Department of Computer Science
Amirkabir University of Technology

Few-shot domain adaptation and cross-domain few-shot learning

Literature Review

Nima Hosseini Dashtbayaz

A thesis presented for the degree of
Bachelor of Science

July 2022 / Mordad 1401

Abstract

In recent years, the development of deep learning methods and increasing computational power has transformed the horizon of machine learning. Convolutional neural networks also revolutionized computer vision, significantly improving performance and offering novel solutions for various problems. However, the high performance of deep models is not achievable unless copious amounts of labeled data are available, which is a big obstacle in applying such models in real-world scenarios where gathering labeled data requires great labor effort. *Transfer learning* and its branches like *domain adaptation* try to overcome this challenge by utilizing the prior knowledge that a model has learned from another task. Specifically, in domain adaptation, a source model trained for a source distribution $\mathcal{D}_s$ is adapted to a new target distribution $\mathcal{D}_t$. While domain adaptation is a well-studied field, most of the works in this domain rely on access to redundant data in the target distribution. The *few-shot domain adaptation* and *cross-domain few-shot learning* settings, where target data is limited, are relatively new problems in the field and have gained increasing attention in recent literature. This work defines these two settings concisely and then reviews the latest approaches and methods in both settings.

1 Introduction

Deep learning methods have caused significant progress in various problems, including computer vision tasks. However, gathering enough data to train these models is not a trivial task in many circumstances. *Transfer learning* is a set of methods for utilizing extracted knowledge from solving a *source* problem to work out a new related *target* problem. *Domain shift* is a related problem that appears in transfer learning when joint distributions $\mathcal{P}(\mathcal{X}_s, \mathcal{Y}_s)$ and $\mathcal{P}(\mathcal{X}_t, \mathcal{Y}_t)$ over the source and target data are not aligned.
While domain shift and transfer learning are well-studied problems, the works in these fields usually need access to plentiful amounts of target data, a constraining requirement in many real-world scenarios where gathering data is expensive or limited due to certain access policies. Thus, Few-shot approaches for overcoming domain shift are critical in real-world problems and have gained great attention in recent literature.

In this work, we focus on these techniques, namely *few-shot domain adaptation* and *cross-domain few-shot learning* settings. In the following sections, we first give a brief definition of transfer learning, domain shift, and related methods, and then we move on to reviewing literature in mentioned settings in depth.

2 Overview

To define the required concepts, we first fix some notations in the next section and then go through the definitions of domain shift, domain adaptations, and few-shot learning.

Figure 1: SVHN (top), USPS (middle), and MNIST (bottom) datasets.

2.1 Notations

Throughout this text, we represent source and target tasks with $\mathcal{T}^s$ and $\mathcal{T}^t$, along with their datasets $\mathcal{D}^s$ and $\mathcal{D}^t$. The source dataset $\mathcal{D}^s$ is composed of pairs (x^s, y^s) while the target dataset might include labels or not, depending on the setting. Finally, the input space of the source and target tasks are represented by $\mathcal{X}^s$ and $\mathcal{X}^t$, while the label spaces are shown with $\mathcal{Y}^s$ and $\mathcal{Y}^t$ respectively.

In computer vision, models usually consist of *feature extractor* and *task head* components. Here, we let F represent a feature extractor, and G represent the task head (e.g., a classifier). The resulting feature vectors from F are shown with f^s and f^t for the source and target domain. Given F and G, the *hypothesis* h is the final prediction function derived from $G \circ F$.

2.2 Domain Shift

In a classification problem, we tend to find a hypothesis h that generalizes well on an input space $\mathcal{X}$. However, minimizing the error on a distribution $\mathcal{P}(\mathcal{X}^s, \mathcal{Y}^s)$ may not generalize well when the distribution changes to $\mathcal{P}(\mathcal{X}^t, \mathcal{Y}^t)$. This change or *shift* of distribution that results in poor performance on the new domain is known as domain shift.

Depending on the change in marginal distributions $\mathcal{P}(\mathcal{X})$ and $\mathcal{P}(\mathcal{Y})$, [21] divides domain shift into the following four variations.

- **Covariate shift** happens when marginal distributions of the source and target input spaces differ while the label space remains the same. Practically, covariate shift happens mostly when we adopt a dataset with the same task as our target domain, but it has been gathered with different tools, from another location, or from another perspective, resulting in visually different images.

MNIST [23], USPS [14], and SVHN [33] digit datasets are well-known examples of covariate shift. (See Figure 1.) While a model trained on MINST reaches high performance on handwritten numbers, applying the same model to USPS results in an accuracy as low as 65% [30]. Figure 2a shows how covariate shift affects the extracted features for each domain. While the classes in the source domain are isolated from each other, the target samples are more scattered and not well-separated. This setting is studied through domain adaptation.

- **Prior shift** is the case where the source and target tasks have different label spaces. Although having different tasks is not surprising in computer vision, this setting is more interesting when there are not enough target data to train a new model, and thus, datasets with other tasks are used to train a model.

  Solving prior shift with limited target data is studied in few-shot learning, where for K new classes, each with N labeled samples, the problem is called *K-way N-shot learning.*

- **General domain shift** is a combination of the two previous settings; The target task has a new label space, and the input space does not match the source inputs either. This setting can be formulated similar to few-shot learning, usually known as the *cross-domain few-shot learning* problem.

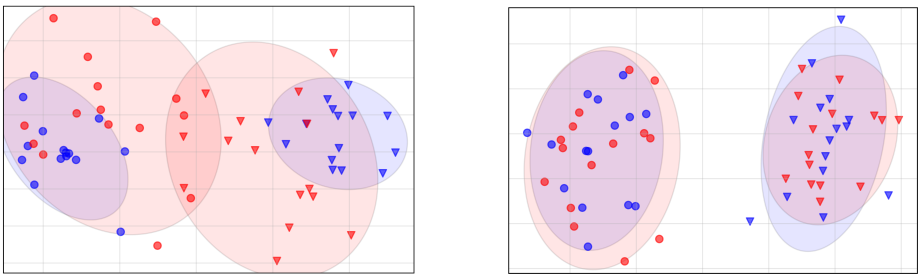
- **Concept shift** is the final case in which neither the input space nor the label space does not change. However, the conditional probability $\mathcal{P}(\mathcal{Y}|\mathcal{X}$ changes as a result of a change in the meaning of the labels. In this work, we do not cover this setting.

2.3 Domain Adaptation

Domain adaptation (DA) is a transfer learning approach for overcoming covariate shift. Conventional DA methods utilize datasets from the source and target domains in order to train a target model with high generalization performance on the target domain, although other settings are possible as well, e.g. model-based DA does not require access to the source dataset while adapting a trained source model.

Most of the work in domain adaptation falls under unsupervised methods (UDA) which rely on a large number of unlabeled target images along with the labeled source dataset [5, 4, 41]. Few approaches use supervised (SDA) [30, 40, 20] or semi-supervised [35, 17] methods to utilize target labels and extract their semantic information. While unsupervised and semi-supervised methods are proved useful, they assume the availability of a large amount of target domain data. This assumption neglects applications where gathering data from different domains is not straightforward, like medical applications [24] and autonomous driving [39, 44].

Supervised domain adaptation (SDA) methods exploit labels from target domain images to extract semantic information, leading to a better performance



(a) (b)

Figure 2: Illustration of misaligned source and target feature space caused by covariate shift (a), and their aligned feature space after domain adaptation (b). Blue points show feature vectors of source samples while red ones represent the target data. Squares and triangles represent classes. The red and blue ellipses show the confidence area for each class and domain.

than UDA methods with the same number of target samples. Thus, recent works [30, 43, 44, 39, 24, 42] have adopted supervised methods to perform domain adaptation in few-shot settings, i.e. only a few images are available from the target domain. Few works also try solving the harder few-shot unsupervised DA [36, 28, 1].

Applications in medicine [24], scene parsing and self-driving cars, and social media monitoring where various different domains, data access restrictions, and expensive labelling procedures limit the data availability for plain domain adaptation, are the among the motivations in studying few-shot approaches.

2.4 Few-shot Learning

Few-shot learning is a machine learning paradigm that aims on learning a new task with limited data or *experience* in machine learning terms. Achieving human-like intelligence, difficulties with data collection and labelling, and handling rare classes are major motivations behind studying FSL in problems like object detection, classification, short-text sentiment analysis, etc.

As mentioned in the previous section, in the context of object classification, FSL is usually formulated as a K-way N-shot learning problem, where there are N labeled samples for each of the K new classes. The $K \times N$ training samples are called *support set*, and they are typically accompanied with Q unlabeled samples called the *query set*.

Episode learning, used instead of usual batch learning, is the go-to training method applied in FSL. To simulate the few-shot conditions in deployment, episode learning generates *episodes* of fake support and query sets from the dataset, with support samples used for learning the task, and query samples for training the model and backpropagation.

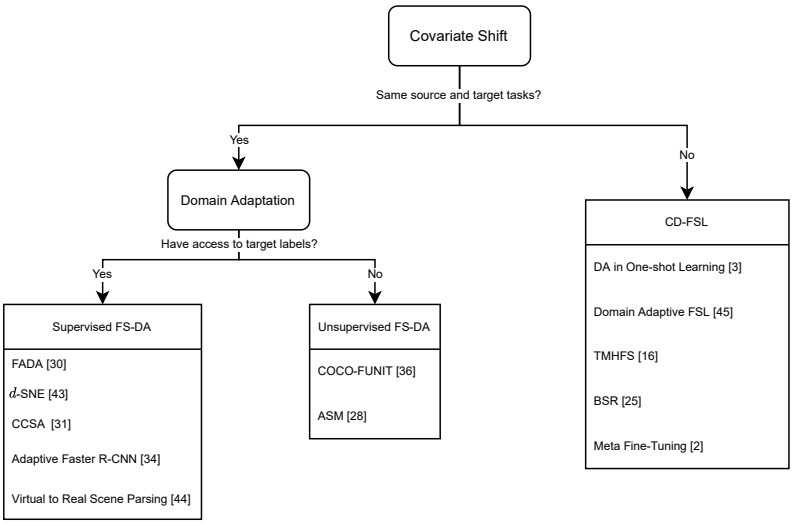


Figure 3: Overview of methods used for solving covariate shift problem with limited target data access.

Figure 3 shows an overview of the problems we review in this work and the methods to solve them.

3 Few-Shot Domain Adaptation

3.1 Supervised

Few-shot supervised domain adaptation (FS-SDA) methods utilize semantic information provided by the labels from the target space to align the two domains. While unsupervised adaptation is a more attractive problem when redundant amounts of data are available, label information becomes crucial with limited data. Metric [31, 43] and adversarial [30, 42] methods are the main approaches to SF-FDA, and we review them in this section.

3.1.1 Metric-based Methods

A large portion of domain adaptation literature focuses on *domain alignment* between source and target domains, forcing source and target images to have feature vectors close to each other in the feature space. Achieving domain alignment makes it possible to use the same task head or classifier for the source and target images, as their feature vectors represent similar patterns. Thus, UDA methods focus on domain alignment as there are no labels from the target domain to fine-tune the task head. In order to align the domains, metric methods rely on minimizing the distance between the domains in the feature space by adding a domain alignment loss $\mathcal{L}_{da}$ term, such as

$$\mathcal{L}_{da} = d(p(f(\mathcal{X}^t)), p(f(\mathcal{X}^s))), \tag{1}$$

where $d(\cdot)$ is a distance function, and $p(f(\cdot))$ is the probability distribution of feature vectors for each domain. Together with a task loss such as cross entropy for the source images, feature extractor f is trained to minimize $\mathcal{L}_{da}$.

Although effective in UDA, this method fails to align domains properly when only a few target images are available. To address this issue [31] and [43] elaborate on metric-based methods to utilize labels from the target domain. [31] replaces the domain alignment loss with a *semantic alignment* loss $\mathcal{L}_{sa}$ which utilizes labels for a better alignment by penalizing source and target images with the same label that are far from each other. $\mathcal{L}_{sa}$ can be formulated as

$$\mathcal{L}_{sa} = \sum_{a=1}^{C} d(p(f(\mathcal{X}_a^t)), p(f(\mathcal{X}_a^s))), \tag{2}$$

where $\mathcal{X}_a^s$ and $\mathcal{X}_a^t$ are source and target images from class a, and C is the number of classes. Along with $\mathcal{L}_{sa}$, [31] applies a *separation loss* $\mathcal{L}_{\mathcal{S}}$ that penalizes similarity between source and target images with different classes, and is formulated as

$$\mathcal{L}_s = \sum_{a,b|a\neq b} k(p(f(X_a^s)), p(f(X_b^t))), \tag{3}$$

where k is a similarity metric. Finally, the feature extractor f is trained by minimizing $\mathcal{L}_{\mathcal{S}}$ and $\mathcal{L}_{\mathcal{SA}}$ together with the classification loss.

With a similar approach, [43] uses class probabilities to achieve domain alignment in the feature space. The probability p_i of making the correct prediction for the target image x_i^t with label $y_i^t = a$ is defined as

$$p_i = \frac{\sum_{x\in\mathcal{X}_a^s} \exp(-d(f_s(x), f_t(x_i^t)))}{\sum_{x\in\mathcal{X}^s} \exp(-d(f_s(x), f_t(x_i^t)))}, \tag{4}$$

where $f_s(\cdot)$ and $f_t(\cdot)$ are feature extractor functions for the source and target domains respectively, and $d(\cdot)$ is a distance measure in the feature space. Since maximizing the probability p_i is desired, log-likelihood of $\frac{1}{p_i}$ can be equivalently minimized for each target image x_j^t with label a, which can be formulated as

$$\mathcal{L} = \log\left(\frac{\sum_{x\in\mathcal{X}_{\not a}^s} \exp(-d(f_s(x), f_t(x_i)))}{\sum_{x\in\mathcal{X}_a^s} \exp(-d(f_s(x), f_t(x_i)))}\right), \tag{5}$$

where $\mathcal{X}_{\not a}^s$ is the set of source images that are not from class a. Minimizing $\mathcal{L}$ would result in minimizing the ratio of intra-class distances to inter-class distances in the latent space. However, the exponential terms in 5 might end in scaling issues with optimization algorithms used for training the model. Thus, Xu et al. [43] relax this formulation to a Hausdorffian distance which minimizes

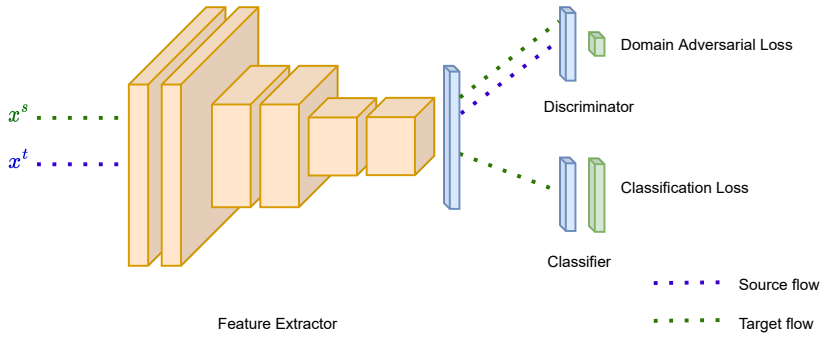


Figure 4: Illustration of a generic unsupervised adversarial DA model. The feature extractor is trained to fool the discriminator while keeping a high classification performance for source images.

only the largest same-class distance and the smallest distance with other classes. The resulting loss is formulated as

$$\tilde{\mathcal{L}} = \sup_{x \in \mathcal{D}_k^s} \{a | a \in d(x, x_j)\} - \inf_{x \in \mathcal{D}_{k'}^s} \{b | b \in d(x, x_j)\} \tag{6}$$

for each target sample x_j from the k-th class.

3.1.2 Domain Adversarial Methods

Introduction of *generative adversarial networks* (GANs) in [7] paved the path for a new line of research in domain adaptation. GANs consist of two generator and discriminator networks. The generator's goal is to create fake images that look real while the discriminator tries to identify the fake images from the real ones. By playing an adversarial game, the generator learns to fool the discriminator, and the discriminator, on the other hand, gets better at identifying the fake images.

Adversarial domain adaptation (ADA) methods [37, 41, 5] adopt discriminators and use them for identifying the domain of the outputs of the feature extractor. Thus, the feature extractor is trained to output domain-invariant features for the source and target domains so that the discriminator can no longer distinguish the domains. As in metric-based methods, domain invariant features result in aligned feature space for both domains, and the task head can be used for both.

Unsupervised adversarial domain adaptation models (see Figure 4) generally apply an adversarial loss $\mathcal{L}_{adv}$ along with the usual task loss such as cross-entropy. The domain discriminator D can be regarded as a domain classifier that predicts whether the outputs of f_s and f_t are from the source domain. $\mathcal{L}_{avd}$ is then formulated as

$$\mathcal{L}_{adv}\left(\mathcal{X}^s, \mathcal{X}^t\right) = \mathbb{E}\left[\log(D(f_s(\mathcal{X}^s)))\right] - \mathbb{E}\left[\log(1 - D(f_t(\mathcal{X}^t)))\right]. \tag{7}$$

The training objective for D is minimizing the above loss, while f_s and f_t are trained to confuse D by maximizing $\mathcal{L}_{adv}$. Note that f_t and f_s might or might not share weights directly.

Despite getting tremendous attention in UDA, when only a few images are available from the target domain, training a domain discriminator is not feasible, as the source data inundate the target domain. To address this problem and effectively train a domain discriminator, [30, 42, 44] use groups of pairs of images to augment the data and train the discriminator to identify groups rather than domains.

Specifically, [30] defines 4 groups $\mathcal{G}_i$ as follows.

1. $\mathcal{G}_1$ contains image pairs from the source distribution with the same label,
2. $\mathcal{G}_2$ contains image pairs from different distribution but with the same label,
3. $\mathcal{G}_3$ contains image pairs from the source distribution with different labels,
4. and $\mathcal{G}_4$ contains image pairs from different distributions with different labels.

A multi-class discriminator then tries to recognize the group each pair is coming from. To confuse the discriminator, the feature extractor learns to minimize the loss

$$\mathcal{L}_{FADA} = \mathbb{E}\left[y_{\mathcal{G}_1}\log(D(f(\mathcal{G}_2))) - y_{\mathcal{G}_3}\log(D(f(\mathcal{G}_4)))\right], \tag{8}$$

so that the first and second groups are not distinguishable, as well as the third and fourth groups. By doing so, pairs from different distributions and the same label are aligned in the feature space, while those with different labels are still semantically separable.

[42] and [44] take this idea to object detection and semantic segmentation problems. [42] introduces *instance-level* and *image-level* adaptation modules on top of the Faster R-CNN [34] model to catch semantic and domain-specific characteristics.

The image-level adaptation module uses split pooling to uniformly extract local feature patches across various locations with different aspect ratios and scales from the feature maps. Each feature window is then pooled to fixed-sized features using RoI pooling. For each window scale, two groups of pairs of images $\mathcal{G}_{tt}$ and $\mathcal{G}_{ts}$ are created, where the pairs in the first group consist of two source image feature patches, and pairs in the second group are made by two feature patches from different domains. A discriminator is trained for each scale size that can tell whether a pair comes from $\mathcal{G}_{tt}$ or $\mathcal{G}_{ts}$, and the feature extractor tries to confuse the discriminator by maximizing an adversarial loss. This module helps extract target domain characteristics present across all images, such as fog.

To further align the domains semantically, [34] applies instance-wise adaptations as well. For each RoI output with a high IoU, the features are passed through intermediate layers between the RoI and classifier layers to get features

o_{id} from domain d with label i. For each class i, two groups $\mathcal{N}_{is}$ and $\mathcal{N}_{it}$ of features pairs are created, where the pairs in the first group are both from the source domain and class i, and those in the second group are from different domains and labeled with i. The *multi-way instance-level* discriminator D^{ins} is then applied with the following adversarial loss

$$\mathcal{L}_{ins} = \sum_{i=1}^{C} -\mathbb{E}\left[\log D^{ins}(\mathcal{N}_{is})\right] - \mathbb{E}\left[\log D^{ins}(\mathcal{N}_{it})\right] \tag{9}$$

where C is the number of classes.

[44] uses a similar approach for adapting a model trained on synthetic to real-world in the context of semantic segmentation by applying two discriminator networks on feature patches from two different layers. At each iteration, three pairs g_{ss}, g_{tt}, and g_{st} are created, where the first pair is made from concatenating features of a source image from current iteration with one from the previous iteration, the second pair is created similar to the first pair but from target images, and the third pair consists of features from two source and target images in the current iteration. The discriminators are then trained to recognize each pair, while the main model learns to confuse the discriminator along with the segmentation task using both source and target data.

In a similar setting to that of [44], [39] also uses adversarial methods to adapt a synthetically trained model. However, [39] does not use image pairs, and instead, proposes a pixel-wise adaptation method inspired by [11] to prevent negative transfer for poorly represented classes. For source and target images x^s and x^t, the discriminator D is trained to minimize the loss

$$\mathcal{L}_D = -\sum_{i \in \mathcal{I}} \log D_i(f(x^s)) + \log(1 - D_i(f(x^t)))\,, \tag{10}$$

where $\mathcal{I}$ is the set of pixels for each image.

3.2 Unsupervised

While the literature on UDA is rich and well-explored, most conventional UDA methods fail in unsupervised few-shot settings. This section reviews the few works within the unsupervised few-shot setting, primarily based on augmenting data by generating new images.

3.2.1 Image Generative Methods

With the introduction of *Conditional GAN*s [29], new research directions evolved in *image-to-image translation* and *style transfer* [15, 26, 46, 27, 36]. Image-to-image (I2I) translation generally focuses on transferring an image to a new style or distribution, while keeping its content, e.g. making a photo look like paintings from Van Gogh. While supervised I2I methods require image pairs from both styles with the same content for training, unpaired I2I relies on datasets from each distribution, without explicitly pairing images in them.

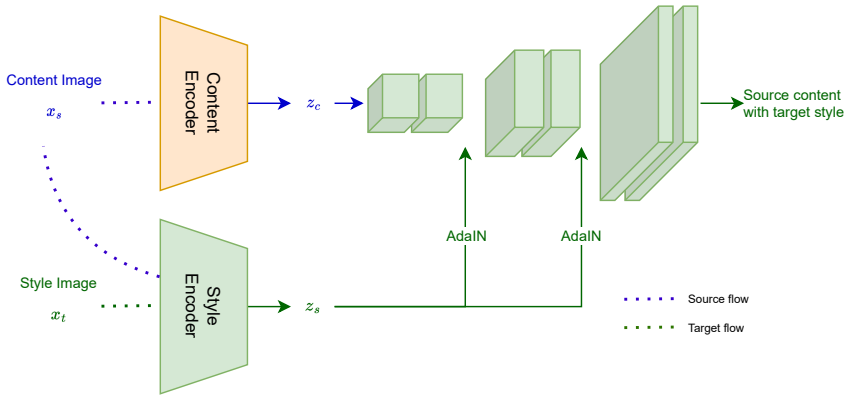


Figure 5: An illustration of COCO-FUNIT [36] model. The style encoder generates the source-conditioned style code z_s, which is applied to the content decoder network as the AdaIN parameter.

Cycle consistent image translation [46] made a big step in unpaired image translation. [46] proposes finding two mappings $F : \mathcal{X}^s \to \mathcal{X}^t$ and $G : \mathcal{X}^t \to \mathcal{X}^s$, where F and G should satisfy the cycle consistency, i.e. for any two images $x \sim \mathcal{X}^s$ and $y \sim \mathcal{X}^t$, $G(F(x)) \approx x$ and $F(G(y)) \approx y$. To achieve this goal, the *cycle consistency loss* is defined as

$$\mathcal{L}_{cyc} = \mathbb{E}_{x\sim\mathcal{X}^s}\left[||G(F(x)) - x||_1\right] + \mathbb{E}_{y\sim\mathcal{X}^t}\left[||F(G(y)) - y||_1\right] , \tag{11}$$

where $||\cdot||_1$ is the 1-norm. While cycle consistency was originally proposed for image translation, [32] adopts a set of losses inspired by cycle consistency to achieve a domain invariant feature space that contains critical information to reconstruct images from their latent representation. Similarly, [10] uses image translation with cycle consistency from [46] to transfer source images to the target domain, and adapt a model pretrained on the source domain using the translated source images.

While the mentioned methods are effective in I2I and UDA, they fail in few-shot settings. An alternative would be entirely translating source images to the target domain and training the whole model with the translated source images. A proper image-to-image translation model should utilize a few images from the target distribution to extract their style and transfer the source images to the target domain. Such a model must preserve the source images' content so that labels remain valid. In tasks like semantic segmentation, where labels are specified pixel by pixel, minor variations in pose or location of content can lead to catastrophically wrong labels.

[43] proposes a few-shot I2I model capable of extracting the content of the source image and the style of a previously unseen target domain with few samples, using two *content encoder* and *style encoder* networks. The resulting *content code* from them content encoder is fed to a decoder network along with the *style code* as the AdaIN[12] parameter to make up the final image. AdaIN proposes

rescaling outputs of each upsampling layer in the encoder network regarding the mean and variance of the style code, assuming that these two parameters control the image's visual characteristics.

[36] proposed a new model built on top of the [43] to further control the content loss. While keeping the content encoder and decoder networks the same, [36] modified the style encoder by conditioning it on the source image content, so that the style code extracted from the target image is controlled by the source image as well.

In the new *content-conditioned style encoder*, source and target images are separately fed into two source and target style encoder networks and are then concatenated to build the final style code. The style code is treated similar to [27] as AdaIN parameters. To train the network, source images are divided into k domains or *classes* which contain similar objects. In each iteration, a pair of images (x_c, x_k) are sampled from two classes, and the network learns to translate x_c to x_k.

3.2.2 Unified Generation and Adaptation

While style transfer and I2I models provide a way to generate images in the target domain and train a model, they are deployed discretely, making it impossible to get feedback from the adaptation process and the task head for the translation network. [28] and [8] address this drawback in a one-shot scenario by using a variational autoencoder [18, 19] (VAE) combined with AdaIN [13].

The proposed *random AdaIN (RAIN)* module [28] includes a VAE that takes as input $\mu(f_t)$ and $\sigma(f_t)$, where $\mu(\cdot)$ and $\sigma(\cdot)$ are channel-wise mean and standard deviation, and f_t is the feature maps of the target image. As a result, the VAE learns a Gaussian distribution $\mathcal{N}$ with mean ψ and standard deviation ξ. The VAE then samples a new mean and standard deviation vector ϵ from $\mathcal{N}$, reconstructs it to get a new style vector, and generates a new image with the content code of a source image and the generated style code as the AdaIN parameter. The generated image keeps the content of the source image with a style similar to the target. As new images are fed to the main network for training, the gradients of the loss function are computed with respect to ϵ to create a style that is harder for the network to process. As a result, styles close to the single target image are discovered adversarially, and the network learns to generalize to the target domain.

3.3 Analytical Comparison

MNIST [23] and USPS [14] are famous digit datasets usually used to evaluate the performance of DA algorithms. Both datasets contain handwritten grayscale digits from 10 classes (0 to 9). Table 1 shows the results achieved by supervised few-shot DA methods compared with unsupervised DA algorithms.

Note that results from UDA methods in Table 1 are achieved using the full MNIST and USPS datasets, while FS-DA methods use n_t-shot support sets for adaptation. The results suggest that the semantic info from a few labeled

Table 1: Accuracy of DA methods in MNIST to USPS ($\mathcal{M} \to \mathcal{U}$) and USPS to MNIST ($\mathcal{U} \to \mathcal{M}$) domain adaptation for image classification task. n_t shows the number of target images available per class, where in n_0 setting, no domain adaptation is conducted.

		UDA methods			Few-shot DA		
Method		[6]	[41]	[37]	CCSA [31]	FADA [30]	d-SNE [43]
$\mathcal{U} \to \mathcal{M}$	$n_t = 0$	73.7%	90.1%	90.8%	58.6%	58.6%	83.37%
	$n_t = 1$				78.4%	81.1%	–
	$n_t = 3$				85.8%	87.5%	–
	$n_t = 5$				88.8%	91.1%	–
	$n_t = 7$				89.4%	91.5%	97.52%
$\mathcal{M} \to \mathcal{U}$	$n_t = 0$	91.8%	89.4%	92.5%	65.4%	65.4%	73.0%
	$n_t = 1$				85.0%	89.1%	92.9%
	$n_t = 3$				90.1%	91.9%	93.5%
	$n_t = 5$				92.4%	93.4%	95.1%
	$n_t = 7$				92.9%	94.4%	96.1%

samples boost the model's performance, outperforming UDA approaches with more data.

While CCSA [31] and d-SNE [43] are both based on a contrastive idea that pulls samples with the same class close together and pushes different classes apart, they implement different metrics. [31] estimates the distance between distributions and their similarity with average pairwise distances, but [43] maximizes the shortest inter-class distance and minimizes the highest intra-class distance, explicitly penalizing scattered samples from the same class and close clusters of different classes. As evident in 1, the second implementation works better in the FS-DA setting.

The adversarial approach proposed by FADA [30] aligns the positive (negative) source-target pairs with the positive (negative) source-source pairs. While the augmentation technique and the adversarial manner outperform CCSA, it does not use semantic info to separate clusters of different classes. Also, the limited target data might still not be enough for the adversarial training of the discriminator.

4 Cross-Domain Few-Shot Learning

Guo et al. [9] discussed the performance of the existing FSL methods in a novel cross-domain benchmark using a variety of datasets with varying levels of dissimilarity with the source images in perspective distortion, semantic content,

and color depth. Their study resulted in a new chapter of FSL research concerning covariate shift. In this section, we will review the CD-FSL methods in recent literature and their contribution to the field.

4.1 Adversarial

While using adversarial methods in conventional domain adaptation is well explored, they are less studied in the few-shot learning context as the sample scarcity and disjoint label spaces make it harder to apply adversarial learning effectively. [3] and [45] apply domain adversarial learning in few-shot learning settings.

[3] utilizes domain adversarial learning along with a data sampling method on top of the prototypical network [38] for the one-shot cross-domain learning problem. The binary domain discriminator is applied right after the feature extractor f_θ, where it aligns features of source and target domains. The discriminator and feature extractor are trained with adversarial loss, while the feature extractor is also optimized on a classification loss for the source data.

Similar to [3], [45] also uses domain adversarial loss to align two domains. However, as per-class alignment is not desired in the few-shot learning settings, where the source and target domains have disjoint label spaces, [45] applies other loss terms to control class isolation. The proposed model includes an *embedding module* right after the feature extractor, consisting of an autoencoder and an attention layer catching domain invariant features. The outputs of the embedding module are then fed to a domain discriminator with an adversarial objective function.

4.2 Fine-tuning

Fine-tuning is a well-known approach in conventional transfer learning approaches when parameters of a pre-trained model are tuned using the new data to fit it. However, applying fine-tuning when data is scarce might result in unwanted results such as over-fitting. [16] and [25] develop models that can be fine-tuned with limited target data.

The *Transductive Multi-head* model proposed in [16], developed based on [22], applies three task heads while training and uses them to fine-tune the model on the target domain. Specifically, the main task head is a prototypical network called the *Meta-confidence Transductive* head. The second task head is a pixel-wise prototypical classification network that classifies each pixel in the feature space using learned prototypes for each class. Finally, the third head is a fully connected classifier. In the training phase, all three heads are used to compute the loss and train the feature extractor along with each head. While fine-tuning on the target domain, only the third head is active, which fine-tunes the feature extractor. Eventually, the auxiliary heads are dropped, and the first

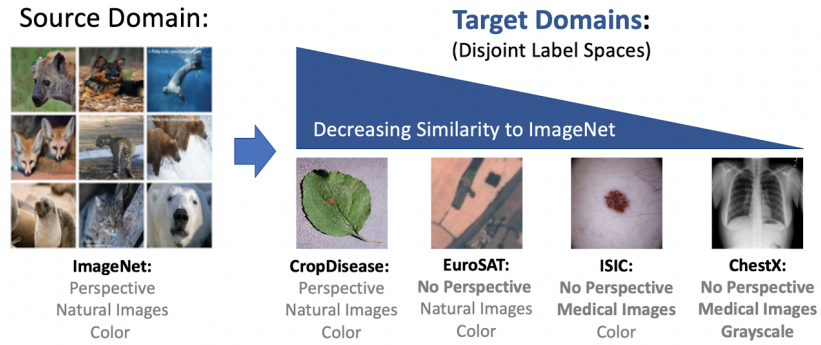


Figure 6: Overview of the CVPR 2020 cross-domain FSL challenge.

prototypical head is used for testing.

[25] proposes an ensemble consisting of M identical models except that a random transformation is applied to the feature space in each model. For each model $i \in \{1, \ldots, M\}$, the feature extractor produces feature vectors f_i^B, and an orthogonal matrix E_i is generated and applied to the feature vectors to get transformed features $f_i^E = E_i F_i^B$. Transformed features are then fed to a softmax classifier c_i, and the models are trained using the cross-entropy loss. After pre-training with the source data, support sets sampled from the augmented few-shot target dataset are used to train target classifiers c_i^t and fine-tune the feature extractors. In order to supply the model with more labeled data, [25] also uses a semi-supervised label propagation process to use the unlabeled test data for further tuning.

Finally, [2] uses a 2-step fine-tuning process by implementing a first-order model-agnostic meta learning algorithms such as episodic learning. During the first step, first k layers of the feature extractor are frozen, and the support samples are used along with a linear classifier to fine-tune the last layers of the feature extractor with the cross entropy loss. During the second step, the linear classifier is replaced with a metric-learning module (such as prototypical network or a GNN FSL model), and all the layers are updated using the training loss. This simple method first catches high level representations in the target domain, and then fine-tunes the model in the FSL way.

4.3 Analytical Comparison

The CD-FSL benchmark [9] and the CVPR 2020 challenge provide a proper benchmark for evaluating the reviewed methods. The test is conducted on four datasets with varying number of samples for each class, and all models are pretrained with the ImageNet dataset.

Table 2 shows the CD-FSL benchmark accuracy scores for each model and support size. As evident in this table, the ProtoNet model trained on ImageNet

Table 2: Accuracy of the FSL methods evaluated with the CD-FSL benchmark [9].

Method		ProtoNet [38, 9]	TMHFS [16]	BSR [25]	Meta Fine-tuning [2]
CropDisease	$n_t = 5$	79.7%	95.2%	96.6%	96.2%
	$n_t = 20$	88.1%	98.5%	99.1%	98.9%
	$n_t = 50$	90.8%	99.2%	99.7%	99.4%
EuroSAT	$n_t = 5$	73.2%	85.3%	88.1%	89.8%
	$n_t = 20$	82.2%	92.4%	94.7%	93.9%
	$n_t = 50$	80.4%	95.6%	96.8%	96.0%
ISIC	$n_t = 5$	39.5%	53.8%	57.4%	61.7%
	$n_t = 20$	49.5%	65.4%	68.0%	65.2%
	$n_t = 50$	51.9%	71.2%	74.0%	75.1%
ChestX	$n_t = 5$	24.0%	27.9%	29.7%	28.4%
	$n_t = 20$	28.2%	37.1%	38.3%	35.6%
	$n_t = 50$	29.3%	43.4%	44.4%	44.7%

loses its performance as domains become more different from the source domain. The other three columns show the effect of adaptation methods in few-shot tasks.

References

[1] F. C. Borlino, S. Polizzotto, A. D'Innocente, S. Bucci, B. Caputo, and T. Tommasi. Self-supervision & meta-learning for one-shot unsupervised cross-domain detection. *CoRR*, abs/2106.03496, 2021.

[2] J. Cai and S. M. Shen. Cross-domain few-shot learning with meta fine-tuning. *CoRR*, abs/2005.10544, 2020.

[3] N. Dong and E. P. Xing. Domain adaption in one-shot learning. In M. Berlingerio, F. Bonchi, T. Gärtner, N. Hurley, and G. Ifrim, editors, *Machine Learning and Knowledge Discovery in Databases - European Conference, ECML PKDD 2018, Dublin, Ireland, September 10-14, 2018, Proceedings, Part I*, volume 11051 of *Lecture Notes in Computer Science*, pages 573–588. Springer, 2018.

[4] Y. Ganin and V. S. Lempitsky. Unsupervised domain adaptation by backpropagation. In F. R. Bach and D. M. Blei, editors, *Proceedings of the 32nd International Conference on Machine Learning, ICML 2015, Lille, France, 6-11 July 2015*, volume 37 of *JMLR Workshop and Conference Proceedings*, pages 1180–1189. JMLR.org, 2015.

[5] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. S. Lempitsky. Domain-adversarial training of neural networks. *J. Mach. Learn. Res.*, 17:59:1–59:35, 2016.

[6] M. Ghifary, W. B. Kleijn, M. Zhang, D. Balduzzi, and W. Li. Deep reconstruction-classification networks for unsupervised domain adaptation. In B. Leibe, J. Matas, N. Sebe, and M. Welling, editors, *Computer Vision - ECCV 2016 - 14th European Conference, Amsterdam, The Netherlands, October 11-14, 2016, Proceedings, Part IV*, volume 9908 of *Lecture Notes in Computer Science*, pages 597–613. Springer, 2016.

[7] I. J. Goodfellow, J. Pouget-Abadie, M. Mirza, B. Xu, D. Warde-Farley, S. Ozair, A. C. Courville, and Y. Bengio. Generative adversarial networks. *CoRR*, abs/1406.2661, 2014.

[8] M. Gu, S. Vesal, R. Kosti, and A. K. Maier. Few-shot unsupervised domain adaptation for multi-modal cardiac image segmentation. *CoRR*, abs/2201.12386, 2022.

[9] Y. Guo, N. Codella, L. Karlinsky, J. V. Codella, J. R. Smith, K. Saenko, T. Rosing, and R. Feris. A broader study of cross-domain few-shot learning. In A. Vedaldi, H. Bischof, T. Brox, and J. Frahm, editors, *Computer Vision - ECCV 2020 - 16th European Conference, Glasgow, UK, August 23-28, 2020, Proceedings, Part XXVII*, volume 12372 of *Lecture Notes in Computer Science*, pages 124–141. Springer, 2020.

[10] J. Hoffman, E. Tzeng, T. Park, J. Zhu, P. Isola, K. Saenko, A. A. Efros, and T. Darrell. Cycada: Cycle-consistent adversarial domain adaptation. *CoRR*, abs/1711.03213, 2017.

[11] J. Hoffman, D. Wang, F. Yu, and T. Darrell. Fcns in the wild: Pixel-level adversarial and constraint-based adaptation. *CoRR*, abs/1612.02649, 2016.

[12] X. Huang and S. J. Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. *CoRR*, abs/1703.06868, 2017.

[13] X. Huang and S. J. Belongie. Arbitrary style transfer in real-time with adaptive instance normalization. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 1510–1519. IEEE Computer Society, 2017.

[14] J. Hull. A database for handwritten text recognition research. *IEEE Transactions on Pattern Analysis and Machine Intelligence*, 16(5):550–554, 1994.

[15] P. Isola, J. Zhu, T. Zhou, and A. A. Efros. Image-to-image translation with conditional adversarial networks. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 5967–5976. IEEE Computer Society, 2017.

[16] J. Jiang, Z. Li, Y. Guo, and J. Ye. A transductive multi-head model for cross-domain few-shot learning. *CoRR*, abs/2006.11384, 2020.

[17] P. Jiang, A. Wu, Y. Han, Y. Shao, M. Qi, and B. Li. Bidirectional adversarial training for semi-supervised domain adaptation. In C. Bessiere, editor, *Proceedings of the Twenty-Ninth International Joint Conference on Artificial Intelligence, IJCAI 2020*, pages 934–940. ijcai.org, 2020.

[18] D. P. Kingma and M. Welling. Auto-encoding variational bayes. In Y. Bengio and Y. LeCun, editors, *2nd International Conference on Learning Representations, ICLR 2014, Banff, AB, Canada, April 14-16, 2014, Conference Track Proceedings*, 2014.

[19] D. P. Kingma and M. Welling. An introduction to variational autoencoders. *CoRR*, abs/1906.02691, 2019.

[20] P. Koniusz, Y. Tas, and F. Porikli. Domain adaptation by mixture of alignments of second- or higher-order scatter tensors. *CoRR*, abs/1611.08195, 2016.

[21] W. M. Kouw and M. Loog. An introduction to domain adaptation and transfer learning. *arXiv preprint arXiv: Arxiv-1812.11806*, 2018.

[22] S. M. Kye, H. Lee, H. Kim, and S. J. Hwang. Transductive few-shot learning with meta-learned confidence. *CoRR*, abs/2002.12017, 2020.

[23] Y. LeCun and C. Cortes. MNIST handwritten digit database. 2010.

[24] S. Li, X. Sui, J. Fu, H. Fu, X. Luo, Y. Feng, X. Xu, Y. Liu, D. S. W. Ting, and R. S. M. Goh. Few-shot domain adaptation with polymorphic transformers. *CoRR*, abs/2107.04805, 2021.

[25] B. Liu, Z. Zhao, Z. Li, J. Jiang, Y. Guo, and J. Ye. Feature transformation ensemble model with batch spectral regularization for cross-domain few-shot classification. *CoRR*, abs/2005.08463, 2020.

[26] M. Liu, T. M. Breuel, and J. Kautz. Unsupervised image-to-image translation networks. In I. Guyon, U. von Luxburg, S. Bengio, H. M. Wallach, R. Fergus, S. V. N. Vishwanathan, and R. Garnett, editors, *Advances in Neural Information Processing Systems 30: Annual Conference on Neural Information Processing Systems 2017, December 4-9, 2017, Long Beach, CA, USA*, pages 700–708, 2017.

[27] M. Liu, X. Huang, A. Mallya, T. Karras, T. Aila, J. Lehtinen, and J. Kautz. Few-shot unsupervised image-to-image translation. *CoRR*, abs/1905.01723, 2019.

[28] Y. Luo, P. Liu, T. Guan, J. Yu, and Y. Yang. Adversarial style mining for one-shot unsupervised domain adaptation. *CoRR*, abs/2004.06042, 2020.

[29] M. Mirza and S. Osindero. Conditional generative adversarial nets. *CoRR*, abs/1411.1784, 2014.

[30] S. Motiian, Q. Jones, S. M. Iranmanesh, and G. Doretto. Few-shot adversarial domain adaptation. *CoRR*, abs/1711.02536, 2017.

[31] S. Motiian, M. Piccirilli, D. A. Adjeroh, and G. Doretto. Unified deep supervised domain adaptation and generalization. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 5716–5726. IEEE Computer Society, 2017.

[32] Z. Murez, S. Kolouri, D. J. Kriegman, R. Ramamoorthi, and K. Kim. Image to image translation for domain adaptation. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 4500–4509. Computer Vision Foundation / IEEE Computer Society, 2018.

[33] Y. Netzer, T. Wang, A. Coates, A. Bissacco, B. Wu, and A. Y. Ng. Reading digits in natural images with unsupervised feature learning. 2011.

[34] S. Ren, K. He, R. B. Girshick, and J. Sun. Faster R-CNN: towards real-time object detection with region proposal networks. *CoRR*, abs/1506.01497, 2015.

[35] K. Saito, D. Kim, S. Sclaroff, T. Darrell, and K. Saenko. Semi-supervised domain adaptation via minimax entropy. *CoRR*, abs/1904.06487, 2019.

[36] K. Saito, K. Saenko, and M. Liu. COCO-FUNIT: few-shot unsupervised image translation with a content conditioned style encoder. *CoRR*, abs/2007.07431, 2020.

[37] S. Sankaranarayanan, Y. Balaji, C. D. Castillo, and R. Chellappa. Generate to adapt: Aligning domains using generative adversarial networks. In *2018 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2018, Salt Lake City, UT, USA, June 18-22, 2018*, pages 8503–8512. Computer Vision Foundation / IEEE Computer Society, 2018.

[38] J. Snell, K. Swersky, and R. S. Zemel. Prototypical networks for few-shot learning. *CoRR*, abs/1703.05175, 2017.

[39] A. Tavera, F. Cermelli, C. Masone, and B. Caputo. Pixel-by-pixel cross-domain alignment for few-shot semantic segmentation. *CoRR*, abs/2110.11650, 2021.

[40] E. Tzeng, J. Hoffman, T. Darrell, and K. Saenko. Simultaneous deep transfer across domains and tasks. In *2015 IEEE International Conference on Computer Vision, ICCV 2015, Santiago, Chile, December 7-13, 2015*, pages 4068–4076. IEEE Computer Society, 2015.

[41] E. Tzeng, J. Hoffman, K. Saenko, and T. Darrell. Adversarial discriminative domain adaptation. In *2017 IEEE Conference on Computer Vision and Pattern Recognition, CVPR 2017, Honolulu, HI, USA, July 21-26, 2017*, pages 2962–2971. IEEE Computer Society, 2017.

[42] T. Wang, X. Zhang, L. Yuan, and J. Feng. Few-shot adaptive faster R-CNN. *CoRR*, abs/1903.09372, 2019.

[43] X. Xu, X. Zhou, R. Venkatesan, G. Swaminathan, and O. Majumder. d-sne: Domain adaptation using stochastic neighborhood embedding. *CoRR*, abs/1905.12775, 2019.

[44] J. Zhang, Z. Chen, J. Huang, L. Lin, and D. Zhang. Few-shot structured domain adaptation for virtual-to-real scene parsing. In *2019 IEEE/CVF International Conference on Computer Vision Workshops, ICCV Workshops 2019, Seoul, Korea (South), October 27-28, 2019*, pages 9–17. IEEE, 2019.

[45] A. Zhao, M. Ding, Z. Lu, T. Xiang, Y. Niu, J. Guan, J. Wen, and P. Luo. Domain-adaptive few-shot learning. *CoRR*, abs/2003.08626, 2020.

[46] J. Zhu, T. Park, P. Isola, and A. A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. In *IEEE International Conference on Computer Vision, ICCV 2017, Venice, Italy, October 22-29, 2017*, pages 2242–2251. IEEE Computer Society, 2017.