ECAI 2024
U. Endriss et al. (Eds.)

doi:10.3233/FAIA240710

Cross-Stage Transfer in Multi-Stage Cascade Ranking and Filtering Systems

Yifan Pan[a], Guibo Luo[a,*] and Yuesheng Zhu[a,*]

[a]Communication and Information Security Lab, Shenzhen Graduate School, Peking University

Abstract. Unsupervised Domain Adaptation (UDA) aims to transfer a model from a labeled source domain to an unlabeled target domain, addressing challenges of distinct data distributions, termed *domain shift*. Existing UDA research primarily focuses on classification-like tasks, but neglects ranking and filtering tasks essential for applications like medical diagnosis and search engines. This paper is the first to notice and identify a new real-world transfer problem: cross-stage transfer in multi-stage cascade ranking and filtering systems, a common issue in diverse applications, including information retrieval systems, medical diagnosis, and other real-world ranking/filtering systems. In this problem, we emphasize the crucial assumption of order-invariance and address the key issue named Cross-stage Class Concept Conflict (C4), highlighting potential inconsistencies in class concepts for the same sample at different stages. To tackle these challenges, we propose a novel method, Unsupervised Rank Adaptation (URA), comprising two key components: order-conditional distribution alignment, characterizing the order-conditional distribution intra-stage and aligning them across stages; and principal projection alignment, aligning the principal component's projection matrix with classifier parameters to ensure order-invariance without guessing pseudo-labels, mitigating the influence of C4. Experimental results show that our approach reaches state-of-the-art performance in various cross-stage transfer tasks.

1 Introduction

Transfer learning aims to help machine learning systems perform well on new tasks by leveraging experiences from previous tasks. Unsupervised Domain Adaptation (UDA), a widely discussed aspect of transfer learning, focuses on adapting a model learned from a labeled source domain to work on an unlabeled target domain. This process addresses the challenge of different data distribution across domains, often called *domain shift* [28]. UDA has been applied to various fields such as video action recognition [37], medical image segmentation [7], text classification [24], multi-object tracking [1], nighttime semantic segmentation [17], and so on.

Traditional UDA research predominantly focuses on tasks such as classification and semantic segmentation (specifically, pixel-level classification) [2, 31, 20, 4]. Previous approaches always learn a domain-invariant latent feature space and employ a shared classifier trained with source semantic supervision [14]. Additionally, some approaches explore the utilization of the self-training strategy [42, 26, 38, 33, 41]. Although prior approaches have achieved remarkable success in classification-like tasks, they always exhibit poor performance in some other real-world scenarios. Specifically, in multi-stage cascade ranking and filtering systems, such as medical diagnosis (Fig. 1) and item retrieval in e-commerce (Fig. 2), it emphasizes the order of results.

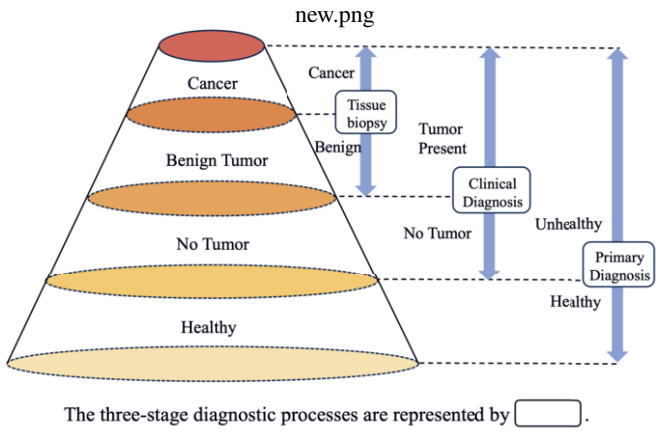


Figure 1. Multi-stage cascade filtering in medical diagnosis. The three-stage diagnostic processes can all be replaced by an AI system.

For instance, as shown in Fig. 1, being diagnosed with cancer represents the final positive concept of the entire filtering system. To balance effectiveness and efficiency, the diagnostic process is organized as a multi-stage cascade: Lower stages prioritize handling extensive data with high recall and efficiency, minimizing time and resource consumption but accepting lower effectiveness (e.g., precision). In contrast, upper stages like tissue biopsy prioritize diagnosis effectiveness (e.g., accuracy) at the expense of efficiency.

However, error accumulation across multi-stages may impair performance, and the inconsistency among stages will exacerbate this issue. Additionally, each stage presents specific advantages: lower-stage models, trained with more data, are less affected by distribution shifts due to selection bias. On the other hand, upper-stage models have more competent architectures and precise supervision, making their predictions more accurate. Hence, an intuitive idea arises: mutual knowledge transfer across stages can leverage their respective strengths and ensure consistency, potentially enhancing the overall performance of the multi-stage cascade ranking and filtering system.

Despite this, traditional approaches are not well-suited to this situation. Cross-stage transfer emphasizes **Order-invariance** rather than distribution-invariance (distribution alignment). Order-invariance signifies the constancy of the relative positional relationship between

* Corresponding Author. Email: {luogb,zhuys}@pku.edu.cn.

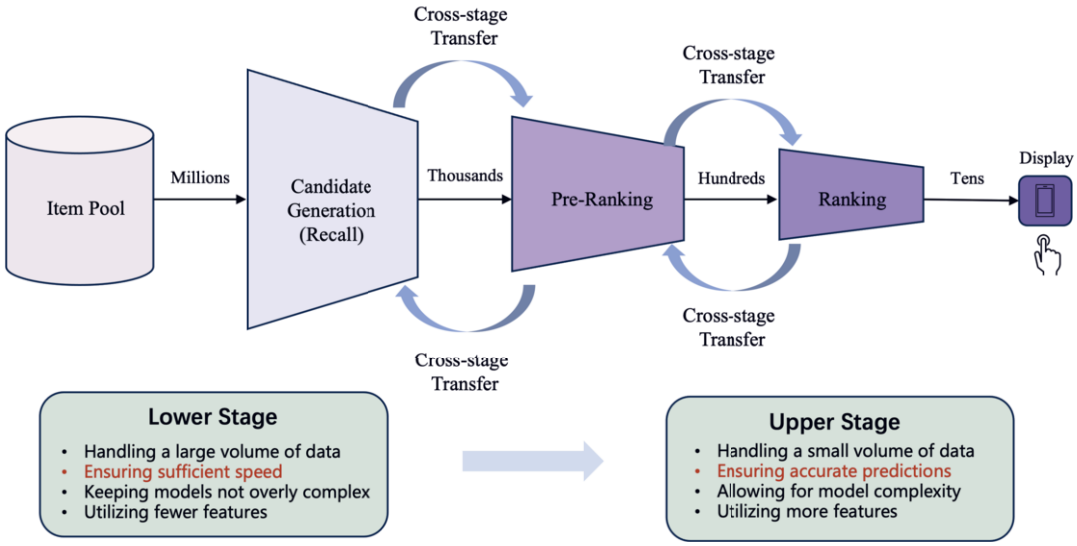


Figure 2. Cross-stage transfer in multi-stage cascade ranking system for item retrieval in e-commerce.

two samples, which will be described in detail in Sec. 2. Besides, there is never an aligned classification boundary between different stages, leading to a situation where the same sample in different stages may acquire different labels. For instance, the absence of a tumor but an unhealthy sample in the primary diagnosis stage is considered positive, whereas it is deemed negative in clinical diagnosis. Specifically, we term this issue as **Cross-stage Class Concept Conflict (C4)**. Traditional UDA approaches never consider the new assumption about order-invariance and the associated problem C4, limiting efficacy in cross-stage transfer.

In this paper, we were the first to notice and identify a new real-world transfer problem with practical application value: cross-stage transfer in multi-stage cascade ranking and filtering systems. We introduce a novel method, Unsupervised Rank Adaptation (URA) to address this issue.

Our primary emphasis is on addressing one of the fundamental challenges in this problem: studying changes in data distribution across different stages. There are two primary technologies in URA: 1) Order-Conditional Distribution Alignment (OCDA): Considering the order-invariance assumption, OCDA models and aligns the relationship distribution of intra-stage samples through the residuals of features across stages, conditioned on the predicted order. 2) Principal Projection Alignment (PPA): Tackling the challenging issue of C4 and drawing inspiration from principal component analysis, we assume that the scores of the samples exhibit a monotonic distribution in the direction of maximum variance. PPA aligns the projection matrix of the principal component with classifier parameters. This alignment ensures order-invariance without relying on guesswork for pseudo-labels. Consequently, we posit that the direction of maximum variance in the target distribution aligns with the sorting direction of the data distribution.

Our main contributions can be summarized as follows:

1. We are the first to notice and formulate the new real-world transfer learning scenario: cross-stage transfer in a multi-stage cascade ranking and filtering system.
2. We highlight the order-invariance assumption, underscoring its critical role in the cross-stage transfer. Furthermore, we consider the key challenge of Cross-stage Class Concept Conflict (C4), exploring its implications within the proposed cross-stage transfer.
3. We propose a novel method URA comprising two key components: order-conditional distribution alignment and principal projection alignment, which can navigate the challenges presented by cross-stage transfer.
4. We conduct extensive empirical studies on datasets CIFAR-10 and CBIS-DDSM to validate the efficacy of the proposed URA.

The remainder of this paper is organized as follows: In Section 2, we present the preliminaries of this paper. In Section 3, we describe our proposed method URA and Section 4 describes our experiments in different datasets. In Section 5, we present a review of related works. Lastly, Section 6 provides a brief conclusion.

2 Preliminaries

2.1 Multi-stage Cascade Ranking and Filtering

Multi-stage cascade ranking/filtering is a prevalent architecture in machine learning applications, such as medical diagnosis and e-commerce. Formally, it can be formulated as $\{M_{\theta_1}, M_{\theta_2}, ..., M_{\theta_N}\}$, where N means the number of stages. Each model M_i can be specifically constructed by distinct parameters and architectures, operating in a cascaded manner.

Considering effectiveness and efficiency, the upper model is always more capable yet slower, whereas the lower one is naive yet faster. During the training process, data go through the upper model are filtered by the lower one:

$$D_u = \{X \in D_l | M_l(X) = 1\}. \tag{1}$$

Here, D_u serves as the input for the upper model, while D_l corresponds to the lower model.

Models at different stages exhibit specific strengths and weaknesses. The upper model is more capable but trained with less data. Thus, it can not perform well on unseen data, which is filtered out by a lower model. This phenomenon is referred to as sample selection bias (SSB) [3]. Conversely, the lower model has more exposure to data but lacks the complexity required for challenging tasks.

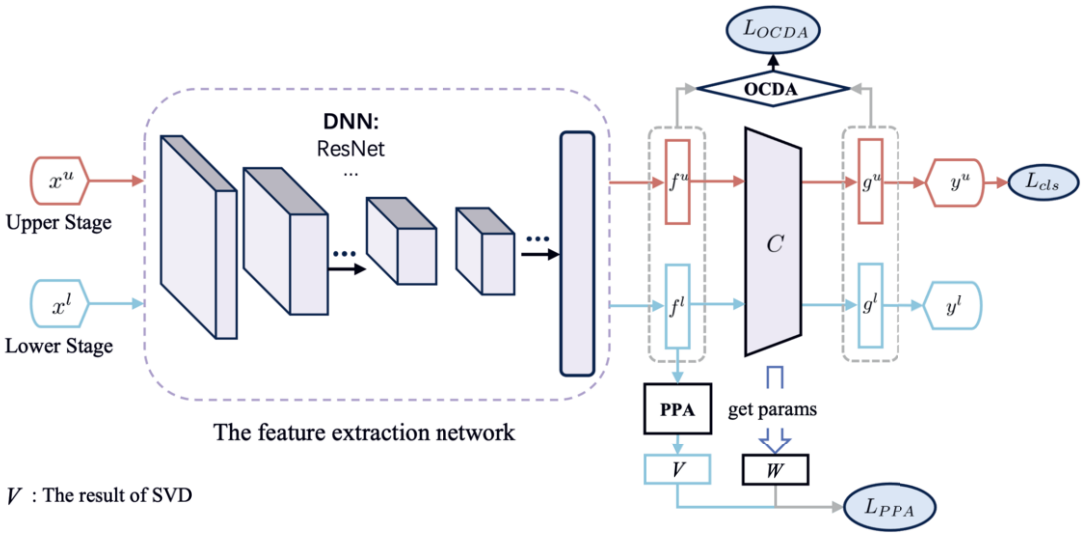


Figure 3. The framework of our method URA: A cross-stage shared single linear layer is employed as the classifier $C : z \mapsto y$ for classification. $(C(z) = softmax(W^T z + b), W \in \mathbb{R}^{2\times c})$. And U means the result of SVD.

2.2 *Cross-stage Transfer*

To make the most of each stage's strengths and address their weaknesses, an intuitive idea is to mutually transfer knowledge between different stages, a process we term *cross-stage transfer*. However, cross-stage transfer introduces entirely different assumptions and key problems with classic UDA, especially the invariance of order relationships between samples and the possible existence of cross-stage class concept conflict problem, which will be explored in the following discussion in detail.

Order-invariance Assumption Existing research on UDA has mainly focused on classification-like tasks, overlooking the specific challenges in multi-stage cascade ranking/filtering tasks, where the primary concern is the ranking order. In contrast to traditional UDA methods emphasizing distribution-invariance across domains, cross-stage transfer should pivot towards emphasizing order-invariance. Specifically, we formulate the order-invariance assumption as:

$$P_l(y|x_1) \leq P_l(y|x_2) \iff P_u(y|x_1) \leq P_u(y|x_2), \forall x_1, x_2\,, \quad (2)$$

which means even if the classification boundary of the task changes, the relative position relationship between x_2 and x_1 remains constant, with x_2 always preceding x_1.

Cross-stage Class Concept Conflict (C4) Traditional UDA approaches can not perform well on cross-stage transfer due to a specific key issue, Cross-stage Class Concept Conflict (C4), which has received insufficient attention in prior research. In cross-stage transfer, the concept (positive or negative) of the same sample in different stages is inconsistent. For instance, as illustrated in Fig. 1, the absence of a tumor but an unhealthy sample in the primary diagnosis stage is considered positive, whereas it is deemed negative in clinical diagnosis. This issue can be formulated as:

$$\exists x, P_u(y|x) \neq P_l(y|x)\,. \quad (3)$$

The distinct order-invariance assumption and the specific challenge posed by C4 hinder the effectiveness of traditional transfer learning in multi-stage cascade ranking and filtering systems. In the upcoming section, we will delve into detailed discussions on solutions for these issues.

3 Methodology

In this section, we introduce an innovative method named Unsupervised Rank Adaptation (URA) to address cross-stage transfer challenges. We focus more on the most basic issue, which is the difference in data distribution across different stages, verifying the effectiveness of essential problems. Illustrated in Fig. 2, cross-transfer can be bidirectionally employed between multiple stages, enabling each stage to share its strengths with the other. To exemplify, we utilize the knowledge transfer from the upper stage $\mathcal{D}_u = \{(x_i^u, y_i^u)\}_{i=0}^{n_u}$ to the lower stage $\mathcal{D}_l = \{x_i^l\}_{i=0}^{n_l}$ as an example for discussion.

As shown in Fig. 3, URA consists of two primary components to address the above issues: Order-Conditional Distribution Alignment (OCDA) for order-invariance and Principal Projection Alignment (PPA) for addressing the C4 problem. Specifically, OCDA aligns the distribution relationship of intra-stage samples through the residuals of features across stages, conditioned on the predicted order, which will be discussed in Sec. 3.1. Additionally, PPA is introduced to align the projection matrix of the principal component with classifier parameters, ensuring order-invariance and reducing reliance on guessing pseudo-labels, to mitigate the influence of C4. The detailed explanation can be found in Sec. 3.2. These two components collectively contribute to the effectiveness of URA in achieving efficient cross-stage transfer.

3.1 *Order-Conditional Distribution Alignment*

In contrast to classic UDA approaches that emphasize distribution invariance across domains, cross-stage transfer should prioritize order-invariance, placing greater emphasis on the order relationship among intra-domain samples. Therefore, a primary challenge in cross-stage transfer is to preserve order-invariance throughout the transfer process. This necessitates the effective characterization of the relative positional relationship between sample pairs.

According to the formulation of order-invariance in Eq. (2), and drawing inspiration from Area Under Curve (AUC) optimization, the main idea of order-conditional distribution alignment is to characterize the feature relationship based on the ranking (order) between pairwise samples. This process involves constructing a new feature

relationship distribution that is conditional on the pairwise order relationship. In any pairwise samples, we can characterize the relationship of their feature $z = F(x)$, extracted by the feature extractor $F : x \mapsto z$. Specifically, our intuitive idea is to construct a new pairwise feature relationship conditional on the order (denoted as $H.$) as:

$$H. = \{h_{ij} = F(x_i) - F(x_j) | \mathbb{I}(y_i > y_j), \forall x_i, x_j \in D.\} \quad (4)$$

where $D.$ represents a specific domain of either D_l or D_u.

However, two challenges arise in constructing this pairwise feature relationship: 1) the label y is unavailable in the target stage, and 2) the pairwise relationship will increase the data size from $\mathcal{O}(n)$ to $\mathcal{O}(n^2)$. In a distribution alignment strategy based on Maximum Mean Discrepancy (MMD) (e.g., DAN [21] and JAN [22]), the computational complexity becomes $\mathcal{O}(n^4)$, making it time and resource consumption. Hence, in practice, URA simplifies the pairwise relationship as the feature relationship between any feature z and the positive feature center z_o^+ and the negative feature center z_o^-. We term this the "order-conditional distribution" and provide a detailed formal discussion as follows.

Order-Conditional Distribution. Firstly, we need to calculate the center of positive z_o^+ and negative z_o^-.

Utilizing the predictions $\hat{y} = C(z) \in \mathbb{R}^{c \times c}$ from the shared classifier $C : z \mapsto y$, we determine these centers as follows:

$$z_o^+ = \mathbb{E}_x[F(x)\hat{P}(\hat{y} = 1|x)] = \frac{F(x) \cdot C(F(x))[:,0]}{\sum_x C(F(x))[:,0]}, \quad (5)$$

$$z_o^- = \mathbb{E}_x[F(x)\hat{P}(\hat{y} = 0|x)] = \frac{F(x) \cdot C(F(x))[:,1]}{\sum_x C(F(x))[:,1]}. \quad (6)$$

Subsequently, we model the relationship between the feature z and its corresponding center using their residuals. Thus, the order-conditional distribution of the lower stage (H_l) and the upper stage (H_u) can be defined as:

$$H_l = \{h_l = [z - z_o^+; z - z_o^-] \mid \forall x \in D_l, z = F(x)\} \quad (7)$$

$$H_u = \{h_u = [z - z_o^+; z - z_o^-] \mid \forall x \in D_u, z = F(x)\} \quad (8)$$

where $[*; *]$ means concat.

Having established the order-conditional distribution via feature residuals, the next step involves aligning the distributions of the upper and lower stages to facilitate cross-stage transfer while preserving order-invariance. And detailed explanation of this process is described in the following section.

The Transfer of Order-Conditional Distribution. To maintain the order-invariant of different stages in cross-stage transfer, we need to align the order-conditional distribution. Specifically, we utilize maximum mean discrepancy to facilitate alignment between the lower stage (H_l) and the upper stage (H_u). The loss function for Order-Conditional Distribution Alignment ($\mathcal{L}_{OCDA}$) is formally expressed as:

$$\begin{aligned}\mathcal{L}_{OCDA} =& \mathcal{MMD}(H_l, H_u) \\ =& \frac{1}{|H_l|^2} \sum_{i=1}^{|H_l|} \sum_{i=1}^{|H_l|} k(h_{l_i}, h_{l_j}) \\ &+ \frac{1}{|H_u|^2} \sum_{i=1}^{|H_u|} \sum_{i=1}^{|H_u|} k(h_{u_i}, h_{u_j}) \\ &- \frac{2}{|H_l||H_u|} \sum_{i=1}^{|H_l|} \sum_{i=1}^{|H_u|} k(h_{l_i}, h_{u_j}),\end{aligned} \quad (9)$$

where $k(\cdot, \cdot)$ means the Gaussian kernel function.

3.2 Principal Projection Alignment

In the above, we have introduced the details of order-conditional distribution alignment, which involves computing positive and negative sample centers based on $P(y|x)$ of upper-stage (i.e., unsupervised) samples. However, the predictions $P(y|x)$ of unsupervised samples may be arbitrary, potentially leading to error accumulation if a mistake is made in the prediction process. This issue is particularly catastrophic when encountering a C4 situation, where conflicts arise in class concepts across stages. For example, as shown in Fig. 1, "Benign Tumor" is labeled as negative in the upper stage "Tissue biopsy" but positive in the lower stage "Clinical Diagnosis". Consequently, the classifier trained in the upper supervised stage may make incorrect predictions in the lower unsupervised stage.

To address these challenges, we propose a novel strategy called PPA. We argue that *the sample scores exhibit a monotonic distribution along the direction of maximum variance.* The weight matrix W of the classifier projects the latent feature z into a 2-dimensional logit space. We can also employ Singular Value Decomposition (SVD) to project the latent feature z into a 2-dimensional subspace, preserving maximum variance. Instead of aligning directly in logit space, we align the projection matrix, which is equivalent and more convenient. Consequently, the projection matrix from SVD and the projection matrix from the classifier will be aligned in the same direction. This alignment ensures order-invariance without the need for guesswork regarding pseudo-labels, making it more robust when facing a C4 scenario. More details will be introduced as follows.

Initially, we utilize SVD to decompose the latent feature matrix Z of lower unsupervised stage samples:

$$Z = U\Sigma V^T. \quad (10)$$

Here $Z \in \mathbb{R}^{d \times n}$, where d represents the feature dimension, and n represents the number of samples.

It is assumed that the results of Eq. (10) are sorted by singular values from largest to smallest. Therefore, to project the latent features into a 2-dimensional space while preserving maximum variance, the projection matrix should be:

$$\widehat{V} = V[:2, :] \in \mathbb{R}^{2 \times n}. \quad (11)$$

The projection matrix from SVD denoted as $\widehat{V}$, and the projection matrix W from the classifier C should align in the same direction, which can be expressed as:

$$W = \lambda \cdot \widehat{V}, \quad \exists \lambda \in \mathbb{R}. \quad (12)$$

Since the result of SVD, V, is an orthogonal matrix, i.e., $VV^T = I$, where I denotes the identity matrix, it follows that $\widehat{V}\widehat{V}^T = I$. Thus:

$$W\widehat{V}^T = \lambda \cdot I \in \mathbb{R}^{2 \times 2}. \quad (13)$$

Building upon this foundation, we design a loss for Eq. (13). Let $M = W\widehat{V}^T \in \mathbb{R}^{2 \times 2}$, and the loss can be defined as:

$$\mathcal{L}_{PPA} = (M_{0,0} - M_{1,1})^2 + \frac{1}{2}(M_{0,1}^2 + M_{1,0}^2), \quad (14)$$

where $M_{i,j}$ denotes the element in the i-th row and the j-th column of M. In this loss equation, the first term aims to ensure that the elements along the diagonal of M ($M_{0,0}$ and $M_{1,1}$) are equal (i.e., both are unknown λ), while the second term constrains the elements outside the diagonal of M ($M_{0,1}$ and $M_{1,0}$) to be 0.

Algorithm 1 Unsupervised Rank Adaptation (URA)

Input: Upper stage dataset $\mathcal{D}_u = \{(x_i^u, y_i^u)\}_{i=0}^{n_u}$ and lower stage dataset $\mathcal{D}_l = \{x_i^l\}_{i=0}^{n_l}$, learning rate η and loss trade-offs α and β.

for iter = 0 to MaxIteration **do**
  x_u, y_u = *RandomSample*($\mathcal{D}_u$) // *Get labeled data*
  x_l = *RandomSample*($\mathcal{D}_l$) // *Get unlabeled data*
  $z_l = F(x_l), z_u = F(x_u)$ // *Extract features*
  $g_l = C(z_l), g_u = C(z_u)$ // *Get logits for classification*
  $H_u = OCDA_{Eq.(7,8)}(f_u, g_u)$,
  $H_l = OCDA_{Eq.(7,8)}(f_l, g_l)$
  $\mathcal{L}_{OCDA} = Eq._{(9)}(H_u, H_l)$ // *Calculate OCDA Loss*
  $\widehat{V}_l = Eq._{(9,10)}(z_l)$ // *Get principal projection*
  $W \leftarrow$ *get parameters form classifier* C
  $\mathcal{L}_{PPA} = Eq._{(14)}(\widehat{V}_l, W)$ // *Calculate PPA Loss*
  $L_{cls} = Eq._{(15)}(y_u, y_u)$
  $\mathcal{L}_{URA} = L_{CLS} + \alpha L_{OCDA} + \beta L_{PPA}$ // *Total Loss*
  $\theta \leftarrow \theta - \eta \frac{\partial \mathcal{L}_{URA}}{\partial \theta}$ // *Update parameters by BP*
end for

3.3 Overall Optimization Objective

In addition to the transfer losses for order-conditional distribution alignment ($\mathcal{L}_{OCDA}$) and principal projection alignment ($\mathcal{L}_{PPA}$), we require a classification loss to train the classifier using supervised data from the upper stage $\mathcal{D}_u = \{(x_i^u, y_i^u)\}_{i=0}^{n_u}$, where $y_i^u \in \{0, 1\}$. This loss is defined as:

$$\mathcal{L}_{CLS} = \frac{1}{n_u} \sum_{i=1}^{n_u} \ell_{ce}(C(F(x_i^u)), y_i^u)) \,, \tag{15}$$

where $\ell_{ce}(\cdot, \cdot)$ is the cross-entropy loss.

In conclusion of this section, we summarize the overall loss as:

$$\mathcal{L}_{URA} = \mathcal{L}_{CLS} + \alpha \mathcal{L}_{OCDA} + \beta \mathcal{L}_{PPA} \,. \tag{16}$$

The optimization objective is to minimize this combined loss, which includes the classification loss as well as the alignment losses weighted by the hyper-parameters α and β. Additionally, we provide the pseudo-code in Algorithm 1.

4 Empirical Studies

In this section, we preprocess and partition subsets of two standard image classification datasets, CIFAR-10 and CBIS-DDSM, by the specified cross-stage transfer problem setting. Subsequently, we conduct a comprehensive empirical study to showcase the effectiveness of our proposed URA algorithm. Further details will be provided in the following.

4.1 Experimental protocol

Datasets for Multi-stage Ranking System. To assess the effectiveness of our proposed URA method in cross-stage transfer scenarios, we extensively conducted experiments on two visual benchmark datasets: a simulated dataset based on CIFAR-10 and a real-world medical image dataset, CBIS-DDSM.

1. **Finding the cat in CIFAR-10:** The original CIFAR-10 dataset consists of 60,000 images divided into 10 classes, each containing 6,000 images. For simulating cross-stage transfer within a multi-stage cascade ranking system, we construct system and datasets to "find the cat": Cat is one of the class in CIFAR-10, and the goal of ranking multi-stage cascade ranking system is to find the cat image. Initially, We train a toy binary classifier using a 1/6 subset of CIFAR-10. We then employ it to split the remaining data into three parts: p_1 is the true negative samples of toy classifier, p_2 is the fake positive of toy classifier, p_3 is the actual cat samples. This allows us to establish a two-stage ranking/filtering system: Stage-1 filters $p_2 \cup p_3$ (considered as positive in this stage) from the entire dataset ($p_1 \cup p_2 \cup p_3$); Stage-2 filters the actual cat images (p_3) from $p_2 \cup p_3$; Additionally, Stage-H (Hyper) directly filters the actual cat images from the entire dataset. With these settings, we construct three mutually-exclusive datasets for each of these stages. Notably, the occurrence of C4 arises during cross-stage transfer between Stage-1 and the other stages.
2. **Diagnosing malignant tumor in CBIS-DDSM:** The CBIS-DDSM dataset is a standardized version of the Digital Database for Screening Mammography (DDSM), containing 2,620 scanned film mammography studies. It encompasses cases of normal, benign, and malignant tumors, all with validated pathological information. Similarly, we can establish a two-stage ranking system to "Diagnose malignant tumors". This will be a natural real-world application, where p_1 represents normal cases, p_2 represents benign tumors, and p_3 represents malignant tumors. Thus, Stage-1 filters all tumors (both benign and malignant) from the all patients; Stage-2 filters malignant tumors from patients with all tumors; Stage-H directly filters malignant tumors from the all patients.

Implementation details. We implement our proposed method, URA, with deep convolutional networks in Pytorch [29] based on the Transfer-Learning-Library [18]. For all the datasets we use, the backbone network is ResNet-50 [13] with parameters finetuned from the model pre-trained on ImageNet. Our experiments were all conducted on NVIDIA A100-PCIE-40GB. For all experiments, we train the models for 30 epochs, and set the batch size to 128. For optimization, we adopt the SGD optimizer to train models with the Nesterov momentum 0.9. The learning rate of the experiment defaults to 0.001. Subsequently, we perform comprehensive experiments to demonstrate the efficacy of URA.

Baselines. We compare URA with state-of-the-art deep unsupervised domain adaptation approaches, including Domain Adversarial Neural Network (DANN) [9], Joint Adaptation Networks (JAN) [22], Conditional Domain Adversarial Network (CDAN) [23], Cycle Self-Training (CST) [19], and the most recent and SOTA baseline in UDA, Invariant Consistency Learning (ICON) [40]. All baseline results are reproduced in the setting of unsupervised ranking adaptation using their official implementations.

Evaluation metrics. To measure the performance of the ranking system, we evaluate the proposed URA and baselines using Area Under Curve (AUC), Normalized Discounted Cumulative Gain (NDCG@K), and Average Precision (AP@K). These metrics are widely used in ranking tasks and are appropriate for evaluating the ability to maintain order-invariance in cross-stage transfer.

4.2 Comparison with state-of-the-art

Tables 1 and 2 show the results compared with the baselines on the simulated dataset based on CIFAR-10 and CBIS-DDSM, respectively. In these tables, we highlight the best and second-best results in bold and underlined. Additionally, all these results are averages obtained from running the experiments 5 times with different random seeds. The results reveal several insightful observations:

Table 1. Results on simulated dataset based on CIFAR-10 with ResNet-50 as backbone (AP and NDCG means AP@5000 and NDCG@5000).

Method	Stage 2 → Stage 1			Stage 1 → Stage 2			Stage 2 → Stage H			Stage H → Stage 2			Avg.		
	AUC	AP	NDCG	AUC	AP	NDCG	AUC	AP	NDCG	AUC	AP	NDCG	AUC	AP	NDCG
ResNet	.685	.512	.673	.706	.751	.621	.912	.675	.715	.684	.511	.669	.747	.612	.670
JAN	.598	.500	.553	.549	.556	.207	.802	.557	.300	.601	.502	.555	.638	.529	.404
DANN	.655	.515	.622	.532	.271	.226	.803	.269	.337	.653	.517	.612	.661	.393	.449
CDAN	.664	.566	.622	.622	.566	.483	.877	.504	.608	.665	.570	.620	.707	.552	.583
CST	.687	.532	.669	.763	.762	.644	.922	.648	.752	.790	.460	.792	.790	.601	.714
ICON	.585	.330	.527	.580	.706	.520	.895	.714	.730	.764	.622	.756	.706	.593	.633
URA	**.740**	**.610**	**.722**	**.800**	**.850**	**.759**	**.938**	**.778**	**.797**	**.798**	**.692**	**.794**	**.819**	**.732**	**.768**

Table 2. Results on CBIS-DDSM with ResNet-50 as backbone (AP and NDCG means AP@3000 and NDCG@3000).

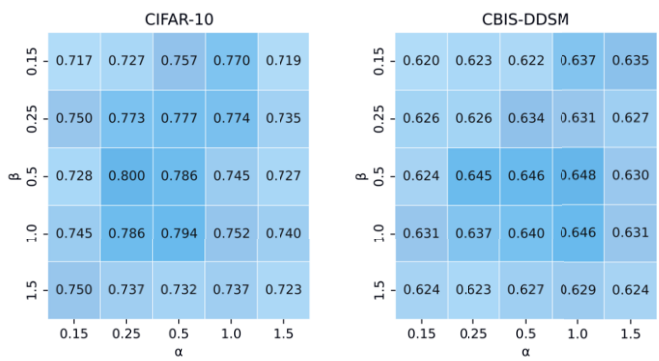
Method	Stage 2 → Stage 1			Stage 1 → Stage 2			Stage 2 → Stage H			Stage H → Stage 2			Avg.		
	AUC	AP	NDCG	AUC	AP	NDCG	AUC	AP	NDCG	AUC	AP	NDCG	AUC	AP	NDCG
ResNet	.612	.440	.672	.814	.952	.890	.653	.464	.720	.638	.837	.794	.679	.673	.769
JAN	.582	.398	.651	.828	.968	.885	.642	.448	.716	.671	.885	.807	.681	.675	.764
DANN	.603	.418	.669	.835	.963	.899	.646	.453	.715	.637	.845	.785	.680	.670	.767
CDAN	.566	.408	.631	.810	.959	.882	.649	.454	.730	.658	.880	.803	.671	.675	.762
CST	.547	.315	.535	.921	.994	.960	.680	.386	**.812**	.643	.850	.795	.698	.636	.776
ICON	.449	.299	.502	.783	.942	.871	.621	.475	.703	.611	.818	.776	.629	.634	.713
URA	**.648**	**.485**	**.690**	**.933**	**.996**	**.974**	**.698**	**.489**	.737	**.715**	**.892**	**.828**	**.764**	**.715**	**.807**

1. Our URA has achieved the best results in all tasks on simulated CIFAR-10 and has outperformed most tasks on CBIS-DDSM. Notably, URA significantly outperforms all compared approaches in average performance across all sub-tasks, including the most recent and SOTA baseline, ICON. Specifically, compared with the second-best results, **URA exhibited improvements of AUC +0.029, AP@5000 +0.131, NDCG@5000 +0.054 in simulated CIFAR-10, and AUC +0.057, AP@3000 +0.04, NDCG@3000 +0.022 in real-world CBIS-DDSM.** This demonstrates that the proposed URA can effectively handle order-invariance and C4, achieving superior performance in cross-stage transfer in multi-stage cascade ranking systems.
2. Compared with the source only without adaptation, many traditional UDA approaches exhibit negative transfer, deteriorating performance upon applying adaptation technologies. Specifically, in terms of average performance, **the most severe negative transfer: AUC -0.109, AP@5000 -0.219, NDCG@5000 -0.266 in simulated CIFAR-10, and AUC -0.05, AP@3000 -0.039, NDCG@3000 -0.056 in real-world CBIS-DDSM.** This underscores that traditional UDA approaches, designed for classification-like tasks, are unsuitable for cross-stage transfer in multi-stage cascade ranking systems. Consequently, addressing order-invariance and C4 is crucial in this specific scenario.

4.3 Ablation Study

The proposed URA consists of two primary components, OCDA and PPA. We conducted ablation studies to assess the individual contributions of these components. The results of these ablations on simulated CIFAR-10 and CBIS-DDSM are reported in Tables 3 and 4. By comparing the average results, two assertions can be made:

1. OCDA and PPA are both beneficial and do not induce negative transfer compared to the source-only baseline.
2. The effects of OCDA and PPA are not conflicting, combining them leads to better performance than employing either one alone.



(a) Results on CIFAR-10. (b) Results on CBIS-DDSM.

Figure 4. The results of hyper-parameter sensitivity of performance (AUC) on two datasets CIFAR-10 and CBIS-DDSM. The X-axis is the hyper-parameter α and the Y-axis is the hyper-parameter β.

4.4 Hyper-Parameter Sensitivity

In the overall optimization loss, Eq. (16), there are two trade-offs involving the weights of OCDA (α) and PPA (β). We employ the grid search technique in our implementation to determine the optimal values for the hyper-parameters α and β, exploring a range of values including $\{0.15, 0.25, 0.5, 1.0, 1.5\}$.

The visualization in Fig. 4 illustrates the impact of different α and β values on AUC for the first task, i.e., Stage 2 → Stage 1. This visualization demonstrates that all variants with positive α and β surpass the source-only baseline, confirming that OCDA and PPA consistently benefit cross-stage transfer in multi-stage cascade ranking systems, and URA is not sensitive to hyper-parameters.

5 Related work

Cascade Ranking. Multi-stage cascade ranking has gained traction as a promising strategy for balancing the efficiency and effec-

Table 3. Ablation results on simulated dataset based on CIFAR-10 with ResNet-50 as backbone (AP and NDCG mean AP@5000 and NDCG@5000).

Method	Stage 2 → Stage 1			Stage 1 → Stage 2			Stage 2 → Stage H			Stage H → Stage 2			Avg.		
	AUC	AP	NDCG	AUC	AP	NDCG	AUC	AP	NDCG	AUC	AP	NDCG	AUC	AP	NDCG
ResNet	.706	.751	.621	.685	.512	.673	.912	.675	.715	.684	.511	.669	.747	.612	.670
OCDA	.778	**.879**	.747	.714	.566	.690	.921	.664	.701	.790	.664	.780	.801	.693	.730
PPA	.771	.791	.677	.728	.589	.708	.936	.776	.793	.794	.689	.787	.807	.711	.741
URA	**.800**	.850	**.759**	**.740**	**.610**	**.722**	**.938**	**.778**	**.797**	**.798**	**.692**	**.794**	**.819**	**.732**	**.768**

Table 4. Ablation results on CBIS-DDSM with ResNet-50 as backbone (AP and NDCG mean AP@3000 and NDCG@3000).

Method	Stage 2 → Stage 1			Stage 1 → Stage 2			Stage 2 → Stage H			Stage H → Stage 2			Avg.		
	AUC	AP	NDCG	AUC	AP	NDCG	AUC	AP	NDCG	AUC	AP	NDCG	AUC	AP	NDCG
ResNet	.612	.440	.672	.814	.952	.890	.653	.464	.720	.638	.837	.794	.679	.673	.769
OCDA	**.648**	.467	**.703**	.927	.980	.935	.678	.368	**.782**	.683	**.906**	.809	.734	.680	**.807**
PPA	.626	.478	.671	.895	.971	.919	.666	.485	.731	.710	.886	.814	.724	.705	.784
URA	**.648**	**.485**	.690	**.933**	**.996**	**.974**	**.698**	**.489**	.737	**.715**	.892	**.828**	**.763**	**.715**	**.807**

tiveness of ranking systems [35], employing distinct models for each stage. Recent research has witnessed a shift towards cascaded ranking models grounded in deep learning. For instance, Gallagher et al. [8] delved into gradient derivation for cascaded classifiers, optimizing them through end-to-end methods, while Fan et al. [5] proposed integrating multiple stages using hard negative sampling. Furthermore, some researchers have noticed that considering different stages together can improve the performance of the system more than optimizing each stage individually. Fei et al. [6] advocated for feature sharing across different stages, and Hron et al. [16] suggested the joint utilization of multiple models during the recall phase, learning to aggregate recall items from diverse channels.

Despite these advancements, these cascading ranking methods still do not fully consider how to utilize the different characteristics of different stages to improve the overall effect of the system.

These approaches failed to recognize the potential for mutual knowledge transfer across stages, which can leverage their respective strengths and ensure consistency. We argue that cross-stage transfer can potentially enhance the overall performance of the multi-stage cascade ranking and filtering system.

Unsupervised Domain Adaptation. UDA is an important subfield of transfer learning that aims to overcome the challenges posed by different data distributions between labeled source and unlabeled target domains. The primary objective of UDA is to leverage knowledge gained from labeled source domains to enhance model performance in the target domain, thereby promoting more accurate predictions and improving overall decision-making capabilities.

Traditional UDA research predominantly focuses on tasks such as classification and semantic segmentation, primarily employing two core techniques: moment matching and adversarial confusion. Moment matching involves quantifying distributional differences using statistical moments, such as the Maximum Mean Discrepancy [12, 21], and its various variations [34, 22, 11]. Furthermore, this method extends its scope to include thoughtful consideration of batch means and variances [36], contributing to a comprehensive understanding of distribution alignment. The second technique, adversarial confusion, utilizes an adversarial training paradigm, in which the domain discriminator is intricately trained alongside the feature extractor to achieve source and target alignment by confusing cross-domain distribution. As a landmark research, Ganin et al. [10] first introduced adversarial learning strategies into deep learning-based domain adaptation. These methods proposed therein have a GAN-like architecture, bringing new possibilities for improving DA using GAN-based techniques [23, 15, 20, 25]. Recently, the research problems in UDA have expanded to include various problem settings, such as [32, 39, 30, 27] and so on.

Despite the remarkable success of these UDA approaches, they always exhibit poor performance in some certain real-world scenarios, such as cross-stage transfer in multi-stage cascade ranking and filtering systems, without considering the characteristics of the ranking task. Cross-stage transfer emphasizes order-invariance rather than distribution-invariance. Additionally, the unaligned classification boundary between different stages may cause the same sample in different stages to obtain different labels, termed C4. Our proposed URA emerges as a notable solution proficient in addressing and resolving the challenges posed by this distinctive problem.

6 Conclusion

In this paper, we first introduce a new real-world transfer learning scenario: cross-stage transfer within multi-stage cascade ranking and filtering systems, which has not been extensively explored in traditional transfer learning methods. We focus on the critical assumption of order-invariance in this problem and address a key issue called C4, which highlights potential inconsistencies in class concepts for the same sample at different stages. To overcome these challenges, we propose a novel method called URA, consisting of two main components: OCDA characterizes the intra-stage order-conditional distribution and aligns them across stages, and PPA aligns the projection matrix of the principal component with classifier parameters to ensure order-invariance without relying on pseudo-labels, thereby mitigating the influence of C4. Experimental results demonstrate the effectiveness of our approach in various cross-stage transfer tasks.

Cross-stage transfer is a vital area within transfer learning, identified and tentatively explored in this paper for the first time. Nonetheless, lots of new research challenges still warrant further investigation in future studies. For example, challenges include improving the efficiency of cross-stage transfer, addressing online test-time cross-stage transfer, and jointly facilitating cross-stage transfer across more stages simultaneously. Overcoming these challenges will help advance the field of transfer learning.

Acknowledgements

This work is supported by the Shenzhen Science and Technology Program (No.JCYJ20230807120800001) and the 2023 Shenzhen Sustainable Supporting Funds for Colleges and Universities (No.20231121165240001). We also acknowledge support from China Unicom Shenzhen's Intelligent Computing Center.

References

[1] C. Abeysinghe, C. Reid, H. Rezatofighi, and B. Meyer. Tracking different ant species: An unsupervised domain adaptation framework and a dataset for multi-object tracking. In *International Joint Conference on Artificial Intelligence*, 2023.

[2] K. Bousmalis, N. Silberman, D. Dohan, D. Erhan, and D. Krishnan. Unsupervised pixel-level domain adaptation with generative adversarial networks. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 3722–3731, 2017.

[3] J. Chen, H. Dong, X. Wang, F. Feng, M. Wang, and X. He. Bias and debias in recommender system: A survey and future directions. *ACM Transactions on Information Systems*, 41(3), 2023.

[4] C. Dong, X. Kang, and A. Ming. Icda: Illumination-coupled domain adaptation framework for unsupervised nighttime semantic segmentation. In *International Joint Conference on Artificial Intelligence*, pages 672–680, 2023.

[5] M. Fan, J. Guo, S. Zhu, S. Miao, M. Sun, and P. Li. MOBIUS: towards the next generation of query-ad matching in baidu's sponsored search. In *ACM SIGKDD International Conference on Knowledge Discovery and Data Mining*, pages 2509–2517, 2019.

[6] H. Fei, J. Zhang, X. Zhou, J. Zhao, X. Qi, and P. Li. Gemnn: Gating-enhanced multi-task neural networks with feature interaction learning for ctr prediction. In *Proceedings of the International ACM SIGIR Conference on Research and Development in Information Retrieval*, page 2166âĂŞ2171, 2021.

[7] W. Feng, L. Ju, L. Wang, K. Song, X. Zhao, and Z. Ge. Unsupervised domain adaptation for medical image segmentation by selective entropy constraints and adaptive semantic alignment. In *AAAI Conference on Artificial Intelligence*, pages 623–631, 2023.

[8] L. Gallagher, R. Chen, R. Blanco, and J. S. Culpepper. Joint optimization of cascade ranking models. In *Proceedings of the International Conference on Web Search and Data Mining (WSDM)*, pages 15–23, 2019.

[9] Y. Ganin and V. S. Lempitsky. Unsupervised domain adaptation by backpropagation. In *International Conference on Machine Learning*, pages 1180–1189, 2015.

[10] Y. Ganin, E. Ustinova, H. Ajakan, P. Germain, H. Larochelle, F. Laviolette, M. Marchand, and V. Lempitsky. Domain-adversarial training of neural networks. *Journal of Machine Learning Research*, 17(1):2096–2030, 2016.

[11] P. Ge, C.-X. Ren, X.-L. Xu, and H. Yan. Unsupervised domain adaptation via deep conditional adaptation network. *Pattern Recognition*, 134: 109088, 2023.

[12] A. Gretton, K. M. Borgwardt, M. J. Rasch, B. Schölkopf, and A. Smola. A kernel two-sample test. *Journal of Machine Learning Research*, 13: 723–773, 2012.

[13] K. He, X. Zhang, S. Ren, and J. Sun. Deep residual learning for image recognition. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 770–778, 2016.

[14] Q. He, S. Xiao, M. Ye, X. Zhu, F. Neri, and D. Hou. Independent feature decomposition and instance alignment for unsupervised domain adaptation. In *International Joint Conference on Artificial Intelligence*, pages 819–827, 2023.

[15] J. Hoffman, E. Tzeng, T. Park, J.-Y. Zhu, P. Isola, K. Saenko, A. Efros, and T. Darrell. Cycada: Cycle-consistent adversarial domain adaptation. In *International Conference on Machine Learning*, pages 1989–1998. PMLR, 2018.

[16] J. Hron, K. Krauth, M. Jordan, and N. Kilbertus. On component interactions in two-stage recommender systems. In *Advances in Neural Information Processing Systems*, pages 2744–2757, 2021.

[17] F. Huang, Z. Yao, and W. Zhou. Dtbs: Dual-teacher bi-directional self-training for domain adaptation in nighttime semantic segmentation. *Frontiers in Artificial Intelligence and Applications*, 372:1084, 2023.

[18] J. Jiang, B. Chen, B. Fu, and M. Long. Transfer-learning-library. https://github.com/thuml/Transfer-Learning-Library, 2020.

[19] H. Liu, J. Wang, and M. Long. Cycle self-training for domain adaptation. In *Advances in Neural Information Processing Systems*, 2021.

[20] Y. Liu, W. Zhang, and J. Wang. Source-free domain adaptation for semantic segmentation. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 1215–1224, 2021.

[21] M. Long, Y. Cao, J. Wang, and M. I. Jordan. Learning transferable features with deep adaptation networks. In *International Conference on Machine Learning*, pages 97–105, 2015.

[22] M. Long, H. Zhu, J. Wang, and M. I. Jordan. Deep transfer learning with joint adaptation networks. In *International Conference on Machine Learning*, pages 2208–2217, 2017.

[23] M. Long, Z. Cao, J. Wang, and M. I. Jordan. Conditional adversarial domain adaptation. In *Advances in Neural Information Processing Systems*, pages 1647–1657, 2018.

[24] M. Lu, Z. Huang, Z. Tian, Y. Zhao, X. Fei, and D. Li. Meta-tsallis-entropy minimization: A new self-training approach for domain adaptation on text classification. In *International Joint Conference on Artificial Intelligence*, pages 5159–5169, 2023.

[25] Y. Ma, Y. Chen, H. Yu, Y. Gu, S. Wen, and S. Guo. Letting go of self-domain awareness: Multi-source domain-adversarial generalization via dynamic domain-weighted contrastive transfer learning. In *European Conference on Artificial Intelligence*, pages 1664–1671, 2023.

[26] K. Mei, C. Zhu, J. Zou, and S. Zhang. Instance adaptive self-training for unsupervised domain adaptation. In *European Conference on Computer Vision*, pages 415–430, 2020.

[27] E. F. Montesuma, F. M. N. Mboula, and A. Souloumiac. Multi-source domain adaptation through dataset dictionary learning in wasserstein space. In *European Conference on Artificial Intelligence*, pages 1739–1746, 2023.

[28] S. J. Pan and Q. Yang. A survey on transfer learning. *IEEE Transactions on Knowledge and Data Engineering*, 22(10):1345–1359, 2010.

[29] A. Paszke, S. Gross, F. Massa, A. Lerer, J. Bradbury, G. Chanan, T. Killeen, Z. Lin, N. Gimelshein, L. Antiga, et al. Pytorch: An imperative style, high-performance deep learning library. In *Advances in Neural Information Processing Systems*, volume 32, 2019.

[30] Q. Peng, Z. Ding, L. Lyu, L. Sun, and C. Chen. Rain: Regularization on input and network for black-box domain adaptation. In *International Joint Conference on Artificial Intelligence*, pages 4118–4126, 2023.

[31] P. O. Pinheiro. Unsupervised domain adaptation with similarity learning. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 8004–8013, 2018.

[32] Z. Qiao, X. Luo, M. Xiao, H. Dong, Y. Zhou, and H. Xiong. Semi-supervised domain adaptation in graph transfer learning. In *International Joint Conference on Artificial Intelligence*, pages 2279–2287, 2023.

[33] H. Sun and M. Li. Enhancing unsupervised domain adaptation by exploiting the conceptual consistency of multiple self-supervised tasks. *Science China Information Sciences*, 66(4):142101, 2023.

[34] E. Tzeng, J. Hoffman, N. Zhang, K. Saenko, and T. Darrell. Deep domain confusion: Maximizing for domain invariance. *arXiv preprint arXiv:1412.3474*, 2014.

[35] L. Wang, J. Lin, and D. Metzler. A cascade ranking model for efficient ranked retrieval. In *Proceedings of the international ACM SIGIR conference on Research and development in Information Retrieval*, pages 105–114, 2011.

[36] X. Wang, Y. Jin, M. Long, J. Wang, and M. I. Jordan. Transferable normalization: Towards improving transferability of deep neural networks. In *Advances in Neural Information Processing Systems*, 2019.

[37] P. Wei, L. Kong, X. Qu, Y. Ren, Z. Xu, J. Jiang, and X. Yin. Unsupervised video domain adaptation for action recognition: A disentanglement perspective. In *Advances in Neural Information Processing Systems*, 2023.

[38] J. Yang, S. Shi, Z. Wang, H. Li, and X. Qi. St3d: Self-training for unsupervised domain adaptation on 3d object detection. In *IEEE Conference on Computer Vision and Pattern Recognition*, pages 10368–10378, 2021.

[39] X. Yang, Y. Gu, K. Wei, and C. Deng. Exploring safety supervision for continual test-time domain adaptation. In *International Joint Conference on Artificial Intelligence*, pages 1649–1657, 2023.

[40] Z. Yue, Q. Sun, and H. Zhang. Make the u in UDA matter: Invariant consistency learning for unsupervised domain adaptation. In *Advances in Neural Information Processing Systems*, 2023.

[41] Y. Zhang, J. Lin, K. Chen, Z. Xu, Y. Wang, and K. Jia. Manifold-aware self-training for unsupervised domain adaptation on regressing 6d object pose. In *International Joint Conference on Artificial Intelligence*, pages 1740–1748, 2023.

[42] Y. Zou, Z. Yu, B. V. Kumar, and J. Wang. Unsupervised domain adaptation for semantic segmentation via class-balanced self-training. In *European Conference on Computer Vision*, 2018.