T-SVDNet: Exploring High-Order Prototypical Correlations for Multi-Source Domain Adaptation

Ruihuang Li[1]*, Xu Jia[2]†, Jianzhong He[3], Shuaijun Chen[4], Qinghua Hu[1]†

[1]College of Intelligence and Computing, Tianjin University, [2]Dalian University of Technology,
[3]Huawei Technologies, [4]Noah's Ark Lab, Huawei Technologies

{liruihuang, huqinghua}@tju.edu.cn, xjia@dlut.edu.cn, chensj1110@163.com,
jianzhong.he@huawei.com

Abstract

Most existing domain adaptation methods focus on adaptation from only one source domain, however, in practice there are a number of relevant sources that could be leveraged to help improve performance on target domain. We propose a novel approach named T-SVDNet to address the task of Multi-source Domain Adaptation (MDA), which is featured by incorporating Tensor Singular Value Decomposition (T-SVD) into a neural network's training pipeline. Overall, high-order correlations among multiple domains and categories are fully explored so as to better bridge the domain gap. Specifically, we impose Tensor-Low-Rank (TLR) constraint on a tensor obtained by stacking up a group of prototypical similarity matrices, aiming at capturing consistent data structure across different domains. Furthermore, to avoid negative transfer brought by noisy source data, we propose a novel uncertainty-aware weighting strategy to adaptively assign weights to different source domains and samples based on the result of uncertainty estimation. Extensive experiments conducted on public benchmarks demonstrate the superiority of our model in addressing the task of MDA compared to state-of-the-art methods. Code is available at `https://github.com/lslrh/T-SVDNet`.

1. Introduction

Deep learning methods have shown superior performance with huge amounts of training data as rocket fuel. However, directly transferring knowledge learned on a certain visual domain to other domains with different distributions would degrade the performance significantly due to

the existence of domain shift [42]. To handle this problem, the prominent approaches such as transfer learning and unsupervised domain adaptation (UDA) endeavor to extract domain-invariant features. Discrepancy-based methods reduce the domain gap by minimizing the discrepancy between source and target distributions, such as Maximum Mean Discrepancy (MMD) [25], correlation alignment [35], and contrastive domain discrepancy [16]. Adversarial methods attempt to align source and target domains through adversarial training [34, 38] or GAN-based loss [14, 46]. These methods only focus on domain adaptation with only single source. However, in many practical application scenarios, there are a number of relevant sources collected in different ways available, which could be used to help improve performance on target domain.

Naively combining various sources into one is not an effective way to fully exploit abundant information within multiple sources, and might even perform worse than single-source methods, because domain gap among multiple sources causes confusion in the learning process [44]. Some Multi-Source Domain Adaptation (MDA) approaches [41, 24, 45, 32, 12] focus on aligning multiple source domains and a target domain by projecting them into a domain-invariant feature space. This is done by either explicitly minimizing the discrepancy of different domains [12, 13, 32] or learning an adversarial discriminator to align distributions of different domains [41, 43, 24]. However, eliminating distribution discrepancy of data has the risk of sacrificing discrimination ability. Moreover, these methods only achieve pair-wise matching, neglecting underlying high-order relations among all domains. Another widely used way in MDA is distribution-weighted combining rule [13, 43, 24], which takes a weighted combination of pre-trained source classifiers as the classifier for target domain. In spite of reasonable performance on MDA task, they do not take into consideration intra-domain weightings

---

among different training samples, so that underlying noisy source data may hurt the performance of learning in the target, which is referred to as "negative transfer" [30].

To address the aforementioned limitations, we propose a novel method named T-SVDNet which incorporates tensor singular value decomposition into a neural network's training pipeline. In MDA tasks, although there is large domain gap between different domains, data belonging to the same category do share essential semantic information across domains. Therefore, we assume that data from different domains should follow a certain kind of category-wise structure. Based on this assumption, we explore high-order relationships among multiple domains and categories in order to enforce the alignment of source and target at the prototypical correlation level. Specifically, we impose Tensor-Low-Rank (TLR) constraint on a tensor which is obtained by stacking up a set of prototypical similarity matrices, so that the relationships between categories are enforced to be consistent across domains by pursuing the lowest-rank structure of tensor. Furthermore, to avoid negative transfer [30] caused by noisy training data, we propose a novel uncertainty-aware weighting strategy to guide the adaptation process. It could dynamically assign weights to different domains and training samples based on the result of uncertainty estimation. To train the whole framework with both classification loss and low-rank regularizer, we adopt an alternative optimization strategy, that is, optimizing network parameters with the low-rank tensor fixed and optimizing the low-rank tensor with network parameters unchanged. We conduct extensive evaluations on several public benchmark datasets, where a significant improvement over existing MDA methods has been achieved. Overall, the main contributions of this paper can be summarized as follows:

• We propose the T-SVDNet to explore high-order relationships among multiple domains and categories from the perspective of tensor, which facilitates both domain-invariance and category-discriminability.

• We devise a novel uncertainty-aware weighting strategy to balance different source domains and samples, so that clean data are fully exploited while negative transfer led by noisy data is avoided.

• We propose an alternative optimization method to train the deep model along with low-rank regularizer. Extensive evaluations on benchmark datasets demonstrate the superiority of our method.

2. Related Work

Single-source Domain Adaptation (SDA). SDA aims to generalize a model learned from a labeled source domain to a related unlabeled domain with different data distribution. Existing SDA methods usually incorporate two terms: one term is task loss like cross-entropy loss which

helps learn a model on the labeled source; the other adaptation term aims to align the distributions of source and target domains. These SDA methods can be roughly categorized into three groups according to the alignment strategies: (1) discrepancy-based methods aim to minimize the discrepancy which is explicitly measured on corresponding layers, including Maximum Mean Discrepancy (MMD) [25], correlation alignment [35], and contrastive domain discrepancy [16]; (2) Some adversarial training-based methods align different data distributions by confusing a well-trained domain discriminator [38, 37].

Multi-source Domain Adaptation (MDA). In practical applications, data may be collected from multiple related domains [2, 36], which involve more abundant information but also bring the difficulty in handling the domain shift. Thus MDA methods become more and more popular. The earlier MDA methods mainly focus on weighted combination of source classifiers [13, 24, 22, 34] based on the assumption that target distribution can be approximated by the mixture of source distributions [3, 1]. Hoffman *et al.* [13] cast distribution combination as a DC-programming and derived a tighter domain generalization bound. Besides classification losses, various domain assignment constraints are devised to reduce the domain gap. In addition to minimizing domain discrepancy between the target and each source domain, Li *et al.* [24] also took into consideration the relationships between pairwise source domains and proposed a tighter bound on the discrepancy among multiple sources. Many explicit measures of discrepancy have been used in MDA methods, such as MMD [12], $\mathcal{L}_2$ distance [33], and moment distance [32]. Some approaches also focus on prototype-based alignment between different domains [31, 40, 39]. As for the adversarial MDA methods which aim to confuse the discriminator so that domain-invariant features are extracted, the optimized objective can be $\mathcal{H}$-divergence [43], Wasserstein distance [45, 24].

Uncertainty Estimation. Quantifying and measuring uncertainty is of great theoretical and practical significance [9, 19]. In Bayesian modeling, there are two main categories of uncertainty [17]: epistemic uncertainty and aleatoric uncertainty. The former is often referred to as model uncertainty, which captures uncertainty in the model parameters, while the latter accounts for noise inherent from the observations. There have been many methods proposed to estimate uncertainty in deep learning [4, 10, 8]. Resorting to these techniques, the robustness and interpretability of many computer vision tasks are improved, such as object detection [7, 21] and face recognition [6].

3. Method

In the MDA setting, there are M labeled source domains $\mathcal{S}_1, \mathcal{S}_2, \cdots, \mathcal{S}_M$ and an unlabeled target domain $\mathcal{T}$. Each source domain $\mathcal{S}_m$ contains N_m observations
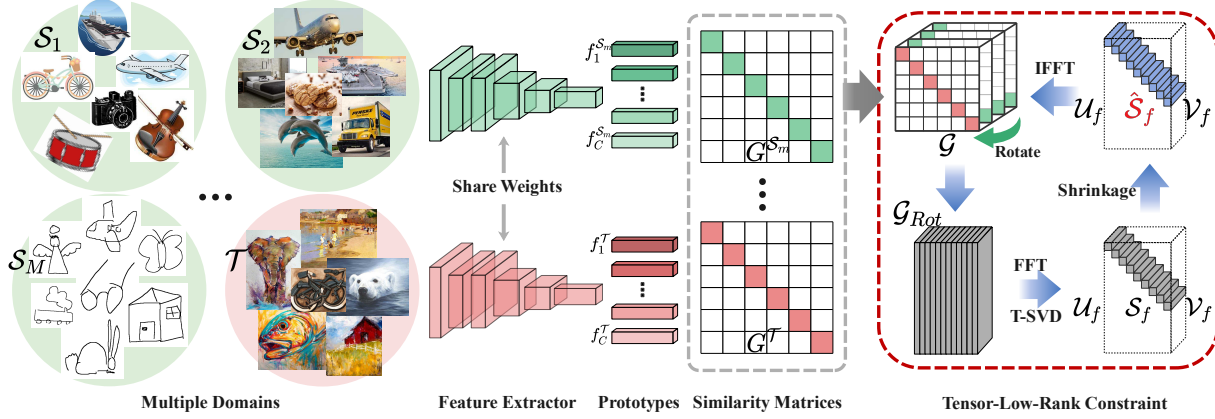
Figure 1. The framework of T-SVDNet. Given M labeled source domains $\mathcal{S}_1, \cdots, \mathcal{S}_M$ and an unlabeled target domain $\mathcal{T}$, we first extract features for input images and compute prototype $f_c^{\mathcal{D}}$ for each category and each domain in an online fashion. Furthermore, the relations between pairwise prototypes are modeled by a group of similarity matrices $G^{\mathcal{S}_1}, \cdots, G^{\mathcal{S}_M}, G^{\mathcal{T}}$. Then we stack these prototypical similarity matrices into a 3-order tensor $\mathcal{G} \in \mathbb{R}^{C \times C \times (M+1)}$ on which tensor-low-rank constraint is imposed in order to explore high-order relationships among different domains. Finally, together with low-rank regularizer, the model is effectively trained in an alternative optimization strategy.

$\{(x_i^{\mathcal{S}_m}, y_i^{\mathcal{S}_m})\}_{i=1}^{N_m}$, where y_i is the desired label, while in the target domain $\mathcal{T}$, the label y is not available. Most existing MDA models can be formulated as the following mapping function:

$$M_{mda}: X^{\mathcal{S}_1} \cup \cdots X^{\mathcal{S}_M} \cup X^{\mathcal{T}} \to Y^{\mathcal{S}_1} \cup \cdots Y^{\mathcal{S}_M}, \quad (1)$$

where M_{mda} is trained on both labeled samples $(X^{\mathcal{S}}, Y^{\mathcal{S}})$ in the source domain and unlabeled samples $X^{\mathcal{T}}$ in the target domain.

In this section, we propose the T-SVDNet which fully explores high-order relationships among all domains by exploiting the tensor obtained by stacking up a set of prototypical similarity matrices (see Fig. 1). In addition, we propose a novel uncertainty-aware weighting strategy to achieve both inter- and intra-domain weightings so that negative transfer is reduced (see Fig. 2). This section is organized as follows: we first construct prototypical similarity matrix in Sec. 3.1. Then we propose the tensor-low-rank constraint and uncertainty-aware weighting strategy in Sec. 3.2 and Sec. 3.3, respectively. Finally, we formulate the total objective function in Sec. 3.4 and propose a novel alternative optimization method in Sec. 3.5.

3.1. Prototypical similarity matrix

In the proposed T-SVDNet, we first map input image into latent space through a feature extractor denoted by $f(\cdot)$, then we update the centroid of each category (prototype) based on the feature embeddings of a mini-batch [31, 40, 39]. For domain $\mathcal{D} \in \{\mathcal{S}_1, \cdots, \mathcal{S}_M, \mathcal{T}\}$, the prototype of the c-th category denoted by $f_c^{\mathcal{D}}$ is computed by:

$$f_c^{\mathcal{D}} = \frac{1}{|\Lambda_c^{\mathcal{D}}|} \sum_{(x_i, y_i) \in \Lambda_c^{\mathcal{D}}} f(x_i), \quad (2)$$

where $\Lambda_c^{\mathcal{D}}$ is the set of training samples belonging to the c-th category in domain $\mathcal{D}$, i.e., $\Lambda_c^{\mathcal{D}} = \{(x_i, y_i) \in \mathcal{D} | y_i = c\}$. It is noteworthy that for unlabeled target domain, we first assign pseudo labels $\hat{y}_i$ to samples with high classification confidence.

In order to reduce the randomness in sampling of each mini-batch and stabilize the training process, the category prototypes are updated according to exponential moving average (EMA) method:

$$f_c^{\mathcal{D}}|_I := \alpha f_c^{\mathcal{D}}|_I + (1 - \alpha) f_c^{\mathcal{D}}|_{I-1}, \quad (3)$$

where α is the exponential decay rate and I denotes current iteration.

Then we employ Gaussian kernel to model inter-class relationships and construct a series of prototypical similarity matrices $G^{\mathcal{S}_1}, \cdots, G^{\mathcal{S}_M}, G^{\mathcal{T}}$:

$$G_{c_i, c_j}^{\mathcal{D}} = \mathcal{K}(f_{c_i}^{\mathcal{D}}, f_{c_j}^{\mathcal{D}}) = exp(-\frac{\left\| f_{c_i}^{\mathcal{D}} - f_{c_j}^{\mathcal{D}} \right\|_2^2}{2\gamma^2}), \quad (4)$$

where $f_{c_i}^{\mathcal{D}}$ and $f_{c_j}^{\mathcal{D}}$ are a pair of category centroids from domain $\mathcal{D}$, and γ is the deviation parameter which is set as 0.05 in experiments.

3.2. Tensor-low-rank constraint via T-SVD

Unlike conventional methods only considering pairwise matching, we achieve high-order alignment of all domains at the prototypical correlation level. Specifically, we stack prototypical similarity matrices into a 3-order tensor $\mathcal{G} \in \mathbb{R}^{C \times C \times (M+1)}$ along the third dimension, where C and M denote the number of classes and domains, respectively. Then we impose the Tensor-Low-Rank (TLR) constraint on

Algorithm 1: T-SVD

Input : $\mathcal{G} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$;
Output: $\mathcal{U}, \mathcal{S}, \mathcal{V}$;
1   $\mathcal{G}_f = \text{FFT}(\mathcal{G}, 3)$;
2   **for** $k = 1 : n_3$ **do**
3 | $[\mathcal{U}_f^{(k)}, \mathcal{S}_f^{(k)}, \mathcal{V}_f^{(k)}] = \text{SVD}(\mathcal{G}_f^{(k)})$;
4   **end**
5   $\mathcal{U}, \mathcal{S}, \mathcal{V} = \text{IFFT}(\mathcal{U}_f, 3), \text{IFFT}(\mathcal{S}_f, 3), \text{IFFT}(\mathcal{V}_f, 3)$;

---

the assembled tensor in order to explore high-order correlations among domains and enforce the relationships between categories to be consistent across domains. Here we first give definitions of T-SVD and tensor rank as follows:

definition 1 *(T-SVD) Given tensor* $\mathcal{G} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$, *the tensor singular value decomposition of* $\mathcal{G}$ *is defined as a finite sum of outer product of matrices [29]:*

$$\mathcal{G} = \sum_{i=1}^{min(n_1, n_2)} \mathcal{U}(:, i, :) * \mathcal{S}(i, i, :) * \mathcal{V}(:, i, :)^T, \quad (5)$$

where $\mathcal{U}$ *and* $\mathcal{V}$ *are orthogonal tensors with size* $n_1 \times n_1 \times n_3$ *and* $n_2 \times n_2 \times n_3$, *respectively.* $\mathcal{S}$ *is a tensor with the size* $n_1 \times n_2 \times n_3$, *each frontal slice of which is a diagonal matrix.* $*$ *denotes tensor product (T-product).*

T-SVD also can be computed more efficiently in the Fourier domain. Specifically, it can be replaced by conducting fast Fourier transformation (FFT) along the third dimension of $\mathcal{G}$ to get $\mathcal{G}_f$, and performing matrix SVDs on each frontal slice of $\mathcal{G}_f$:

$$\mathcal{G}_f^{(k)} = \mathcal{U}_f^{(k)} \cdot \mathcal{S}_f^{(k)} \cdot \mathcal{V}_f^{(k)T}, k = 1, \cdots, n_3 \quad (6)$$

where $\cdot$ means matrix product. We use $\mathcal{G}_f^{(k)}$ to denote the k-th frontal slice of $\mathcal{G}$, *i.e.*, $\mathcal{G}_f^{(k)} = \mathcal{G}(:, :, k)$. The result of T-SVD is finally obtained by taking the inverse FFT on $\mathcal{U}_f, \mathcal{S}_f, \mathcal{V}_f$ along the third dimension (see Alg. 1).

definition 2 *(Tensor rank) [18, 28] The rank of* $\mathcal{G} \in \mathbb{R}^{n_1 \times n_2 \times n_3}$ *is a vector* $\mathbf{p} \in \mathbb{R}^{n_3 \times 1}$ *with the* k*-th element equal to the rank of the* k*-th frontal slice of* $\mathcal{G}_f^{(k)}$.

However, we need an adequate convex relaxation to ℓ_1 norm of tensor rank in optimization process. To this end we formulate it as tensor nuclear norm, which is defined as the sum of the singular values of all frontal slices $\mathcal{S}_f^{(k)}$:

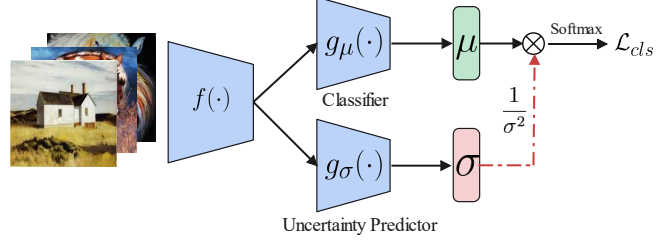$$\|\mathcal{G}\|_{TNN} = \sum_{k=1}^{n_3} \sum_{i=1}^{min(n_1, n_2)} |\mathcal{S}_f^{(k)}(i, i)|. \quad (7)$$

Figure 2. Uncertainty-aware weighting strategy. Each sample is modeled as a Gaussian distribution parameterized by mean μ and variance σ. The classification loss is weighted by estimated data uncertainty σ.

Tensor rotation. In view of each frontal slice $\mathcal{G}^{(k)}$ of tensor $\mathcal{G}$ only contains information from single domain, we rotate it horizontally (or vertically) to obtain $\mathcal{G}_{Rot}$ (see Fig. 1). In this way, each frontal slice $\mathcal{G}_{Rot}^{(k)}$ will involve information from different domains. In the end, imposing tensor-low-rank constraint on the rotated tensor $\mathcal{G}_{Rot}$ benefits the exploration of high-order relationships among different domains. The inter-category data structure is enforced to be consistent across domains by pursuing the lowest rank of each frontal slice $\mathcal{G}_f^{(k)}$ in Fourier domain.

3.3. Uncertainty-aware weighting strategy

Instead of equally treating each source domain and sample, we propose a novel uncertainty-aware weighting strategy to adaptively balance different sources and alleviate negative transfer led by noisy data. Considering that data uncertainty could capture the noise inherent in the data, *i.e.*, it reflects the reliability of output [17, 6], we could weigh different sources and samples based on the result of uncertainty estimation. As shown in Fig. 2, $g_\mu(\cdot)$ and $g_\sigma(\cdot)$ serve on a classifier and uncertainty predictor, respectively. The output of network is modeled as a Gaussian distribution parameterized by mean μ and variance σ. Specifically, the mean is acted by original feature vector, while the variance quantifies uncertainty of training samples. For regression tasks, the Gaussian likelihood is defined as:

$$p(y_i|x_i) = \mathcal{N}(\mu_i, \sigma_i^2), \quad (8)$$

with $\mu_i = h(x_i) = f(x_i) \circ g_\mu(x_i)$ and $\sigma_i = f(x_i) \circ g_\sigma(x_i)$. For classification task, we often squash the model output through a softmax function and obtain a scaled classification likelihood:

$$p(y_i|x_i, \sigma_i) = Softmax\left(\frac{1}{\sigma_i^2} h(x_i)\right). \quad (9)$$

This can be interpreted as a Boltzmann distribution (Gibbs distribution) and σ_i^2 works as temperature for re-scaling in-

put. The log likelihood of output is:

$$log\, p(y_i = c|x_i, \sigma_i) = \frac{1}{\sigma_i^2} h_c(x_i) - log \sum_{c' \neq c} exp(\frac{1}{\sigma_i^2} h_{c'}(x_i)), \tag{10}$$

where $h_c(x_i)$ denotes the c-th element of vector $h(x_i)$. Then the total classification loss is defined as:

$$\begin{aligned}\mathcal{L}_{cls}(\theta) &= -\frac{1}{M}\sum_{m=1}^{M}\frac{1}{N_m}\sum_{i=1}^{N_m} log\, p(y_i = c|x_i, \sigma_i) \\ &= \frac{1}{M}\sum_{m=1}^{M}\frac{1}{N_m}\sum_{i=1}^{N_m}\frac{1}{\sigma_i^2}\mathcal{L}_{CE}(\theta) + log\frac{\sum_{c'} exp\left(\frac{1}{\sigma_i^2}h_{c'}(x_i)\right)}{\left(\sum_{c'} exp\left(h_{c'}(x_i)\right)\right)^{\frac{1}{\sigma_i^2}}} \\ &\approx \frac{1}{M}\sum_{m=1}^{M}\frac{1}{N_m}\sum_{i=1}^{N_m}\frac{1}{\sigma_i^2}\mathcal{L}_{CE}(\theta) + log\sigma_i, \end{aligned} \tag{11}$$

where $\mathcal{L}_{CE}(\theta)$ denotes classification cross entropy loss with $h(x_i)$ not scaled. M and N_m denote the number of domains and samples, respectively. $log\sigma_i$ prevents σ from getting too large. Noisy data with large uncertainty would be assigned less weights, *i.e.*, $1/\sigma^2$. The derivation process of Eq. 11 is provided in supplementary material.

3.4. Objective function

The overall objective function of the proposed model is as follows:

$$\begin{aligned}\mathcal{L}_{total} &= \mathcal{L}_{cls}(\theta) + \lambda \|\mathcal{G}\|_{\circledast}, \\ s.t.\ \mathcal{G} &= \Psi_R(G^{\mathcal{S}_1}, \cdots, G^{\mathcal{S}_M}, G^{\mathcal{T}}), \end{aligned} \tag{12}$$

where $\circledast$ denotes two operations: tensor rotation and tensor nuclear norm, and Ψ_R represents the operation of stacking up all the domain-specific prototypical similarity matrices into a tensor. θ denotes neural network parameters. The first term is classification loss and the second term imposes TLR constraint on the stacked tensor, aiming at achieving high-order alignment of domains.

3.5. Optimization of T-SVDNet

The optimization of T-SVDNet is presented in Alg. 2. In order to make the problem tractable, we introduce an auxiliary variable and alternatively update it along with the network parameters till convergence.

Auxiliary variable. To optimize the objective function in Eq. 12, we first introduce an auxiliary tensor $\mathcal{A}$ to replace $\mathcal{G}$, which converts the original optimization problem into the following one:

$$\min_{\theta, \mathcal{A}} \mathcal{L}_{cls} + \lambda \|\mathcal{A}\|_{\circledast} + \frac{\eta}{2} \|\mathcal{A} - \mathcal{G}\|_F^2, \tag{13}$$

where η is a penalty parameter. It starts from a small initial positive scalar η_0, and gradually increases to the maximum

---

Algorithm 2: Optimization of T-SVDNet

Input : Training data $\mathcal{S}_1, \cdots, \mathcal{S}_M, \mathcal{T}$;
Output: Model parameters θ of T-SVDNet ;

1 **for** $iter = 1$ *to* max_iter **do**
2   • **Updating** θ
3    $\theta_f \leftarrow \theta_f - \partial(\mathcal{L}_{cls} + \frac{\eta}{2}\|\mathcal{A} - \mathcal{G}\|_F^2)/\partial\theta_f$;
4    $\theta_\mu \leftarrow \theta_\mu - \partial\mathcal{L}_{cls}/\partial\theta_\mu$; $\theta_\sigma \leftarrow \theta_\sigma - \partial\mathcal{L}_{cls}/\partial\theta_\sigma$;
5   • **Updating** $\mathcal{A}$
6    $\mathcal{G}_{Rot} = Rotate(\mathcal{G})$;
7    $\mathcal{G}_f = \text{FFT}(\mathcal{G}_{Rot}, 3)$;
8   **for** $k = 1 : n_3$ **do**
9      $[\mathcal{U}_f^{(k)}, \mathcal{S}_f^{(k)}, \mathcal{V}_f^{(k)}] = \text{SVD}(\mathcal{G}_f^{(k)})$;
10      $\mathcal{A}_f^{(k)} = \mathcal{U}_f^{(k)} \cdot D_{\frac{\lambda}{\eta}}(\mathcal{S}_f^{(k)}) \cdot \mathcal{V}_f^{(k)T}$;
11   **end**
12    $\mathcal{A}_{Rot} = \text{IFFT}(\mathcal{A}_f, 3)$;
13    $\mathcal{A} = Rotate(\mathcal{A}_{Rot})$;
14    $\eta = min(\rho\eta, \eta_{max})$
15 **end**

---

truncated value η_{max} with iterations, *i.e.*, it is updated by $\eta = min(\rho\eta, \eta_{max})$, where ρ represents the rate of increase which is set as 1.1 in all experiments. The reason why we update η in such a incremental fashion is that randomly initialized $\mathcal{A}$ may lead to the wrong direction of gradient descent at the beginning of training process.

Update of network parameters θ. The parameters of feature extractor θ_f, classifier θ_μ, and uncertainty predictor θ_σ are updated through gradient descent with auxiliary tensor $\mathcal{A}$ fixed.

Update of auxiliary variable $\mathcal{A}$. When network parameters are fixed, we optimize the subproblem associated with $\mathcal{A}$ as follows:

$$\min_{\mathcal{A}} \lambda \|\mathcal{A}\|_{\circledast} + \frac{\eta}{2}\|\mathcal{A} - \mathcal{G}\|_F^2. \tag{14}$$

We solve this problem in Fourier domain with basic procedure similar to Alg. 1. We first transform tensor $\mathcal{G}_{Rot}$ to Fourier domain $\mathcal{G}_f$, and perform matrix SVD on each k-th frontal slice $\mathcal{G}_f^{(k)}$ and obtain the $\mathcal{U}_f^{(k)}$, $\mathcal{S}_f^{(k)}$, $\mathcal{V}_f^{(k)}$. Then, each frontal slice $\mathcal{A}_f^{(k)}$ of auxiliary variable can be updated by *shrinkage operation* [5, 20] on $\mathcal{G}_f^{(k)}$ in the Fourier domain defined as follows:

$$\mathcal{A}_f^{(k)} = \mathcal{U}_f^{(k)} \cdot D_{\frac{\lambda}{\eta}}(\mathcal{S}_f^{(k)}) \cdot \mathcal{V}_f^{(k)T}, \tag{15}$$

where $D_{\lambda/\eta}(\mathcal{S}_f^{(k)}) = \mathcal{S}_f^{(k)} \cdot \mathcal{J}_f^{(k)}$ is singular value shrinkage operator. $\mathcal{J}_f^{(k)}$ is a diagonal matrix with the i-th diagonal element to be $\mathcal{J}_f^{(k)}(i,i) = (1 - \frac{\lambda}{\eta\mathcal{S}_f^{(k)}(i,i)})_+$. Finally, updated $\mathcal{A}$ is obtained by inverse fast Fourier transform from $\mathcal{A}_f$.

Standards	Methods	→ mm	→ mt	→ up	→ sv	→ syn	Avg
Single Best	Source-Only	52.90 ± 0.60	97.20 ± 0.60	84.70 ± 0.80	77.70 ± 0.80	85.20 ± 0.60	80.80
	DAN	63.78 ± 0.71	96.31 ± 0.54	94.24 ± 0.87	62.45 ± 0.72	85.43 ± 0.77	80.44
	DANN	71.30 ± 0.56	97.60 ± 0.75	92.33 ± 0.85	63.48 ± 0.79	85.34 ± 0.84	82.01
	CORAL	62.53 ± 0.69	97.21 ± 0.83	93.45 ± 0.82	64.40 ± 0.72	82.77 ± 0.69	80.07
	ADDA	71.57 ± 0.52	97.89 ± 0.84	92.83 ± 0.74	75.48 ± 0.48	86.45 ± 0.62	84.84
Source Combination	Source-Only	63.37 ± 0.74	90.50 ± 0.83	88.71 ± 0.89	63.54 ± 0.93	82.44 ± 0.65	77.71
	DAN	67.91 ± 0.82	97.52 ± 0.60	93.45 ± 0.41	67.79 ± 0.62	86.90 ± 0.50	82.69
	DANN	70.80 ± 0.77	97.91 ± 0.69	93.53 ± 0.76	68.54 ± 0.52	87.40 ± 0.90	83.61
	ADDA	72.32 ± 0.74	97.88 ± 0.60	93.10 ± 0.79	75.02 ± 0.80	86.69 ± 0.55	85.02
	MCD	72.50 ± 0.67	96.21 ± 0.81	95.33 ± 0.74	78.89 ± 0.78	87.47 ± 0.65	86.10
	JAN	65.88 ± 0.68	97.21 ± 0.73	95.42 ± 0.77	75.27 ± 0.71	86.55 ± 0.64	84.07
Multi-Source	MDAN	69.48 ± 0.30	98.04 ± 0.89	92.39 ± 0.71	69.23 ± 0.62	87.44 ± 0.45	83.30
	MDDA	78.63 ± 0.61	98.78 ± 0.42	93.91 ± 0.48	79.33 ± 0.80	89.71 ± 0.72	88.12
	DCTN	70.53 ± 1.24	96.23 ± 0.82	92.81 ± 0.27	77.61 ± 0.41	86.77 ± 0.78	84.79
	M³SDA	72.82 ± 1.13	98.43 ± 0.68	96.14 ± 0.81	81.32 ± 0.86	89.58 ± 0.56	87.65
	T-SVDNet$_{part}$	90.05 ± 0.91	99.24 ± 0.08	98.61 ± 0.16	84.03 ± 1.22	94.92 ± 0.17	93.37
	T-SVDNet$_{all}$	91.22 ± 0.74	99.28 ± 0.11	98.63 ± 0.22	84.86 ± 1.47	95.71 ± 0.30	93.94

Table 1. Classification results on Digits-Five. The top value is highlighted in blue bold font and the second best in green bold font.

4. Experiments

In this section, we perform extensive evaluations on several benchmark datasets with state-of-the-art methods.

4.1. Datasets

Digits-Five [15] contains 5 different domains including MNIST (mt), MNIST-M (mm), SVHN (sv), USPS (up), and Synthetic Digits (syn). Each domain consists of 10 numerals from '0' to '9'. There will be further performance gain if all data are employed for training. For a fair comparison, we report the results on both settings (T-SVDNet$_{part}$ and T-SVDNet$_{all}$ in Tab. 1).

PACS [23] is a small-scale multi-domain dataset containing 9991 images from 4 domains: photo (P), art-painting (A), cartoon (C), sketch (S) whose styles are different. These domains share the same seven categories.

DomainNet [32] is a large-scale dataset for Multi-Source Domain Adaptation. Due to the great number of categories and samples (345 categories, around 0.6 million images) and large domain shift, DomainNet is by far the most difficult dataset which contains 6 different domains: clipart (clp), infograph (inf), painting (pnt), quickdraw (qdr), real (rel), and sketch (skt).

4.2. Compared methods

For all experiments, we compare our method with state-of-the-art single-source and multi-source domain adaptation algorithms. Specifically, two strategies are adopted to train the single-source model: Single Best and Source Combination. The former reports the best result among all domains, while the latter simply combines all source domains together. Overall, these compared methods can be roughly categorized into two main groups: (1) adversarial-based methods include DANN [11], ADDA [38], MCD [34], DCTN [41], MDAN [43] and MDDA [45]; (2) another

Methods	→ A	→ C	→ S	→ P	Avg
Source-Only	75.97	73.34	64.23	91.65	76.30
MDAN	83.54	82.34	72.42	92.91	82.80
MDDA	86.73	86.24	77.56	93.89	86.11
DCTN	84.67	86.72	71.84	95.60	84.71
M³SDA	84.20	85.68	74.62	94.47	84.74
T-SVDNet	90.43	90.61	85.49	98.50	91.25

Table 2. Classification results on PACS. The top value is highlighted in blue bold font and the second best in green bold font.

typical strategy is discrepancy minimization, the representative methods involve DAN [25], JAN [27], RTN [26], CORAL [35], and M³SDA [32]. Source-Only directly transfers the model trained in source domain to target domain. For a fair comparison, we use the same model architecture and data pre-processing routines as compared methods in all experiments. More implementation details are provided in supplementary materials.

4.3. Experimental results

The results on **Digits-Five** are shown in Tab. 1. Overall, our method tops the list in all domains and achieves 93.37% average accuracy, around 5.25% higher than the second best method MDDA. In particular, a performance improvement about 11.42% and 5.21% over MDDA is achieved on '→ mm' and '→ syn' tasks, respectively. The performance will be further boosted to 93.94% if all training data is used, outperforming other algorithms by a large margin.

The results on **PACS** are shown in Tab. 2. Our method T-SVDNet achieves the best performance on all domains and gets 91.25% average accuracy, outperforming the second best method MDDA by 5.14%. Especially on '→ S' task, our method achieves a 7.93% performance gain over MDDA.

The experimental results on **DomainNet** are reported in Tab. 3. Overall, T-SVDNet achieves the best performance

Standards	Methods	→clp	→inf	→pnt	→qdr	→rel	→skt	Avg
Single-Best	Source-Only	39.6 ± 0.6	8.2 ± 0.8	33.9 ± 0.6	11.8 ± 0.7	41.6 ± 0.8	23.1 ± 0.7	26.4
	DAN	39.1 ± 0.5	11.4 ± 0.8	33.3 ± 0.6	16.2 ± 0.4	42.1 ± 0.7	29.7 ± 0.9	28.6
	RTN	35.3 ± 0.7	10.7 ± 0.6	31.7 ± 0.8	13.1 ± 0.7	40.6 ± 0.6	26.5 ± 0.8	26.3
	JAN	35.3 ± 0.7	9.1 ± 0.6	32.5 ± 0.7	14.3 ± 0.6	43.1 ± 0.8	25.7 ± 0.6	26.7
	ADDA	39.5 ± 0.8	14.5 ± 0.7	29.1 ± 0.8	14.9 ± 0.5	41.9 ± 0.8	30.7 ± 0.7	28.4
	DANN	37.9 ± 0.7	11.4 ± 0.9	33.9 ± 0.6	13.7 ± 0.6	41.5 ± 0.7	28.6 ± 0.6	27.8
	MCD	42.6 ± 0.3	19.6 ± 0.8	42.6 ± 1.0	3.8 ± 0.6	50.5 ± 0.4	33.8 ± 0.9	32.2
Source Combination	Source-Only	47.6 ± 0.5	13.0 ± 0.4	38.1 ± 0.5	13.3 ± 0.4	51.9 ± 0.9	33.7 ± 0.5	32.9
	DAN	45.4 ± 0.5	12.8 ± 0.9	36.2 ± 0.6	15.3 ± 0.4	48.6 ± 0.7	34.0 ± 0.5	32.1
	RTN	44.2 ± 0.6	12.6 ± 0.7	35.3 ± 0.6	14.6 ± 0.8	48.4 ± 0.7	31.7 ± 0.7	31.1
	ADDA	47.5 ± 0.8	11.4 ± 0.7	36.7 ± 0.5	14.7 ± 0.5	49.1 ± 0.8	33.5 ± 0.5	32.2
	JAN	40.9 ± 0.4	11.1 ± 0.6	35.4 ± 0.5	12.1 ± 0.7	45.8 ± 0.6	32.3 ± 0.6	29.6
	MCD	54.3 ± 0.6	22.1 ± 0.7	45.7 ± 0.6	7.6 ± 0.5	58.4 ± 0.7	43.5 ± 0.6	38.5
Multi-Source	MDAN	52.4 ± 0.6	21.3 ± 0.8	46.9 ± 0.4	8.6 ± 0.6	54.9 ± 0.6	46.5 ± 0.7	38.4
	MDDA	59.4 ± 0.6	23.8 ± 0.8	53.2 ± 0.6	12.5 ± 0.6	61.8 ± 0.5	48.6 ± 0.8	43.2
	DCTN	48.6 ± 0.7	23.5 ± 0.6	48.8 ± 0.6	7.2 ± 0.5	53.5 ± 0.6	47.3 ± 0.5	38.2
	M³SDA	58.6 ± 0.5	26.0 ± 0.9	52.3 ± 0.6	6.3 ± 0.6	62.7 ± 0.5	49.5 ± 0.8	42.6
	T-SVDNet	66.1 ± 0.4	25.0 ± 0.8	54.3 ± 0.7	16.5 ± 0.9	65.4 ± 0.5	54.6 ± 0.6	47.0

Table 3. Classification results on DomainNet. The top value is highlighted in blue bold font and the second best in green bold font.

Methods	→ mm	→ mt	→ up	→ sv	→ syn	Avg	Gain
Source-Only	67.25 ± 0.81	98.88 ± 0.49	97.87 ± 0.43	77.76 ± 0.92	92.41 ± 0.57	86.83	–
T-SVDNet (+E)	73.85 ± 0.84	98.96 ± 0.35	97.87 ± 0.65	77.86 ± 0.84	92.44 ± 0.39	88.19	1.36 ↑
T-SVDNet (+E+T)	88.76 ± 0.41	99.16 ± 0.26	98.09 ± 0.14	82.94 ± 0.90	94.47 ± 0.62	92.68	5.85 ↑
T-SVDNet (+E+T+U)	91.22 ± 0.74	99.28 ± 0.11	98.63 ± 0.22	84.86 ± 1.47	95.71 ± 0.30	93.94	7.11 ↑

Table 4. Ablation study on key components of model on Digits-Five.

on five out of six tasks. It obtains average accuracy of 47.0% on six domains and ranks the first in the list, with 3.8% performance improvement over MDDA, which is mainly attributed to the thorough exploration of high-order relations between different domains and categories. It is noteworthy that the performances of many MDA methods drop obviously compared to Single Best on '→ qdr' task due to negative transfer, while our method still attains better performance because of uncertainty-aware weighting strategy. Negative transfer is avoided by filtering out noisy source samples near decision boundaries for training, while clean data with low noise intensity are fully exploited.

5. Analysis

Ablation study. We further validate the effects of some key components in our framework. Tab. 4 shows the results of controlled experiments on Digits-Five dataset. As a reference, we report the performance of Source-Only that directly transfers the model trained on source domains to target domain. For convenience, '+E', '+T', '+U' denote entropy minimization constraint on target domain, tensor-low-rank constraint, and uncertainty-aware weighting, respectively. In general, we have the following observations according to Tab. 4: (1) Entropy minimization boosts performance obviously due to the exploitation of target domain; (2) It is remarkable that Tensor-Low-Rank constraint significantly improves the performance by 14.91% on '→ mm' task. This is attributed to the high-order alignment between different domains and the extraction of domain-invariant features; (3) Uncertainty-aware weighting strat-
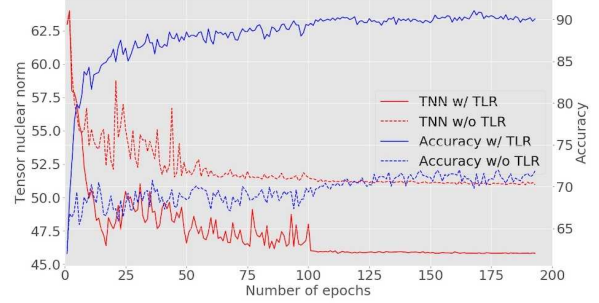

Figure 3. Tensor nuclear norm and classification accuracy curves on "→ mm" task.

egy further improves the performance by 1.26% on average, which suggests that our model is able to learn more transferable features across domains.

The effect of TLR constraint. We compute tensor nuclear norm (TNN) which is usually used as an approximate measure of tensor rank. As shown in Fig. 3, we compare the TNN curves w/ and w/o TLR constraint. We find that TNN w/ TLR drops significantly during the first several epochs and stabilizes at around 46, while the baseline w/o TLR drops slowly and becomes stable earlier. This demonstrates that our proposed TLR constraint is effective and brings large performance improvement.

Feature visualization. To demonstrate the transfer ability of our model, we visualize the feature embeddings of different models on '→ C' task on PACS. As shown in Fig. 4 (a), the target features learned by Source-Only almost mismatch with source domain and different classes in
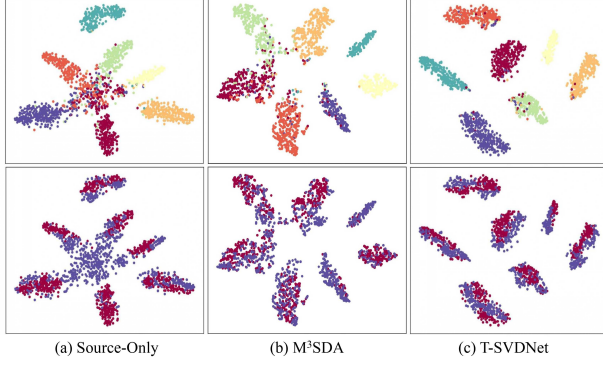
(a) Source-Only   (b) M³SDA   (c) T-SVDNet

Figure 4. The t-SNE visualizations of feature embeddings on '→C' task on PACS. The top row represents category information (each color denotes a class). The bottom row represents domain information (red: source domain; purple: target domain).
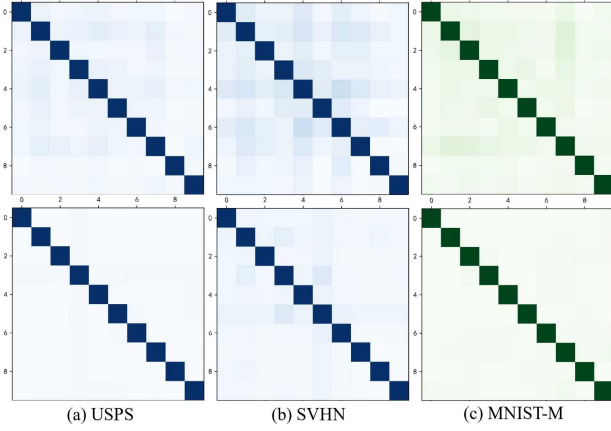

(a) USPS   (b) SVHN   (c) MNIST-M

Figure 5. Visualizations of similarity matrices on '→mm' task on Digits-Five. The top row denotes the model w/o TLR constraint, and the bottom row is T-SVDNet. Blue and Green represent source and target domain, respectively.

target domain are entirely mixed up. Compared to M3SDA and Source-Only, our method produces clusters with clearer boundaries, which suggests that T-SVDNet possesses better transfer ability on target and is able to eliminate domain discrepancy without sacrificing discrimination ability.

Visualizations of similarity matrices. To further validate the effect of TLR constraint, we visualize the prototypical similarity matrices of three domains on Digits-Five dataset in Fig. 5. Compared to the baseline without TLR constraint (the top row), our method (the bottom row) could capture clearer category-wise data structure. Specifically, matrices in the bottom row contain less domain-specific noise, because we search for a lowest-rank structure of tensor and enforce prototypical correlations to be consistent across domains. Especially on the target domain (MNIST-M), the noise is reduced by a large margin compared to Source-Only. These results indicate the effectiveness of TLR constraint on aligning source and target domains.
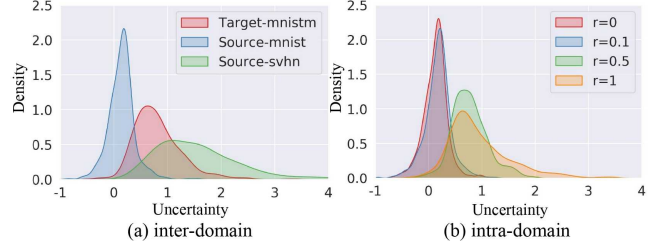

(a) inter-domain   (b) intra-domain

Figure 6. (a) Uncertainty distributions of different domains on Digits-Five. (b) Uncertainty distribution of single domain varies with the increasing noise intensity r on MNIST.

Uncertainty estimation. We conduct qualitative and quantitative experiments to demonstrate the ability of model to measure noise intensity (data uncertainty).

(1) Inter-domain weighting. The uncertainty distributions of different domains on '→mm' task are shown in Fig. 6 (a). Overall, the estimated uncertainty is highly correlated with domain quality. *e.g.*, the uncertainty distribution of high-quality domain MNIST (blue curve) is more concentrated than low-quality domain SVHN (Green curve).

(2) Intra-domain weighting. As shown in Fig. 6 (b), we add noise sampled from Gaussian distribution $\mathcal{N}(0, \mathbf{I})$ to original images, *i.e.*, $\tilde{x}_i = x_i + r\epsilon_i$, where ϵ denotes noise and r controls the intensity of noise. According to Fig. 6 (b), when noise intensity is small ($r = 0.1$), the curves of noisy and clean samples ($r = 0$) are highly overlapped. However, with the increase of noise intensity ($r = 0.5, 1$), the uncertainty distributions get more dispersed.

6. Conclusion

In this paper, we propose the T-SVDNet for multi-source domain adaptation, which is featured by incorporating tensor singular value decomposition into neural network training process. Category-wise relations are modeled by prototypical similarity matrix, aiming at capturing complex data structure. Furthermore, high-order relations between different domains are fully explored by imposing tensor-low-rank constraint on the tensor stacked by domain-specific similarity matrices. In addition, a novel uncertainty-aware weighting strategy is proposed to combine data distributions of different domains, which reduces negative transfer led by noisy data. We adopt alternative optimization algorithm to train T-SVDNet efficiently. Extensive experiments on three public benchmark datasets demonstrate the favorable performance against state-of-the-art methods.

7. Acknowledgement

References

[1] Shai Ben-David, John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman Vaughan. A theory of learning from different domains. *Machine Learning*, 79(1):151–175, 2010.

[2] Himanshu S Bhatt, Arun Rajkumar, and Shourya Roy. Multi-source iterative adaptation for cross-domain classification. pages 3691–3697, 2016.

[3] John Blitzer, Koby Crammer, Alex Kulesza, Fernando Pereira, and Jennifer Wortman. Learning bounds for domain adaptation. pages 129–136, 2007.

[4] Charles Blundell, Julien Cornebise, Koray Kavukcuoglu, and Daan Wierstra. Weight uncertainty in neural networks. *arXiv preprint arXiv:1505.05424*, 2015.

[5] Jian-Feng Cai, Emmanuel J. Candès, and Zuowei Shen. A singular value thresholding algorithm for matrix completion. *Siam Journal on Optimization*, 20(4):1956–1982, 2010.

[6] Jie Chang, Zhonghao Lan, Changmao Cheng, and Yichen Wei. Data uncertainty learning in face recognition. In *2020 IEEE/CVF Conference on Computer Vision and Pattern Recognition (CVPR)*, pages 5710–5719, 2020.

[7] Jiwoong Choi, Dayoung Chun, Hyun Kim, and Hyuk-Jae Lee. Gaussian yolov3: An accurate and fast object detector using localization uncertainty for autonomous driving. In *2019 IEEE/CVF International Conference on Computer Vision (ICCV)*, pages 502–511, 2019.

[8] Roberto Cipolla, Yarin Gal, and Alex Kendall. Multi-task learning using uncertainty to weigh losses for scene geometry and semantics. In *2018 IEEE/CVF Conference on Computer Vision and Pattern Recognition*, pages 7482–7491, 2018.

[9] Michael Havbro Faber. On the treatment of uncertainties and probabilities in engineering decision analysis. *Journal of Offshore Mechanics and Arctic Engineering-transactions of The Asme*, 127(3):243–248, 2005.

[10] Yarin Gal and Zoubin Ghahramani. Dropout as a bayesian approximation: representing model uncertainty in deep learning. In *ICML'16 Proceedings of the 33rd International Conference on International Conference on Machine Learning - Volume 48*, pages 1050–1059, 2016.

[11] 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.

[12] Jiang Guo, Darsh J Shah, and Regina Barzilay. Multi-source domain adaptation with mixture of experts. In *Proceedings of the 2018 Conference on Empirical Methods in Natural Language Processing*, pages 4694–4703, 2018.

[13] Judy Hoffman, Mehryar Mohri, and Ningshan Zhang. Algorithms and theory for multiple-source adaptation. pages 8246–8256, 2018.

[14] Judy Hoffman, Eric Tzeng, Taesung Park, Jun-Yan Zhu, Phillip Isola, Kate Saenko, Alexei A. Efros, and Trevor Darrell. Cycada: Cycle-consistent adversarial domain adaptation. pages 1989–1998, 2018.

[15] Jonathan J. Hull. A database for handwritten text recognition research. 16(5):550–554, 1994.

[16] Guoliang Kang, Lu Jiang, Yi Yang, and Alexander G. Hauptmann. Contrastive adaptation network for unsupervised domain adaptation. pages 4893–4902, 2019.

[17] Alex Kendall and Yarin Gal. What uncertainties do we need in bayesian deep learning for computer vision. In *NIPS'17 Proceedings of the 31st International Conference on Neural Information Processing Systems*, volume 30, pages 5580–5590, 2017.

[18] Misha Elena Kilmer, Karen S. Braman, Ning Hao, and Randy C. Hoover. Third-order tensors as operators on matrices: A theoretical and computational framework with applications in imaging. *SIAM Journal on Matrix Analysis and Applications*, 34(1):148–172, 2013.

[19] Armen Der Kiureghian and Ove Dalager Ditlevsen. Aleatory or epistemic? does it matter? *Structural Safety*, 31(2):105–112, 2009.

[20] Tamara G Kolda and Brett W Bader. Tensor decompositions and applications. *SIAM review*, 51(3):455–500, 2009.

[21] Florian Kraus and Klaus Dietmayer. Uncertainty estimation in one-stage object detection. In *2019 IEEE Intelligent Transportation Systems Conference (ITSC)*, pages 53–60, 2019.

[22] Chen-Yu Lee, Tanmay Batra, Mohammad Haris Baig, and Daniel Ulbricht. Sliced wasserstein discrepancy for unsupervised domain adaptation. pages 10285–10295, 2019.

[23] Da Li, Yongxin Yang, Yi-Zhe Song, and Timothy M Hospedales. Deeper, broader and artier domain generalization. In *Proceedings of the IEEE international conference on computer vision*, pages 5542–5550, 2017.

[24] Yitong Li, michael Murias, geraldine Dawson, and David E. Carlson. Extracting relationships by multi-domain matching. pages 6798–6809, 2018.

[25] Mingsheng Long, Yue Cao, Jianmin Wang, and Michael Jordan. Learning transferable features with deep adaptation networks. pages 97–105, 2015.

[26] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Unsupervised domain adaptation with residual transfer networks. pages 136–144, 2016.

[27] Mingsheng Long, Han Zhu, Jianmin Wang, and Michael I Jordan. Deep transfer learning with joint adaptation networks. pages 2208–2217. PMLR, 2017.

[28] Canyi Lu, Jiashi Feng, Yudong Chen, Wei Liu, Zhouchen Lin, and Shuicheng Yan. Tensor robust principal component analysis with a new tensor nuclear norm. 42(4):925–938, 2019.

[29] Carla D. Martin, Richard Shafer, and Betsy LaRue. An order-p tensor factorization with applications in imaging. *SIAM Journal on Scientific Computing*, 35(1), 2013.

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

[31] Yingwei Pan, Ting Yao, Yehao Li, Yu Wang, Chong-Wah Ngo, and Tao Mei. Transferrable prototypical networks for unsupervised domain adaptation. pages 2239–2247, 2019.

[32] Xingchao Peng, Qinxun Bai, Xide Xia, Zijun Huang, Kate Saenko, and Bo Wang. Moment matching for multi-source domain adaptation. pages 1406–1415, 2019.

[33] Sayan Rakshit, Biplab Banerjee, Gemma Roig, and Subhasis Chaudhuri. Unsupervised multi-source domain adaptation driven by deep adversarial ensemble learning. In *German Conference on Pattern Recognition*, pages 485–498, 2019.

[34] Kuniaki Saito, Kohei Watanabe, Yoshitaka Ushiku, and Tatsuya Harada. Maximum classifier discrepancy for unsupervised domain adaptation. pages 3723–3732, 2018.

[35] Baochen Sun, Jiashi Feng, and Kate Saenko. Correlation alignment for unsupervised domain adaptation. *Domain Adaptation in Computer Vision Applications*, pages 153–171, 2017.

[36] Shiliang Sun, Honglei Shi, and Yuanbin Wu. A survey of multi-source domain adaptation. *Information Fusion*, 24:84–92, 2015.

[37] Yi-Hsuan Tsai, Wei-Chih Hung, Samuel Schulter, Kihyuk Sohn, Ming-Hsuan Yang, and Manmohan Chandraker. Learning to adapt structured output space for semantic segmentation. pages 7472–7481, 2018.

[38] Eric Tzeng, Judy Hoffman, Kate Saenko, and Trevor Darrell. Adversarial discriminative domain adaptation. 2017.

[39] Hang Wang, Minghao Xu, Bingbing Ni, and Wenjun Zhang. Learning to combine: Knowledge aggregation for multi-source domain adaptation. pages 727–744. Springer, 2020.

[40] Shaoan Xie, Zibin Zheng, Liang Chen, and Chuan Chen. Learning semantic representations for unsupervised domain adaptation. pages 5423–5432, 2018.

[41] Ruijia Xu, Ziliang Chen, Wangmeng Zuo, Junjie Yan, and Liang Lin. Deep cocktail network: Multi-source unsupervised domain adaptation with category shift. pages 3964–3973, 2018.

[42] Jason Yosinski, Jeff Clune, Yoshua Bengio, and Hod Lipson. How transferable are features in deep neural networks? pages 3320–3328, 2014.

[43] Han Zhao, Shanghang Zhang, Guanhang Wu, José M. F. Moura, João Paulo Costeira, and Geoffrey J. Gordon. Adversarial multiple source domain adaptation. pages 8559–8570, 2018.

[44] Sicheng Zhao, Bo Li, Pengfei Xu, and Kurt Keutzer. Multi-source domain adaptation in the deep learning era: A systematic survey. *arXiv preprint arXiv:2002.12169*, 2020.

[45] Sicheng Zhao, Guangzhi Wang, Shanghang Zhang, Yang Gu, Yaxian Li, Zhichao Song, Pengfei Xu, Runbo Hu, Hua Chai, and Kurt Keutzer. Multi-source distilling domain adaptation. 34(7):12975–12983, 2020.

[46] Jun-Yan Zhu, Taesung Park, Phillip Isola, and Alexei A. Efros. Unpaired image-to-image translation using cycle-consistent adversarial networks. pages 2242–2251, 2017.