Transporting Causal Mechanisms for Unsupervised Domain Adaptation

Zhongqi Yue\textsuperscript{1,3}  Qianru Sun\textsuperscript{2}  Xian-Sheng Hua\textsuperscript{3}  Hanwang Zhang\textsuperscript{1}
\textsuperscript{1}Nanyang Technological University  \textsuperscript{2}Singapore Management University  \textsuperscript{3}Damo Academy, Alibaba Group
yuez0003@ntu.edu.sg  qianrusun@smu.edu.sg
xiansheng.hxs@alibaba-inc.com  hanwangzhang@ntu.edu.sg

Abstract

Existing Unsupervised Domain Adaptation (UDA) literature adopts the covariate shift and conditional shift assumptions, which essentially encourage models to learn common features across domains. However, due to the lack of supervision in the target domain, they suffer from the semantic loss: the feature will inevitably lose non-discriminative semantics in source domain, which is however discriminative in target domain. We use a causal view—transportability theory [40]—to identify that such loss is in fact a confounding effect, which can only be removed by causal intervention. However, the theoretical solution provided by transportability is far from practical for UDA, because it requires the stratification and representation of the unobserved confounder that is the cause of the domain gap. To this end, we propose a practical solution: Transporting Causal Mechanisms (TCM), to identify the confounder stratum and representations by using the domain-invariant disentangled causal mechanisms, which are discovered in an unsupervised fashion. Our TCM is both theoretically and empirically grounded. Extensive experiments show that TCM achieves state-of-the-art performance on three challenging UDA benchmarks: ImageCLEF-DA, Office-Home, and VisDA-2017. Codes are available at https://github.com/yue-zhongqi/tcm.

1. Introduction

Machine learning is always challenged when transporting training knowledge to testing deployment, which is generally known as Domain Adaptation (DA) [37]. As shown in Figure 1, when the target domain \((S = t)\) is drastically different (e.g., image style) from the source domain \((S = s)\), deploying a classifier trained in \(S = s\) results in poor performance due to the large domain gap [5]. To narrow the gap, conventional supervised DA requires a small set of labeled data in \(S = t\) [13, 45], which is expensive and sometimes impractical. Therefore, we are interested in a more practical setting: Unsupervised DA (UDA), where we can leverage the abundant unlabelled data in \(S = t\) [37, 29]. For example, when adapting an autopilot system trained in one country to another, where the street views and road signs are different, one can easily collect unlabelled street images from a camera-equipped vehicle cruise.

Existing UDA literature widely adopts the following two assumptions on the domain gap\textsuperscript{1}: 1) Covariate Shift [50, 4]: \(P(X|S = s) \neq P(X|S = t)\), where \(X\) denotes the samples, e.g., real-world vs. clip-art images; and 2) Conditional Shift [49, 31]: \(P(Y|X, S = s) \neq P(Y|X, S = t)\), where \(Y\) denotes the labels, e.g., in clip-art domain, the “pure background” feature extracted from \(X\) is no longer a strong visual cue for \(Y = “speaker”\) as in real-world domain. To turn “\(\neq\)” into “\(=\)” almost all existing UDA solutions rely on learning invariant (or common) features in both source and target domains [50, 29, 20]. Unfortunately, due to the lack of supervision in target domain, it is challenging to capture such domain invariance.

For example, in Figure 1, when \(S = s\), the distribution of “Background = pure” and “Background = cluttered” is already informative for “speaker” and “bed”, a classifier may recklessly tend to downplay the “Shape” features (or attributes) as they are not as “invariant” as “Background”; but, it does not generalize to \(S = t\), where all clip-art images

\textsuperscript{1}A few early works [57] also consider target shift of \(P(Y)\), which is now mainly discussed in other settings like long-tailed classification [27].
have “Background = pure” that is no longer discriminative. A popular approach to make up for the above loss of “Shape” is by imposing unsupervised reconstruction [6]—any feature loss hurts reconstruction. However, it is well-known that discrimination and reconstruction are adversarial objectives, leaving the finding of their ever-elusive trade-off parameter per se an open problem [9, 16, 23].

To systematically understand how domain gap causes the feature loss, we propose to use the transportability theory [40] to replace the above “shift” assumptions that overlook the explicit role of semantic attributes. Before we introduce the theory, we first abstract the DA problem in Figure 1 into the causal graph in Figure 2. We assume that the classification task $X \rightarrow Y$ is also affected by the unobserved semantic feature $U$ (e.g., shape and background), where $U \rightarrow X$ denotes the generation of pixel-level image samples $X$ and $U \rightarrow Y$ denotes the definition process of semantic class. Note that these causalities have been already shown valid in their respective areas, e.g., $U \rightarrow X$ and $U \rightarrow Y$ can be image [17] and language generation [7], respectively. In particular, the introduction of the domain selection variable $S$ reveals that the fundamental reasons of the covariate shift and conditional shift are both due to $P(U|S = s) \neq P(U|S = t)$.

Note that the domain-aware $U$ is the confounder that prevents the model from learning the domain-invariant causality $X \rightarrow Y$. It has been theoretically proven that the confounding effect cannot be eliminated by statistical learning without causal intervention [39]. Fortunately, the transportability theory offers a principled DA solution based on the causal intervention in Figure 2:

$$P(Y|do(X), S) = \sum_u P(Y|X, U = u)P(U = u|S).$$

(1)

Note that the goal of DA is achieved by causal intervention $P(Y|do(X), S)$ using the $do$-operator [41]. To appreciate the merits of the calculus on the right-hand side of Eq. (1), we need to understand the following two points. First, $P(Y|X, U)$ is domain-agnostic as $S$ is separated from $X$ and $Y$ given $U = u$ [39], thus $P(Y|X, U)$ generalizes to $S = t$ in testing even if it is trained on $S = s$. Second, every stratum of $U$ is fairly adjusted subject to the domain prior $P(U|S)$, i.e., forcing the model to respect “shape” as it is the only “invariance” that distinguishes between “speaker” and “bed” in controlled “cluttered” or “pure” backgrounds. Therefore, the semantic loss is eliminated.

However, Eq. (1) is purely declarative and far from practical, because it is still unknown how to implement the stratification and representation of the unobserved $U$, not mentioning to transport it across domains. To this end, we propose a novel approach to make Eq. (1) practical in UDA:

**Identify the number of $U$ by disentangled causal mechanisms.** In practice, the number of $U$ can be large due to the combinations of multiple attributes, making Eq. (1) computationally prohibitive (e.g., many network passes). Fortunately, if the attributes are disentangled, we can stratify $U$ according to the much smaller number of disentangled attributes, as the effect of each feature is independent with each other [19], e.g., $P(Y|X, U = (u_1, u_2)) \propto P(Y|X, u_1) + P(Y|X, u_2)$ if $u_1$ and $u_2$ are disentangled. As detailed in Section 3.1, this motivates us to discover a small number of Disentangled Causal Mechanisms (DCMs) in unsupervised fashion [38], each of which corresponds to a feature-specific intervention between $X|S = s$ and $X|S = t$, e.g., mapping a real-world “speaker” image to its clip-part counterpart by changing “Background” from “cluttered” to “pure”.

**Represent $U$ by proxy variables.** Yet, DCMs do not provide vector representations of $U$, leaving difficulties in implementing Eq. (1) as neural networks. In Section 3.2, we show that the transformed output $\hat{X}$ of each DCM taking $X$ as input can be viewed as a proxy of $U$ [35], which provides a theoretical guarantee that we can replace the unobserved $U$ with the observed $\hat{X}$ to make Eq. (1) computable.

Overall, instead of transporting the abstract and unobserved $U$ in the original transportability theory of Eq. (1), we transport the concrete and learnable disentangled causal mechanisms, which generate the observed proxy of $U$. Therefore, we term the approach as **Transporting Casual Mechanisms (TCM)** for UDA. Through extensive experiments, our approach achieves state-of-the-art performance on UDA benchmarks: 70.7% on Office-Home [53], 90.5% on ImageCLEF-DA [1] and 75.8% on VisDA-2017 [43]. Specifically, we show that learning disentangled mechanisms and capturing the causal effect through proxy variables are the keys to improve the performance, which validates the effectiveness of our approach.

2. Related Work

Existing UDA works fall into the three categories: 1) **Sample Reweighing** [50, 10]. This line-up adopts the covariate shift assumption. It first models $P(X|S = s)$ and $P(X|S = t)$. When minimizing the classification loss, each sample in $S = s$ is assigned an importance weight $P(X|S = t)/P(X|S = s)$, i.e., a target-like sample has a larger weight, which effectively encourages $P(X|S = s) = P(X|S = t)$. 2) **Domain Mapping** [20, 36]. This approach focuses on the conditional shift assumption. It first learns a mapping function that transforms samples from $S = s$ to $S = t$ through unpaired image-to-image transla-
tion techniques such as CycleGAN [63]. Then, the transformed source domain samples are used to train a classifier.

3) Invariant Feature Learning. This is the most popular approach in recent literature [15, 30, 55]. It maps the samples in \( S = s \) and \( S = t \) to a common feature space, denoted as domain \( S = c \), where the source and target domain samples are indistinguishable. Some works [37, 29] adopted the covariate shift assumption, and aim to minimize the differences in \( P(X|S = c) \) using distance measures like Maximum Mean Discrepancy [52, 32] or using adversarial training to fool a domain discriminator [15, 6]. Others used the conditional shift assumption and aim to align \( P(Y|X,S = c) \) [31, 30]. As the target domain samples have no labels, their \( P(Y|X,S = c) \) is either estimated through clustering [21, 58] or using a classifier trained with the source domain samples [46, 60].

They all aim to make the source and target domain alike while learning a classifier with the labelled data in the source domain, leading to the semantic loss. In fact, there are existing works that attempt to Alleviate Semantic Loss by learning to capture \( U \) in an unsupervised fashion: One line exploits that \( U \) generates \( X \) (via \( U \rightarrow X \)) and learns a latent variable to reconstruct \( X \) [6, 16]. The other line [8, 11] exploits \( S \rightarrow U \), and aims to learn a domain-specific representation for each \( S = s, S = t \). However, this leads to an ever-illusive trade-off between the adversarial discrimination and reconstruction objectives. Our approach aims to fundamentally eliminate the semantic loss by providing a practical implementation of the transportability theory [40] based on causal intervention [41].

3. Approach

Though Eq. (1) provides a principled solution for DA, it is impractical due to the unobserved \( U \). To this end, the proposed TCM is a two-stage approach to make it practical. The first stage identifies the \( U \) stratification by discovering Disentangled Causal Mechanisms (DCMs) (Section 3.1) and the second stage represents each stratification of \( U \) with the proxy variable generated from the discovered DCMs (Section 3.2). The training and inference procedures are summarized in Algorithm 1 and 2, respectively.

3.1. Disentangled Causal Mechanisms Discovery

Physicists believe that real-world observations are the outcome of the combination of independent physical laws. In causal inference [19, 51], we denote these laws as disentangled generative factors, such as shape, color, and position. For example, in our causal graph of Figure 2, if the semantic attribute \( U \) can be disentangled into generative causal factors (\( U_1, \ldots, U_k \)), each one will independently contribute to the observation: \( P(Y|X) \propto \sum_{i=1}^{k} P(Y|X, U_i) \). Even though \( |U| \) is large, \( i.e. \), the number of sum in Eq. (1) is expensive, the disentanglement can still significantly reduce the number to a small \( k \).

However, learning a disentangled representation of \( U = (U_1, \ldots, U_k) \) without supervision is challenging [28]. Fortunately, we can observe the outcomes of \( U \): the source domain samples \( X_s \) and the target domain samples \( X_t \) gen-

Algorithm 1 Two-stage Training of TCM

1: Input: Labelled domain \( S = s \), unlabelled domain \( S = t \), pre-trained backbone parameters \( \beta \)
2: Output: DCMs \( \{(M_i, M_i^{-1})\}_{i=1}^{k} \), fine-tuned backbone parameters \( \beta \) and linear functions parameters \( \omega \)
3: Randomly initialize \( \{(M_i, M_i^{-1})\}_{i=1}^{k} \), \( \omega \), VAE parameters \( \theta \), discriminator parameters \( \gamma \) (see Eq. (8))
4: repeat
5: // See Section 3.1 for details
6: Sample \( x \) randomly from \( S = s \) or \( S = t \)
7: Calculate \( \mathcal{L}_{\text{CycleGAN}}^i \) for \( i \in \{1, \ldots, k\} \)
8: Update \( \{(M_i, M_i^{-1})\}_{i=1}^{k} \) with Eq. (2)
9: until convergence
10: repeat
11: // See Section 3.2 for details
12: Sample \( x_s, y_s \) from \( S = s \), \( x_t \) from \( S = t \)
13: Obtain \( X_s = \{M_i(x_s)\}_{i=1}^{k}, X_t = \{M_i^{-1}(x_t)\}_{i=1}^{k} \)
14: Calculate \( h_y(x_s, x_t), \forall x_s \in \hat{X}_s \) with Eq. (7)
15: Obtain \( P(Y|\text{do}(X = x_s), S = s) \) with Eq. (5)
16: Update \( \omega, \beta, \gamma \) with Eq. (8)
17: until convergence

Algorithm 2 Inference with TCM

1: Input: \( x_t, P(X|S = t) \), DCMs \( \{M_i^{-1}\}_{i=1}^{k} \), backbone parameters \( \beta \) and linear functions parameters \( \omega \)
2: Output: Predicted label \( y_t \)
3: Obtain \( \hat{X}_t = \{M_i^{-1}(x_t)\}_{i=1}^{k} \)
4: Calculate \( h_y(x_t, \hat{x}), \forall \hat{x} \in \hat{X}_t \) with Eq. (7)
5: \( y_t \leftarrow \arg\max P(Y|\text{do}(X = x_t), S = t) \) with Eq. (5)

Figure 3: (a) Our DCMs \( \{(M_i, M_i^{-1})\}_{i=1}^{k} \), where each \( M_i : X_s \rightarrow X_t \) and \( M_i^{-1} : X_t \rightarrow X_s \) correspond to intervention on the disentangled attribute \( U_i \). (b) CycleGAN [63] loss \( \mathcal{L}_{\text{CycleGAN}} \) on each \( (M_i, M_i^{-1}) \).
generated from $P(X|U)$. This motivate us to discover $k$ pairs of end-to-end functions $\{(M_i, M_i^{-1})\}_{i=1}^k$ in unsupervised fashion, where $M_i : X_s \rightarrow X_t$ and $M_i^{-1} : X_t \rightarrow X_s$, as shown in Figure 3 (a). Each $(M_i, M_i^{-1})$ corresponds to a counterfactual mapping that transforms a sample to the counterpart domain by intervening a disentangled factor $U_i$, i.e., modifying $U_i$ w.r.t. the domain shift while fixing the values of other factors $U_j\neq i$—this is essentially the definition of independent causal mechanisms [38], each of which corresponds to a disentangled causal factor. Hence we refer to these mapping functions as DCMs.

We first make a strong assumption that the one-to-one correspondence between $(M_i, M_i^{-1})$ and $U_i$ has been already established, then we relax this assumption by removing the correspondence for the proposed practical solution. Without loss of generality, we consider a source domain sample $x_s$. For each $M_i : X_s \rightarrow X_t$, we obtain a transformed sample $M_i(x_s)$, corresponding to the interventional outcome of $U_i$ drawn from $P(U_i|S = s)$ to $P(U_i|S = t)$. To ensure that the interventions are disentangled, we use the Counterfactual Faithfulness theorem [3, 56], which guarantees that $M_i$ is a disentangled intervention if and only if $M_i(x_s) \sim P(X_t)$ (proof in Appendix). Hence, as shown in Figure 3 (b), we use the CycleGAN loss [63] denoted as $L_{CycleGAN}^i$ to make the transformed samples from each $M_i : X_s \rightarrow X_t$ indistinguishable from the real samples in $S = t$. Likewise, we apply the CycleGAN loss to each $M_i^{-1} : X_t \rightarrow X_s$ such that the transformed samples are close to real samples in $S = s$.

Now we relax the above one-to-one correspondence assumption. We begin with a sufficient condition (proof in Appendix): if $(M_i, M_i^{-1})$ intervenes $U_i$, the $i$-th mapping function outputs the counterfactual faithful generation, i.e., $i = \arg\min_j \{ 1, \ldots, k \} L_{CycleGAN}^j$. To “guess” the one-to-one correspondence, we adopt a practical method by using the necessary conclusion: if $i = \arg\min_j \{ 1, \ldots, k \} L_{CycleGAN}^j$, then $(M_i, M_i^{-1})$ corresponds to $U_i$ intervention. Specifically, training samples are fed into all $(M_i, M_i^{-1})$ in parallel to compute $L_{CycleGAN}$ for each pair. Only the winning pair with the smallest loss is updated. The objective is given by:

$$\min_{(M_i, M_i^{-1})} L_{CycleGAN}^i,$$

where $i = \arg\min_j L_{CycleGAN}^j$.

The functions (e.g., $M_i$) in the optimization objective denote their parameters for simplicity. Note that this is not sufficient, hence our approach has limitations. Still, this practice has been empirically justified in [38], and we leave a more theoretically complete approach as future work.

After training, we obtain $k$ DCMs $\{(M_i, M_i^{-1})\}_{i=1}^k$, where the $i$-th pair corresponds to $U_i$. Hence we identify the number of $U$ as $k$. Figure 4 shows an example with $k = 4$, where the 4 DCMs correspond to “Brightness”, “Exposure”, “Temperature” and “Sharpness”, respectively.

### 3.2. Representing $U$ by Proxy Variables

While the learned DCMs identify the number of $U$, they do not provide a vector representation of $U$. Hence $P(U|S)$ and $P(Y|X,U)$ in Eq. (1) are still unobserved. Fortunately, $U$ generates two observed variables $X$ and $Z$ as shown in Figure 5, which are called proxy variables [35] of $U$ and make $P(U|S)$ and $P(Y|X,U)$ estimable. We will first explain $X$ and $Z$ as well as their related causal links.

$\hat{X} \rightarrow X \rightarrow Y$. $\hat{X}$ represents the DCMs outputs, i.e., for a sample $x_s$ in $S = s$, $\hat{X}|S = s$ takes value from $\hat{X}_s = \{M_i(x_s)\}_{i=1}^k$; and for a sample $x_t$ in $S = t$, $\hat{X}|S = t$ takes value from $\hat{X}_t = \{M_i^{-1}(x_t)\}_{i=1}^k$. As detailed in Appendix, $M_i, M_i^{-1}$ generates counterfactuals in the other domain by fixing $U_j \neq i$ and intervening $U_i$, i.e., $\hat{X}$ is generated from $U$. Hence $U \rightarrow \hat{X}$ is justified. Moreover, as the other domain is also labelled, $\hat{X} \rightarrow Y$ denotes the predictive effects from the generated counterfactuals.

$Z \rightarrow U \rightarrow X \rightarrow Y$. $Z$ is a latent variable encoded from $X$ by a VAE [22]. The link $Z \rightarrow X$ is because the latent

---

Note that our DCM learning is orthogonal to the GAN models used. One can use more advanced GAN models to discover complex mechanisms such as shape or viewpoint changes.
variable is trained to reconstruct $X$. Furthermore, the latent variable of VAE is shown to capture some information of the underlying $U$ [51], justifying $U \rightarrow Z$.

Note that the networks to obtain $\tilde{X}$ and $Z$ are trained in an unsupervised fashion. Hence $\tilde{X}, Z$ are observed in both $S = s$ and $S = t$. Under this causal graph, we have a theoretically grounded solution to Eq. (1) using the theorem below (proof in Appendix as corollary to [35]).

**Theorem (Proxy Function).** Under the causal graph in Figure 5, any solution $h_y(X, \tilde{X})$ to Eq. (3) satisfies Eq. (4).

$$P(Y|Z, X, S) = \sum_{\tilde{x}} h_y(X, \tilde{x}) P(\tilde{X} = \tilde{x}|Z, X, S) \tag{3}$$

$$P(Y|X, U) = \sum_{\tilde{x}} h_y(X, \tilde{x}) P(\tilde{X} = \tilde{x}|U) \tag{4}$$

Notice that we can set $S = s$ in Eq. (3) and use the labelled data in $S = s$ to learn $h_y$ (details given below). We prove in the Appendix that $h_y$ is invariant across domains. This leads to the following inference strategy.

**3.2.1 Inference**

Taking expectation over $U|S = t$ on both sides of Eq. (4) leads to a practical solution to Eq. (1):

$$P(Y|do(X), S = t) = \sum_{\tilde{x} \in \tilde{X}} h_y(X, \tilde{x}) P(\tilde{X} = \tilde{x}|S = t), \tag{5}$$

where $P(\tilde{X}|S = t)$ is estimable as $\tilde{X}$ is observed in $S = t$, and $h_y$ is trained in $S = s$ as given below.

**3.2.2 Learning $h_y$**

We detail the training of $h_y$, including the representation of its inputs $X$ and $\tilde{X}$, the function forms of the terms in Eq. (3) and the training objective.

**Representation of $X, \tilde{X}$**. When $X$ and $\tilde{X}$ are images, directly evaluating Eq. (5) in image space can be computationally expensive, not to mention the tendency for severe over-fitting. Therefore, we follow the common practice in UDA [30, 21] to introduce a pre-trained feature extractor backbone (e.g., ResNet [18]). Hereinafter, $X, \tilde{X}$ denote their respective n-d features. Note that $Z$ of l dimensions is encoded from $X$’s feature form ($l < n$).

**Function Forms**. We adopt an isotropic Gaussian distribution for $P(\tilde{X}|S)$. We implement $P(Y|Z, X, S = s), P(\tilde{X}|Z, X, S = s)$ in Eq. (3) with the following linear model $f_y(Z, X)$ and $f_{\tilde{x}}(Z, X)$, respectively:

$$f_y(Z, X) = W_1 Z + W_2 X + b_1$$

$$f_{\tilde{x}}(Z, X) = W_3 Z + W_4 X + b_2, \tag{6}$$

where $f_y : (Z, X) \rightarrow \mathbb{R}^c$ produces logits for $c$ classes, $f_{\tilde{x}} : (Z, X) \rightarrow \mathbb{R}^n$ predicts $X$, $W_1 \in \mathbb{R}^{c \times l}, W_2 \in \mathbb{R}^{c \times n}, W_3 \in \mathbb{R}^{n \times l}, W_4 \in \mathbb{R}^{n \times n}, b_1 \in \mathbb{R}^c$ and $b_2 \in \mathbb{R}^n$. With this function form, we can solve $h_y : (X, \tilde{X}) \rightarrow \mathbb{R}^c$ from Eq. (3) as (derivation in Appendix):

$$h_y(X, \tilde{X}) = b_1 - W_1 W_3^+ b_2 + W_1 W_3^+ \tilde{X} + (W_2 - W_1 W_3^+ W_4) X, \tag{7}$$

where $(\cdot)^+$ denotes pseudo-inverse.

**Overall Objective**. The training objective is given by:

$$\min_{\omega, \beta, \theta} (L_c + L_v) + \min_{\beta, \gamma} \alpha L_p, \tag{8}$$

where $\omega$ denotes the parameters of the linear functions $f_y$ and $f_{\tilde{x}}$, $\beta$ denotes the parameters of the backbone, $\theta$ denotes the parameters of the VAE, $L_c$ denotes the sum of Cross-Entropy loss to train $f_y$ and the mean squared error loss to train $f_{\tilde{x}}$ (see Eq. (6)). $L_v$ is the VAE loss, $L_p$ is the proxy loss, $\gamma$ denotes the parameters of the discriminators used in $L_p$, and $\alpha$ is a trade-off parameter.

**VAE Loss**. $L_v$ is used to train the VAE, which contains an encoder $Q_{\theta}(Z|X)$ and a decoder $P_{\theta}(X|Z)$. Given a feature $x_s$ in $S = s$, $L_v$ is given by:

$$L_v = -E_{Q_{\theta}(Z|X=x_s)}[P_{\theta}(X = x_s|Z)] + D_{KL}(Q_{\theta}(Z|X = x_s) \parallel P(Z)), \tag{9}$$

where $D_{KL}$ denotes KL-divergence and $P(Z)$ is set to the isotropic Gaussian distribution. Note that we only need to learn a VAE in $S = s$, as $h_y(X, \tilde{X})$ is domain-agnostic.

**Proxy Loss**. In practice, the generated images from the DCMs $\{(M_i, M_i^{-1})\}_{i=1}^k$ may contain artifacts that contaminate the feature outputs from the backbone [33], making $X$ as feature no longer resembles the features in the counterpart domain. Hence we regularize the backbone with a proxy loss to enforce feature-level resemblance between the samples transformed by the DCMs and the samples in the counterpart domain. Given feature $x_s$ in $S = s, x_t$ in $S = t$ and their corresponding DCMs outputs sets $\tilde{X}_s, \tilde{X}_t$ (as n-d features), the loss is given by:

$$L_p = \min_{\tilde{x}_s \in \tilde{X}_s} \sum_{\tilde{x}_s \in \tilde{X}_s} \log(1 - D_t(\tilde{x}_s)) + \min_{\tilde{x}_t \in \tilde{X}_t} \sum_{\tilde{x}_t \in \tilde{X}_t} \log(1 - D_s(\tilde{x}_t)), \tag{10}$$

where $D_s, D_t$ are discriminators parameterized by $\gamma$ that return a large value for features of real samples. Through min-max adversarial training, generated sample features $X$ become similar to the sample features in the counterpart domain and fulfill the role as a proxy.

**4. Experiment**

**4.1. Datasets**

We validated TCM on three standard benchmarks for visual domain adaptation:
**ImageCLEF-DA** [1] is a benchmark dataset for ImageCLEF 2014 domain adaptation challenge, which contains three domains: 1) Cutsel-256 (C), 2) ImageNet ILSVRC 2012 (I) and 3) Pascal VOC 2012 (P). For each domain, there are 12 categories and 50 images in each category. We permuted all the three domains and built six transfer tasks, i.e., I→P, P→I, I→C, C→I, C→P, P→C.

**Office-Home** [53] is a very challenging dataset for UDA with 4 significantly different domains: Artistic images (A), Clipart (C), Product images (P) and Real-World images (R). It contains 15,500 images from 65 categories of everyday objects in the office and home scenes. We evaluated TCM in all 12 permutations of domain adaptation tasks.

**VisDA-2017** [43] is a challenging simulation-to-real dataset that is significantly larger than the other two datasets with 280k images in 12 categories. It has two domains: Synthetic, with renderings of 3D models from different angles and with different lighting conditions; and Real with natural real-world images. We followed the common protocol [30, 12] to evaluate on Synthetic→Real task.

### 4.2. Setup

**Evaluation Protocol.** We followed the common evaluation protocol for UDA [30, 60, 12], where all labeled source samples and unlabeled target samples are used to train the model, and the average classification accuracy is compared in each dataset based on three random experiments. Following [60, 58], we reported the standard deviation on ImageCLEF-DA. For fair comparison, our TCM and all comparative methods used the backbone ResNet-50 [18] pre-trained on ImageNet [44]. Please refer to Appendix for implementation details.

**Baseline.** As existing domain-mapping methods either focus on the image segmentation task [26], or only evaluate on toy settings (e.g., digit) [20], we implemented a domain-mapping baseline. Specifically, we trained a CycleGAN on toy settings (e.g., digit) and with different lighting conditions; and Real with natural real-world images. We followed the common protocol to evaluate on Synthetic→Real task.
Then we learned a classifier on the transformed samples $X_{s} \rightarrow X_t$ with the standard cross-entropy loss, while using the loss in Eq. (10) to align the features of the transformed samples with the target domain features. In inference, we directly used the learned classifier.

4.3. Results

Overall Results. As shown in Table 1, 2, 3, our method achieves the state-of-the-art average classification accuracy on Office-Home [53], ImageCLEF-DA [1], and VisDA-2017 [43]. Specifically, 1) ImageCLEF-DA has the smallest domain gap, where the 3 domains correspond to real-world images from 3 datasets. Our TCM outperforms existing methods on 4 out of 6 UDA tasks. 2) Office-Home has a much larger domain gap, such as the Artistic domain (A) and Clipart domain (C). Besides improvements in average accuracy, our method significantly outperforms existing methods on the most difficult tasks, e.g., +1.6% on A→C, +1.1% on P→C. 3) VisDA-2017 is a large-scale dataset with 280k images, where TCM performs competitively. Note that the training complexity and convergence speed of TCM are comparable with existing methods (details in Appendix). Hence overall, our TCM is generally applicable to small to large-scale UDA tasks with various sizes of domain gaps.

Comparison with Baseline. We have three observations: 1) From Table 1, 2, 3, we notice that the Baseline does not perform competitively on 3 benchmarks. This shows that the improvements from TCM are not the results of superior performance. We have three observations: 1) Among each step of $k$, the largest difference occurs between $k = 1$ and $k = 2$. This means that accounting for 2 disentangled factors can explain much of the domain shift due to $P(U|S = s) \rightarrow P(U|S = t)$ in ImageCLEF-DA. 2) Across all tasks, the best performance is usually achieved with $k = 5$. This supports the effectiveness of stratifying $U$ according to a small number of disentangled attributes. 3) A large $U$ (e.g., 10) leads to a slight performance drop. In practice, we observed that when $k$ is large, a few DCMs seldom win any sample from Eq. (2), and hence are poorly trained. This can impact their image generation quality, leading to reduced performance. We argue that this is because the chosen $k$ is larger than the number of factors that can be disentangled from the dataset. For example, if there exist $k' < k$ factors $(U_1, \ldots, U_{k'})$, once $k'$ DCMs establish correspondence with them, the rest $(k - k')$ DCMs will never win any sample, as they can never generate images that are more counterfactual faithful compared with intervention on each $U_i$ (see Counterfactual Faithfulness theorem in Appendix).

Table 3: Accuracy (%) of Synthetic→Real task on the VisDA-2017 dataset [43].

<table>
<thead>
<tr>
<th>k</th>
<th>I→P</th>
<th>P→I</th>
<th>I→C</th>
<th>C→I</th>
<th>C→P</th>
<th>P→C</th>
<th>P→C Avg</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>77.6</td>
<td>91.6</td>
<td>96.7</td>
<td>92.3</td>
<td>78.4</td>
<td>95.3</td>
<td>88.7</td>
</tr>
<tr>
<td>2</td>
<td>79.7</td>
<td>93.6</td>
<td>97.5</td>
<td>93.5</td>
<td>79.2</td>
<td>96.6</td>
<td>89.9</td>
</tr>
<tr>
<td>3</td>
<td>79.9</td>
<td>94.2</td>
<td>97.6</td>
<td>93.7</td>
<td>79.6</td>
<td>96.5</td>
<td>90.3</td>
</tr>
<tr>
<td>5</td>
<td>79.7</td>
<td>93.9</td>
<td><strong>97.8</strong></td>
<td><strong>93.8</strong></td>
<td>79.8</td>
<td><strong>96.9</strong></td>
<td>90.3</td>
</tr>
<tr>
<td>7</td>
<td>79.5</td>
<td>94.1</td>
<td>97.7</td>
<td>93.6</td>
<td><strong>79.9</strong></td>
<td>96.6</td>
<td>90.2</td>
</tr>
<tr>
<td>10</td>
<td>79.2</td>
<td>93.6</td>
<td>97.5</td>
<td>93.5</td>
<td>79.6</td>
<td><strong>96.4</strong></td>
<td>90.0</td>
</tr>
</tbody>
</table>

Table 4: Ablation on the number of DCMs $k$ with accuracy (%) on the 6 UDA tasks from ImageCLEF-DA [1].

Figure 7: t-SNE [34] plot of features in $S = s$ (red) and in $S = t$ (blue) after training using method: (a) GVB-GD [12]. (b) Our TCM. Only samples from the first 15 classes are plotted to avoid clutter.
the features in $S = s$ and $S = t$ in Office-Home $A \rightarrow C$ task after training, for both the current SOTA GVB-GD [12] and our TCM. GVB-GD is based on invariant-feature-learning and focuses on making $X_s$ and $X_t$ alike, while our TCM does not explicitly align them. We indeed observe a better alignment between the features in $S = s$ and $S = t$ in GVB-GD. However, as shown in Table 1, our method outperforms GVB-GD on $A \rightarrow C$. This shows that the current common view on the solution to UDA based on [2], i.e., making domains alike while minimizing source risk, does not necessarily lead to the best performance. In fact, this is in line with the conclusion from recent theoretical works [61]. Our approach is orthogonal to existing approaches and we demonstrate the practicality of a more principled solution to UDA, i.e., establishing appropriate causal assumption on the domain shift and solving the corresponding transport formula (e.g., Eq. (1)).

**Alleviating Semantic Loss.** We use Figure 8 to reveal the semantic loss problem on existing methods, and how our method can alleviate it. The figure shows the CAM [62] responses on images in $S = s$ and $S = t$ from the $A \rightarrow C$ task in Office-Home dataset. We compared with two lines of existing approaches: invariant-feature-learning method (GVB-GD [12]) and domain-mapping method (our Baseline), and we generated CAM for each of them as shown in the red dotted box. For our TCM, the prediction is a linear combination of the effects from the two images: $X$ and the DCM outputs $\hat{X}$ (see Eq. (7)). As the locations of objects tend to remain the same in $\hat{X}$ as in $X$ (see Figure 4), we visualize the overall CAM of our TCM in the blue box by combining the CAM from $X$ and $\hat{X}$ weighted by their contributions towards softmax prediction probability. On the left, we show the CAMs in $S = s$. By looking at the CAM of GVB-GD and the Baseline, we observe that they focus on the contextual object semantics (e.g., food that commonly appears together with “fork”) to distinguish “marker” and “fork”, where the effect from the object shape is mostly lost. In contrast, our TCM focuses on the marker and fork, i.e., the shape semantic is preserved in training. On the right in $S = t$, the contextual object semantic is no longer discriminative for the two classes. Hence GVB-GD and Baseline either focus on the wrong semantic (e.g., plate) or become confused (focusing on a large area), leading to the wrong prediction. Thanks to the preserved shape semantic, our TCM focuses on the object and makes the correct prediction. This provides an intuitive explanation of how our TCM alleviates the semantic loss (more examples in Appendix).

5. Conclusion

We presented a practical implementation of the transportability theory for UDA called Transporting Causal Mechanisms (TCM). It systematically eliminates the semantic loss from the confounding effect. In particular, we proposed to identify the confounder stratification by discovering disentangled causal mechanisms and represent the unknown confounder representation by proxy variables. Extensive results on UDA benchmarks showed that TCM is empirically effective. It is worth highlighting that TCM is orthogonal to the existing UDA solutions—instead of making the two domains similar, we aim to find how to transport and what can be transported with a causality theoretic viewpoint of the domain shift. We believe that the key bottleneck of TCM is the quality of causal mechanism disentanglement. Therefore, we will seek more effective unsupervised feature disentanglement methods and investigate their causal mechanisms.

6. Acknowledgements

The authors would like to thank all reviewers and ACs for their constructive suggestions, and thank Alibaba City Brain Group for the donations of GPUs. This research is partly supported by the Alibaba-NTU Singapore Joint Research Institute, Nanyang Technological University (NTU), Singapore; A*STAR under its AME YIRG Grant (Project No. A20E6c0101); and the Singapore Ministry of Education (MOE) Academic Research Fund (AcRF) Tier 2 grant.
References

[12] Shuhao Cui, Shuhui Wang, Junbao Zhuo, Chi Su, Qingming Huang, and Qi Tian. Gradually vanishing bridge for adversarial domain adaptation. In CVPR, 2020. 6, 7, 8


[57] Yuchen Zhang, Tianle Liu, Mingsheng Long, and Michael Jordan. Bridging theory and algorithm for domain adaptation. In IML, 2019. 6, 7


[59] Han Zhao, Tan Wang, Hanwang Zhang, Qianru Sun, and Xian-Sheng Hua. Counterfactual zero-shot and open-set visual recognition. In CVPR, 2021. 4