Multimedia Tools and Applications (2022) 81:39545–39576
https://doi.org/10.1007/s11042-022-12630-8

A survey of deep domain adaptation based on label set classification

Min Fan[1] · **Ziyun Cai**[1] · **Tengfei Zhang**[1] · **Baoyun Wang**[1]

Received: 18 December 2020 / Revised: 12 March 2021 / Accepted: 9 February 2022 /
Published online: 29 April 2022

Abstract
Traditional machine learning requires good tags to obtain excellent performance, while manual tagging usually consumes a lot of time and money. Due to the influence of domain shift, using the trained model on the source domain directly on the target domain is not good. Domain adaptation is used to solve the above problems. The deep domain adaptation method uses deep neural networks to complete domain adaptation. This article has carried out a comprehensive review of the deep domain adaptation method of image classification. The main contributions are the following four aspects. Firstly, we divided the deep domain adaptation into several categories based on the label set of the source domain and the target domain. Secondly, we summarized various methods of Closed-set domain adaptation. Thirdly, we discussed current methods of multi-source domain adaptation. Finally, we discussed future research directions, challenges, and possible solutions.

Keywords Deep learning · Domain adaptation · Transfer learning · Label set

1 Introduction

In recent years, traditional machine learning and its related applications have achieved great success [37, 41, 49, 84], but these successes require good labeling support. Labeling

✉ Ziyun Cai
caiziyun@163.com

Min Fan
fanmin0330@163.com

Tengfei Zhang
tfzhang@126.com

Baoyun Wang
bywang@njupt.edu.cn

[1] College of Automation, Nanjing University of Posts and Telecommunications, Nanjing 210023, China

in machine learning is quite complicated and tedious, especially when labeling samples of new domains or tasks. Manual labeling will cost a lot of time and money. Semi-supervised [88, 103] learning alleviates the problem to a certain extent. However, Semi-supervised learning also needs a certain amount of labeled instances and a large number of unlabeled instances. A large number of unlabeled instances are difficult to obtain in real-life application scenarios. This usually makes the training model challenging to converge.

Unlike traditional machine learning, transfer learning [62, 109] allows different domains, tasks, and distributions to be used in training and testing. The original intention of transfer learning is to use the previously labeled domain to label the new domain. Just like some people can play the violin, maybe the cello can be learned quickly. Although the data distribution of the source domain and target domain is different, their tasks are the same. This unique transfer learning is domain adaptation. For example, a police officer investigating a crime can use a citizenship photo recorded in the system to quickly and accurately locate a target in a surveillance video [104]. Banks use standard fonts in their databases to help identify a target's handwriting [59].

Domain adaptation (DA) is a particular case of transfer learning (TL). In the last few decades, various shallow domain adaptation methods have been proposed to solve domain transfer between the source and target domains. Shallow domain adaptation can usually be divided into three categories. 1) Instance-based domain adaptation. It is achieved by adjusting the weights of the instances so that the distributions of the two domains are similar [7, 20], 2) Feature-based domain adaptation. It achieves domain adaptation by adjusting the features of two domains [32, 61]. 3) Parameter-based Domain Adaptation. It performs better results by adjusting the model parameters [7, 99]. With the advancement of technology [38], more and more new fields and new tasks require suitable labels. The performance of shallow domain adaptation can no longer meet today's requirements for accuracy. Deep neural networks are widely used in computer vision [1, 2, 47, 69] and natural language processing [39, 44, 76, 83] applications. Deep neural networks have more computing units and more robust non-linear representations [3], which can establish better decision-making boundaries. Therefore, the idea of combining domain adaptation and deep neural networks was born. Recently commonly used deep neural network models today include convolutional neural networks (CNNs) [3, 18, 22, 29, 40, 43], deep belief networks (DBNs) [23, 36, 56–58], and stacked autoencoders (SAEs) [30, 90, 107].

In this paper, we analyze and discuss the deep DA methods.To summarize, the main contributions are:

- We divided the deep domain adaptation into several categories based on the label set of the source domain and the target domain.
- We summarized various methods of Closed-set domain adaptation.
- We discussed current methods of multi-source domain adaptation.
- We discussed future research directions, challenges, and possible solutions.

The remainder of this survey is structured as follows. In Section 2, we reviewed the related work. In Section 3, we first define some notations, and then we categorize deep DA into different settings (given in Fig. 1). In the next two sections, other approaches are discussed for each setting, which is given in Tables 1 and 5 in detail. Finally, the conclusion of this paper and discussion of future work is presented in Section 6.

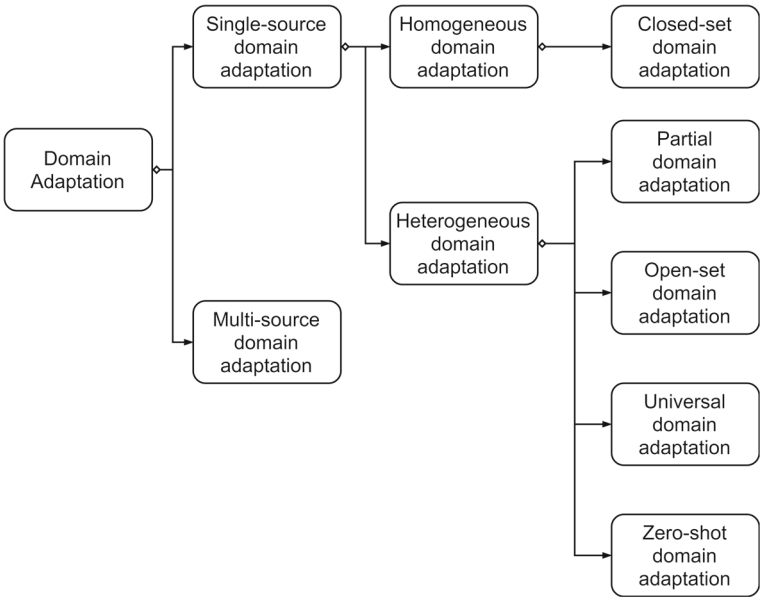

Fig. 1 Classification of domain adaptation

2 Related work

Over the past few years, there have been many reviews or surveys on transfer learning and domain adaptation. Pan et al. [62] divided transfer learning into three cases: inductive TL, transductive TL, and unsupervised TL, but they only studied homogeneous feature spaces. Patel et al. [64] focused only on domain adaptation. Csurka et al. [21] briefly described shallow domain adaptation for each case adaptation method, as well as categorizing deep domain adaptation methods into categories based on training loss: classification loss, discrepancy loss, and adversarial loss. However, Csurka et al. only studied deep domain adaptation in visual application scenarios. Wang et al. [93] divided deep domain adaptation methods into single and multi-step, and single-step domain adaptation methods based on training loss into difference-based, adversarial loss-based, and reconstruction-based. Four criteria were proposed in difference-based domain adaptation: class criterion, statistic criterion, architecture criterion, and geometric criterion. Adversarial loss-based domain adaptation consists of generative models and non-generative models. Reconstruction-based domain adaptation consisted of encoder-decoder reconstruction and adversarial reconstruction. Multi-step domain adaptation methods are divided into three categories based on how intermediate domains are selected and utilized: including Hand-crafted, Instance-based, and Representation-based. Sun et al. [82] mainly reviewed some theoretical results and well-established algorithms for multi-source domain adaptation problems. Kouw et al. [42] introduced dataset shifting in transfer learning and domain adaptation and the treatment of domain transfers. Wang et al. [94] analyzed existing work on Zero-shot learning at that time from three perspectives, which are semantic spaces, methods, and applications. Sematic spaces consist of engineered semantic spaces and learned semantic spaces. Zero-shot learning methods are divided into classifier-based methods and instance-based methods. Wilson et al. [96] compare unsupervised deep domain adaptation by examining alternative methods,

Table 1 Classification of Closed-set DA

Closed-set DA	Brief description	Advantages	Disadvantages
Discrepancy-Based methods	Alignment of feature levels by measuring differences in the corresponding activation levels	The structure is simple and easy to understand.	The calculations are large, especially when calculating higher order moments.
Adversarial-Based methods	Based on generative adversarial networks (GANs) and their variants, spurious data is generated to align source and target domains at the pixel or feature level.	With a generator that can produce an unlimited number of synthetic target data, you can train directly on the synthetic data. Without generators can learn domain invariant representation using gradient reversal layer.	Easily explode or vanish in a gradient.
Reconstruction-Based methods	Using data reconstruction as an auxiliary task to ensure feature invariance	Domain invariance of features is easily ensured by reconstruction of the self-autoencoder.	The performance is not very high when the domain discrepancy is large.

Table 2 Accuracy (%) of different domain adaptation methods on the Office-31 datasets

Method	A → W	D → W	W → D	A → D	D → A	W → A	AVG
ResNet	68.4	96.7	99.3	68.9	62.5	60.7	76.1
DDC	75.6	96.0	98.2	76.5	62.2	61.5	78.3
DAN	80.5	97.1	99.6	78.6	63.6	62.8	80.4
RTN	84.5	96.8	99.4	77.5	66.2	64.8	81.6
JAN	85.4	97.4	99.8	84.7	68.6	70.0	84.3
D-CORAL	66.4	95.7	99.2	66.8	52.8	51.5	72.1
CMD	77.0	96.3	99.2	79.6	63.8	63.3	79.9
HoMM	91.7	98.8	100.0	89.1	71.2	70.6	86.9
DCAN	95.0	97.5	100.0	92.6	77.2	74.9	89.5

the unique and common elements, results, and theoretical insights. Cai et al. [8] Give a comprehensive description of the available RGB-D data sets to guide researchers in choosing the right data set to evaluate their algorithms. Chu et al. [19] compared domain adaptation techniques for neural machine translation (NMT) with the techniques beingstudied in statistical machinetranslation (SMT), which has been the main research area in the last two decades (Tables 2, 3, and 4).

After reviewing the above literature, we study deep domain adaptation methods for various scenarios. Firstly, there is the consistently studied closed-set DA, which is the base scenario of most algorithms. In recent years, Partial DA, Open set DA, Universal DA, and Zero-shot DA have been proposed to address domain adaptation in various scenarios.

3 Overview

3.1 Notations and definitions

In this section, we introduce some of the symbols and definitions that will be used in this survey, and the symbols and definitions match those in the survey papers of [21, 93, 94]to maintain consistency across surveys. A domain consists of feature space $\mathcal{X}$ and a marginal probability distribution $P(X)$,where $X = \{x_1, \ldots, x_n\} \in \mathcal{X}$. Given a specific domain $\mathcal{D} = \{\mathcal{X}, P(X)\}$, a task $\mathcal{T}$ consists of label space $\mathcal{Y}$ and an objective predictive function $f(\cdot)$, which can also be viewed as a conditional probability distribution $P(Y|X)$ from a probabilistic perspective. In general, we can learn $P(Y|X)$ in a supervised manner from the

Table 3 Accuracy (%) of different unsupervised domain adaptation methods on the digits datasets

Method	MN → US	US → MN	SV → MN	AVG
PixelDA	95.9			
GTA	95.3	90.8	92.4	92.8
ADR	96.1	93.1	95.0	94.7
DM-ADA	96.7	94.2	95.5	95.4
ALDA	98.6	95.6	98.7	97.6

Table 4 Accuracy (%) of different without generator adversarial domain adaptation methods on the Office-31 datasets

Method	A → W	D → W	W → D	A → D	D → A	W → A	AVG
DANN	82.0	96.9	99.1	79.7	68.2	67.4	82.2
ADDA	86.2	96.2	98.4	77.8	69.5	68.9	82.9
MADA	90.0	97.4	99.6	87.8	70.3	66.4	85.2
DADA	92.3	99.2	100.0	93.9	74.4	74.2	89.0

labeled data $\{x_i, y_i\}$,where $x_i \in \mathcal{X}$ and $y_i \in \mathcal{Y}$. Suppose L_s and L_t are the label sets in the source and target domains.

Assume that we have two domains: the training dataset with sufficient labeled data is the source domain $\mathcal{D}^s = \{\mathcal{X}^s, P(X)^s\}$, and the test dataset with a small amount of labeled data or no labeled data even no data in the traditional sense is the target domain $\mathcal{D}^t = \left\{\mathcal{X}^t, P(X)^t\right\}$. Firstly, we consider the target domain where the label exists, mark the labeled parts as $\mathcal{D}^{tl}$ and the unlabeled parts as $\mathcal{D}^{tu}$, form the entire target domain, $\mathcal{D} = \mathcal{D}^{tl} \cup \mathcal{D}^{tu}$. The task of the source domain is $\mathcal{T}^s = \{\mathcal{Y}^s, P(Y^s|X^s)\}$, and the one of target domain is $\mathcal{T}^t = \left\{\mathcal{Y}^t, P(Y^t|X^t)\right\}$. Similarly, $P(Y^s|X^s)$ can be learned from the source labeled data $\left\{x_i^s, y_i^s\right\}$, and $P(Y^t|X^t)$ can be learned from the target labeled data $\left\{x_i^{tl}, y_i^{tl}\right\}$ and unlabeled data $\left\{x_i^{tu}\right\}$. Then, we do not have a traditional sample of the target domain available, we need to introduce a semantic representation $a_c \in \mathbb{R}^Q$ to aid in network training. The commonness between two domains is defined as the Jaccard distance between two label sets, $\xi = \frac{|\mathcal{L}_s \cap \mathcal{L}_t|}{|\mathcal{L}_s \cup \mathcal{L}_t|}$.

3.2 Dataset

In this subsection, we introduce some usual datasets for DA. **Office-31** [70] is relatively small, with 4,652 images in 31 classes. Three domains, namely A, D, W, are collected by downloading from amazon.com (A), taking from DSLR (D), and from web camera (W). Six domain adaptation tasks: A→W, D→W, W→D, A→D, D→A, and W→A. **Office-Home** [89] is a larger dataset, with 4 domains of distinct styles: Artistic, Clip Art, Product, and Real-World. Each domain contains images of 65 object categories. Denoting them as Ar, Cl, Pr, Rw, we obtain twelve domain adaptation tasks: Ar→Cl, Ar→Pr, Ar→Rw, Cl→Ar, Cl→Pr, Cl→Rw, Pr→Ar, Pr→Cl, Pr→Rw, Rw→Ar, Rw→Cl, and Rw→Pr. **VisDA2017** [68] (VD) comprises of 12 categories with synthenic (S) and real-world (R) domains. **Office-Caltech** [33] utilizes the shared classes in **Office-31** and **Caltech** as whole dataset.

3.3 Different scenarios for domain adaptation

The case for traditional machine learning is $\mathcal{D}^s = \mathcal{D}^t$ and $\mathcal{T}^s = \mathcal{T}^t$. As for transfer learning, Pan et al. [61] divide data set divergence into divergence in the domain itself and divergence brought about by the task. The former is generally caused by distribution shifts or feature space divergence, while the latter is caused by a divergence in the conditional distribution or label space. Based on these two types of divergence, Pan et al. classify transfer learning into three categories: inductive, transductive, and unsupervised transfer learning. In Pan et al.'s classification, domain adaptation falls into transductive transfer learning. It is characterized by the same task $\mathcal{T}^s = \mathcal{T}^t$ but there is a domain divergence $\mathcal{D}^s \neq \mathcal{D}^t$.

First, according to the number of source domains, domain adaptation can be classified into single source domain adaptation and multi-source domain adaptation. Secondly, it is classified into homogeneous domain adaptation and heterogeneous domain adaptation according to the divergence of domains. Under the setting of homogenous domain adaptation, the feature spaces of the target domain and the source domain are almost the same ($\mathcal{X}^s = \mathcal{X}^t$) and ($d^s = d^t$). The main difference lies in the difference in the edge distribution of the target domain and the source domain ($P(X)^s \neq P(X)^t$). However, there is a big difference ($\mathcal{X}^s \neq \mathcal{X}^t$) or ($d^s \neq d^t$) between the feature space of the target domain and the source domain under the heterogeneous domain adaptation setting. In this paper, we do not use the presence or absence of supervision as a classification criterion. We classify source and target domains according to their label sets. The classification of Single-source domain adaptation based on label set is shown in Fig. 2

4 Single-source domain adaptation

4.1 Homogeneous domain adaptation

The first consideration is single-source domain adaptation, *i.e.*, learning a model from a tagged source domain and then generalizing it to other different but related target domains. The feature spaces of the target and source domains are essentially the same. The label sets of the target and source domains are also consistent. We refer to the domain adaptation in this setting as closed-set domain adaptation. Most of the current methods are divided into three main categories. Discrepancy-Based methods, Adversarial-Based methods, and Reconstruction-Based methods. The mentioned network properties are listed in Tables 2, 3 and 4.

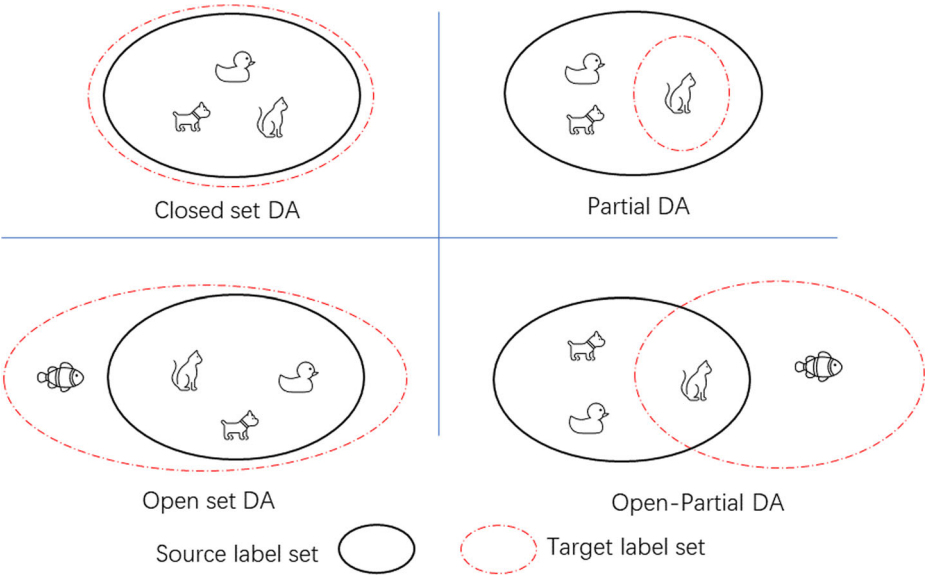

Fig. 2 The classification of Single-source domain adaptation: closed-set DA ($L_s = L_t$), Partial DA ($L_t \subset L_s$), Open set DA ($L_s \subset L_t$), and Open-Partial DA

Closed-set domain adaptation In the Closed-set DA setting, it is supposed that the source and target domains only contain images of the same set of object classes. It does not include images of unknown classes, or classes that do not exist in other domains. And the images should be of the same type. The first thing that comes to mind is to align the source and target domains, then reduce the classification loss, and finally fine-tune [14, 100] the classification case for the target domain. However, direct fine-tuning of the parameters of the deep network is very problematic.

1) Discrepancy-Based methods: In the past, many network structures have been proposed to solve the classification task, such as LeNet-5 [46], AlexNet [43], and VGG [79]. Due to the domain shift between the two domains, these network models' accuracy will be significantly reduced. It is particularly true when the model has been trained in a domain and then used directly in the new domain.

Gretton et al. proposed Maximum Mean Discrepancy (MMD) to measure the discrepancy of the two different domains. MMD is essentially the supremum of the expected difference between two data distribution after the mapping function change. MMD is a very effective way to measure the distance between two distributions. Given two distributions s and t, the MMD is defined as follows,

$$MMD^2(s,t) = \sup_{\|\phi\|_{\mathcal{H}} \leq 1} \|E_{\mathrm{x}^s \sim s}[\phi(\mathrm{x}^s) - E_{\mathrm{x}^t \sim t}[\phi(\mathrm{x}^t)]\|_{\mathcal{H}}^2, \tag{1}$$

where ϕ represents the kernel function that maps the original data to a reproducing kernel Hilbert space (RKHS) and $\|\phi\|_{\mathcal{H}} \leq 1$ defines a set of functions in the unit ball of RKHS.

Based on MMD, Tzeng et al. [87] proposed a new network structure called deep domain confusion (DDC) to solve the DA problem. An adaptation layer is added between the feature layers of the shared weight network. This layer takes MMD between the source and target domain features as a loss and reduces the discrepancy between the source and target domains by minimizing MMD. The MMD also determines the location of the adaptation layer. According to [100], the adaptation layer is more useful at higher layers of features because lower layer features are usually general features that do not carry a higher level of discrimination. Therefore, the adaptation layer of the DDC is placed after fc7. The network architecture of DDC is shown in Fig. 3.

Based on DDC, Long et al. [52] proposed the Deep Adaptive Network (DAN), which differs from DDC in two main ways. 1) Only one layer is adapted in DDC, and multiple layers are adapted in DAN. 2) Only a single kernel function is used in DDC, and a multicore with weighted kernel function is used in DAN. DAN improves the performance through multilayer adaptation and Multi-Kernel Maximum Mean Discrepancy (MK-MMD) [34]. In 2016, long et al. Proposed RTN by imitating residual networks. The classifier layer of RTN connects the source classifier and the target classifier end-to-end. However, the above model assumes that the conditional distributions in the two domains are consistent. In real-world scenarios, this assumption of condition is too strong. For this reason, further research by Long et al. [53] proposed Joint Adaptation Network (JAN), which adjusts the joint distribution of source and target domains using classification loss and Joint Maximum Mean Discrepancy (JMMD) as a function of loss.

Because of the enormous computational effort required to compute the MK-MMD, Sun et al. [81] utilized CORAL loss to measure the distance between the two domains. Moreover, it can be seamlessly integrated into different layers or architectures. CORAL loss is defined

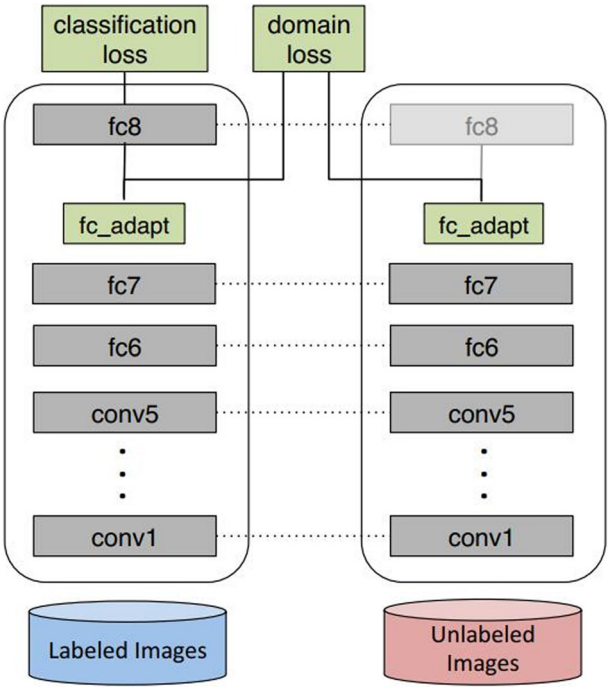

Fig. 3 Using unsupervised domain adaptation as an example, a sample training network with labeled source domains is entered on the left, and the right network has the same weights as the left. The source and target domains are aligned by minimizing the classification loss and MMD distance. Thus, the classifier on the source domain can also be applied to the target domain

as the distance between the second-order statistics (covariance) of the source and target domain features.

$$\mathcal{L}_{CORAL} = \frac{1}{4d^2}\|C_S - C_T\|_F^2, \tag{2}$$

where $\|\cdot\|_F^2$ denotes the squared matrix Frobenius norm. C_S and C_T denote the covariance matrices of the source and target data, respectively. The goal of domain adaptation is achieved by optimizing both classification loss and Correlation Alignment (CORAL) loss simultaneously. Zellinger et al. [102] proposed the Central Moment Discrepancy (CMD) based on MMD and KL divergence. CMD consists of a vector of empirical expectations and a vector of k-order sample center distances. In simple words, if the probability distributions of samples of source and target domains are similar, then their per-order center distances are also similar. The more similar the sample probability distributions are, the smaller the value of CMD. CMD contains higher-order moment information than KL divergence and reduces the computational effort compared to MMD because there is no need to compute the kernel matrix. Unlike CMD matching higher-order central moment, Higher-order Moment Matching (HoMM) [16] matches higher-order cumulant tensor. Because a higher-order moment tensor contains more information to represent feature distributions better. HoMM can be matched with arbitrary moment tensor, with first-order HoMM and second-order HoMM are equivalent to MMD and CORAL, respectively. Third- and fourth-order moment tensor matching helps achieve global alignment, as higher-order statistics can be adapted to

more complex non-Gaussian distributions. The final objective function of HoMM is as follows,

$$\mathcal{L} = \mathcal{L}_s + \lambda_d \mathcal{L}_d + \lambda_{dc} \mathcal{L}_{dc}, \tag{3}$$

where $\mathcal{L}_s$ is the classification loss in the source domain, $\mathcal{L}_d$ is the domain discrepancy loss measured by the higher-order moment matching, and $\mathcal{L}_{dc}$ denotes the discriminative clustering loss. Note that to obtain reliable discrimination of clustered pseudo-labels, set λ_{dc} to 0 in the initial iteration and enable clustering loss λ_{dc} after the total loss has stabilized. The domain discrepancy loss can be given as,

$$\mathcal{L}_d = \frac{1}{b^2}\sum_{i=1}^{b}\sum_{j=1}^{b} k(\boldsymbol{h}_{sp}^i, \boldsymbol{h}_{sp}^j) - \frac{2}{b^2}\sum_{i=1}^{b}\sum_{j=1}^{b} k(\boldsymbol{h}_{sp}^i, \boldsymbol{h}_{tp}^j) + \frac{1}{b^2}\sum_{i=1}^{b}\sum_{j=1}^{b} k(\boldsymbol{h}_{tp}^i, \boldsymbol{h}_{tp}^j), \tag{4}$$

Where b is the batch size, $k(\boldsymbol{x}, \boldsymbol{y}) = \exp(-\gamma \|\boldsymbol{x} - \boldsymbol{y}\|_2)$ is the RBF kernel function, $\boldsymbol{h}_{sp}^i$ denotes a randomly sampled value in the p-level tensor. The discriminative clustering loss can be given as,

$$\mathcal{L}_{dc} = \frac{1}{n_t}\sum_{i=1}^{n_t} \|\boldsymbol{h}_t^i - \boldsymbol{c}_{\hat{y}_t^i}\|_2^2, \tag{5}$$

where $\hat{y}_t^i$ is the assigned pseudo-labels of x_t^i, $\boldsymbol{c}_{\hat{y}_t^i} \in \mathbb{R}^L$ denotes its estimated class center. Group moment matching and random sample matching to perform compact tensor matching in HoMM. Li et al. [48] introduced the attention mechanism in domain adaptation. This mechanism can simulate the independence between source and target convolution channels. Furthermore, it does facilitate the alignment of cross-domain features.

2) Adversarial-Based methods: Unlike previous Discrepancy-Based methods, the adversarial approach's basic idea is a minimax game. The game ordinary takes place between the domain discriminator and the feature extractor. The domain discriminator identifies whether an instance comes from the target domain. The purpose of the feature extractor is to extract features that can cheat domain discriminators. The whole network iterates between the training domain discriminator and the feature extractor until the whole model converges.

Adversarial domain adaptation networks with generators generally synthesize the source data with labels (or pseudo-labels) into the target data and keep the labels (or pseudo-labels). The synthesized target data is then used to train the network. Unsupervised pixel-level domain adaptation (PixelDA) [5] employed pixel-space cross-domain transformation achieve domain adaptation. Unlike classical GANs, the input to PixelDA contains not only noise vectors but also source images. An almost infinite amount of training data can be synthesized using the noise vector and the source image. The PixelDA model maps the source domain image to the target domain image at the pixel level. It can change the architecture of a particular task without having to retrain the domain adaptation component. However, The downside of pixelDA is that it can only deal with the low-level differences between the source domain and the target domain, mainly noise, resolution, lighting, color. If the object type changes, geometric changes are difficult to deal with. Rather than using GANs as a data enhancement step as before, Sankaranarayanan et al. [75] utilized GANs to obtain rich gradient information that bridges the gap between the source and target domains. The joint adversarial-discriminative approach transfers the information of the target distribution to the learned embedding using a generator-discriminator pair (Fig. 4).

Saito et al. [73] proposed a novel adversarial alignment technique to avoid misclassification of samples near the decision boundary. The model is composed of a feature extractor

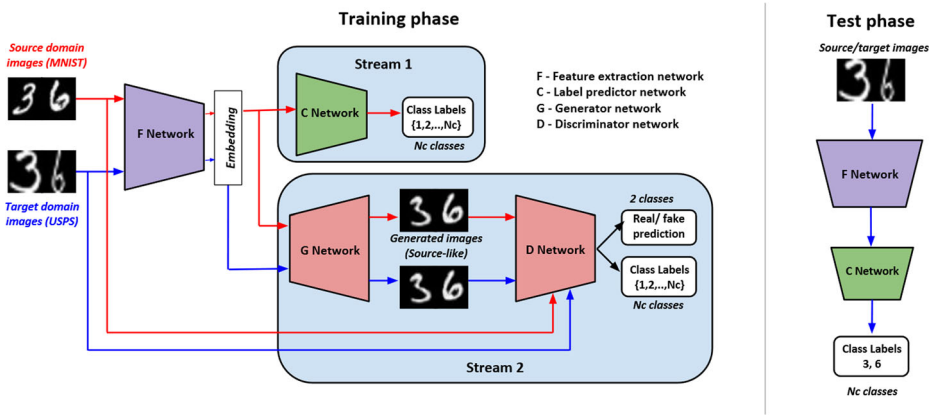

Fig. 4 The main components of the GTA network are illustrated. During the training phase, the pipeline consists of two parallel streams. 1) Stream 1 is updated using supervised classification loss; 2) Stream 2 keeps the images from the target and source domains more similar via a GAN. In the testing phase, Stream2 is removed and classified using the F-C pair

G and a classifier C. Different from previous work, this classifier also acts as a discriminator. C classfies input x into K class j by $p(y = j|x) = \frac{\exp(l_j)}{\sum_{k=1}^{K} exp(l_k)}$. In the confrontation training of mixed domain samples, the discriminator can detect the instances close to the boundary. The feature extractor pushes these samples away from the boundary to generate discriminative domain invariant features. The goal of Adversarial Dropout Regularization (ADR) is to learn G and C by solving the optimization problem:

$$\min_{G,C} L(X_s, Y_s) = -\mathbb{E}_{(x_s,y_s)\sim(X_s,Y_s)} \sum_{k=1}^{K} \mathbb{1}_{[k=y_s]} \log C(G(x_s))_k, \tag{6}$$

$$\max_{G} \min_{C} L(X_s, Y_s) - L_{adv}(X_t), \tag{7}$$

$$L_{adv}(X_t) = \mathbb{E}_{x_t\sim X_t}[d(C_1(G(x_t)), (C_2(G(x_t)))], \tag{8}$$

$$d(p_1, p_2) = \frac{1}{2}(D_{kl}(p_1|p_2) + D_{kl}(p_2|p_1)), \tag{9}$$

where $L(X_s, Y_s)$ is standard classification loss, $L_{adv}(X_t)$ is the loss of between $C1$ and $C2$, $d(p1, p2)$ is represent the difference between $p1$ and $p2$, $D_{kl}()$ is KL divergence. Inspired by VAE-GAN [45], Xu et al. [97] proposed Adversarial Domain Adaptation with Domain Mixup (DM-ADA), which makes the source and target domains consistently distributed through VAE and discriminators. The pixel-level and feature-level domain mixture and well-designed soft domain labels improve the generalization capability. The classifier is optimized with cross-entropy loss. Namely, The source domain classifier loss is $\mathcal{L}_C = -\mathbf{E}_{x^s\sim P_s} \sum_{i=1}^{K} y_i^s \log(C([\cdot]))$, where K is the numbers of classes. Chen et al. [17] combine domain adversarial learning with self-learning to proposed Adversarial-Learned Loss for Domain Adaptation (ALDA). The confusion matrix is used to eliminate (or reduce) the effect of noise in the pseudo labels. In contrast to ordinary domain adversarial learning, this adversarial loss incorporates classifier predictions and label information into the optimization. In this way, the model enables level-by-level feature alignment. The noise-corrected can align the features between the source and target domains. According to the theory of [4], the expected error of the target sample can be defined by the expected error

in the source domain and the difference in features between the domains. Therefore, the expected error of the target for noise-corrected is theoretically bounded. Therefore, the expected error of ALDA is theoretically bounded.

The key to adversarial domain adaptation networks without generators is to learn domain invariant representations from the source and target samples. These representations are used to deceive the classifier (discriminator) and introduce domain confusion losses to improve the performance of the network. Domain Adaptive Neural Network (DANN) was first proposed at the 2014 Pacific Rim AI Conference. Ganin et al. [28] formally proposed DANN to address the unsupervised domain adaptation (UDA) problem. DANN can be easily implemented using deep learning packages in the Deep Learning Framework. DANN is powered by the feature extractor $G_f(\cdot;\theta_f)$, the label predictor $G_y(\cdot;\theta_y)$, the domain classifier $G_d(\cdot;\theta_d)$, and the gradient inversion layer (GRL) comprise. It uses GRL for back-propagation training so that the distribution of source and target domains is consistent. The optimization goal of the entire network has two components: minimizing the source domain classification error, maximizing the domain classification error, and introducing λ as a trade-off parameter. Generally speaking, aligning the source domain and the target domain is to map the target domain to the source domain and then classify the target domain through a classifier trained on the source domain. However, Adversarial Discriminative Domain Adaptation (ADDA) [86] maps both the source domain and the target domain to a shared space and reduces the distance uses a trained classifier to classify the mapped target domain. ADDA minimizes the source and target representation distance by iteratively minimizing the following functions, which is most similar to the original GAN:

$$\min_{M^s,C} \mathcal{L}_{cls}(X^s, Y^s) = -\mathbb{E}_{(x^s,y^s)\sim(X^s,Y^s)} \sum_{k=1}^{k} \mathbb{1}_{[k=y^s]} \log C(M^s(x^s)), \tag{10}$$

$$\min_{D} \mathcal{L}_{advD}(X^s, X^t, M^s, M^t) = -\mathbb{E}_{(x^s)\sim(X^s)}[\log D(M^s(x^s))] - \mathbb{E}_{(x^t)\sim(X^t)}[\log(1 - D(M^t(x^t)))], \tag{11}$$

$$\min_{M^s,M^t} \mathcal{L}_{advM}(M^s, M^t) = -\mathbb{E}_{(x^t)\sim(X^t)}[\log D(M^t(x^t))], \tag{12}$$

where the mappings M^s and M^t are learned from the source data X^s and target data X^t, C represents a classifier working on the source domain. $\mathcal{L}_{cls}$ is optimized by training the source model using the labeled source data. $\mathcal{L}_{advD}$ is minimized to train the discriminator, while $\mathcal{L}_{advM}$ is learning a representation that is domain invariant. After ADDA, Multi-Adversarial Domain Adaptation (MADA) [65] utilize more than one class discriminator, and this change may bring three benefits. It avoids rigidly assigning each point to only one domain discriminator, similar to using soft labels to increase information. It avoids negative transfer because each moment is only aligned with the most relevant class, and irrelevant classes are filtered out. By weighting, these domain discriminators with different parameters facilitate the positive transfer of each instance. This structure also has an obvious shortcoming, that is, specifying a discriminator for each class, and the computational cost is quite high. The structure of MADA is shown in Fig. 5. To make the classification on the target domain more precise, the DADA [85] proposed by Tang et al. makes the joint distribution alignment between the two domains more explicit. They proposed a target loss based on the design of an integrated classifier by using conditional category probability weighted domain prediction. The entropy minimization principle was used for the regularization term. Inspired by Wasserstein GAN, Shen et al. [77] proposed a novel approach to

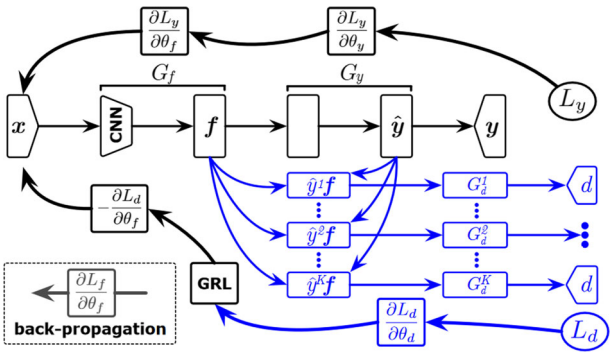

Fig. 5 The architecture of the Multi-Adversarial Domain Adaptation (MADA) approach, where f is the extracted deep features, $\hat{y}$ is the predicted data label, and $\hat{d}$ is the predicted domain label; G_f is the feature extractor, G_y and L_y are the label predictor and its loss,G_d^K and L_d^K are the domain discriminator and its loss; GRL stands for Gradient Reversal Layer. The blue part shows the multiple adversarial networks (each for a class, K in total).The network uses each discriminator to determine which domain a sample belongs to. After the discriminator is trained, it is classified by the classifier. Best viewed in color

learn domain invariant feature representations, namely Wasserstein Distance Guided Representation Learning (WDGRL). WDGRL utilizes neural networks to estimate the empirical Wasserstein distance between the source and target samples and optimizes the network of feature extractors to minimize the estimated Wasserstein distance.

Fang et al. [25] introduced a perturbation function in the label classifier to simulate changes in the distribution of labels in different domains, and insert ResNet to learn the perturbation function. By learning the perturbation function, the label classifier will be more robust and accurate. And the joint distribution of image features and class labels is used to align the source and target domains to obtain a more robust and differentiated feature representation. An intuitive illustration is shown in Fig. 6, where the target classifier with the perturbation functioncorrectly classifies image samples from the target domain.

3) Reconstruction-Based Approaches: The goal of the reconstruction-based approach is to extract domain invariant representations. The Deep Reconstruction Classification Network (DRCN) proposed in Ghifary et al. [31] learns a shared encoding representation.The DRCN is a CNN architecture that combines two pipelines with a shared encoder. The shared encoder can be considered as a feature extractor. The first pipeline performs supervised classification in the source domain, while the second pipeline performs unsupervised reconstruction on the target domain. Domain separation networks (DSNs) [6] model the private and shared components for domain representations. It uses a scale-invariant mean squared error reconstruction loss.

4.2 Heterogeneous domain adaptation

In a heterogeneous domain adaptation scenario, there are many situations in the relationship between the source domain and the target domain. When domain adaptation is applied to a real-world scenario, it is more likely to encounter a situation where the source domain and the target domain have large differences. We divide heterogeneous domain adaptation into three categories based on the shared category of the source domain and target domain. The category of the target domain is included in the source domain ($L_t \subset L_s$)is Partial DA. The category of the source domain is included in the target domain ($L_s \subset L_t$) is Open set DA.

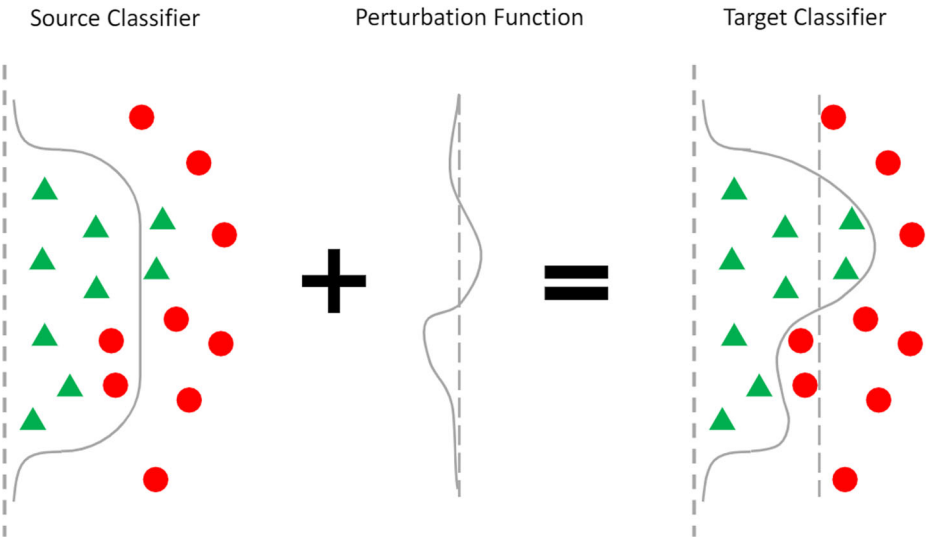

Fig. 6 A well-trained source classifier may fail to classify images in the target domain correctly. By adding a perturbation function, the target classifier corrects the mistakes made by the source classifier on the target domain. Here the red dots and green triangles denote image samples from the target domain. Best viewed in color

Part of the source domain and target domain category is shared category is Open-Partial DA. For open-Partial DA is not studied separately, it is integrated into the Universal DA, and additional Zero-shot DA for special cases is added (Table 5).

Partial domain adaptation With the advent of the Big Data era, the label set of the target domain is usually included in one of the source domains. Previous approaches for homogeneous domain adaptation have performed poorly. While the Partial DA proposed by Long et al. applies to a more general case. The label set of the target domain is contained in the source domain (*i.e.*, the target domain label space is just a subspace of the source domain label space). In this case, there is an obvious problem. Those labels (or samples) that exist only in the source domain will result in a negative transfer.

Table 5 Classification of heterogeneous DA

Heterogeneous DA	Brief description
Partial DA	The category of the target domain is included in the source domain. Avoid negative transfer by reducing the impact of unique category samples in the source domain through either an adversarial-based approach or a difference-based approach.
Open set DA	The category of the source domain is included in the target domain. Classify samples that are unique to the target domain by clustering. Improve performance by learning domain invariant representation.
Universal DA	No need to know the category relationship for source and target domains. Maximize the inter-class distance and minimize the intra-class distance.
Zero-shot DA	No sample in the conventional sense. Auxiliary classification through semantic information

Cao et al. [11] proposed partial transfer learning to address partial transfer learning from large-scale domains to small-scale domains. The architecture of the proposed Selective Adversarial Networks (SAN) for partial transfer learning is shown in Fig. 7. The final objective of Selective Adversarial Network (SAN) is,

$$C(\theta_f, \theta_y, \theta_d^k|_{k=1}^{|\mathcal{C}_s|}) = \frac{1}{n_s}\sum_{\mathrm{x}_i\in\mathcal{D}s} L_y(G_y(G_f(\mathrm{x}_i)), y_i) + \frac{1}{n_t}\sum_{\mathrm{x}_i\in\mathcal{D}_t} H(G_y(G_f(\mathrm{x}_i))) - \frac{\lambda}{n_s+n_t}\sum_{k=1}^{|\mathcal{C}_s|}[(\frac{1}{n_t}\sum_{\mathrm{x}_i\in\mathcal{D}_t}\hat{y}_i^k) \times (\sum_{\mathrm{x}_i\in\mathcal{D}_s\cup\mathcal{D}_t}\hat{y}_i^k L_d^k(G_d^k(G_f(\mathrm{x}_i)), d_i))], \quad (13)$$

where λ is a hyper-parameter that trade-offs the two objectives in the unified optimization problem. $H(\cdot)$ is the conditional-entropy loss functional, $H(G_y(G_f(\mathrm{x}_i))) = -\sum_{k=1}^{|\mathcal{C}_s|}\hat{y}_i^k \log \hat{y}_i^k$. SAN down-weight the domain discriminators responsible for the outlier source classes as follows,

$$\mathcal{L}_d = \frac{1}{n_s+n_t}\sum_{k=1}^{|\mathcal{C}_s|}[(\frac{1}{n_t}\sum_{\mathrm{x}_i\in\mathcal{D}_t}\hat{y}_i^k) \times (\sum_{\mathrm{x}_i\in\mathcal{D}_s\cup\mathcal{D}_t}\hat{y}_i^k L_d^k(G_d^k(G_f(\mathrm{x}_i)), d_i))]. \quad (14)$$

The optimization problem is to find the network parameters $\hat{\theta}_f$, $\hat{\theta}_y$ and $\hat{\theta}_d^k (k = 1, 2, \ldots, |\mathcal{C}_s|)$ that satisfy the following functions,

$$(\hat{\theta}_f, \hat{\theta}_y) = \arg\min_{\theta_f,\theta_y} C(\theta_f, \theta_y, \theta_d^k|_{k=1}^{|\mathcal{C}_s|}), \quad (15)$$

$$(\hat{\theta}_d^1, \ldots, \hat{\theta}_d^{|\mathcal{C}_s|}) = \arg\min_{\theta_d^1,\ldots,\theta_d^{|\mathcal{C}_s|}} C(\theta_f, \theta_y, \theta_d^k|_{k=1}^{|\mathcal{C}_s|}), \quad (16)$$

SAN reduces negative transfer due to categories that do not belong to the target domain by weighting the instances and weighting the categories. SAN preliminary addresses Partial DA, which simultaneously circumvents negative transfer by filtering the outlier source class $\mathcal{C}_s \backslash \mathcal{C}_t$ and promotes positive transfer by maximizing the data distribution $p_{\mathcal{C}_t}$ and q in

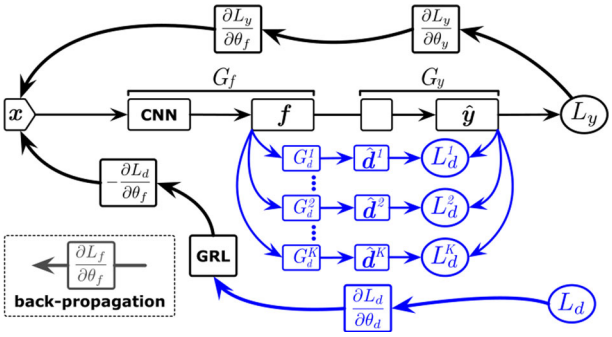

Fig. 7 f is the extracted deep features, $\hat{y}$ is the predicted data label, and $\hat{d}$ is the predicted domain label; G_f is the feature extractor, G_y and L_y are the label predictor and its loss, G_d^k and L_d^k are the domain discriminator and its loss; GRL stands for Gradient Reversal Layer. The blue part shows the class-wise adversarial networks ($|\mathcal{C}_s|$ in total). Best viewed in color

the shared tag space $\mathcal{C}_t$. Modified from SAN, Cao et al. [12] proposed Partial Adversarial Domain Adaptation (PADA).The architecture of PADA is shown in Fig. 8

$$C(\theta_f, \theta_y, \theta_d) = \frac{1}{n_s} \sum_{\mathrm{x}_i \in \mathcal{D}_s} \gamma_{y_i} L_y(G_y(G_f(\mathrm{x}_i)), y_i) - \frac{\lambda}{n_s} \sum_{\mathrm{x}_i \in \mathcal{D}_s} \gamma_{y_i} L_d(G_d(G_f(\mathrm{x}_i)), d_i) - \frac{\lambda}{n_t} \sum_{\mathrm{x}_i \in \mathcal{D}_t} L_d(G_d(G_f(\mathrm{x}_i)), d_i), \tag{17}$$

$$\gamma = \frac{1}{n_t} \sum_{i=1}^{n_t} \hat{\mathrm{y}}_i, \tag{18}$$

where γ is a $|\mathcal{C}_s|$-dimensional weight vector quantifying the contribution of each source class, y_i is the ground truth label of source point x_i while γy_i is the corresponding class weight, and λ is a hyper-parameter that trade-offs the source label classifier and the partial adversarial domain discriminator in the optimization problem. PADA averages the label predictions and all target data to eliminate the effects of possible errors.

Improved based on DANN, Zhang et al. [105] proposed a two-domain classifier strategy named Importance Weighted Adversarial Nets (IWAN) to solve partial DA. The network consists of two feature extractors F_s and F_t, two domain classifiers D and D_0. The green parts are the feature extractors for source and target domains. The network architecture is shown in Fig. 9. The overall objective of the weighted adversarial nets-based method is,

$$\min_{F_s, C} \mathcal{L}_s(F_s, C) = -\mathbb{E}_{\mathrm{x}, y \sim p_s(\mathrm{x}, y)} \sum_{k=1}^{K} \mathbb{1}_{[k=y]} \log C(F_s(\mathrm{x})), \tag{19}$$

$$\min_{D} \mathcal{L}_D(D, F_s, F_t) = -(\mathbb{E}_{\mathrm{x} \sim p_s(\mathrm{x})}[\log D(F_s(\mathrm{x}))] + \mathbb{E}_{\mathrm{x} \sim p_t(\mathrm{x})}[\log(1 - D(F_t(\mathrm{x})))]) \tag{20}$$

$$, \min_{F_t} \max_{D_0} \mathcal{L}_w(C, D_0, F_s, F_t) = \gamma \mathbb{E}_{\mathrm{x} \sim p_t(\mathrm{x})} H(C(F_t(\mathrm{x}))) + \lambda(\mathbb{E}_{\mathrm{x} \sim p_s(\mathrm{x})}[w(\mathrm{z}) \log D_0(F_s(\mathrm{x}))] + \mathbb{E}_{\mathrm{x} \sim p_t(\mathrm{x})}[\log(1 - D_0(F_t(\mathrm{x})))]), \tag{21}$$

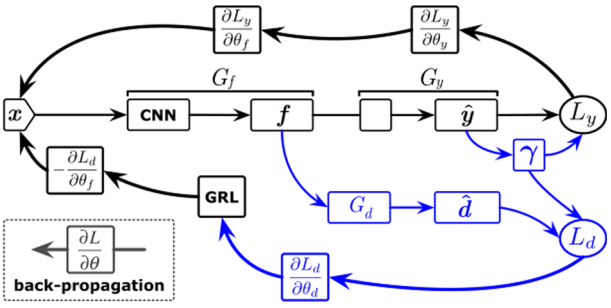

Fig. 8 Overview of the architecture of PADA, f is the extracted deep features, $\hat{y}$ is the predicted data label, and $\hat{d}$ is the predicted domain label by softmax probability; G_f is the feature extractor, G_y and L_y are the label predictor and its loss, G_d^k and L_d^k are the domain discriminator and its loss, respectively, γ is the class weights averaged over the label predictions of target data. Best viewed in color

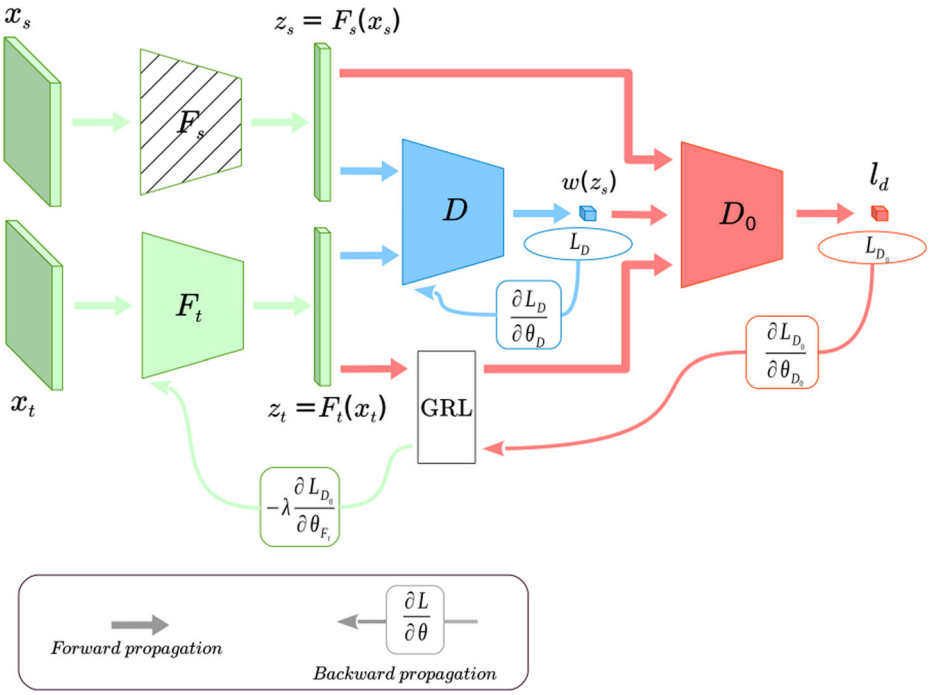

Fig. 9 F_s and F_t are feature extractors for the source and target domains, respectively. The parameters of F_s are pre-learned and are not updated during training. D is the domain classifier that gets the importance weights w of the source samples and does not participate in the minimax game. D_0 is another classifier that uses a weighted source domain sample and a target sample for the maximal-minimal game. GRL stands for Gradient Reversal Layer. Best viewed in color

where λ is the trade-off parameter, $\mathcal{L}_s$ is the loss of source domain classifier, $\mathcal{L}_D$ is the loss of domain classifier D, $\mathcal{L}_w$ is the sum of loss of domain classifier D_0 and entropy of target classes, The objectives are optimized in stages. F_s and C are pre-trained on the source domain data and fixed afterwards. Then the D, D_0 and F_t are optimized simultaneously without the need of revisiting F_s and C. The relative importance of the source sample is given by $w(\mathbf{z}) = \frac{\tilde{w}(\mathbf{z})}{\mathbb{E}_{\mathbf{z}\sim p_s(\mathbf{z})}\tilde{w}(\mathbf{z})}$, which $\tilde{w}(\mathbf{z}) = \frac{1}{\frac{p_s(\mathbf{z})}{p_t(\mathbf{z})}+1}$. The essence of IWAN is to reduce the Jensen-Shannon divergence between the weighted source data distribution and the target data distribution in the feature space. Cao et al. [13] proposed the Example Transfer Network (ETN) gradually reduces the weight of irrelevant samples of non-shared categories on the source classifier and employs a domain classifier to quantify the transferability of instances. The architecture of ETN is shown in Fig. 10. The goal of ETN model is finding saddle-point solutions $\hat{\theta}_f$, $\hat{\theta}_y$, $\hat{\theta}_d$ and $\hat{\theta}_{\tilde{y}}$ to model parameters as follows,

$$(\hat{\theta}_f, \hat{\theta}_g) = \arg\min_{\theta_f, \theta_y} \mathbb{E}_{G_y} - \mathbb{E}_{G_d}, \tag{22}$$

$$(\hat{\theta}_d) = \arg\min_{\theta_d} \mathbb{E}_{G_d}, \tag{23}$$

$$(\hat{\theta}_{\tilde{y}}) = \arg\min_{\theta_{\tilde{y}}} \mathbb{E}_{\tilde{G}_y} + \mathbb{E}_{\tilde{G}_d}, \tag{24}$$

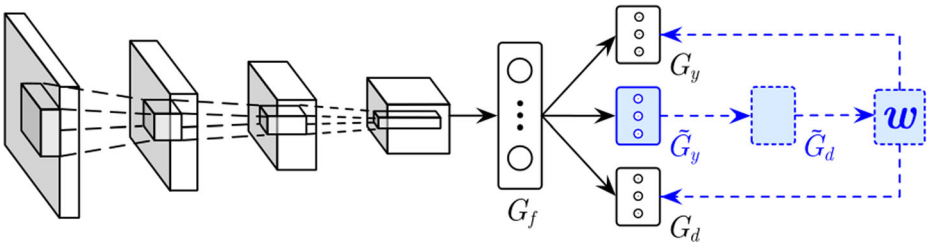

Fig. 10 The architecture of ETN is shown in the figure, where G_f is the feature extractor, G_y is the source classifier, and G_d is the domain identifier for domain alignment; $\tilde{G}_d$ is the auxiliary domain identifier that quantifies the transferability w of each source example, and $\tilde{G}_y$ is the auxiliary predictive label that encodes the distinguishing information as the auxiliary domain discriminator $\tilde{G}_d$. Best viewed in color

$$\mathbb{E}_{G_y} = \frac{1}{n_s}\sum_{i=1}^{n_s} w(\mathrm{x}_i^s) L(G_y(G_f(\mathrm{x}_i^s), \mathrm{y}_i^s)) + \frac{\gamma}{n_t}\sum_{j=1}^{n_t} H(G_y(G_f(\mathrm{x}_j^t))), \tag{25}$$

$$\mathbb{E}_{G_d} = -\frac{1}{n_s}\sum_{i=1}^{n_s} w(\mathrm{x}_i^s)\log(G_d(G_f(\mathrm{x}_i^s))) - \frac{1}{n_t}\sum_{j=1}^{n_t}\log(1 - G_d(G_f(\mathrm{x}_i^t))), \tag{26}$$

$$\mathbb{E}_{\tilde{G}_y} = -\frac{\lambda}{n_s}\sum_{i=1}^{n_s}\sum_{c=1}^{|\mathcal{C}_s|}[y_{i,c}^s \log \tilde{G}_y^c(G_f(\mathrm{x}_i^s)) + (1 - y_{i,c}^s)\log \tilde{G}_y^c(G_f(\mathrm{x}_i^s))], \tag{27}$$

$$\mathbb{E}_{\tilde{G}_d} = -\frac{1}{n_s}\sum_{i=1}^{n_s}\log(\tilde{G}_d(G_f(\mathrm{x}_i^s))) - \frac{1}{n_t}\sum_{j=1}^{n_t}\log(1 - \tilde{G}_d(G_f(\mathrm{x}_i^t))), \tag{28}$$

where $w(\mathrm{x}_i^s) = 1 - \tilde{G}_d(G_f(\mathrm{x}_i^s))$ is the weight of each source example x_i^s, which quantifies the example's transferability, γ is a trade-off parameter. Equation (25) and (26) proposed transferability weighting framework. From (27) and (28), with the help of $\tilde{G}_y$ (leaky-softmax activation function) ,$\tilde{G}_d$, which is trained with label information and domain information, resolving the ambiguity between shared and unshared classes. ETN can derive more accurate and discriminative weights to quantify the transferability of each source example. The accuracy of the above network is listed in Table 6.

Open set domain adaptation Open set DA case is the opposite of Partial DA mentioned above, where the source domain label set is only a small fraction of the one of the target domain. The specific setup for the Open set DA problem is that the target domain contains

Table 6 Accuracy (%) of different Partial domain adaptation methods on the Office-31 datasets

Method	A → W	D → W	W → D	A → D	D → A	W → A	AVG
SAN	93.9	99.3	99.4	94.3	94.2	88.7	95.0
PADA	86.5	99.3	100.0	82.2	92.7	95.4	92.7
IWAN	89.2	99.3	99.4	90.5	95.6	94.3	94.7
ETN	94.5	100.0	100.0	95.0	96.2	94.6	96.7

Note that there are 31 categories in the source domain and10 categories in the target domain

all the classes in the source domain. We need to classify the data correctly for the known classes in the target domain (common to both target and source domains), and the data for all unknown classes (only in the target domain) is classified as "unknown" because we don't have information about these classes.

Busto et al. [63] first proposed a novel problem scenario. Considering the actual scene, there is usually an intersection between the source domain and the target domain, rather than the closed set previously set (Fig. 11).

The method used in [63] is to project the target domain and source domain into the same space-based on distance and then classify the target samples by SVM. In the unsupervised scenario, the objective functions to be optimized are as follows,

$$
\begin{aligned}
\min_{x_{ct}, w_{ct}, o_t} \quad & \sum_t (\sum_c d_{ct} x_{ct} + \sum_c w_{ct} + \lambda o_t), \\
s.t. \quad & \sum_c x_{ct} + o_t = 1 \\
& \sum x_{ct} \geq 1 \\
& a_{ct} x_{ct} + \sum_{t' \in N_t} \sum_{c'} d_{cc'} x_{c't'} - w_{ct} \leq a_{ct} \\
& x_{ct}, o_t \in \{0, 1\} \\
& w_{ct} \geq 0
\end{aligned} \tag{29}
$$

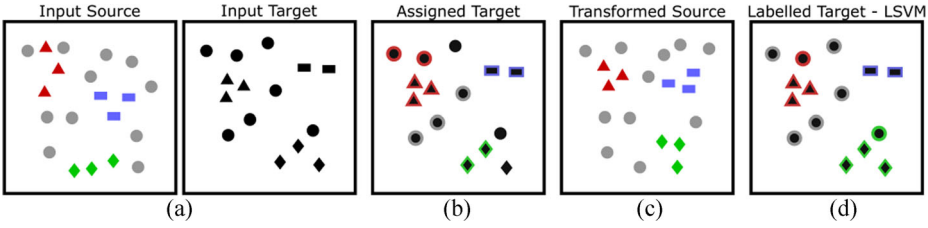

Fig. 11 Overview of Unsupervised Open Set Domain Adaptation Methods. **a** The source domain contains labeled and unlabeled images, where the same color means the labels are the same and gray means they belong to an unknown category. For the samples in the target domain, there are no labels. **b** As a first step, the category labels are assigned to some target samples, while the outliers have no labels. **c** By minimizing the distance between the source and the target domain samples by the same category. Then iterate between (b) and (c) until convergence to a local minimum. **d** shows the classification results of the algorithm. Best viewed in color

where $w_{ct} = x_{ct}(\sum_{t'=N_t} \sum_{c'} x_{c't'} dcc')$, $a_{ct} = \sum_{t' \in N_t} \sum_{c'} d - cc'$. Compared with unsupervised scenario, semi-supervised scenario adds some restrictions to ensure that labeled target samples are not misclassified. After solving the assignment problem by iterating, the source domain is transferred to the target domain by a linear transformation, which is represented by a matrix $W \in \mathbb{R}^{D \times D}$. It could be estimated by minimizing the following loss function: $f(W) = \frac{1}{2}\|WP_S - P_T\|_F^2$. After the approach has converged, Classification of data in the target domain using linear SVM trained in the source domain. Later, based on the idea of Generative Adversarial Networks (GAN), Saito et al. [74] Proposed a new method for open set domain adaptation. The network architecture is shown in the Fig. 12. The goal is to correctly categorize known target samples into corresponding known classes and recognize unknown target samples as unknown. The objective function is as follows,

$$\min_C L_s(x_s, y_s) + L_{adv}(x_t), \tag{30}$$

$$\min_G L_s(x_s, y_s) - L_{adv}(x_t), \tag{31}$$

$$L_s(x_s, y_s) = -\log(p(y = y_s|x_s)), \tag{32}$$

$$p(y = y_s|x_s) = (C \circ G(x_s))_{y_s}, \tag{33}$$

$$L_{adv}(x_t) = -t \log(p(y = K+1|x_t)) \\ -(1-t)\log(1 - p(y = K+1|x_t)), \tag{34}$$

where $L_s(x_s, y_s)$ is the loss of classifier C, t is set as 0.5. The generator attempts to maximize the value of $L_{adv}(x_t)$. Saito uses the symmetric KL divergence as a new binary cross-entropy loss formula. Liu et al. [50] developed a method Separate to Adapt (STA), a progressive separation mechanism consisting of a coarse-to-fine separation pipeline. First, a multi-binary classifier is trained with the source data to estimate the similarity between the data in the target domain and each of the source classes; second, data with extremely high and low similarity are selected as the boundary data for the known and unknown classes, and they are further used to train a fine-grained binary classifier to perform fine-grained separation of all target domain samples. Finally, iterate between the above two steps and use weights to reject samples with unknown domain adaptation classes. The network struc-

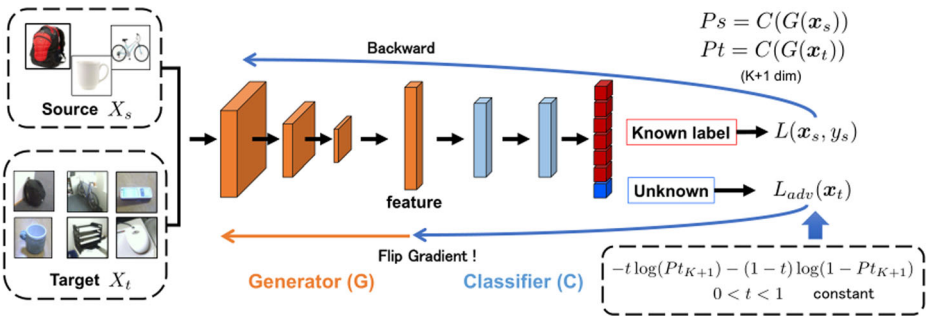

Fig. 12 Overview of the network architecture. The network has been trained to classify source samples. For the target sample, through the minimax game of the classifier and the classifier, the probability that the sample belongs to the unknown class or the correct class is obtained. Best viewed in color

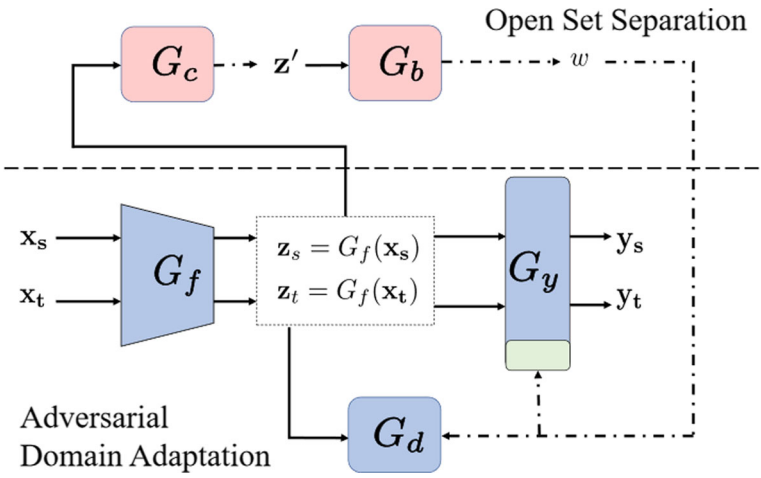

Fig. 13 The separate to Adapt (STA) approach for open set domain adaptation is split into two parts by the dotted line. Above the dotted line consists of a multi-binary classifier $G_c|_{C=1}^{|\mathcal{C}_s|}$ and a binary classifier G_b, which will generate the weights w for rejecting target samples in the unknown classes $\mathcal{C}_t \backslash \mathcal{C}_s$. Below the dotted line is a feature extractor G_f, a classifier G_y, and a domain discriminator G_d to perform adversarial domain adaptation between source and target data in the shared label space. z_s and z_t is the extracted deep features of source and target domains. $\hat{y}_s$ and $\hat{y}_t$ are the predicted labels. z' is the feature selected by G_c. Best viewed in color

ture is shown in Fig. 13. This paper introduces a new concept: openness, which is used to measure how much the target domain class is compared to the source domain class. It is defined as $\mathbb{O} = 1 - \frac{|\mathcal{C}_s|}{|\mathcal{C}_t|}$. There's no need to select the threshold hyperparameters throughout the process manually, so we don't need to adjust them when the openness changes manually. The accuracy of the above network is listed in Table 7.

Universal domain adaptation Universal Domain Adaptation (UDA) does not require a priori knowledge of the label set. In the UDA setting, given a labeled source domain, any related target domain regardless of how its label set differs from the source domain's label set, requires to be appropriately classified if it belongs to any of the categories in the source label set. Otherwise, it is labeled as "unknown". You et al. [101] proposed Universal Adaptation Network (UAN). It quantifies transferability at the sample level by sharing label sets and private label sets for each domain, thus facilitating the adaptation of automatically discovered public label sets and the successful identification of "unknown" samples. In the training phase, E_G, $E_{D'}$ and E_D represent the error for label classifier G, non-adversarial domain dis-

Table 7 Classification accuracy (%) of open set domain adaptation tasks on VisDA-2017 (VGGNet)

Method	bicycle	bus	car	motorcycle	train	truck	UNK	OS	OS*
OSVM	31.7	51.6	66.5	70.4	88.5	20.8	38.0	52.5	54.9
OSBP	51.1	67.1	42.8	84.2	81.8	28.0	85.1	62.9	59.2
STA	52.4	69.6	59.9	87.8	86.5	27.2	84.1	66.8	63.9

criminator D' and adversarial domain discriminator D, which are formally defined as (Fig. 14),

$$E_G = \mathbb{E}_{(\mathrm{x},\mathrm{y})\sim p} L(\mathrm{y}, G(F(\mathrm{x}))), \tag{35}$$

$$\begin{aligned} E_{D'} = &-\mathbb{E}_{\mathrm{x}\sim p} \log D'(F(\mathrm{x})) \\ &-\mathbb{E}_{\mathrm{x}\sim q} \log(1 - D'(F(\mathrm{x}))), \end{aligned} \tag{36}$$

$$\begin{aligned} E_D = &-\mathbb{E}_{\mathrm{x}\sim p} w^s(\mathrm{x}) \log D'(F(\mathrm{x})) \\ &-\mathbb{E}_{\mathrm{x}\sim q} w^t(\mathrm{x}) \log(1 - D'(F(\mathrm{x}))), \end{aligned} \tag{37}$$

where L is the standard cross-entropy loss, $w^s(\mathrm{x}) = \frac{H(\hat{\mathrm{y}})}{\log|\mathcal{C}_s|} - \hat{d}'(\mathrm{x})$ indicates the probability of a source sample x belonging to the common label set $\mathcal{C}$, $w^t(\mathrm{x}) = \hat{d}'(\mathrm{x}) - \frac{H(\hat{\mathrm{y}})}{\log|\mathcal{C}_s|}$ indicates the probability of a target sample x belonging to the common label set $\mathcal{C}$. With well-established weighting $w^s(\mathrm{x})$ and $w^t(\mathrm{x})$, the adversarial domain discriminator D is confined to distinguish the source and target data in the common label set $\mathcal{C}$. Non-adversarial domain discriminator D' is trained to get good weights $w^s(\mathrm{x})$ and $w^t(\mathrm{x})$, and Then conduct adversarial training on the adversarial domain discriminator D and label classifier G. After the training, the target sample class is judged by the value of weight $w^t(\mathrm{x})$.

Motivated by the domain similarity and uncertainty criteria proposed in [101], Saito et al. proposed Domain Adaptive Neighborhood Clustering via Entropy optimization (DANCE) in [72]. DANCE utilizes a classifier based on the prime center (prototype). This mapper maps the samples close to their true class prime centers (prototypes) and away from other classes. The target samples are first clustered in the target domain using Self-Supervision. Because of neighbor clustering, DANCE can extract different feature representations for "unknown" samples unsupervised. Next, align the target point with the source class prototype or reject it as "unknown" by entropy separation loss. Also, it utilizes domain-specific batch normalization [15, 17, 71] to eliminate domain style information as a form of weak domain alignment. It is worth noting that DANCE extracts discriminative feature representations for "unknown" class examples without any supervision on the target domain. Prediction entropy and output of the auxiliary domain classifier are not robust and discriminable enough. Fu et al. [27] proposed Calibrated Multiple Uncertainties (CMU) as the mixture of entropy, consistency, and confidence. And designed a deep ensemble model

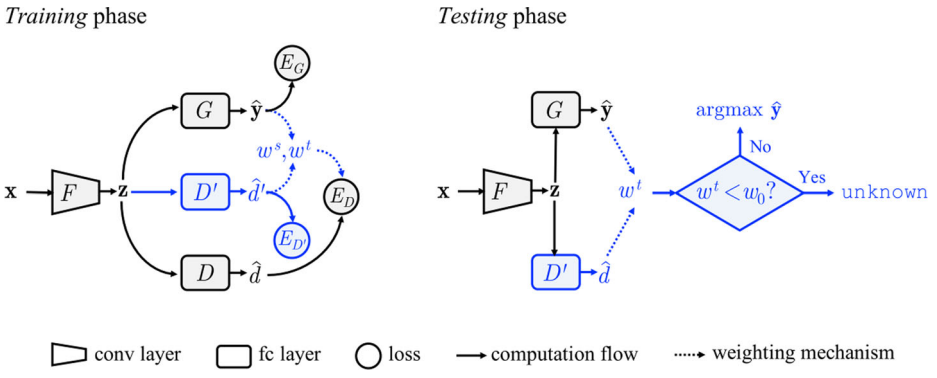

Fig. 14 The architecture of (Universal Adaptation Network) UAN is composed of a feature extractor F, an opposite domain discriminator D, a non opposite domain discriminator D' and a label classifier G. Best viewed in color

to characterizes different degrees of uncertainty and distinguishes target data in the common label set from those in the private label set. Fu et al. further proposed a novel H-score to compensate for the previous per-class accuracy for ignorance of open classes. H-score is the harmonic mean of the instance accuracy on common class $a_{\mathcal{C}}$ and accuracy on the"unknown" class $a_{\bar{\mathcal{C}}t}$ as $h = 2 \cdot \frac{a_{\mathcal{C}} \cdot a_{\bar{\mathcal{C}}t}}{a_{\mathcal{C}} + a_{\bar{\mathcal{C}}t}}$. Summary of the Universal comparisons is listed in Table 8.

Zero-shot domain adaptation In some extreme cases, we can not get a sample of the target domain. Zero-shot DA, which was gradually promoted from Few-shot DA, made its appearance. In the case of a zero sample domain adaptation setting, only the source domain data is available for the task of interest. Sometimes semantic information about the target domain classification is used, known as generalized Zero-shot learning (GZSL). DeViSE [26] is initialized from two pre-trained neural network models: a skip-gram text model and a visual model (AlexNet without its softmax prediction layer in this paper). A combination of dot-product similarity and hinge rank loss was used in the paper. The network is trained by reducing the distance between the image and the corresponding label and expanding the distance between the image and the non-corresponding labels. Based on the main idea of DeViSE, [60] adopts the framework of CNN and word2vec. The nearest category weight (probability) of the image is obtained by standard CNN, and the category is obtained by word2vec. Then the similarity is calculated with the test category to predict the label of the image. Zhang et al. [106] proposed a new embedding model of ZSL based on a deep neural network. There are two main differences between the model and the previous model, 1)It uses the visual feature space output from the CNN subnet as the embedding space. The projection direction is from semantic space to visual feature space, reducing the pivot point problem. 2)It realizes the end-to-end learning of semantic space representation. The architecture of the model is shown in Fig. 15.

The purpose of least squares embedding loss is to minimize the difference between visual features and their class representation embedding vectors in the visual feature space. With these three losses, our objective function is as follows,

$$\mathcal{L}(W_1, W_2) = \frac{1}{N} \sum_{i=1}^{N} \|\phi(I_i) - f_1(W_2 f_1(W_1 y_i^u))\|^2 + \lambda(\|W_1\|^2 + \|W_2\|^2), \tag{38}$$

where $W_1 \in \mathbb{R}^{L \times M}$ are the weights to be learned in the first FC layer and $W_2 \in \mathbb{R}^{M \times D}$ for the second FC layer. λ is a hyperparameter that weights the two parameters relative to

Table 8 Summary of the Universal comparisons

Method	Closed set DA			Partial DA			Open set DA			AVG
	Office31	OH	VD	OC	OH	VD	Office31	OH	VD	
DANN	85.9	62.7	69.1	42.2	40.9	38.7	88.7	72.8	48.2	61.0
UAN	84.4	58.8	66.4	52.9	34.2	39.7	91.0	74.6	50.0	61.3
DANCE	85.5	69.1	70.2	84.9	71.1	73.7	94.1	78.1	65.3	76.9

Each dataset (Office31, OC, OH, VisDA) has multiple domains and adaptation scenarios and we provide the average accuracy over all scenarios

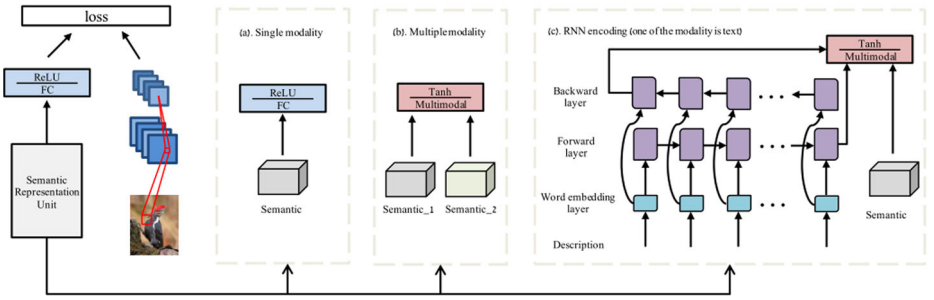

Fig. 15 The entire network consists of a visual coding branch and a semantic coding branch, where the visual coding branch takes the image as input and outputs feature vectors. The space in which the feature vectors are located will be considered as the embedding space. The semantic coding branch takes the one-dimensional semantic representation vector as input and outputs a three-dimensional semantic embedding vector after two fully connected linear unit layers. The two branches are connected by a least-squares embedding loss. Best viewed in color

the embedding loss after regularization. $f_1(\cdot)$ is the Rectified Linear Unit which introduces nonlinearity in the encoding subnet. The classification of the test image I_j in the visual feature space can be achieved by merely calculating its distance to the embedded prototypes. It is illustrated that visual feature space as an embedding space is much better than semantic space as an embedding space. Liu et al. [51] proposed a novel Deep Calibration Network (DCN) approach towards this generalized Zero-shot learning paradigm, which enables simultaneous calibration of deep networks on the confidence of source classes and uncertainty of target classes. Two scenarios are given in this paper, Zero-shot Learning(ZSL) and Generalized Zero-shot Learning (GZSL). The biggest difference between ZSL and GZSL is that GZSL can utilize the available semantic representation of the target domain, But GZSL needs to classify over both source and target classes. The network architecture is shown in Fig. 16.

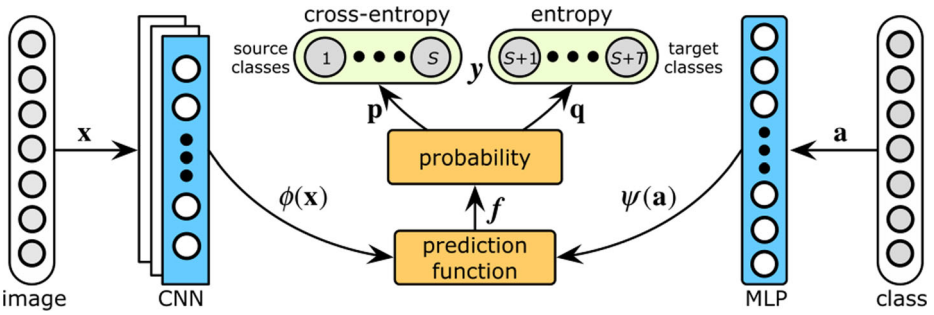

Fig. 16 Deep Calibration Network (DCN) consists of four modules. A CNN for earning deep embedding $\phi(\mathrm{x})$ for each image x and an MLP for learning deep embedding $\psi(\mathrm{a})$ for each class a.a prediction function f made by nearest prototype classifier (NPC).two probabilities p and q that transform the prediction function f into distributions over the source and target classes. A cross-entropy loss minimizes the overfitting to the source domain. The entropy loss minimizes the uncertainty of target classes. Best viewed in color

The optimization problem of the deep calibration network (DCN) for generalized Zero-shot learning can be formulated by integrating the empirical risk minimization and uncertainty calibration,

$$\min_{\phi,\psi} L + \lambda H + \gamma \Omega(\phi, \psi), \tag{39}$$

$$L = -\sum_{n=1}^{N}\sum_{c=1}^{S} y_{n,c} \log p_c(\mathrm{x}_n), \tag{40}$$

$$p_c(\mathrm{x}_n) = \frac{\exp(f_c(\mathrm{x}_n)/\tau)}{\sum_{c'=1}^{S} \exp(f_{c'}(\mathrm{x}_n)/\tau)}, \tag{41}$$

$$H = -\sum_{n=1}^{N}\sum_{c=S+1}^{S+T} q_c(\mathrm{x}_n) \log q_c(\mathrm{x}_n), \tag{42}$$

$$q_c(\mathrm{x}_n) = \frac{\exp(f_c(\mathrm{x}_n)/\tau)}{\sum_{c'=S+1}^{S+T} \exp(f_{c'}(\mathrm{x}_n)/\tau)}, \tag{43}$$

$$f_c(\mathrm{x}_n) = \mathrm{sim}(\phi(\mathrm{x}_n), \psi(\mathrm{a}_c)), \tag{44}$$

$$y(\mathrm{x}_n) = \arg\max_{c} fc(\mathrm{x}), \tag{45}$$

where $\Omega(\phi, \psi)$ is the penalty to control model complexity; λ and γ are hyper-parameters; In deep learning, weight decay can be used to replace the penalty term $\gamma\Omega(\phi, \psi)$; L is the empirical risk; H is uncertainty calibration; $\mathrm{sim}(\cdot)$ is a similarity function, *e.g.* inner product and cosine similarity. The ultimate goal is to minimize entropy to correctly classify, but this requires that the category of the target domain is known. [66] is the first domain adaptation and sensor fusion method that does not require relevant target domain data. [66] relies on the correspondences between source and target domain data samples in the irrelevant task to train the model. In contrast, Conditional Coupled Generative Adversarial Networks for Zero-shot DomainAdaptation (CoCoGAN) [91] does not rely on such information thanks to it captures the joint distribution of source and target domain data samples. Wang et al. [92] presented Adversarial Learning for Zero-shot Domain Adaptation (ALZSDA) to extend the scope of applications further. ALZSDA can learn the domain shift from an irrelevant task and transfer it to multiple different tasks of interest.

5 Multi-source domain adaptation

In practical scenarios, labeled data can be collected from multiple sources with different distributions. In this case, the above single-source domain adaptation(SSDA) approach can be applied simply by combining multiple source domains into a single source domain. However, merging multiple source domains and then using the SSDA approach usually results in more unsatisfactory performance than merely utilizing one of the source domains and discarding the others. Since domain transfer exists between each source domain and the target domain and between different source domains, merging source domain data from various sources may interfere with each other during the learning process. Therefore, to utilize all available data, multi-source domain adaptation (MSDA) is required (Fig. 17).

Xu et al. [98] proposed a deep cocktail network (DCTN) to cope with domain and category shifts among multiple sources. According to the theoretical results in [55], the target distribution can be represented as a weighted combination of the source distributions.

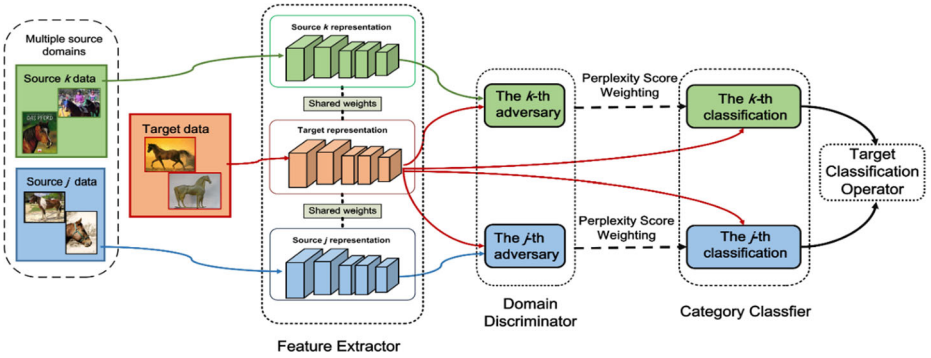

Fig. 17 Overview of the Deep Cocktail Network (DCTN). The framework receives multi-source instances of ground truth with annotations and adaptively classifies the target samples. For simplicity, it is illustrated with the source domains j and k. Firstly, the feature extractor maps the target domain, source domain j, and source domain k into a common feature space. Secondly, The category classifier receives the target feature and produces the j-th and k-th classifications based upon the categories in source domain j and k, respectively. Thirdly, The domain discriminator receives features from source j, k and target, and then provides an adversary between each source and target domain pair. Finally, The target classifier integrates all the weighted classification results and then predicts the target class. Best viewed in color

MSDA is executed in two iterative steps: first, the differences between the target source domain and multiple source domains are minimized through adversarial learning, and a confusion score is obtained for each source domain, which represents the likelihood that the target sample belongs to different source domains. In the second step, a multi-source category classifier is combined with confusion scores to classify the target samples and update the multi-source category classifier and feature extractor with pseudo label target and source samples (Fig. 18).

In contrast to [98] which symmetrically maps multiple sources and targets to the same space, proposes Multi-source Distillation Domain Adaptive (MDDA), which asymmetrically maps targets to individual source domains. Get more distinguished representations of the target by using respective feature extractors. Adversarial training using Wasserstein distance also produces more stable gradients.

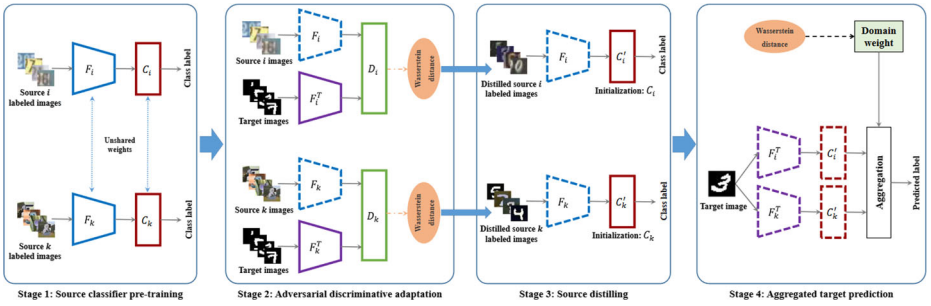

Fig. 18 Overview of MDDA. Firstly, pre-train the classifiers of each source domain. Then map the features of the extracted target domain to each source domain for adversarial training. Then based on the Wasserstein distance to select samples close to the target domain to fine-tune the classifier. Finally, the prediction of each target sample is weighted. Best viewed in color

The Stage 1 is the pre-training of the source domain classifier. The objective function in the Stage 2 is,

$$\max_{D_i} \mathcal{L}_{wd_D}(D_i) - \alpha \mathcal{L}_{grad}(D_i), \tag{46}$$

where α is a balancing coefficient, the value of which can be empirically set. $\mathcal{L}_{wd_D}(D_i)$ is the Wasserstein distance loss. To make sure the Lipschitz constraint is enforced, the gradient penalty is introduced for the parameters of each discriminator D_i as in [35]. Unlike the above methods, the model of Peng et al. [67] directly matches all the distributions by matching the moments. Moreover, they provide concrete proof of why matching the moments of multiple distributions works for MSDA. The Domain AggRegation Network (DARN) proposed by Wen et al. [95] dynamically adjusts the weights of each source domain during the training process. The weights are determined by the discrepancy between the source domain and the target domain. Unlike previous works, the aggregation scheme is direct optimizing our generalization upper bound without resorting to surrogates.

6 Conclusion

The Deep DA methods mainly refer to the domain adaptation algorithm based on deep network end-to-end training optimization. This survey paper focuses on this definition, and we mainly have reviewed deep DA techniques on visual categorization tasks.

We classify source and target domains based on their label set status. We do not use the supervised state as a basis for classification; we consider unsupervised and weakly supervised to be the way forward. Supervised domain adaption to a bridge to understanding adaptability better.

Firstly, we classify DA into single-source DA and multi-source DA. Further, according to whether the feature space is the same or not, the domain adaptation of single-source DA is divided into homogeneous domain adaptation and heterogeneous domain adaptation.

Furthermore, We introduce the label set as a classification indicator and classify domain adaptation into, Closed-set DA, Partial DA, Open set DA, Universal DA, and Zero-shot DA. There are three main approaches to solve the Closed-set DA problem. Discrepancy-Based methods, Adversarial-Based methods, and Reconstruction-Based methods. The better solution for deep DA is a comprehensive approach. For Partial DA and Open set DA, it is essentially a matter of blocking the negative transfer caused by "irrelevant samples" and extracting an invariant representation of the domain to promote positive transfer. For the Universal DA, self-supervised auxiliary domain adaptation is usually introduced. Intra-class distance is reduced by clustering. The inter-class distance is increased by entropy maximization. For Zero-shot DA, the primary research is still in the semantic representation of the class and the visual embedding of images.

Besides, we also study the multi-source DA. The current deep multi-source DA can be divided into two main categories. 1) using a shared network of feature extractors to symmetrically map multiple source and target domains into the same space. A discriminator is then trained for each source-target pair to distinguish between source domain features and target domain features. Based on the classifiers from different source domains, final predictions are made for the target image either on average or on weights. 2) Using a non-shared feature extractor to obtain the feature representation of each source domain, target domain features are asymmetrically matched to each source domain feature space. Pre-trained

classifiers are extracted with selected representative samples, and the classification is performed using a weighted approach.

Despite the recent success of deep DA, there are still many problems to be solved. First, the importance of each class is consistent in most datasets. However, in the actual application scenario, the importance may be inconsistent. How to reduce or eliminate the deviation caused by this inconsistency may become a future research topic. In addition, there are few studies on Universal DA and Zero-shot DA, and there will be more studies in the future.

In addition, deep DA have been successfully applied to many real-world applications, including image classification and object detection. The datasets for these tasks are 2D. For some task-specific 3D/4D data [78, 80], it is challenging to design DA networks to capture their 3D/4D features.

Finally, most of the existing deep DA methods are single modality. However, to take advantage of complementary but heterogeneous data, such as 2D images and 3D point clouds, Magnetic Resonance Imaging (MRI) and Computed Tomography (CT) images. It is meaningful to consider the heterogeneity between modalities and the difference between domains when designing the DA model. Recently, some papers [9, 10, 24, 54, 108] began to focus on this issue, and it is worth more research.

Acknowledgments This work is partly supported by the Natural Science Foundation of China (Grant No. 62006127, 61833011 and 62073173), partly supported by NUPTSF under Grant NY218120 and Grant NY220021, and partly supported by Jiangsu Shuang-Chuang Project under Grant CZ005SC19019 and Nanjing Overseas Innovation Project Grant RK005NLX20001. It is also supported by National Science Foundation of Jiangsu Province, China (Grant No. BK20191376 and BK20190728).

References

1. Ahmed A, Yousif H, He Z (2021) Ensemble diversified learning for image classification with noisy labels. Multimed Tools and Appl. https://doi.org/10.1007/s11042-021-10760-z
2. Alyafeai Z, Ghouti L (2020) A fully-automated deep learning pipeline for cervical cancer classification. Expert Syst Appl 141:112951. https://doi.org/10.1016/j.eswa.2019.112951
3. Aquino G, Rubio JDJ, Pacheco J, Gutierrez GJ, Ochoa G, Balcazar R, Cruz DR, Garcia E, Novoa JF, Zacarias A (2020) Novel nonlinear hypothesis for the delta parallel robot modeling. IEEE Access 8:46324–46334. https://doi.org/10.1109/ACCESS.2020.2979141
4. Ben-David S, Blitzer J, Crammer K, Kulesza A, Pereira F, Vaughan JW (2010) A theory of learning from different domains. Mach Learn 79(1):151–175
5. Bousmalis K, Silberman N, Dohan D, Erhan D, Krishnan D (2017) Unsupervised pixel-level domain adaptation with generative adversarial networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 3722–3731
6. Bousmalis K, Trigeorgis G, Silberman N, Krishnan D, Erhan D (2016) Domain separation networks. In: advances in neural information processing systems, pp 343–351
7. Bruzzone L, Marconcini M (2009) Domain adaptation problems: A DASVM classification technique and a circular validation strategy. IEEE Trans Pattern Anal Mach Intell 32(5):770–787
8. Cai Z, Han J, Liu L, Shao L (2017) RGB-D datasets using microsoft kinect or similar sensors: A survey. Multimed Tools Appl 76(3):4313–4355. https://doi.org/10.1007/s11042-016-3374-6
9. Cai Z, Jing X-Y, Shao L (2020) Visual-depth matching network: deep rgb-D domain adaptation with unequal categories. IEEE Trans Cybern:1–13. https://doi.org/10.1109/TCYB.2020.3032194
10. Cai Z, Long Y, Shao L (2018) Adaptive RGB image recognition by visual-depth embedding. IEEE Trans Image Process 27(5):2471–2483. https://doi.org/10.1109/TIP.2018.2806839
11. Cao Z, Long M, Wang J, Jordan MI (2018) Partial transfer learning with selective adversarial networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2724–2732
12. Cao Z, Ma L, Long M, Wang J (2018) Partial adversarial domain adaptation. In: Proceedings of the European conference on computer vision, pp 135–150

13. Cao Z, You K, Long M, Wang J, Yang Q (2019) Learning to transfer examples for partial domain adaptation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2985–2994
14. Chakraborty S, Mondal R, Singh PK, Sarkar R, Bhattacharjee D (2021) Transfer learning with fine tuning for human action recognition from still images. Multimed Tools Appl. https://doi.org/10.1007/s11042-021-10753-y
15. Chang W-G, You T, Seo S, Kwak S, Han B (2019) Domain-specific batch normalization for unsupervised domain adaptation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 7354–7362
16. Chen C, Fu Z, Chen Z, Jin S, Cheng Z, Jin X, Hua X-S (2020) HoMM: Higher-order moment matching for unsupervised domain adaptation. Order 1(10):20
17. Chen M, Zhao S, Liu H, Cai D (2020) Adversarial-learned loss for domain adaptation. In: AAAI, pp 3521–3528
18. Chiang H-S, Chen M-Y, Huang Y-J (2019) Wavelet-Based EEG Processing for Epilepsy Detection Using Fuzzy Entropy and Associative Petri Net. IEEE Access 7:103255–103262. https://doi.org/10.1109/ACCESS.2019.2929266
19. Chu C, Wang R (2018) A survey of domain adaptation for neural machine translation. arXiv:1806.00258
20. Chu W-S, De la Torre F, Cohn JF (2013) Selective transfer machine for personalized facial action unit detection. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 3515–3522
21. Csurka G (2017) Domain adaptation for visual applications: A comprehensive survey. arXiv:1702.05374
22. de Jesus Rubio J (2009) SOFMLS: online self-organizing fuzzy modified least-squares network. IEEE Trans Fuzzy Syst 17(6):1296–1309. https://doi.org/10.1109/TFUZZ.2009.2029569
23. de Rubio JJ (2020) Stability analysis of the modified levenberg-marquardt algorithm for the artificial neural network training. IEEE Trans Neural Netw Learn Syst:1–15. https://doi.org/10.1109/TNNLS.2020.3015200
24. Dou Q, Ouyang C, Chen C, Chen H, Heng P-A (2018) Unsupervised cross-modality domain adaptation of convnets for biomedical image segmentations with adversarial loss. arXiv:1804.10916
25. Fang X, Bai H, Guo Z, Shen B, Hoi S, Xu Z (2020) Dart: Domain-adversarial residual-transfer networks for unsupervised cross-domain image classification. Neural Netw
26. Frome A, Corrado GS, Shlens J, Bengio S, Dean J, Ranzato M, Mikolov T (2013) Devise: A deep visual-semantic embedding model. In: advances in neural information processing systems, pp 2121–2129
27. Fu B, Cao Z, Long M, Wang J (2020) Learning to detect open classes for universal domain adaptation. In: european conference on computer vision. Springer, pp 567–583
28. Ganin Y, Lempitsky V (2015) Unsupervised domain adaptation by backpropagation. In: international conference on machine learning. PMLR, pp 1180–1189
29. Gatys LA, Ecker AS, Bethge M (2016) Image style transfer using convolutional neural networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2414–2423
30. Gehring J, Miao Y, Metze F, Waibel A (2013) Extracting deep bottleneck features using stacked auto-encoders. In: 2013 IEEE International Conference on Acoustics, Speech and Signal Processing. IEEE, pp 3377–3381
31. Ghifary M, Kleijn WB, Zhang M, Balduzzi D, Li W (2016) Deep reconstruction-classification networks for unsupervised domain adaptation. In: european conference on computer vision. Springer, pp 597–613
32. Gong B, Grauman K, Sha F (2013) Connecting the dots with landmarks: Discriminatively learning domain-invariant features for unsupervised domain adaptation. In: international conference on machine learning, pp 222–230
33. Gong B, Shi Y, Sha F, Grauman K (2012) Geodesic flow kernel for unsupervised domain adaptation. In: 2012 IEEE conference on computer vision and pattern recognition. IEEE, pp 2066–2073
34. Gretton A, Borgwardt KM, Rasch MJ, Schölkopf B, Smola A (2012) A kernel two-sample test. J Mach Learn Res 13(1):723–773
35. Gulrajani I, Ahmed F, Arjovsky M, Dumoulin V, Courville AC (2017) Improved training of wasserstein gans. In: advances in neural information processing systems, pp 5767–5777
36. Hernández G, Zamora E, Sossa H, Téllez G, Furlán F (2020) Hybrid neural networks for big data classification. Neurocomputing 390:327–340. https://doi.org/10.1016/j.neucom.2019.08.095
37. Huang Z, Wang X, Huang L, Huang C, Wei Y, Liu W (2019) Ccnet: Criss-cross attention for semantic segmentation. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 603–612
38. Huh J-H, Seo Y-S (2019) Understanding edge computing: engineering evolution with artificial intelligence. IEEE Access 7:164229–164245. https://doi.org/10.1109/ACCESS.2019.2945338

39. Iwendi C, Srivastava G, Khan S, Maddikunta PKR (2020) Cyberbullying detection solutions based on deep learning architectures. Multimed Syst. https://doi.org/10.1007/s00530-020-00701-5
40. Kim Y (2014) Convolutional neural networks for sentence classification. arXiv:1408.5882
41. Kong T, Sun F, Liu H, Jiang Y, Li L, Shi J (2020) Foveabox: Beyound anchor-based object detection. IEEE Trans Image Process 29:7389–7398
42. Kouw WM, Loog M (2018) An introduction to domain adaptation and transfer learning. arXiv:1812.11806
43. Krizhevsky A, Sutskever I, Hinton GE (2017) Imagenet classification with deep convolutional neural networks. Commun ACM 60(6):84–90
44. Kutia S, Chauhdary SH, Iwendi C, Liu L, Yong W, Bashir AK (2019) Socio-technological factors affecting user's adoption of eHealth functionalities: a case study of China and Ukraine eHealth systems. IEEE Access 7:90777–90788. https://doi.org/10.1109/ACCESS.2019.2924584
45. Larsen ABL, Sønderby SK, Larochelle H, Winther O (2016) Autoencoding beyond pixels using a learned similarity metric. In: international conference on machine learning. PMLR, pp 1558–1566
46. LeCun Y et al (2015) LeNet-5, convolutional neural networks. http://yann.lecun.com/exdb/lenet 20(5):14
47. Lee H, Park S-H, Yoo J-H, Jung S-H, Huh J-H (2020) Face recognition at a distance for a stand-alone access control system. Sensors 20(3):785. https://doi.org/10.3390/s20030785
48. Li S, Liu CH, Lin Q, Xie B, Ding Z, Huang G, Tang J (2020) Domain conditioned adaptation network. In: AAAI, pp 11386–11393
49. Li W, Zhu X, Gong S (2018) Harmonious attention network for person re-identification. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2285–2294
50. Liu H, Cao Z, Long M, Wang J, Yang Q (2019) Separate to adapt: Open set domain adaptation via progressive separation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2927–2936
51. Liu S, Long M, Wang J, Jordan MI (2018) Generalized zero-shot learning with deep calibration network. In: advances in neural information processing systems, pp 2005–2015
52. Long M, Cao Y, Wang J, Jordan M (2015) Learning transferable features with deep adaptation networks. In: international conference on machine learning. PMLR, pp 97–105
53. Long M, Zhu H, Wang J, Jordan MI (2017) Deep transfer learning with joint adaptation networks. In: international conference on machine learning. PMLR, pp 2208–2217
54. Ma X, Zhang T, Xu C (2019) Deep multi-modality adversarial networks for unsupervised domain adaptation. IEEE Trans Multimed 21(9):2419–2431
55. Mansour Y, Mohri M, Rostamizadeh A (2009) Domain adaptation with multiple sources. In: advances in neural information processing systems, pp 1041–1048
56. Meda-Campana JA (2018) On the estimation and control of nonlinear systems with parametric uncertainties and noisy outputs. IEEE Access 6:31968–31973. https://doi.org/10.1109/ACCESS.2018.2846483
57. Mohamed A-r, Dahl GE, Hinton G (2011) Acoustic modeling using deep belief networks. IEEE Trans Audio Speech Lang Process 20(1):14–22
58. Mohamed A, Hinton G, Penn G (2012) Understanding how deep belief networks perform acoustic modelling. In: 2012 IEEE international conference on acoustics, speech and signal processing. IEEE, pp 4273–4276
59. Narang SR, Kumar M, Jindal MK (2021) DeepNetDevanagari: A deep learning model for Devanagari ancient character recognition. Multimed Tools Appl. https://doi.org/10.1007/s11042-021-10775-6
60. Norouzi M, Mikolov T, Bengio S, Singer Y, Shlens J, Frome A, Corrado GS, Dean J (2013) Zero-shot learning by convex combination of semantic embeddings. arXiv:1312.5650
61. Pan SJ, Tsang IW, Kwok JT, Yang Q (2010) Domain adaptation via transfer component analysis. IEEE Trans Neural Netw 22(2):199–210
62. Pan SJ, Yang Q (2009) A survey on transfer learning. IEEE Trans Knowl Data Eng 22(10):1345–1359
63. Panareda Busto P, Gall J (2017) Open set domain adaptation. In: Proceedings of the IEEE international conference on computer vision, pp 754–763
64. Patel VM, Gopalan R, Li R, Chellappa R (2015) Visual domain adaptation: A survey of recent advances. IEEE Signal Process Mag 32(3):53–69
65. Pei Z, Cao Z, Long M, Wang J (2018) Multi-adversarial domain adaptation. arXiv:1809.02176
66. Peng K-C, Wu Z, Ernst J (2018) Zero-shot deep domain adaptation. In: Proceedings of the European conference on computer vision, pp 764–781
67. Peng X, Bai Q, Xia X, Huang Z, Saenko K, Wang B (2019) Moment matching for multi-source domain adaptation. In: Proceedings of the IEEE international conference on computer vision, pp 1406–1415
68. Peng X, Usman B, Kaushik N, Hoffman J, Wang D, Saenko K (2017) Visda: The visual domain adaptation challenge. arXiv:1710.06924

69. Rakshit RD, Kisku DR, Gupta P, Sing JK (2021) Cross-resolution face identification using deep-convolutional neural network. Multimed Tools Appl. https://doi.org/10.1007/s11042-021-10745-y
70. Saenko K, Kulis B, Fritz M, Darrell T (2010) Adapting visual category models to new domains. In: European conference on computer vision. Springer, pp 213–226
71. Saito K, Kim D, Sclaroff S, Darrell T, Saenko K (2019) Semi-supervised domain adaptation via minimax entropy. In: Proceedings of the IEEE international conference on computer vision, pp 8050–8058
72. Saito K, Kim D, Sclaroff S, Saenko K (2020) Universal domain adaptation through self supervision. arXiv:2002.07953
73. Saito K, Ushiku Y, Harada T, Saenko K (2017) Adversarial dropout regularization. arXiv:1711.01575
74. Saito K, Yamamoto S, Ushiku Y, Harada T (2018) Open set domain adaptation by backpropagation. In: Proceedings of the European conference on computer vision, pp 153–168
75. Sankaranarayanan S, Balaji Y, Castillo CD, Chellappa R (2018) Generate to adapt: Aligning domains using generative adversarial networks. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 8503–8512
76. Seo Y-S, Huh J-H (2019) Automatic emotion-based music classification for supporting intelligent IoT applications. Electronics 8(2):164. https://doi.org/10.3390/electronics8020164
77. Shen J, Qu Y, Zhang W, Yu Y (2017) Wasserstein distance guided representation learning for domain adaptation. arXiv:1707.01217
78. Shi H, Lin G, Wang H, Hung T-Y, Wang Z (2020) Spsequencenet: Semantic segmentation network on 4d point clouds. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 4574–4583
79. Simonyan K, Zisserman A (2014) Very deep convolutional networks for large-scale image recognition. arXiv:1409.1556
80. Sothmann T, Gauer T, Werner R (2018) Influence of 4d ct motion artifacts on correspondence model-based 4d dose accumulation. In: medical imaging 2018: image-guided procedures, robotic interventions, and modeling, vol 10576. International Society for Optics and Photonics, p 105760F
81. Sun B, Saenko K (2016) Deep coral: Correlation alignment for deep domain adaptation. In: European conference on computer vision. Springer, pp 443–450
82. Sun S, Shi H, Wu Y (2015) A survey of multi-source domain adaptation. Inf Fusion 24:84–92
83. Syed AM, Anjum A, Khan S, Mohan S, Srivastava G (2020) N-Sanitization: A semantic privacy-preserving framework for unstructured medical datasets. Comput Commun 161:160–171. https://doi.org/10.1016/j.comcom.2020.07.032
84. Tan M, Pang R, Le QV (2020) Efficientdet: Scalable and efficient object detection. In: Proceedings of the IEEE/CVF conference on computer vision and pattern recognition, pp 10781–10790
85. Tang H, Jia K (April 2020) Discriminative adversarial domain adaptation. Proc AAAI Conf Artif Intell 34(04):5940–5947. https://doi.org/10.1609/aaai.v34i04.6054
86. Tzeng E, Hoffman J, Saenko K, Darrell T (2017) Adversarial discriminative domain adaptation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 7167–7176
87. Tzeng E, Hoffman J, Zhang N, Saenko K, Darrell T (2014) Deep domain confusion: Maximizing for domain invariance. arXiv:1412.3474
88. Van Engelen JE, Hoos HH (2020) A survey on semi-supervised learning. Mach Learn 109(2):373–440
89. Venkateswara H, Eusebio J, Chakraborty S, Panchanathan S (2017) Deep hashing network for unsupervised domain adaptation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 5018–5027
90. Vincent P, Larochelle H, Lajoie I, Bengio Y, Manzagol P-A, Bottou L (2010) Stacked denoising autoencoders: Learning useful representations in a deep network with a local denoising criterion. J Mach Learn Res 11(12)
91. Wang J, Jiang J (2019) Conditional coupled generative adversarial networks for zero-shot domain adaptation. In: Proceedings of the IEEE international conference on computer vision, pp 3375–3384
92. Wang J, Jiang J (2020) Adversarial learning for zero-shot domain adaptation. In: European conference on computer vision. Springer, pp 329–344
93. Wang M, Deng W (2018) Deep visual domain adaptation: A survey. Neurocomputing 312:135–153
94. Wang W, Zheng VW, Yu H, Miao C (2019) A survey of zero-shot learning: Settings, methods, and applications. ACM Trans Intell Syst Technol 10(2):1–37
95. Wen J, Greiner R, Schuurmans D (2020) Domain aggregation networks for multi-source domain adaptation. In: international conference on machine learning. PMLR, pp 10214–10224
96. Wilson G, Cook DJ (2020) A survey of unsupervised deep domain adaptation. ACM Trans Intell Syst Technol 11(5):1–46. https://doi.org/10.1145/3400066
97. Xu M, Zhang J, Ni B, Li T, Wang C, Tian Q, Zhang W (2020) Adversarial domain adaptation with domain mixup. In: Proceedings of the AAAI conference on artificial intelligence, vol 34, pp 6502–6509

98. Xu R, Chen Z, Zuo W, Yan J, Lin L (2018) Deep cocktail network: Multi-source unsupervised domain adaptation with category shift. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 3964–3973
99. Yang J, Yan R, Hauptmann AG (2007) Cross-domain video concept detection using adaptive svms. In: Proceedings of the 15th ACM international conference on multimedia, pp 188–197
100. Yosinski J, Clune J, Bengio Y, Lipson H (2014) How transferable are features in deep neural networks? In: advances in neural information processing systems, pp 3320–3328
101. You K, Long M, Cao Z, Wang J, Jordan MI (2019) Universal domain adaptation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2720–2729
102. Zellinger W, Grubinger T, Lughofer E, Natschläger T, Saminger-Platz S (2017) Central moment discrepancy (cmd) for domain-invariant representation learning. arXiv:1702.08811
103. Zhai X, Oliver A, Kolesnikov A, Beyer L (2019) S4l: Self-supervised semi-supervised learning. In: Proceedings of the IEEE/CVF international conference on computer vision, pp 1476–1485
104. Zhang G, Jiang T, Yang J, Xu J, Zheng Y (2021) Cross-view kernel collaborative representation classification for person re-identification. Multimed Tools Appl. https://doi.org/10.1007/s11042-021-10671-z
105. Zhang J, Ding Z, Li W, Ogunbona P (2018) Importance weighted adversarial nets for partial domain adaptation. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 8156–8164
106. Zhang L, Xiang T, Gong S (2017) Learning a deep embedding model for zero-shot learning. In: Proceedings of the IEEE conference on computer vision and pattern recognition, pp 2021–2030
107. Zhang M, Hu H, Li Z, Chen J (2021) Attention-based encoder-decoder networks for workflow recognition. Multimed Tools Appl. https://doi.org/10.1007/s11042-021-10633-5
108. Zhang W, Xu D, Zhang J, Ouyang W (2021) Progressive modality cooperation for multi-modality domain adaptation. IEEE Trans Image Process 30:3293–3306
109. Zhuang F, Qi Z, Duan K, Xi D, Zhu Y, Zhu H, Xiong H, He Q (2020) A comprehensive survey on transfer learning. Proc IEEE 109(1):43–76