Independent Feature Decomposition and Instance Alignment for Unsupervised Domain Adaptation

Qichen He[1], **Siying Xiao**[1], **Mao Ye**[1*], **Xiatian Zhu**[2], **Ferrante Neri**[3] and **Dongde Hou**[4]
[1]School of CSE, University of Electronic Science and Technology of China, Chengdu, China
[2]Surrey Institute for People-Centred Artificial Intelligence, CVSSP, University of Surrey, Guildford, UK
[3]NICE Research Group, Department of Computer Science, University of Surrey, Guildford, UK
[4]Advanced Research Institute, Southwest University of Political Science&Law, Chongqing, China
{202121081209, 2018270101016}@std.uestc.edu.cn, maoye@uestc.edu.cn, {xiatian.zhu, f.neri}@surrey.ac.uk, gxin_001@163.com

Abstract

Existing Unsupervised Domain Adaptation (UDA) methods typically attempt to perform knowledge transfer in a *domain-invariant space* explicitly or implicitly. In practice, however, the obtained features are often mixed with domain-specific information which causes performance degradation. To overcome this fundamental limitation, this article presents a novel ***independent feature decomposition and instance alignment method*** (**IndUDA** in short). Specifically, based on an invertible flow, we project the base feature into a decomposed latent space with domain-invariant and domain-specific dimensions. To drive semantic decomposition independently, we then swap the domain-invariant part across source and target domain samples with the same category and require their inverted features are consistent in class-level with the original features. By treating domain-specific information as noise, we replace it by Gaussian noise and further regularize source model training by instance alignment, i.e., requiring the base features close to the corresponding reconstructed features, respectively. Extensive experiment results demonstrate that our method achieves state-of-the-art performance on popular UDA benchmarks. The appendix and code are available at https://github.com/ayombeach/IndUDA.

1 Introduction

Recently, deep learning has achieved great success in a variety of applications, particularly in computer vision and natural language processing. Despite such a great achievement, most deep learning methods require a large amount of manually labelled data. However, in many real-world applications, such an amount of data with labels is usually hard to collect [Pan and Yang, 2009]. In the majority of cases, we can only have access to a model pre-trained on a labelled source domain and fit it into an unlabelled target domain [Long *et al.*, 2016; Pan *et al.*, 2010]. Due to domain shift, the source domain model often suffers from performance degradation on target data. To solve this problem, the research of Unsupervised Domain Adaptation (UDA) comes into play [Sugiyama *et al.*, 2007]. Usually, UDA methods project the source and target samples into a common feature space to enable knowledge transfer cross domains. This feature space is often called a *domain-invariant* space. Existing methods typically derive this domain-invariant space by three categories of approaches. The first category is based on *traditional distribution alignment* [Long *et al.*, 2015; Long *et al.*, 2016; Ganin *et al.*, 2016], which minimizes domain discrepancy by aligning the distributions between two domains. Domain-invariant features are extracted without considering the impact of domain-specific information (playing a role as noise as they are largely class irrelatively). The second approach is *self-supervised learning* which implicitly learns a common space based on pseudo labels or prototypes [Saenko *et al.*, 2010; French *et al.*, 2018]. Similar to the first category of methods, the impact of domain-specific noise is not

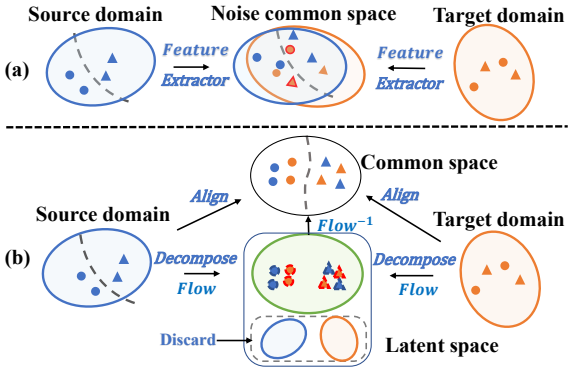

Figure 1: Comparison between previous methods and our IndUDA. (a) Domain-invariant features extracted by previous methods are typically mixed with domain-specific noise. To mitigate this limitation, (b) IndUDA first decomposes a feature into domain-invariant and domain-specific parts in a latent space, followed by replacing the domain-specific part with Gaussian noise for instance alignment between the original feature and inverted noised features.

*Mao Ye is the corresponding author. This work was supported by National Natural Science Foundation of China (62276048).

considered. Hence, the ideal domain-invariant space cannot be achieved. The last category *disentangles* the feature into domain-invariant feature and domain-specific feature [Cai *et al.*, 2019; Lee *et al.*, 2021; Zhou *et al.*, 2022a]. Although good performance can be achieved, the independence relationship between the disentangled domain-invariant feature and domain-specific feature is largely overlooked.

In summary, previous approaches presented in the literature can only obtain a noisy domain-invariant space, as illustrated in Fig.1(a). To address this limitation, we propose to decompose a sample feature into two independent parts in channel: a domain-invariant part and a domain-specific part. It is not so easy since the extracted features are dependent in channels. To that end, we exploit normalizing flow [Lugmayr *et al.*, 2020] that can transfer a distribution to another distribution while holds an important invertible property. As shown in Fig.1(b), our key idea is that, through swapping domain-invariant parts between a source sample and a target sample with the same category in the latent space whilst requiring the inverted features after swapping are consistent with the original feature in class-level, the independence between domain-invariant and domain-specific information can be achieved to high degree. Then, if we discard the domain-specific part, and require the features from the source model close to the corresponding inverted features, the model is endowed with the ability of extracting domain invariant features.

With the above analysis, we propose a novel method, dubbed as *Independent feature decomposition and instance alignment UDA* (IndUDA). Specifically, IndUDA consists of an independent feature decomposition module and an instance feature alignment module. For the first module, an Invertible Neural Network (INN) [Lugmayr *et al.*, 2020] is exploited to project the original feature into a compact latent space. A channel mask is further used to split the projected feature into domain-invariant part and domain-specific part. After that, for the target samples with high-confidence pseudo-labels, the instance cross-domain swap strategy exchanges the domain-invariant parts with respect to the source and target samples, then requires consistency in class-level between the original feature and inverted feature through a backward process of INN flow. For the instance feature alignment module, the domain-specific part is dropped in latent space and replaced by Gaussian noise. An ideal domain-invariant feature is obtained through the reverse process of INN flow again. Then, the source and target features are aligned with the ideal features respectively. In this way, domain-invariant parts in the original space can be obtained.

Our contribution can be summarized as follows: (1) We propose a novel unsupervised domain adaptation framework, namely IndUDA. Different from previous approaches, the domain-specific noise can be better eliminated by requiring independence between domain-invariant parts and domain-specific part. So the performance is boosted. (2) A new instance-level cross-domain feature swap strategy is proposed to ensure independence, along with exploiting the invertible flow. By swapping the domain-invariant parts between the source and target sample features and requiring the inverted features consistent, the independence degree between domain-invariant part to domain-specific part can be maximized. (3) Extensive experiments demonstrate the efficacy of our method on three benchmarks. Ablation analysis explains the performance superiority of our model in the component level. Furthermore, the adaptation networks are only used in the training phase, while during the test process, only the adapted model is needed.

2 Related Work

2.1 Unsupervised Domain Adaptation

Unsupervised domain adaptation (UDA) is a technique that aims to improve the generalization ability of a model on an unlabelled target domain [Pan *et al.*, 2010; Patel *et al.*, 2015; Tang *et al.*, 2020]. From the point of finding domain-invariant features, the existing solutions can be roughly divided into three categories. The first approach is based on *self-supervised learning*, such as Self-Ensembling (SE) [French *et al.*, 2018] and Transferrable Prototypical Networks (TPN) [Pan *et al.*, 2019], which attempt to apply pseudo-labels or other supervision information like a prototype to retrain the source model. Such a self-training solution is also a way to find a common representation space of two domains [Saenko *et al.*, 2010].

The second approach regards domain adaptation as a *distribution alignment* problem. There are two routes to address this problem: statistical moment matching and adversarial learning. The statistical moment matching method aims to minimize the statistical discrepancy to align the distributions between two domains, such as Maximum Mean Discrepancy (MMD) [Long *et al.*, 2015], DAN [Long *et al.*, 2015], and CAN [Kang *et al.*, 2019]. Adversarial learning-based methods use a min-max training approach between feature extractor and discriminator to find domain-invariant features, such as MCD [Saito *et al.*, 2018], DANN [Ganin and Lempitsky, 2015] and ADDA [Tzeng *et al.*, 2017].

The final approach aims to *disentangle* features into domain-invariant and domain-specific features [Cai *et al.*, 2019; Lee *et al.*, 2021; Liu *et al.*, 2018; Deng *et al.*, 2022]. There are two ways. The first is based on a classifier prototype. For example, MSDTR [Zhou *et al.*, 2022a] uses classifier prototypes and domain prototypes to split features. Another one trains an encoder-decoder network to decompose and reconstruct features. For example, DRANet [Lee *et al.*, 2021] adds a content adaptive domain transfer module in an encoder-decoder network that helps to retain scene structure. Despite the great performance achieved, all of these methods do not systematically consider independence between domain-invariant and domain-specific features. So domain-specific noise is not eliminated very well.

2.2 Normalizing Flow

Normalized flow was originally designed for unsupervised learning of probabilistic models [Dinh *et al.*, 2016]. The flow model regards the origin dataset as a complex distribution with an invertible neural network, which can map data sampled from a simple distribution (e.g. Gaussian). Previous methods explore the exact negative log-likelihood based on conditional distribution [Kingma and Dhariwal, 2018] which

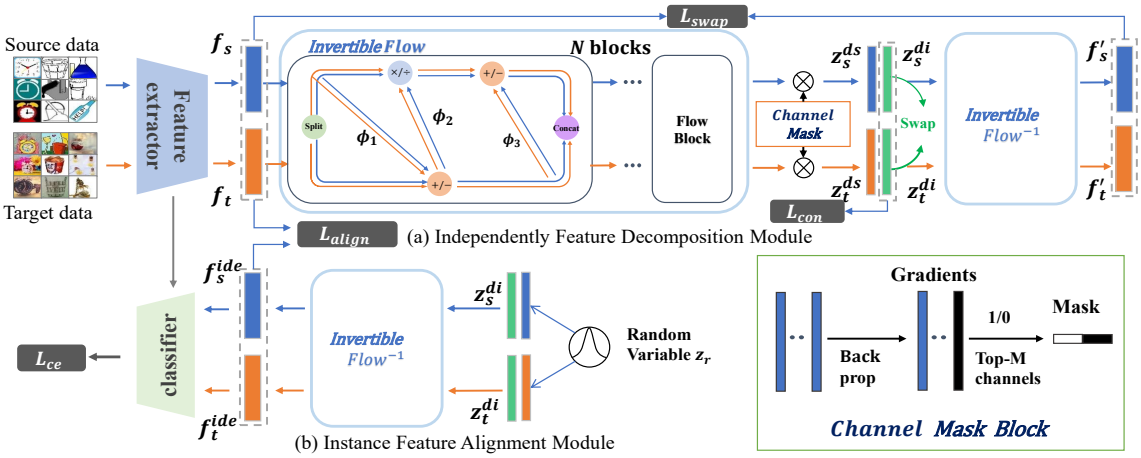

Figure 2: The proposed framework. (a) Independent feature decomposition maps the features to a latent space, then splits them and obtains compact and independent domain-invariant features. (b) Instance feature alignment reconstructs ideal domain-invariant features through inverse feature mapping and aligns the extracted features with the corresponding ideal features.

forces the simple distribution to approximate the prior distribution. A series of works exploit the flow model on many low-level visual tasks such as InvDN [Liu *et al.*, 2021b] for image denoising tasks and SRFlow [Lugmayr *et al.*, 2020] for super-resolution. Different from previous works, we employ normalizing flow to implement the independence between the domain-invariant part and domain-specific part by sufficiently using the invertible property.

3 Methodology

3.1 Problem Statement

In the UDA setting, we have a set of labelled source domain samples $\mathcal{S} = \left\{\left(\boldsymbol{x}_1^s, y_1^s\right), \cdots, \left(\boldsymbol{x}_{N_s}^s, y_{N_s}^s\right)\right\}$, and an unlabelled target domain sample set $\mathcal{T} = \left\{\boldsymbol{x}_1^t, \cdots, \boldsymbol{x}_{N_t}^t\right\}$, where $\boldsymbol{x}^s, \boldsymbol{x}^t$ represent the source and target input sample, and $y^s \in R^M$ denotes the source ground truth label with M classes. N_s and N_t are the numbers of source samples and target samples respectively. The source domain and target domain shares the same label space $\{0, 1, \cdots, M-1\}$ but with different distributions. The goal of UDA is to mitigate the distribution discrepancy and make accurate predictions $\{\hat{y}^t\}$ on the target domain $\mathcal{T}$.

Overview. Our method IndUDA consists of two parts, which are illustrated in Fig.2. One is an independent feature decomposition module, while the other is an instance feature alignment module. For the Independent Feature Decomposition (IFD) module as shown in Fig.2(a), the source feature f_s and target feature f_t which has high-confidence pseudo-label, are first mapped to a latent space z by an invertible neural network G_{flow} as $z_s = G_{flow}(f_s)$, $z_t = G_{flow}(f_t)$. Then, z_s and z_t are split into domain-invariant parts z_s^{di}, z_t^{di} and domain-specific parts z_s^{ds}, z_t^{ds} respectively by a channel mask block. To guarantee the independence and seek semantic domain-invariant information, an instance cross-domain swap strategy is employed. If the extracted domain-invariant parts are independent of the corresponding domain-specific parts, then after swapping the domain-invariant parts, the inverted features should maintain the same class memberships as the original features. In the end, a constraint loss L_{con} is used to align the domain-invariant parts between source and target domains which further reduces the correlations between the domain-invariant part and domain-specific part.

For the instance feature alignment module as shown in Fig.2(b), domain-specific parts can be considered as noise because they are not related to the classification task. Our idea is to reconstruct the ideal feature which contain no domain-specific noise to guide the training of the feature extractor. So we replace z_s^{ds}, z_t^{ds} with a Gaussian noise r. Here, the reason we use a Gaussian noise r instead of zero is that it can help to improve the generalization ability. Then the extracted features f_s, f_t are required to be aligned to the corresponding ideal features respectively by the loss L_{align}, i.e., a better domain-invariant space is obtained.

3.2 Independent Feature Decomposition

To achieve independent feature decomposition, this module consists of three parts: invertible feature mapping, channel mask block and instance cross-domain swap strategy.

Invertible feature mapping. After feature extractor F, domain-invariant features and domain-specific features are highly correlated in feature channel dimension. So it is better to map the extracted feature f into a latent space such that the domain-invariant and domain-specific parts can be separated easily in channel dimension. Since Invertible Neural Network (INN) [Lugmayr *et al.*, 2020] can link two distributions because of its invertible property, we adapt an INN as the feature mapping to the latent space so that we can back to the origin feature space via its reverse process after obtaining domain-invariant channels in latent space. We define this feature mapping as a function G_{flow}, and the inverse process as G_{flow}^{-1}.

Channel mask block. To split the mapped features into domain-invariant part and domain-specific part in a reason-

able way, without any doubts, the domain-invariant part is relevant to classification task [Huang *et al.*, 2020] while domain-specific part has nothing to do with the classification performance. For simplicity, we decompose features according to feature channel dimension. The gradient of classifier respect to the i-th average pooled channel feature is used as an evaluation measure, which is defined as

$$g_x^i = \frac{\partial y_x}{\partial f_x^i}, \tag{1}$$

where y_x denotes the classifier output of sample x and f_x^i is the i-th average pooled channel feature. Since the extracted feature $f_x \in \mathbb{R}^{D\times H\times W}$, where D denotes the channel number, after average pooling for each channel, the gradient $g_x \in \mathbb{R}^{D\times 1\times 1}$. The gradient distributions on channels for different samples of the same category are different. So the union of all samples is considered and domain-invariant channels are chosen for each category.

A memory bank $M_{bank} \in \mathbb{R}^{C\times D}$ is established as a channel mask, where C is the category number, and all components in M_{bank} are initialized with 1. For a sample x belongs to the category c, the mask for the category c and the i-th channel is defined as follows,

$$mask_x^{c,i} = \begin{cases} 1, & g_x^i > \tau, \\ 0, & else, \end{cases} \tag{2}$$

where τ is a threshold. Then $mask_x^{c,i}$ is used to update M_{bank}. The mask of class c in M_{bank} is updated as

$$M_{bank}[c,i] \leftarrow M_{bank}[c,i] \odot mask_x^{c,i}, \tag{3}$$

where $\odot$ represents the element-wise multiplication. The element of M_{bank} is valued at 1 only in the case that the mask of every source sample in this class is 1, otherwise is 0 for other cases. The value of threshold τ will affect feature decomposition. If τ is too small, most of the channels are reserved and domain-specific features cannot be de-noised well; while if τ is too large, most of the domain-invariant channels will be dismissed. The value of τ is decided such that just D/2 channels are selected as domain invariant part. After passing all source samples, we can obtain a memory bank M_{bank} for each category. For selecting semantic domain-invariant channels, we only need to query M_{bank} without calculating the gradient again.

Remark. There is another way to split the feature, i.e., directly bisecting the feature and forcing one half as domain-invariant channels and the other half becomes domain-specific channels. But using this method is not easy to obtain a stable model because domain invariant channels are different for each category. It should be noted that the mask bank is only calculated based on the source samples because they have accurate labels. Experiments in Appendix also show that the masks are the same with the confident pseudo-labelled target samples.

Instance cross-domain swap strategy. First, the target samples are pseudo-labelled by a spherical K-means method as used in CAN [Kang *et al.*, 2019]. Suppose the c-th cluster center in the target domain is represented as O^c and only a subset is labelled as confidence samples $\tilde{\mathcal{T}} = \{(x^t, \hat{y}^t) | dist(f_t, O^{\hat{y}^t}) < D_0\}$, where $\hat{y}^t$ is assigned by the nearest cluster center and D_0 is a constant which is set to 0.05. The distance is measured as cosine dissimilarity. Besides, if the number of samples satisfying the condition in one category is less than three, this category will not be sampled in the current epoch.

After that, by invertible feature mapping, the extracted source and confidence target features f_s and f_t are mapped as z_s and z_t. Then, the features z_s and z_t are further split into $[z_s^{di}, z_s^{ds}]$ and $[z_t^{di}, z_t^{ds}]$ by M_{bank}, where z_s^{di}, z_t^{di} composed of channels with mask value 1. Since we assume that the domain-invariant features should follow the same distribution if we exchange z_s^{di} and z_t^{di} to get two features z_s' and z_t', their distributions should also be the same. This process can be written as follows:

$$z_s' = z_t^{di} \oplus z_s^{ds}, \quad z_t' = z_s^{di} \oplus z_t^{ds}, \tag{4}$$

where $\oplus$ denotes the concatenate operation. To our analysis, the inverted features f_s', f_t' of z_s', z_t' should have the same distributions as f_s and f_t respectively. So for training invertible feature mapping, we propose the following swap loss,

$$\min_{G_{flow}} \mathcal{L}_{swap} = \mathcal{D}_H\left(f_s, f_s'\right) + \mathcal{D}_H\left(f_t, f_t'\right), \tag{5}$$

where D_H is a function that measures the distribution difference. Here, contrastive domain discrepancy (CDD) loss [Kang *et al.*, 2019] is used.

To further guarantee that domain-invariant parts from the source and target samples obey the same distribution, we add a constraint term $\mathcal{L}_{con}$ as

$$\min_{G_{flow}} \mathcal{L}_{con} = \mathcal{D}_H(z_s^{di}, z_t^{di}). \tag{6}$$

We use Eq.(6) to constrain z_s^{di} and z_t^{di} become more compact in latent space z, because the CDD loss can make the intra-class distance smaller and the inter-class distance larger.

Remark. By invertible property, swap strategy reduces the correlation between domain-invariant part and domain-specific part, thus the independence of domain-invariant part is achieved in some sense. Furthermore, the distance is measured by CDD loss in the class level instead of L_2 loss. There are two reasons. First, the domain-invariant part for each category cannot be exactly the same, otherwise, the diversity of samples is lost. Second, CDD loss can obtain a more compact latent space.

3.3 Instance Feature Alignment

In the above section, domain-invariant part is obtained in latent space. If we project back this part to the original feature space and ask for the extracted features close to the corresponding reconstructed features, then the feature extractor can extract domain-invariant features in the original feature space. However, it is not suitable to directly drop the domain-specific channels and set them to zero. Because normalizing flow is a manifold mapping, i.e., distribution-to-distribution. If the domain-specific part is simply set as zero, the correspondence becomes one-on-one which reduces the generalization ability.

Method	Venue	Ar→Cl	Ar→Pr	Ar→Rw	Cl→Ar	Cl→Pr	Cl→Rw	Pr→Ar	Pr→Cl	Pr→Rw	Rw→Ar	Rw→Cl	Rw→Pr	*Avg.*
ResNet-50	CVPR16	34.9	50.0	58.0	37.4	41.9	46.2	38.5	31.2	60.4	53.9	41.2	59.9	46.1
CAN	CVPR19	58.7	78.1	82.1	67.4	75.7	78.1	67.2	54.2	82.5	73.4	60.9	83.5	71.8
ALDA	AAAI20	53.7	70.1	76.4	60.2	72.6	71.5	56.8	51.9	77.1	70.2	56.3	82.1	66.6
ATDOC	CVPR21	58.3	78.8	82.3	69.4	78.2	78.2	67.1	56.0	82.7	72.0	58.2	**85.5**	72.2
SIDA	IJCAI22	57.2	79.1	81.7	67.1	74.5	77.3	67.2	53.9	82.5	71.4	58.7	83.3	71.2
CDAN	NIPS18	49.0	69.3	74.5	54.4	66.0	68.4	55.6	48.3	75.9	68.4	55.4	80.5	63.8
MetaAlign	CVPR21	59.3	76.0	80.2	65.7	74.7	75.1	65.7	56.5	81.6	74.1	61.1	85.2	71.3
DALN	CVPR22	57.8	79.9	82.0	66.3	76.2	77.2	66.7	55.5	81.3	73.5	60.4	85.2	71.8
CDAL	ACMMM22	59.5	77.8	80.0	67.0	77.1	76.6	66.6	56.2	81.8	74.3	60.6	84.6	71.8
SRADA	IJCAI20	57.2	79.1	81.7	67.1	74.5	77.3	67.2	53.9	82.5	71.4	58.7	83.3	71.2
SDAT+ELF	ICLR23	58.2	79.7	82.5	67.5	77.2	77.2	64.6	57.9	82.2	75.4	**63.1**	**85.5**	72.6
DSR	IJCAI19	53.4	71.6	77.4	57.1	66.8	69.3	56.7	49.2	75.7	68.0	54.0	79.5	64.9
IC^{2}FA	ACMMM21	56.7	78.6	81.0	64.8	73.7	74.9	65.5	53.9	81.7	74.1	59.8	**85.5**	70.8
IndUDA	Ours	**61.9** ±0.2	**80.2** ±0.1	**83.0** ±0.1	**70.3** ±0.2	**78.9** ±0.1	**78.8** ±0.1	**70.9** ±0.2	**59.6** ±0.3	**84.0** ±0.2	**76.1** ±0.3	63.0 ±0.2	85.4 ±0.2	**74.3**

Table 1: Comparisons with state-of-the-art methods on *Office-Home* dataset. Metric: classification accuracy (%); Backbone: ResNet-50.

So a random variable $r \sim \mathcal{N}(0, I)$ is employed to replace the domain-specific part, which can improve the generalization ability of the reconstructed features while also maintaining the distribution consistency between z_s^{di} and z_t^{di}. In this way, we can make the domain-invariant part follow a distribution and it is also domain-agnostic because the random variable does not contain any domain-specific noise.

Formally, two denoised features with respect to two domain-invariant parts are

$$f_s^{ide} = G_{flow}^{-1}(z_s^{di} \oplus r), f_t^{ide} = G_{flow}^{-1}(z_t^{di} \oplus r). \quad (7)$$

An instance feature alignment method is performed to align the source and target features respectively. The alignment loss can be written as follows,

$$\min_F \mathcal{L}_{align} = \mathcal{D}_H\left(f_s, f_s^{ide}\right) + \mathcal{D}_H\left(f_t, f_t^{ide}\right), \quad (8)$$

where $\mathcal{D}_H$ is also the CDD loss [Kang *et al.*, 2019]. This alignment paradigm is theoretically reasonable. For minimizing distribution divergence $D_H(f_s, f_t)$ between the source and target domains, based on triangle inequality, we can write $D_H(f_s, f_t)$ as follows,

$$D_H(f_s, f_t) \leq D_H(f_s, f_s^{ide}) + D_H(f_t, f_t^{ide}) + D_H(f_s^{ide}, f_t^{ide})$$

where the term $D_H(f_s, f_s^{ide}) + D_H(f_t, f_t^{ide})$ corresponds the loss $\mathcal{L}_{align}$; while $D_H(f_s^{ide}, f_t^{ide})$ corresponds to the loss $\mathcal{L}_{con}$ in Eq.(7) since $D_H(f_s^{ide}, f_t^{ide}) \simeq D_H(z_s^{di}, z_t^{di})$ and G_{flow} is a bijection. So we only need to align the features with their corresponding reconstructed features as shown in Eq.(8). In this way, we can minimize $D_H(f_s, f_t)$ through $\mathcal{L}_{align}$ and $D_H(f_s^{ide}, f_t^{ide})$.

Remark. The previous methods always align the target features with the source features directly without taking independence into consideration. The performance is hugely degraded by domain-specific noise. As IndUDA aligns the source features and target features with their denoised features at the instance level,the feature extractor can efficiently obtain domain-agnostic features. Here, CDD loss is used instead of L_2 because CDD can better guarantee the consistency of class distribution.

3.4 Overall Loss Function

In the independent feature decomposition part, the parameters of G_{flow} are updated by gradient descent via $\mathcal{L}_{flow}$ as follows,

$$\min_{G_{flow}} \mathcal{L}_{flow} = \mathcal{L}_{swap} + \mathcal{L}_{con}. \quad (9)$$

In the instance feature alignment part, the feature extractor F is updated by minimizing $\mathcal{L}_{align}$ in Eq. (6).

Besides, we adapt the cross entropy loss $\mathcal{L}_{ce}$ to maintain the performance on source model,

$$\min_{F,C} \mathcal{L}_{ce} = \mathrm{E}_{x_s^i \in D_s} \mathcal{L}_{ce}(f_s^i, y_s^i). \quad (10)$$

So the overall object function $\mathcal{L}_{total}$ can be written as

$$\min_{G_{flow},F,C} \mathcal{L}_{total} = \mathcal{L}_{ce} + \gamma \mathcal{L}_{flow} + \lambda \mathcal{L}_{align}. \quad (11)$$

where λ and γ are trade-off parameters to weight the loss functions.

Because the feature extractor F and the invertible flow G_{flow} are two separate networks, it is not easy to optimize them together. So the training consists of two phases: first, the G_{flow} is trained, and then the trained G_{flow} is used to guide the training of F. For training G_{flow}, pseudo-labels are needed for target samples. The updated model will be iteratively used. In the test procedure, only the adapted networks F and C are used.

4 Experiments

4.1 Experimental Setup

Datasets. We conduct experiments on three standard domain adaptation datasets. ***Office-31*** [Saenko *et al.*, 2010] is a popular dataset with a total of 4652 photos, including 31 object categories in three domains: Amazon (A), DSLR (D) and Webcam (W). ***Office-Home*** [Venkateswara *et al.*, 2017] is a more challenging dataset, which contains 15500 images with an average of 70 images per class. It consists of 65 categories in 4 domains: Art (A), Clipart (C), Product (P) and Real-World (R). ***VisDA-2017*** [Peng *et al.*, 2018] is a simulation-to-real dataset across 12 categories. The source domain train-

Method	Venue	plane	bcycl	bus	car	horse	knife	mcycl	person	plant	sktbrd	train	truck	*Avg.*
ResNet-101	CVPR16	55.1	53.3	61.9	59.1	80.6	17.9	79.7	31.2	81.0	26.5	73.5	8.5	52.4
CAN	CVPR19	97.0	87.2	82.5	74.3	**97.8**	96.2	**90.8**	80.7	**96.6**	**96.3**	87.5	59.9	87.2
ALDA	AAAI20	93.8	74.1	82.4	69.4	90.6	87.2	89.0	67.6	93.4	76.1	87.7	22.2	77.8
ATDOC	CVPR21	93.7	83.0	76.9	58.7	89.7	95.1	84.4	71.4	89.4	80.0	86.7	55.1	80.3
SIDA	IJCAI22	95.4	83.1	77.1	64.6	94.5	97.2	88.7	78.4	93.8	89.9	85.2	59.4	84.0
SUDA	CVPR22	88.3	79.3	66.2	64.7	87.4	80.1	85.9	78.3	86.3	87.5	78.8	**74.5**	79.8
CaCo	CVPR22	90.4	80.7	78.8	57.0	88.9	87.0	81.3	79.4	88.7	88.1	86.8	63.9	80.9
CDAN	NIPS18	85.2	66.9	83.0	50.8	84.2	74.9	88.1	74.5	83.4	76.0	81.9	38.0	73.9
DWL	CVPR21	90.7	80.2	**86.1**	67.6	92.4	81.5	86.8	78.0	90.6	57.1	85.6	28.7	77.1
CLS	ICCV21	92.6	84.5	73.7	72.7	88.5	83.3	89.1	77.6	89.5	89.2	85.8	72.7	81.6
CDAL	ACMMM22	97.5	84.9	81.0	70.5	97.1	**97.3**	90.6	80.9	96.2	94.9	88.2	48.7	85.7
SRADA	IJCAI20	95.4	83.1	77.1	64.6	94.5	97.2	88.7	78.4	93.8	89.9	85.2	59.4	84.0
IC^{2}FA	ACMMM21	89.7	70.6	79.8	**84.3**	96.5	72.1	90.4	65.3	92.7	63.3	86.5	36.0	77.3
DTA	ICCV19	93.7	82.2	85.6	83.8	93.0	81.0	90.7	**82.1**	95.1	78.1	86.4	32.1	81.5
IndUDA	Ours	**97.6**	**89.6**	**86.1**	81.3	96.6	96.0	90.0	81.8	95.4	92.2	**88.9**	60.3	**88.0**

Table 2: Comparison with state-of-the-art methods on *Visda-17* dataset. Metric: per-class classification accuracy (%); Backbone: ResNet-101.

ing set contains 152397 synthetic images and the target domain validation set contains 55388 real-world images. Further datasets are shown in Appendix.

Implementation details. All experiments are carried out on the Pytorch platform. The invertible flow INN consists of 5 blocks, whose structure is shown in Appendix, and the subnet $\phi_{1,2,3}(\cdot)$ of each INN block consists of 3 convolution layers with LeakyReLu function only in the first layer. In order to compare the state-of-the-art methods fairly across various datasets, we use the commonly pre-trained ResNet-50/50/101 [He *et al.*, 2016a; He *et al.*, 2016b] on ImageNet [Deng *et al.*, 2009] as feature extractors for *Office-31, Office-Home* and *VisDA-2017* respectively. The classifier consists of a fully connected layer. With the exception of the batch normalization layers' domain-specific characteristics, all network parameters are shared by the source domain and target domain data. The initial learning rate η_0 is set as 0.001 for the first convolutional layers and 0.01 for the rest including the entire INN network and no pre-trained layers in the feature extractor. We adopt the same learning rate scheduler $\eta_p = \eta_0(1 + ap)^{-b}$. For Office-31 and Office-home datasets, we set $a = 10$ and $b = 0.75$, while for Visda-2017 we set $a = 10$ and $b = 2.25$. We select half of the channels as domain-invariant feature channels in *channel mask block* and choose 0.5, 0.3 as the value of γ and λ respectively.

Competitors. We compare our method with three types of unsupervised domain adaptation methods. The first type of method adapts implicit alignment based on adversarial learning. They are CDAN [Long *et al.*, 2018], DALN [Chen *et al.*, 2022], CDAL [Zhou *et al.*, 2022b], DWL [Xiao and Zhang, 2021], CLS [Liu *et al.*, 2021a], SRADA [Wang and Zhang, 2020] and SDAT+ELS [Zhang *et al.*, 2023]. The second type of method is explicitly alignment based on statistic moment matching. They are SUDA [Zhang *et al.*, 2022], CaCo [Huang *et al.*, 2022] ATDOC [Liang *et al.*, 2021], SIDA [Zhao *et al.*, 2022], ALDA [Chen *et al.*, 2020] and CAN [Kang *et al.*, 2019]. The third type of method is based on feature decomposition, which decomposes features into domain-invariant features and domain-specific features. They are DSR [Cai *et al.*, 2019], IC^{2}FA [Deng *et al.*, 2021] and DTA [Lee *et al.*, 2019].

4.2 Comparison with State-of-the-Art

Tables 1-2 show experiment results of our IndUDA and other state-of-the-art methods on datasets *Office-home, Visda-2017*, respectively. The total results on *Office-31* are not shown here because the performance is saturated and limited page size, which can be found in Appendix. Especially, on *Office-31*, our method IndUDA reaches a mean accuracy of 91.4%, which is 0.8% higher than the best competitor (CAN). As for the *Office-home*, IndUDA boosts the mean accuracy to 74.3% than other state-of-the-art methods. It is 2.1% higher than the second highest method ATDOC [Liang *et al.*, 2021]. Specifically, our results achieve the best results in 11 of the 12 tasks and only 0.1% less than the best method on task $Rw \rightarrow Pr$. On *Visda-2017*, IndUDA also achieves the best results compared with other state-of-the-art methods.

More specifically, for adversarial learning methods, IndUDA outperforms the best approach CDAL [Zhou *et al.*, 2022b] by 2.4% in an average on *Office-Home* and *visda-17* datasets. Compared with the best statistic moment matching based methods CAN [Kang *et al.*, 2019], the result of IndUDA is 1.7% higher in an average on these two datasets. Both of these alignment methods do not consider the effectiveness of domain-specific noise, while our IndUDA makes the domain-invariant part and domain-specific part independent and denoise the domain-specific feature before alignment, which can achieve better alignment. For the feature decomposition-based domain adaptation methods, our method also outperforms the corresponding competitors, because they cannot decompose the features without domain-specific noise. All in all, the experiment results confirm our previous analysis: splitting out the domain-specific part by independence is advantageous; instance cross-domain swap strategy can obtain domain-invariant part and make the feature space more compact. These properties boost the adaptation performance.

Due to space limitations, the visual analysis is shown in Appendix. There are two basic observations. Compared with

$\mathcal{L}_{con}$	$\mathcal{L}_{swap}$	$\mathcal{L}_{align}$	Office-31				Office-home
			A→D	A→W	D→A	W→A	*Avg.*
✓		✓	90.0	90.7	74.8	74.1	72.7
	✓	✓	94.4	95.2	77.3	77.8	73.3
✓	✓	✓	**96.6**	**95.8**	**78.9**	**78.1**	**74.3**

Table 3: Ablation study on loss function.

Networks	Office-31				Office-home
	A→D	A→W	D→A	W→A	*Avg.*
Two INNs	94.4	92.8	77.0	76.0	70.8
Doublemap	95.6	95.4	77.8	77.2	73.5
IndUDA	**96.6**	**95.8**	**78.9**	**78.1**	**74.3**

Table 4: Abalation study on using invertible neural network.

the existing methods, (1) IndUDA can extract more compact features and (2) the features focus more on the category-specific local and global parts in the image.

4.3 Further Analysis

Ablation study on loss function. The ablation analysis is performed on *Office-31* and *Office-home* dataset. For *Office-31*, the experiments are evaluated on the tasks $A \rightarrow W$, $A \rightarrow D$, $D \rightarrow A$ and $W \rightarrow A$, as the performance on other two tasks is saturated. To explore the effectiveness of different loss functions (components), we do experiments without $\mathcal{L}_{con}$ and without $\mathcal{L}_{swap}$ respectively. Without $\mathcal{L}_{con}$ means that the domain-invariant part is obtained only via swap strategy, i.e. using the loss $\mathcal{L}_{swap}$. While for the experiment without $\mathcal{L}_{swap}$, it means we only align the domain-invariant part by the consistency loss $\mathcal{L}_{con}$. It can be seen from Tab.3 that swap strategy $\mathcal{L}_{swap}$ can reach a better alignment performance on the domain-invariant feature. The performance is 1.6% higher on each task than those without $\mathcal{L}_{swap}$. Because the input and output of invertible feature mapping are strictly one-to-one corresponding, it adds a strong constraint such that the domain-invariant part and domain-specific part are well separated. Furthermore, from the last two rows in Tab.3, without the constrain loss $\mathcal{L}_{con}$, the performance decreases by 1%, because the loss $\mathcal{L}_{con}$ makes the features in the latent space more compact such that the reconstructed feature is denoised better.

Ablation analysis on invertible neural networks. To verify the effectiveness of using INN, we experiment with different variants, i.e., two INNs, Doublemap and One INN (IndUDA). Two INNs mean two invertible neural networks are trained corresponding to the source and target features respectively, which do not share the weight but with the same structure. Doublemap means double mapping based on an autoencoder scheme which is not invertible. The encoder and decoder share the same structure which consists of three 3×3 convolutional layers with the feature output dimensions as 64, 64 and 2048 respectively, while the input of the third convolutional layer is the concatenation of the extracted feature and the output of the second convolutional layer. Besides, each convolutional layer is followed by a Leaky-RELU layer.

From Tab.4, it can be observed that using One INN (IndUDA) reaches the best result. The role of INN is to find the domain-invariant part while denoising the domain-specific noise. We swap domain-invariant parts to make the reconstructed features consistent with the original features, so as to realize the separation of the domain-invariant parts and domain-specific parts. One INN ensures the minimum reconstruction error, so the domain-invariant part found in this way is better. Two INNs can not obtain domain-invariant part because the correspondences between the constructed swap features and the original features are not guaranteed. The features reconstructed by double mapping have more errors compared with One INN, resulting in lower accuracy.

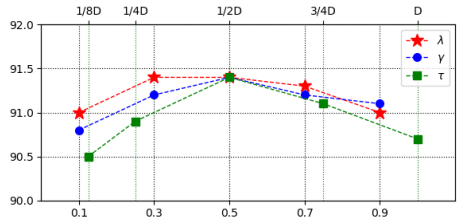

Figure 3: Hyper-parameter sensitivity analysis of λ, γ and τ on *Office-31* dataset.

Sensitivity analysis of the parameters. There are three hyper-parameters, which are γ, λ in Eq.(11) and τ in Eq.(2), respectively. Experiments are performed on *Office-31*. First, with different values of γ, λ in the range of $[0.1, 2.0]$, as shown in Fig.3, the performance of our method consistently outperforms the baseline over a wide range and is stable as the value gets larger. Besides, the bell-shaped curves also prove the robustness of our method. According to the performance curves, we set γ and λ equal to 0.5, and 0.3 respectively.

Regarding τ, its value is set such that $D/8$, $D/4$, $D/2$, $3D/4$, D channels are selected as domain-invariant channels, where $D = 2048$ is the amount of feature channels. It can be observed that the accuracy is highest when $D/2$ channels are exactly selected. .The performance of our method is not good when only $D/8$ and $D/4$ channels are chosen. This case misses many domain-invariant channels and dismisses some indispensable channels for classification; while choosing $3D/4$ and D channels means almost all channels are reserved and domain-specific channels are not dismissed. So half channels for the domain-invariant part and another half for the domain-specific part is a reasonable choice.

5 Conclusion

In this paper, we proposed a novel framework, named Independent feature decomposition and alignment for Unsupervised Domain Adaptation (IndUDA). By considering domain-specific information contained in the features as noise, and sufficiently using the property of invertible flow, a swap strategy is proposed to guarantee the separated domain-invariant and domain-specific parts in the latent space are independent. After discarding domain-specific parts, an independent alignment module is used to align the inverted ideal domain-invariant features with the original features. Thus domain-invariant features in the original feature space are obtained. Experiments on real-world benchmarks demonstrate the superiority and effectiveness of our method compared with the state-of-the-art baselines.

Ethical Statement

There are no ethical issues.

References

[Cai *et al.*, 2019] Ruichu Cai, Zijian Li, Pengfei Wei, Jie Qiao, Kun Zhang, and Zhifeng Hao. Learning disentangled semantic representation for domain adaptation. In *Proceedings of the International Joint Conference on Artificial Intelligence*, volume 2019, page 2060. NIH Public Access, 2019.

[Chen *et al.*, 2020] Minghao Chen, Shuai Zhao, Haifeng Liu, and Deng Cai. Adversarial-learned loss for domain adaptation. In *Proceedings of the AAAI Conference on Artificial Intelligence*, volume 34, pages 3521–3528, 2020.

[Chen *et al.*, 2022] Lin Chen, Huaian Chen, Zhixiang Wei, Xin Jin, Xiao Tan, Yi Jin, and Enhong Chen. Reusing the task-specific classifier as a discriminator: Discriminator-free adversarial domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7181–7190, 2022.

[Deng *et al.*, 2009] Jia Deng, Wei Dong, Richard Socher, Li-Jia Li, Kai Li, and Li Fei-Fei. Imagenet: A large-scale hierarchical image database. In *2009 IEEE conference on Computer Vision and Pattern Recognition*, pages 248–255. Ieee, 2009.

[Deng *et al.*, 2021] Wanxia Deng, Yawen Cui, Zhen Liu, Gangyao Kuang, Dewen Hu, Matti Pietikäinen, and Li Liu. Informative class-conditioned feature alignment for unsupervised domain adaptation. In *Proceedings of the 29th ACM International Conference on Multimedia*, pages 1303–1312, 2021.

[Deng *et al.*, 2022] Wanxia Deng, Lingjun Zhao, Qing Liao, Deke Guo, Gangyao Kuang, Dewen Hu, Matti Pietikäinen, and Li Liu. Informative feature disentanglement for unsupervised domain adaptation. *IEEE Transactions on Multimedia*, 24:2407–2421, 2022.

[Dinh *et al.*, 2016] Laurent Dinh, Jascha Sohl-Dickstein, and Samy Bengio. Density estimation using real nvp. *arXiv preprint arXiv:1605.08803*, 2016.

[French *et al.*, 2018] Geoff French, Michal Mackiewicz, and Mark Fisher. Self-ensembling for visual domain adaptation. In *International Conference on Learning Representations*, 2018.

[Ganin and Lempitsky, 2015] Yaroslav Ganin and Victor Lempitsky. Unsupervised domain adaptation by backpropagation. In *International Conference on Machine Learning*, pages 1180–1189. PMLR, 2015.

[Ganin *et al.*, 2016] Yaroslav Ganin, Evgeniya Ustinova, Hana Ajakan, Pascal Germain, Hugo Larochelle, François Laviolette, Mario Marchand, and Victor Lempitsky. Domain-adversarial training of neural networks. *The journal of Machine Learning Research*, 17(1):2096–2030, 2016.

[He *et al.*, 2016a] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Deep residual learning for image recognition. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[He *et al.*, 2016b] Kaiming He, Xiangyu Zhang, Shaoqing Ren, and Jian Sun. Identity mappings in deep residual networks. In *European Conference on Computer Vision*, pages 630–645. Springer, 2016.

[Huang *et al.*, 2020] Zeyi Huang, Haohan Wang, Eric P Xing, and Dong Huang. Self-challenging improves cross-domain generalization. In *European Conference on Computer Vision*, pages 124–140. Springer, 2020.

[Huang *et al.*, 2022] Jiaxing Huang, Dayan Guan, Aoran Xiao, Shijian Lu, and Ling Shao. Category contrast for unsupervised domain adaptation in visual tasks. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 1203–1214, 2022.

[Kang *et al.*, 2019] Guoliang Kang, Lu Jiang, Yi Yang, and Alexander G Hauptmann. Contrastive adaptation network for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 4893–4902, 2019.

[Kingma and Dhariwal, 2018] Durk P Kingma and Prafulla Dhariwal. Glow: Generative flow with invertible 1x1 convolutions. *Advances in Neural Information Processing Systems*, 31, 2018.

[Lee *et al.*, 2019] Seungmin Lee, Dongwan Kim, Namil Kim, and Seong-Gyun Jeong. Drop to adapt: Learning discriminative features for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF International Conference on Computer Vision (ICCV)*, October 2019.

[Lee *et al.*, 2021] Seunghun Lee, Sunghyun Cho, and Sunghoon Im. Dranet: Disentangling representation and adaptation networks for unsupervised cross-domain adaptation. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pages 15252–15261, 2021.

[Liang *et al.*, 2021] Jian Liang, Dapeng Hu, and Jiashi Feng. Domain adaptation with auxiliary target domain-oriented classifier. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 16632–16642, 2021.

[Liu *et al.*, 2018] Yen-Cheng Liu, Yu-Ying Yeh, Tzu-Chien Fu, Sheng-De Wang, Wei-Chen Chiu, and Yu-Chiang Frank Wang. Detach and adapt: Learning cross-domain disentangled deep representation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition*, pages 8867–8876, 2018.

[Liu *et al.*, 2021a] Xiaofeng Liu, Zhenhua Guo, Site Li, Fangxu Xing, Jane You, C.-C. Jay Kuo, Georges El Fakhri, and Jonghye Woo. Adversarial unsupervised domain adaptation with conditional and label shift: Infer, align and iterate. In *Proceedings of the IEEE/CVF International Conference on Computer Vision*, pages 10367–10376, October 2021.

[Liu *et al.*, 2021b] Yang Liu, Zhenyue Qin, Saeed Anwar, Pan Ji, Dongwoo Kim, Sabrina Caldwell, and Tom Gedeon. Invertible denoising network: A light solution for real noise removal. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pages 13365–13374, 2021.

[Long *et al.*, 2015] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. In *International Conference on Machine Learning*, pages 97–105. PMLR, 2015.

[Long *et al.*, 2016] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Unsupervised domain adaptation with residual transfer networks. *Advances in Neural Information Processing Systems*, 29, 2016.

[Long *et al.*, 2018] Mingsheng Long, Zhangjie Cao, Jianmin Wang, and Michael I Jordan. Conditional adversarial domain adaptation. *Advances in Neural Information Processing Systems*, 31, 2018.

[Lugmayr *et al.*, 2020] Andreas Lugmayr, Martin Danelljan, Luc Van Gool, and Radu Timofte. Srflow: Learning the super-resolution space with normalizing flow. In *European Conference on Computer Vision*, pages 715–732. Springer, 2020.

[Pan and Yang, 2009] Sinno Jialin Pan and Qiang Yang. A survey on transfer learning. *IEEE Transactions on knowledge and data engineering*, 22(10):1345–1359, 2009.

[Pan *et al.*, 2010] Sinno Jialin Pan, Ivor W Tsang, James T Kwok, and Qiang Yang. Domain adaptation via transfer component analysis. *IEEE Transactions on Neural Networks*, 22(2):199–210, 2010.

[Pan *et al.*, 2019] Yingwei Pan, Ting Yao, Yehao Li, Yu Wang, Chong-Wah Ngo, and Tao Mei. Transferrable prototypical networks for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, June 2019.

[Patel *et al.*, 2015] Vishal M Patel, Raghuraman Gopalan, Ruonan Li, and Rama Chellappa. Visual domain adaptation: A survey of recent advances. *IEEE signal processing magazine*, 32(3):53–69, 2015.

[Peng *et al.*, 2018] Xingchao Peng, Ben Usman, Neela Kaushik, Dequan Wang, Judy Hoffman, and Kate Saenko. Visda: A synthetic-to-real benchmark for visual domain adaptation. In *Proceedings of the IEEE Conference on Computer Vision and Pattern Recognition (CVPR) Workshops*, June 2018.

[Saenko *et al.*, 2010] Kate Saenko, Brian Kulis, Mario Fritz, and Trevor Darrell. Adapting visual category models to new domains. In *European Conference on Computer Vision*, pages 213–226. Springer, 2010.

[Saito *et al.*, 2018] Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum classifier discrepancy for unsupervised domain adaptation. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 3723–3732, 2018.

[Sugiyama *et al.*, 2007] Masashi Sugiyama, Shinichi Nakajima, Hisashi Kashima, Paul Buenau, and Motoaki Kawanabe. Direct importance estimation with model selection and its application to covariate shift adaptation. *Advances in Neural Information Processing Systems*, 20, 2007.

[Tang *et al.*, 2020] Hui Tang, Ke Chen, and Kui Jia. Unsupervised domain adaptation via structurally regularized deep clustering. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pages 8725–8735, 2020.

[Tzeng *et al.*, 2017] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 7167–7176, 2017.

[Venkateswara *et al.*, 2017] Hemanth Venkateswara, Jose Eusebio, Shayok Chakraborty, and Sethuraman Panchanathan. Deep hashing network for unsupervised domain adaptation. In *Proceedings of the IEEE conference on Computer Vision and Pattern Recognition*, pages 5018–5027, 2017.

[Wang and Zhang, 2020] Shanshan Wang and Lei Zhang. Self-adaptive re-weighted adversarial domain adaptation. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 3181–3187, 2020.

[Xiao and Zhang, 2021] Ni Xiao and Lei Zhang. Dynamic weighted learning for unsupervised domain adaptation. In *Proceedings of the IEEE/CVF conference on Computer Vision and Pattern Recognition*, pages 15242–15251, 2021.

[Zhang *et al.*, 2022] Jingyi Zhang, Jiaxing Huang, Zichen Tian, and Shijian Lu. Spectral unsupervised domain adaptation for visual recognition. In *Proceedings of the IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 9829–9840, 2022.

[Zhang *et al.*, 2023] YiFan Zhang, Xue Wang, Jian Liang, Zhang Zhang, Liang Wang, Rong Jin, and Tieniu Tan. Free lunch for domain adversarial training: Environment label smoothing. In *The Eleventh International Conference on Learning Representations*, 2023.

[Zhao *et al.*, 2022] Haiteng Zhao, Chang Ma, Qinyu Chen, and Zhi-Hong Deng. Domain adaptation via maximizing surrogate mutual information. In *Proceedings of the International Joint Conference on Artificial Intelligence*, pages 1700–1706, 2022.

[Zhou *et al.*, 2022a] Lihua Zhou, Mao Ye, Dan Zhang, Ce Zhu, and Luping Ji. Prototype-based multisource domain adaptation. *IEEE Transactions on Neural Networks and Learning Systems*, 33(10):5308–5320, 2022.

[Zhou *et al.*, 2022b] Lihua Zhou, Mao Ye, Xiatian Zhu, Shuaifeng Li, and Yiguang Liu. Class discriminative adversarial learning for unsupervised domain adaptation. In *Proceedings of the 30th ACM International Conference on Multimedia*, pages 4318–4326, 2022.