39] Leon A Gatys, Alexander S Ecker, and Matthias Bethge. Image style transfer using convolutional neural networks. In *CVPR*, 2016. 37, 39

[40] Andrew Gelman, John B. Carlin, Hal S. Stern, David B. Dunson, Aki Vehtari, and Donald B. Rubin. *Bayesian Data Analysis*. Chapman and Hall/CRC, Boca Raton, FL, third edition, 2013. 84

[41] Muhammad Ghifary, W Bastiaan Kleijn, Mengjie Zhang, David Balduzzi, and Wen Li. Deep reconstruction-classification networks for unsupervised domain adaptation. In *ECCV*, 2016. 24, 25, 26, 82

[42] Behnam Gholami, Pritish Sahu, Ognjen Rudovic, Konstantinos Bousmalis, and Vladimir Pavlovic. Unsupervised multi-target domain adaptation: An information theoretic approach. *TIP*, 2020. 57, 59

[43] Ryan Gomes, Andreas Krause, and Pietro Perona. Discriminative clustering by regularized information maximization. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2010. 81, 85, 86

[44] Boqing Gong, Yuan Shi, Fei Sha, and Kristen Grauman. Geodesic flow kernel for unsupervised domain adaptation. In *CVPR*, 2012. 31, 41, 42

[45] Ian Goodfellow, Jean Pouget-Abadie, Mehdi Mirza, Bing Xu, David Warde-Farley, Sherjil Ozair, Aaron Courville, and Yoshua Bengio. Generative adversarial nets. In *NIPS*, 2014. 30, 33, 59, 82

[46] Marco Gori, Gabriele Monfardini, and Franco Scarselli. A new model for learning in graph domains. In *Proc. IJNN*, 2005. 60

[47] Y. Grandvalet and Y. Bengio. Semi-supervised learning by entropy minimization. In *NIPS*, 2004. 14

[48] Yves Grandvalet, Yoshua Bengio, et al. Semi-supervised learning by entropy minimization. In

Advances in Neural Information Processing Systems (NeurIPS), pages 281–296, 2005. 87

[49] Gregory Griffin, Alex Holub, and Pietro Perona. Caltech-256 object category dataset. 2007. 42

[50] Sorin Grigorescu, Bogdan Trasnea, Tiberiu Cocias, and Gigel Macesanu. A survey of deep learning techniques for autonomous driving. *Journal of Field Robotics*, 37(3):362–386, 2020. 1

[51] Philip Haeusser, Thomas Frerix, Alexander Mordvintsev, and Daniel Cremers. Associative domain adaptation. In *ICCV*, 2017. 12, 13, 24, 25

[52] P. Haeusser, T. Frerix, A. Mordvintsev, and D. Cremers. Associative domain adaptation. In *ICCV*, 2017. 14

[53] Bo Han, Quanming Yao, Xingrui Yu, Gang Niu, Miao Xu, Weihua Hu, Ivor Tsang, and Masashi Sugiyama. Co-teaching: Robust training of deep neural networks with extremely noisy labels. In *Proc. NeurIPS*, 2018. 60, 67, 68

[54] Ligong Han, Yang Zou, Ruijiang Gao, Lezi Wang, and Dimitris Metaxas. Unsupervised domain adaptation via calibrating uncertainties. In *CVPR Workshops*, 2019. 84

[55] Trevor Hastie, Robert Tibshirani, Jerome H Friedman, and Jerome H Friedman. *The elements of statistical learning: data mining, inference, and prediction*, volume 2. Springer, 2009. 1

[56] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Delving deep into rectifiers: Surpassing human-level performance on imagenet classification. In *Proceedings of the IEEE international conference on computer vision*, pages 1026–1034, 2015. 1

[57] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *CVPR*, 2016. 22, 25, 43, 45, 69, 70, 90

[58] Matthias Hein, Maksym Andriushchenko, and Julian Bitterwolf. Why relu networks yield high-confidence predictions far away from the training data and how to mitigate the problem. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 41–50, 2019. 81, 85, 87

[59] Martin Heusel, Hubert Ramsauer, Thomas Unterthiner, Bernhard Nessler, and Sepp Hochreiter. Gans trained by a two time-scale update rule converge to a local nash equilibrium. In *NIPS*, 2017. 44, 49

[60] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 1989–1998, 2018. 41, 80, 82

[61] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *Proc. ICML*, 2017. 3, 30, 33, 53, 56, 59

[62] Haoshuo Huang, Qixing Huang, and Philipp Krahenbuhl. Domain transfer through deep activation matching. In *ECCV*, 2018. 24, 25

[63] Lei Huang, Dawei Yang, Bo Lang, and Jia Deng. Decorrelated batch normalization. In *CVPR*, 2018. 15, 17, 19, 20, 24

[64] Xun Huang, Ming-Yu Liu, Serge Belongie, and Jan Kautz. Multimodal unsupervised image-to-image translation. In *ECCV*, 2018. 48

[65] Wei-Chih Hung, Varun Jampani, Sifei Liu, Pavlo Molchanov, Ming-Hsuan Yang, and Jan Kautz. Scops: Self-supervised co-part segmentation. In *CVPR*, 2019. 40

[66] Sergey Ioffe and Christian Szegedy. Batch normalization: Accelerating deep network training by

reducing internal covariate shift. In *ICML*, 2015. 12, 32, 36, 37, 48

[67] Phillip Isola, Jun-Yan Zhu, Tinghui Zhou, and Alexei A Efros. Image-to-image translation with conditional adversarial networks. In *CVPR*, 2017. 33

[68] Max Jaderberg, Karen Simonyan, Andrew Zisserman, et al. Spatial transformer networks. In *NIPS*, 2015. 40

[69] Ying Jin, Ximei Wang, Mingsheng Long, and Jianmin Wang. Minimum class confusion for versatile domain adaptation. In *Proc. ECCV*, 2020. 59, 76

[70] Alex Kendall and Yarin Gal. What uncertainties do we need in Bayesian deep learning for computer vision? In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 5574–5584, 2017. 83

[71] A. Kessy, A. Lewin, and K. Strimmer. Optimal whitening and decorrelation. *The American Statistician*, 2017. 17

[72] Diederik P Kingma and Jimmy Ba. Adam: A method for stochastic optimization. In *ICLR*, 2014. 21, 44

[73] Thomas N. Kipf and Max Welling. Semi-supervised classification with graph convolutional networks. In *Proc. ICLR*, 2017. 57, 60, 61

[74] J. Kohler, H. Daneshmand, A. Lucchi, M. Zhou, K. Neymeyr, and T. Hofmann. Towards a Theoretical Understanding of Batch Normalization. *arXiv:1805.10694*, 2018. 19

[75] Agustinus Kristiadi, Matthias Hein, and Philipp Hennig. Being Bayesian, even just a bit, fixes overconfidence in relu networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 5436–5446, 2020. 85, 87, 89

[76] Alex Krizhevsky. Learning multiple layers of features from tiny images. Master's thesis, University of Tronto, Toronto, Canada, 2009. 92

[77] Jogendra Nath Kundu, Naveen Venkat, R Venkatesh Babu, et al. Universal source-free domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 4544–4553, 2020. 81, 83

[78] Vinod Kumar Kurmi, Vipul Bajaj, Venkatesh K Subramanian, and Vinay P Namboodiri. Curriculum based dropout discriminator for domain adaptation. *arXiv*, 2019. 66

[79] Vinod Kumar Kurmi, Shanu Kumar, and Vinay P Namboodiri. Attending to discriminative certainty for domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 491–500, 2019. 84

[80] Balaji Lakshminarayanan, Alexander Pritzel, and Charles Blundell. Simple and scalable predictive uncertainty estimation using deep ensembles. In *Advances in Neural Information Processing Systems (NeurIPS)*, 2017. 83

[81] Qicheng Lao, Xiang Jiang, and Mohammad Havaei. Hypothesis disparity regularized mutual information maximization. In *Proceedings of the AAAI Conference on Artificial Intelligence*, 2021. 81, 83, 84

[82] Yann LeCun, Léon Bottou, Yoshua Bengio, and Patrick Haffner. Gradient-based learning applied to document recognition. *Proceedings of the IEEE*, 86(11):2278–2324, 1998. 13, 20, 42, 68

[83] Yann LeCun, Léon Bottou, Genevieve B. Orr, and Klaus-Robert Müller. Efficient backprop. In *Neural Networks: Tricks of the Trade - Second Edition*, pages 9–48. 2012. 19

[84] June-Goo Lee, Sanghoon Jun, Young-Won Cho, Hyunna Lee, Guk Bae Kim, Joon Beom Seo, and

Namkug Kim. Deep learning in medical imaging: general overview. *Korean journal of radiology*, 18(4):570–584, 2017. 1

[85] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *Proc. ICCV*, 2017. 68, 69

[86] Rui Li, Qianfen Jiao, Wenming Cao, Hau-San Wong, and Si Wu. Model adaptation: Unsupervised domain adaptation without source data. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 9641–9650, 2020. 94

[87] Yijun Li, Chen Fang, Jimei Yang, Zhaowen Wang, Xin Lu, and Ming-Hsuan Yang. Universal style transfer via feature transforms. *Advances in neural information processing systems*, 30, 2017. 34

[88] Yanghao Li, Naiyan Wang, Jianping Shi, Jiaying Liu, and Xiaodi Hou. Revisiting batch normalization for practical domain adaptation. *arXiv:1603.04779*, 2016. 12, 13, 14, 17, 23, 32, 59

[89] Jian Liang, Ran He, Zhenan Sun, and Tieniu Tan. Exploring uncertainty in pseudo-label guided unsupervised domain adaptation. *Pattern Recognition*, 96:106996, 2019. 89

[90] Jian Liang, Dapeng Hu, and Jiashi Feng. Do we really need to access the source data? source hypothesis transfer for unsupervised domain adaptation. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 6028–6039, 2020. 3, 81, 82, 83, 85, 87, 89, 90, 93, 94, 95, 96

[91] Jian Liang, Yunbo Wang, Dapeng Hu, Ran He, and Jiashi Feng. A balanced and uncertainty-aware approach for partial domain adaptation. In *Proceedings of European Conference on Computer Vision (ECCV)*, pages 123–140. Springer, 2020. 89

[92] Hong Liu, Zhangjie Cao, Mingsheng Long, Jianmin Wang, and Qiang Yang. Separate to adapt: Open set domain adaptation via progressive separation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 2927–2936, 2019. 83, 95

[93] Ming-Yu Liu, Thomas Breuel, and Jan Kautz. Unsupervised image-to-image translation networks. In *NIPS*, 2017. 53

[94] Ming-Yu Liu and Oncel Tuzel. Coupled generative adversarial networks. In *Proc. NeurIPS*, 2016. 3, 24, 25, 30, 33, 53, 56, 59

[95] Ziwei Liu, Zhongqi Miao, Xingang Pan, Xiaohang Zhan, Dahua Lin, Stella X Yu, and Boqing Gong. Open compound domain adaptation. In *Proc. CVPR*, 2020. 59

[96] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. In *Proc. NeurIPS*, 2018. 3, 56, 59, 61, 63, 72, 74, 75, 76, 77, 78, 82, 93, 94

[97] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. *arXiv:1705.10667v2*, 2018. 22, 25, 27

[98] Mingsheng Long and Jianmin Wang. Learning transferable features with deep adaptation networks. In *Proc. ICML*, 2015. 3, 14, 25, 32, 46, 47, 56, 59, 74, 75, 80, 82, 94

[99] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Deep transfer learning with joint adaptation networks. *ICML*, 2017. 14, 25, 32, 59, 75

[100] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Deep transfer learning with joint adaptation networks. In *Proceedings of the International Conference on Machine Learning (ICML)*, pages 2208–2217, 2017. 82

[101] Yadan Luo, Zijian Wang, Zi Huang, and Mahsa Baktashmotlagh. Progressive graph learning for

open-set domain adaptation. *In Proc. ICML*, 2020. 60, 62, 67

[102] Xinhong Ma, Tianzhu Zhang, and Changsheng Xu. Gcan: Graph convolutional adversarial network for unsupervised domain adaptation. In *Proc. CVPR*, 2019. 60

[103] David JC MacKay. A practical Bayesian framework for backpropagation networks. *Neural Computation*, 4(3):448–472, 1992. 85

[104] David JC MacKay. *Information Theory, Inference and Learning Algorithms*. Cambridge University Press, 2003. 81

[105] Wesley J Maddox, Pavel Izmailov, Timur Garipov, Dmitry P Vetrov, and Andrew Gordon Wilson. A simple baseline for Bayesian uncertainty in deep learning. In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 13132–13143, 2019. 83

[106] Massimiliano Mancini, Samuel Rota Bulò, Barbara Caputo, and Elisa Ricci. Adagraph: Unifying predictive and continuous domain adaptation through graphs. In *CVPR*, 2019. 14

[107] Massimiliano Mancini, Hakan Karaoguz, Elisa Ricci, Patric Jensfelt, and Barbara Caputo. Kitting in the wild through online domain adaptation. *IROS*, 2018. 13

[108] Massimiliano Mancini, Lorenzo Porzi, Samuel Rota Bulò, Barbara Caputo, and Elisa Ricci. Boosting domain adaptation by discovering latent domains. *CVPR*, 2018. 12, 14, 32, 59

[109] Yishay Mansour, Mehryar Mohri, and Afshin Rostamizadeh. Domain adaptation with multiple sources. In *NIPS*, 2009. 30, 33

[110] Takeru Miyato and Masanori Koyama. cGANs with projection discriminator. In *ICLR*, 2018. 36, 40, 43, 44

[111] Pietro Morerio, Jacopo Cavazza, and Vittorio Murino. Minimal-entropy correlation alignment for unsupervised deep domain adaptation. *ICLR*, 2018. 12, 13, 14, 24, 25, 32, 59, 82

[112] Rafael Müller, Simon Kornblith, and Geoffrey Hinton. When does label smoothing help? In *Advances in Neural Information Processing Systems (NeurIPS)*, pages 4694–4703, 2019. 86

[113] Zak Murez, Soheil Kolouri, David Kriegman, Ravi Ramamoorthi, and Kyungnam Kim. Image to image translation for domain adaptation. In *CVPR*, 2018. 30, 33, 53

[114] Radford M Neal. *Bayesian Learning for Neural Networks*. Springer Science & Business Media, 2012. 83

[115] Yuval Netzer, Tao Wang, Adam Coates, Alessandro Bissacco, Bo Wu, and Andrew Y Ng. Reading digits in natural images with unsupervised feature learning. In *NIPS workshop on deep learning and unsupervised feature learning*, 2011. 13, 20, 42, 68

[116] Le Thanh Nguyen-Meidine, Madhu Kiran, Jose Dolz, Eric Granger, Atif Bela, and Louis-Antoine Blais-Morin. Unsupervised multi-target domain adaptation through knowledge distillation. In *Proc. WACV*, 2021. 75, 76

[117] Ian Osband. Risk versus uncertainty in deep learning: Bayes, bootstrap and the dangers of dropout. In *NIPS workshop on Bayesian deep learning*, 2016. 81, 89

[118] Ian Osband, John Aslanides, and Albin Cassirer. Randomized prior functions for deep reinforcement learning. In *Advances in Neural Information Processing Systems (NIPS)*, pages 8617–8629, 2018. 81, 89

[119] Sinno Jialin Pan, Qiang Yang, et al. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2010. 11

[120] Pau Panareda Busto and Juergen Gall. Open set domain adaptation. In *Proceedings of the IEEE*

International Conference on Computer Vision (ICCV), pages 754–763, 2017. 83, 95

[121] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. *ICCV*, 2019. 3, 30, 33, 45, 46, 47

[122] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. In *Proc. ICCV*, 2019. 58, 68, 69, 89

[123] Xingchao Peng, Zijun Huang, Ximeng Sun, and Kate Saenko. Domain agnostic learning with disentangled representations. *arXiv*, 2019. 57, 59, 69, 76

[124] Xingchao Peng and Kate Saenko. Synthetic to real adaptation with generative correlation alignment networks. In *Proc. WACV*, 2018. 3, 32, 56, 59

[125] 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. 89

[126] Alec Radford, Jong Wook Kim, Chris Hallacy, Aditya Ramesh, Gabriel Goh, Sandhini Agarwal, Girish Sastry, Amanda Askell, Pamela Mishkin, Jack Clark, et al. Learning transferable visual models from natural language supervision. In *International Conference on Machine Learning*, pages 8748–8763. PMLR, 2021. 98

[127] Sayan Rakshit, Biplab Banerjee, Gemma Roig, and Subhasis Chaudhuri. Unsupervised multi-source domain adaptation driven by deep adversarial ensemble learning. In *DAGM*, 2019. 33

[128] Tobias Ringwald and Rainer Stiefelhagen. Unsupervised domain adaptation by uncertain feature alignment. *The British Machine Vision Conference (BMVC)*, 2020. 81, 84

[129] Hippolyt Ritter, Aleksandar Botev, and David Barber. A scalable laplace approximation for neural networks. In *Proceedings of the International Conference on Learning Representations (ICLR)*, 2018. 83, 88

[130] Subhankar Roy, Evgeny Krivosheev, Zhun Zhong, Nicu Sebe, and Elisa Ricci. Curriculum graph co-teaching for multi-target domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 5351–5360, 2021. 55

[131] Subhankar Roy, Aliaksandr Siarohin, Enver Sangineto, Samuel Rota Bulo, Nicu Sebe, and Elisa Ricci. Unsupervised domain adaptation using feature-whitening and consensus loss. *CVPR*, 2019. 11, 31, 32, 37, 38, 49, 56, 59, 80, 82, 93

[132] Subhankar Roy, Aliaksandr Siarohin, Enver Sangineto, Nicu Sebe, and Elisa Ricci. Trigan: Image-to-image translation for multi-source domain adaptation. *Machine vision and applications*, 2021. 29, 59

[133] Subhankar Roy, Aliaksandr Siarohin, and Nicu Sebe. Unsupervised domain adaptation using full-feature whitening and colouring. In *Proc. ICIAP*, 2019. 59

[134] Subhankar Roy, Martin Trapp, Andrea Pilzer, Juho Kannala, Nicu Sebe, Elisa Ricci, and Arno Solin. Uncertainty-guided source-free domain adaptation. In *European Conference on Computer Vision*, 2022. 79

[135] Olga Russakovsky, Jia Deng, Hao Su, Jonathan Krause, Sanjeev Satheesh, Sean Ma, Zhiheng Huang, Andrej Karpathy, Aditya Khosla, Michael Bernstein, et al. Imagenet large scale visual recognition challenge. *IJCV*, 2015. 45

[136] Paolo Russo, Fabio Maria Carlucci, Tatiana Tommasi, and Barbara Caputo. From source to target and back: symmetric bi-directional adaptive gan. In *Proc. CVPR*, 2018. 3, 14, 24, 25, 26, 30, 33, 41, 53, 56, 59

[137] H. Ney S. Wiesler. A convergence analysis of log-linear training. In *NIPS*, 2011. 19

[138] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *Proc. ECCV*, 2010. 68, 89

[139] Kuniaki Saito and Kate Saenko. Ovanet: One-vs-all network for universal domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 9000–9009, 2021. 83, 96

[140] Kuniaki Saito, Yoshitaka Ushiku, and Tatsuya Harada. Asymmetric tri-training for unsupervised domain adaptation. *arXiv:1702.08400*, 2017. 12, 13, 14, 18, 24, 25

[141] Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *Proc. CVPR*, 2018. 67, 68, 76

[142] Kuniaki Saito, Shohei Yamamoto, Yoshitaka Ushiku, and Tatsuya Harada. Open set domain adaptation by backpropagation. In *Proceedings of the European Conference on Computer Vision (ECCV)*, pages 153–168, 2018. 83

[143] Mehdi Sajjadi, Mehran Javanmardi, and Tolga Tasdizen. Regularization with stochastic transformations and perturbations for deep semi-supervised learning. In *NIPS*, 2016. 12, 14, 18, 19

[144] Swami Sankaranarayanan, Yogesh Balaji, Carlos D Castillo, and Rama Chellappa. Generate to adapt: Aligning domains using generative adversarial networks. In *CVPR*, 2018. 14

[145] Swami Sankaranarayanan, Yogesh Balaji, Carlos D Castillo, and Rama Chellappa. Generate to adapt: Aligning domains using generative adversarial networks. In *CVPR*, 2018. 30, 33, 53, 59, 74

[146] Devashish Shankar, Sujay Narumanchi, HA Ananya, Pramod Kompalli, and Krishnendu Chaudhury. Deep learning based large scale visual recommendation and search for e-commerce. *arXiv preprint arXiv:1703.02344*, 2017. 1

[147] Ashish Shrivastava, Tomas Pfister, Oncel Tuzel, Josh Susskind, Wenda Wang, and Russ Webb. Learning from simulated and unsupervised images through adversarial training. *arXiv:1612.07828*, 2016. 14

[148] Rui Shu, Hung H Bui, Hirokazu Narui, and Stefano Ermon. A dirt-t approach to unsupervised domain adaptation. *arXiv preprint arXiv:1802.08735*, 2018. 19, 20, 25, 26, 59, 70

[149] Yang Shu, Zhangjie Cao, Mingsheng Long, and Jianmin Wang. Transferable curriculum for weakly-supervised domain adaptation. In *Proc. AAAI*, 2019. 59

[150] A. Siarohin, E. Sangineto, and N. Sebe. Whitening and Coloring transform for GANs. In *ICLR*, 2019. 15, 17, 31, 36, 37, 49

[151] Baochen Sun, Jiashi Feng, and Kate Saenko. Return of frustratingly easy domain adaptation. In *AAAI*, 2016. 12, 13, 24, 25, 46

[152] Baochen Sun and Kate Saenko. Deep coral: Correlation alignment for deep domain adaptation. In *Proc. ECCV*, 2016. 3, 12, 13, 14, 32, 56, 59, 80, 82

[153] Yaniv Taigman, Adam Polyak, and Lior Wolf. Unsupervised cross-domain image generation. *ICLR*, 2017. 14

[154] Antti Tarvainen and Harri Valpola. Mean teachers are better role models: Weight-averaged consistency targets improve semi-supervised deep learning results. In *NIPS*, 2017. 12, 23

[155] Jiayi Tian, Jing Zhang, Wen Li, and Dong Xu. VDM-DA: Virtual domain modeling for source data-free domain adaptation. *IEEE Transactions on Circuits and Systems for Video Technology*,

2021. 81

[156] Luke Tierney and Joseph B Kadane. Accurate approximations for posterior moments and marginal densities. *Journal of the American Statistical Association*, 81(393):82–86, 1986. 81, 85, 87

[157] Antonio Torralba and Alexei A Efros. Unbiased look at dataset bias. In *CVPR 2011*, pages 1521–1528. IEEE, 2011. 2, 11, 30, 56, 80

[158] Eric Tzeng, Judy Hoffman, Trevor Darrell, and Kate Saenko. Simultaneous deep transfer across domains and tasks. In *ICCV*, 2015. 14, 24, 25

[159] Eric Tzeng, Judy Hoffman, Trevor Darrell, and Kate Saenko. Adversarial discriminative domain adaptation. In *Proc. CVPR*, 2017. 3, 33, 53, 56, 59, 74, 75, 80, 82

[160] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *CVPR*, 2017. 24, 25, 46

[161] Eric Tzeng, Judy Hoffman, Ning Zhang, Kate Saenko, and Trevor Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv*, 2014. 3, 32, 56, 59, 80, 82

[162] Dmitry Ulyanov, Andrea Vedaldi, and Victor S. Lempitsky. Instance normalization: The missing ingredient for fast stylization. *arXiv:1607.08022*, 2016. 37, 48

[163] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *CVPR*, 2017. 13, 14, 21, 32, 59, 68, 69, 89

[164] Mei Wang and Weihong Deng. Deep visual domain adaptation: A survey. *Neurocomputing*, 312:135–153, 2018. 82

[165] Jun Wen, Nenggan Zheng, Junsong Yuan, Zhefeng Gong, and Changyou Chen. Bayesian uncertainty matching for unsupervised domain adaptation. In *International Joint Conference on Artificial Intelligence (IJCAI)*, 2019. 84

[166] Andrew Gordon Wilson. The case for Bayesian deep learning. Technical report, New York University, 2019. 83

[167] Zonghan Wu, Shirui Pan, Fengwen Chen, Guodong Long, Chengqi Zhang, and S Yu Philip. A comprehensive survey on graph neural networks. *TNNLS*, 2020. 60

[168] Haifeng Xia, Handong Zhao, and Zhengming Ding. Adaptive adversarial network for source-free domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 9010–9019, 2021. 93, 94

[169] Ruijia Xu, Ziliang Chen, Wangmeng Zuo, Junjie Yan, and Liang Lin. Deep cocktail network: Multi-source unsupervised domain adaptation with category shift. In *CVPR*, 2018. 30, 31, 33, 41, 42, 46, 47, 59, 68

[170] Ruijia Xu, Guanbin Li, Jihan Yang, and Liang Lin. Larger norm more transferable: An adaptive feature norm approach for unsupervised domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 1426–1435, 2019. 93, 94

[171] Luyu Yang, Yogesh Balaji, Ser-Nam Lim, and Abhinav Shrivastava. Curriculum manager for source selection in multi-source domain adaptation. *Proc. ECCV*, 2020. 59, 60

[172] Shiqi Yang, Yaxing Wang, Joost van de Weijer, Luis Herranz, and Shangling Jui. Generalized source-free domain adaptation. In *Proceedings of the IEEE International Conference on Computer Vision (ICCV)*, pages 8978–8987, 2021. 93, 96

[173] Xu Yang, Cheng Deng, Tongliang Liu, and Dacheng Tao. Heterogeneous graph attention network

for unsupervised multiple-target domain adaptation. *TPAMI*, 2020. 57, 59, 60, 65, 74, 75

[174] Yanchao Yang and Stefano Soatto. Fda: Fourier domain adaptation for semantic segmentation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4085–4095, 2020. 34

[175] Yi Yao and Gianfranco Doretto. Boosting for transfer learning with multiple sources. In *CVPR*, 2010. 30, 33

[176] Hao-Wei Yeh, Baoyao Yang, Pong C Yuen, and Tatsuya Harada. Sofa: Source-data-free feature alignment for unsupervised domain adaptation. In *Proceedings of the IEEE Winter Conference on Applications of Computer Vision (WACV)*, pages 474–483, 2021. 83

[177] Hongjie Zhang, Ang Li, Xu Han, Zhaoming Chen, Yang Zhang, and Yanwen Guo. Improving open set domain adaptation using image-to-image translation. In *IEEE International Conference on Multimedia and Expo (ICME)*, pages 1258–1263. IEEE, 2019. 83

[178] Wen Zhang, Lingfei Deng, and Dongrui Wu. Overcoming negative transfer: A survey. *arXiv*, 2020. 56

[179] Xu Zhang, Felix Xinnan Yu, Shih-Fu Chang, and Shengjin Wang. Deep transfer network: Unsupervised domain adaptation. *arXiv*, 2015. 75

[180] Sicheng Zhao, Bo Li, Xiangyu Yue, Yang Gu, Pengfei Xu, Runbo Hu, Hua Chai, and Kurt Keutzer. Multi-source domain adaptation for semantic segmentation. In *NeurIPS*, 2019. 33

[181] Sicheng Zhao, Guangzhi Wang, Shanghang Zhang, Yang Gu, Yaxian Li, Zhichao Song, Pengfei Xu, Runbo Hu, Hua Chai, and Kurt Keutzer. Multi-source distilling domain adaptation. In *AAAI*, 2020. 33

[182] Zhedong Zheng and Yi Yang. Rectifying pseudo label learning via uncertainty estimation for domain adaptive semantic segmentation. *International Journal of Computer Vision (IJCV)*, 2021. 81, 84

[183] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *CVPR*, 2017. 33, 40, 44, 48

[184] Xiaojin Zhu. Semi-supervised learning literature survey. 2005. 14